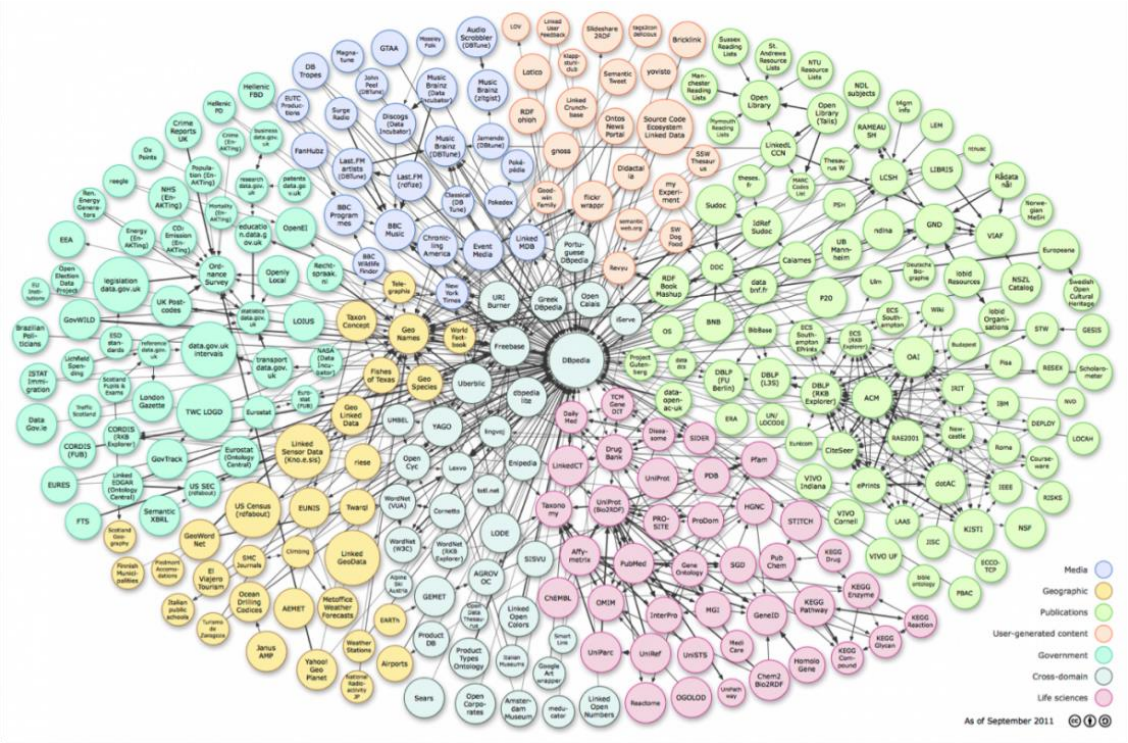
Hes·so VALAIS WALLIS
Haute Ecole de Gestion & Tourisme
Hochschule für Wirtschaft & Tourismus

Filière Informatique de gestion
Studiengang Wirtschaftsinformatik

Travail de Bachelor 2014

Interface de visualisation innovante du Linked Data



Étudiant : Quentin Oberson

Professeure : Anne Le Calvé

Déposé, le 28 juillet 2014

Résumé

Deux applications résultent de ce travail de Bachelor. La première est un outil utilisé pour extraire des données provenant du web sémantique. La deuxième application est capable de créer des visualisations en se basant sur des données au format JSON.

Figure 1 Processus de création de visualisations

L'outil d'administration est capable de récupérer des données issues de plusieurs endpoints à la fois. Il permet à l'utilisateur de naviguer à l'intérieur de ces données et d'extraire les informations qu'il juge pertinentes.

L'application de développement de visualisations importe des données avec lesquelles elle crée différentes visualisations. Par la suite, ces visualisations peuvent être déployées sur des sites internet et être consultées par les visiteurs.

L'objectif principal de ces deux outils est de permettre d'utiliser les données appartenant au web sémantique d'une façon simple. Sans grande connaissance dans le domaine, l'utilisateur doit être en mesure de parcourir chacune des étapes jusqu'à la publication de visualisations.

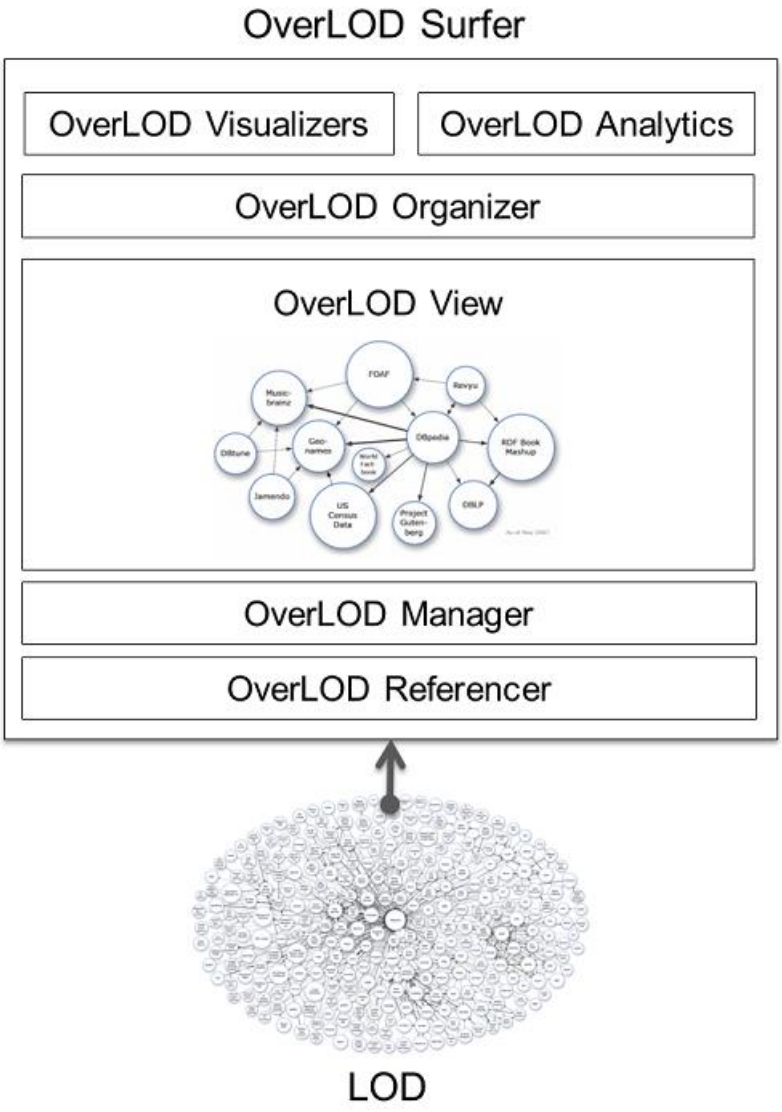
Mots-clés : linked data, web sémantique, visualisation, RDF, SPARQL, JavaScript

Avant-propos

Ce document a pour but de synthétiser et de présenter le travail effectué dans le cadre du travail de Bachelor de la filière Informatique de Gestion de la Haute-École spécialisée de Suisse occidentale (HES-SO). L'étudiant a suivi sa formation sur le site de Sierre en Valais pendant 3 années à temps plein.

Le sujet proposé pour ce travail de Bachelor a attisé mon intérêt car je désirais en savoir plus sur le fonctionnement des données liées et la façon dont les ordinateurs pourront à l'avenir alléger le travail de traitement de l'information à la place des êtres humains.

Ce travail de Bachelor s'inscrit dans le Projet OverLOD Surfer (voir annexe). Le web sémantique est une technologie puissante mais difficile à utiliser pour les personnes non-initiées. L'objectif du projet OverLOD Surfer est de créer un outil permettant d'analyser les données du web sémantique le plus facilement possible en cachant la complexité technique aux utilisateurs.



Cette image décrit l'acheminement des données extraites du web sémantique jusqu'à la présentation finale. Après que les données aient été rendues accessibles sur le web, le « referencer », configuré à l'aide du « manager », extrait ces données et standardise leurs structures en vue d'être utilisées par des développeurs. Ces derniers vont alors utiliser les données pour les analyser ou créer des visualisations.

Remerciements

Les personnes suivantes sont remerciées pour leur aide, leur participation et/ou leur soutien dans l'élaboration de ce travail de Bachelor :

Anne Le Calvé, pour son soutien, son optimisme et sa confiance accordée.

Zhan Liu, pour ses conseils avisés et son temps investi.

Fabian Cretton, pour sa documentation de départ.

David Russo, qui prend toujours la peine d'apporter son aide en cas de problème.

Mes collègues du TechnoPôle, pour leur soutien, leur aide et leur bonne humeur.

Table des matières

1 Introduction

Le web sémantique est une technologie prometteuse qui permet de donner une partie de l'étape d'analyse des informations faite par les humains aux ordinateurs. Les avantages qui en découlent sont une économie de temps, d'argent et d'effort. Bien que les entrepreneurs comprennent les intérêts que peut apporter le web sémantique dans le monde économique, la complexité technique de cette nouvelle technologie empêche son utilisation à grande échelle.

Dans ce contexte, ce travail de Bachelor tente d'apporter une approche nouvelle dans la façon d'exploiter le web sémantique. En cachant la complexité de la technologie, son objectif est de faciliter l'exploitation de données provenant du web sémantique. À terme, de l'extraction à la visualisation des données, l'utilisateur bénéficie des atouts du web sémantique sans s'en rendre compte.

Pour parvenir à ce résultat, ce travail de Bachelor s'articule autour des deux applications. Le rôle de la première est de pouvoir extraire des données issues du web sémantique. Tandis que la deuxième application permet d'étudier les données grâce à des visualisations.

1.1 Introduction au web sémantique

Ce chapitre introductif a pour but de familiariser le lecteur avec les concepts de base employés dans la technologie du web sémantique.

Le web sémantique est un mouvement guidé par le W3C [1] qui vise à transformer le web actuel, composé principalement de documents, en un « web des données ». La manipulation des données avec un niveau de granularité très faible peut profiter à un très grand nombre d'applications. L'objectif fondamental de cette démarche est de promouvoir le partage des données et leur réutilisabilité dans de nombreuses applications.

Malgré quelques détracteurs qui doutent de la faisabilité de cette nouvelle idéologie à terme, le reste du monde scientifique semble plutôt apercevoir de nouvelles opportunités dans le web sémantique. Des applications dans le domaine de l'industrie, de la biologie et

autres recherches dans les sciences humaines apportent d'ors et déjà la preuve que le web sémantique peut apporter une dimension nouvelle dans la recherche d'informations [2].

Toutefois, le web sémantique est confronté à quelques problèmes qui l'empêchent de s'intégrer aisément parmi le grand public.

Tout d'abord, la quantité immense de données que représente le web à l'heure actuelle est un défi. En effet, il est difficile d'appréhender une telle diversité de concepts, de sens et de langues que le web sémantique doit englober en totalité.

Ensuite, la désambiguïsation de chaque mot est une lourde tâche. C'est parce que un même mot dans un contexte différent peut radicalement changer de sens qu'il est difficile de classer les mots peu précis.

De plus, dans les domaines médicaux par exemple, il est crucial de traiter les données incertaines avec une extrême prudence. Se fier uniquement à des données statistiques résultant de calculs de probabilité complexes ne peut pas remplacer le sixième sens des médecins.

En outre, l'inconsistance des données, c'est-à-dire des incohérences, est un problème qui peut surgir à l'intérieur de domaine large et complexe. Lorsque la machine rencontre une incohérence, il en résulte généralement des informations erronées dont la cause est difficile à découvrir.

Enfin, au centre de toutes ces difficultés, se trouvent des personnes qui, intentionnellement ou non, rendent des données incorrectes publiques et contredisent d'autres sources d'informations [2].

Le « Linked Data », qui est l'une des composantes du web sémantique, est un mécanisme qui vise à relier entre elles les données pour les rendre réutilisables au maximum. Par exemple, l'entité « Genève » serait reliée au concept « Suisse » avec une propriété « ville de ». De cette manière, par déduction logique, les machines seraient capables d'affirmer que tout ce qui se trouve à Genève, est situé à l'intérieur de la Suisse. Ainsi, l'entité « Genève » serait réutilisée à chaque description et toutes les données y faisant référence pourraient être extraites [3].

Le « Linked Open Data » consiste à rendre des données reliées libres de droit. Dans la philosophie libre, les restrictions apportées par les droits d'auteur sont abolies. Ainsi, ces données peuvent être utilisées, adaptées et republiées par tout le monde [4].

Pour parvenir aux objectifs du web sémantique décrits plus haut, le W3C a mis en place plusieurs protocoles qui ont pour but de standardiser la manière d'utiliser le web sémantique à travers le monde. RDF, OWL, SPARQL, JSON-LD sont quelques-unes des standardisations que ce travail va utiliser. Pour pouvoir mieux aborder la suite de ce document, des explications succinctes sur ces notions vont être apportées dans les sections qui suivent.

1.1.1 RDF

[7] Pour que des données puissent être utilisables efficacement, les ordinateurs ont besoin de pouvoir se repérer grâce à la façon dont elles ont été structurées. Dans le cadre du web sémantique, les structures de données suivent le format RDF. Ce format permet de décrire des triplets. Les triplets sont formés d'un sujet, d'un prédicat et d'un objet. Par exemple, dans la phrase suivante : « L'auteur du Seigneur des Anneaux est J.R.R. Tolkien », le triplet est facilement reconnaissable. Voici comment représenter le triplet [5]:

Sujet	Predicat	Objet
Le Seigneur des Anneaux	auteur	J.R.R. Tolkien

Figure 2 Représentation d'un triplet

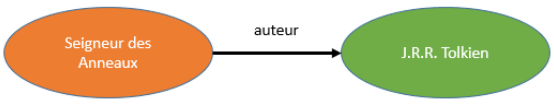


Figure 3 Autre représentation d'un triplet

De plus, pour que les données puissent être liées entre elles et facilement utilisables par tout le monde, le format RDF considère chaque partie d'un triplet comme une ressource en ligne. Pour les identifier, RDF fait appel à des URIs. Comme chaque URI est unique, elles permettent d'éviter toute ambiguïté quant au sens d'un triplet.

http://dbpedia.org/resource/Amazon_River

Figure 4 Exemple d'URI

Les données RDF sont le point de départ de l'application. Les interfaces à l'intention de l'utilisateur final seront construites à partir de ces données brutes.

1.1.2 Ontologies

[8] Les ontologies servent à donner des renseignements sur la signification exacte d'un concept. En outre, elles définissent les relations possibles avec d'autres concepts. Les ontologies sont utilisées pour identifier précisément les ressources. Elles désambigüisent les termes de façon à éviter les erreurs d'interprétation.

Illustrons cette explication: Imaginons des données servant à comparer les températures sur les planètes du système solaire. Sans une ontologie, le terme « Mercure » est ambigu. S'agit-il de la planète, de l'élément chimique ou du Dieu romain ? L'ontologie indiquera lequel dispose d'une température atmosphérique, d'une distance par rapport au Soleil, de planètes voisines, etc.

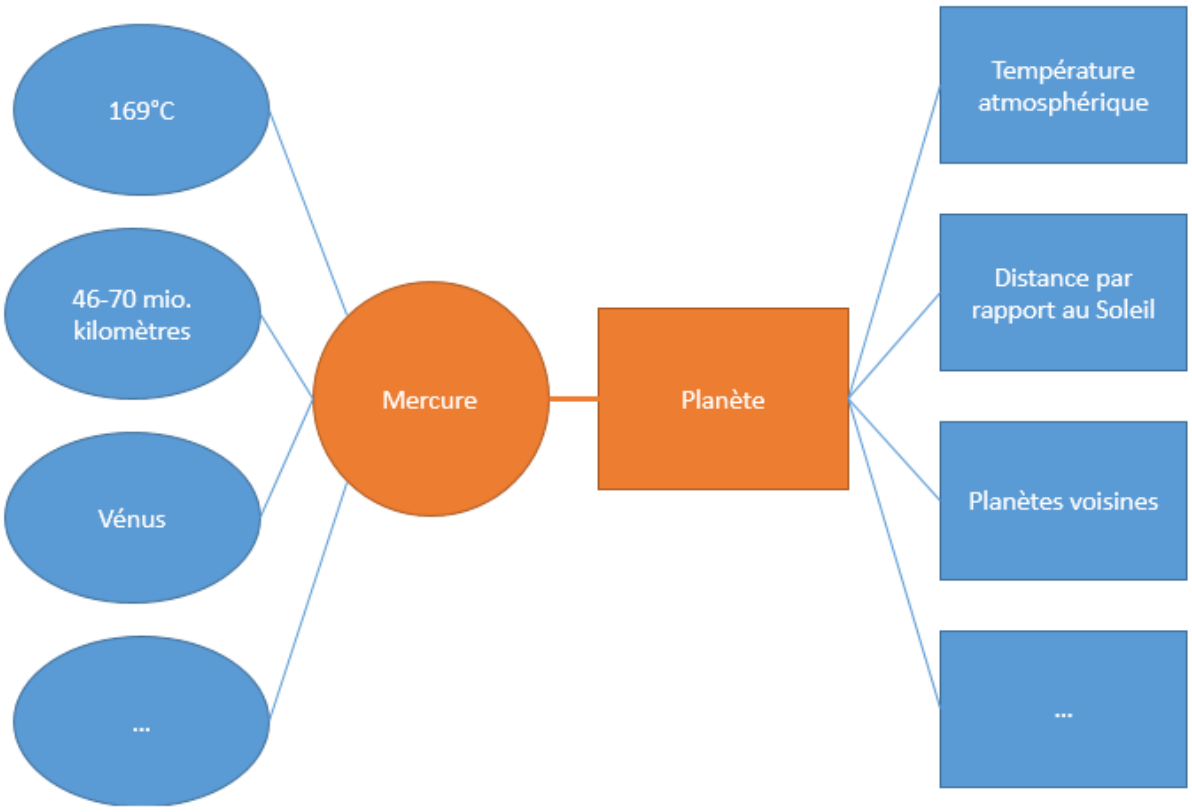


Figure 5 Mercure possède les propriétés de l'ontologie Planète

1.1.3 OWL

[6] OWL est un format standardisé de publication et de partage d'ontologies soutenu par le W3C. Les schémas RDF qui décrivent la structure des données contenues dans un document RDF sont soumis à plusieurs contraintes. Par exemple, ils ne détectent pas d'erreur si les parents d'une personne sont des animaux ou si une personne a plusieurs mères. Si une personne est le fils d'une autre personne, ils ne sont pas capables de déduire que cette dernière est automatiquement le parent de la première personne.

En raison de ces limitations, les ontologies sont écrites avec OWL qui est un langage plus complet que les schémas RDF. OWL est suffisamment détaillé pour passer outre ces limitations et parvenir à décrire très exactement les rapports que possèdent les concepts entre eux.

OWL est indirectement utilisé dans le projet car l'application ne gère pas les ontologies, mais s'en sert pour renseigner l'utilisateur sur le sens des données qui lui sont présentées.

1.1.4 SPARQL

[9] SPARQL est un langage de requêtage promu par le W3C. Il a été créé pour extraire des données RDF.

Les données se trouvant dans des fichiers RDF sont stockées dans des bases de données sémantiques appelés « triplestore » [39]. La porte d'entrée, qui reçoit les requêtes destinées à la base de données, est un « endpoint » [40]. Une requête SPARQL peut demander des données à un endpoint en utilisant le protocole HTTP.

Il est intéressant de noter que dans sa version 1.0, SPARQL n'est capable que de lire les données contenues dans des triplestores. La version 1.1, disponible depuis 2013 [41], a amené une solution à quelques limitations de la version précédente tel que les agrégations, les sous-requêtes ou les négations et permet dorénavant de transformer les données stockées à distance.

Afin de comprendre un peu plus concrètement comment SPARQL fonctionne, un exemple simple peut être expliqué.

Voici à quoi ressemble une requête SPARQL :

```
PREFIX rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
PREFIX dbp: http://dbpedia.org/ontology/
SELECT *
WHERE {
   ?city rdf:type <http://dbpedia.org/class/yago/CitiesInTexas> ;
        dbp:populationTotal ?popTotal ;
        dbp:populationMetro ?popMetro .
}
ORDER BY DESC(?popMetro)
```

Les deux premières lignes spécifient à quoi fait référence les raccourcis « rdf » et « dbp ». Ces raccourcis sont utilisés afin que la requête soit moins verbeuse et plus facilement compréhensible. Le « SELECT * » signifie que toutes les variables utilisées dans la requête seront affichées au final. La clause « WHERE {} » qui suit sert à orienter la recherche. Dans le cas présenté, la variable « ?city » sert à stocker toutes les ressources qui ont pour type « CitiesInTexas ». Les variables « popTotal » et « popMetro » servent à mémoriser respectivement la population totale et la population métropolitaine de chaque ville.

L'application se sert de SPARQL pour extraire des données qui seront filtrées puis synthétisées dans un graphique.

1.1.5 JSON-LD

[10] JSON est un format de sérialisation et de partage de données similaire à XML, RDF et CSV par exemple. Il permet d'apporter une structure logique qui est facilement compréhensible par un humain et aisément manipulable par un ordinateur. Cette double capacité en fait un format très apprécié des technologies du web.

JSON-LD dispose d'une structure particulière étudiée pour faciliter le traitement de données liées (Linked Data). Concrètement, chaque ressource décrite dans le document se verra attribué un contexte. Ce contexte, qui se réfère à une ontologie, permet de clarifier le concept de la ressource décrite.

```
{
  "@context": {
    "name": "http://schema.org/name",
    "description": "http://schema.org/description",
    "image": {
      "@id": "http://schema.org/image",
      "@type": "@id"
    },
    "geo": "http://schema.org/geo",
    "latitude": {
      "@id": "http://schema.org/latitude",
      "@type": "xsd:float"
    },
    "longitude": {
      "@id": "http://schema.org/longitude",
      "@type": "xsd:float"
    },
    "xsd": "http://www.w3.org/2001/XMLSchema#"
  },
  "name": "The Empire State Building",
  "description": "The Empire State Building is a 102-story landmark in New York City.",
  "image": "http://www.civil.usherbrooke.ca/cours/gci215a/empire-state-building.jpg",
  "geo": {
    "latitude": "40.75",
    "longitude": "73.98"
  }
}
```

Figure 6 Le contexte donne un sens aux propriétés que la ressource utilise

JSON-LD sera utilisé dans l'application pour échanger des données entre l'interface d'administration et l'interface d'utilisation.

1.2 Introduction aux interfaces de visualisation

Les interfaces, au sens informatique du terme, servent aux humains à communiquer avec les ordinateurs. Par exemple, les souris et les claviers permettent aux utilisateurs d'effectuer des actions sur les machines.

Les interfaces graphiques sont utilisées pour décrire visuellement ce qu'il se passe à l'intérieur de l'ordinateur. Ainsi, après une action, l'utilisateur voit directement le résultat s'afficher sur son moniteur. Pour des raisons de simplicité, l'interface graphique a remplacé la ligne de commande dans les années 1980 [11].

Le domaine spécifique des interfaces graphiques de ce travail est la visualisation de données. L'objectif principal de cette application va être de répondre à la question : **« Comment représenter efficacement une quantité importante de données ? »**.

1.2.1 Création de visualisations

La première étape pour afficher des informations est de disposer de données. Comme expliqué précédemment, le projet OverLOD Surfer s'intéresse essentiellement aux données liées. C'est pourquoi la première tâche du programme va être d'extraire des données et de les filtrer.

Une condition importante doit être atteinte avant le traitement de ces informations. En effet, l'ordinateur a besoin de connaître le format des données utilisé sans quoi il est incapable d'effectuer des opérations dessus. JSON-LD est le format qui a été choisi pour transporter les données. Le format CSV, utilisé pour représenter des tableaux avec des données séparées par des virgules et des points-virgules, ne permet pas de construire des arborescences qui font parties intégrante du web sémantique. Bien que XML supporte également les structures de données hiérarchiques, JSON possède quelques propriétés avantageuses dans le cas d'utilisation du projet [12]:

- ✓ JSON est plus facile à lire en tant qu'humain. La structure de données est moins verbeuse.
- ✓ JSON est plus orienté « données » que XML. Il s'intègre plus facilement dans un environnement orienté objet.
- ✓ Il est plus facile de convertir du texte au format XML en JSON que l'inverse.

Enfin, comme expliqué dans le chapitre sur le format JSON-LD, cette extension de JSON permet d'intégrer des ontologies aux données.

Pour toutes ces raisons, JSON-LD est le format privilégié pour les échanges de données issues du web sémantique.

Une fois les données acquises, il faut penser à la façon de les représenter graphiquement. Un très large panel de visualisations est disponible gratuitement sur le web. Chacune met l'accent sur un aspect à mettre en évidence, que ce soit la proportionnalité entre les données, la temporalité, l'évolution des données, les positions géographiques, l'analyse de

beaucoup de paramètre en même temps, etc. Le développeur qui se chargera de cette étape est familiarisé avec les visualisations de base.

Une fois la visualisation sélectionnée, il va falloir trouver une manière de la dessiner sur la page web en partant des données récoltées. À cette étape, deux options sont disponibles : coder cette étape soi-même ou faire appel à une librairie existante. Si la fonctionnalité nécessaire existe déjà gratuitement, il est plus préférable de tirer profit d'un outil ayant déjà été développé. L'application se servira par conséquent de plusieurs librairies, chacune consacrée à une tâche bien précise.

1.2.2 Règles à respecter

Développer une interface graphique demande des compétences dans plusieurs domaines [13]:

- ✓ De la discipline (maîtrise de la plateforme, des outils, etc.)
- ✓ De la science (fonctionnement interne complexe)
- ✓ De l'art (l'interface doit être intuitive, plaisante, facile à prendre en main)

De manière générale, les développeurs d'interfaces sont confrontés à trois problèmes récurrents :

- ✓ Il est difficile de se concentrer sur un problème et sur le design en même temps
- ✓ Il est difficile de comprendre et d'exploiter la programmation événementielle
- ✓ Il est difficile d'utiliser des objets de façon instinctive sur un écran, par exemple la disquette symbolise la sauvegarde, un bouton donne l'impression que l'on peut appuyer dessus, etc.

Une application efficace devrait susciter une compréhension instinctive et non demander une longue phase d'apprentissage. Pour y parvenir, l'une des méthodes consiste à se mettre à la place de l'utilisateur final avant de s'intéresser au fonctionnement du programme.

Dans l'absolu, l'utilisateur est capable de dialoguer avec l'ordinateur instinctivement et de voir les résultats de ses actions clairement. Autrement dit la complexité du programme est cachée à l'utilisateur. Pour qu'un élément visuel soit instinctif à utiliser, il faut utiliser des métaphores et des analogies avec le monde réel. Par exemple, une porte symbolise la « sortie » d'un bâtiment et par analogie, l'utilisateur comprend automatiquement que ce bouton sert à « sortir » du programme. Pour découvrir quels sont les symboles que les utilisateurs comprennent facilement, les observer pendant qu'ils utilisent le programme permet efficacement de se rendre compte de quelles techniques font sens chez eux.

Enfin, voici 10 règles de base à respecter pour créer une interface graphique [13**Erreur ! Source du renvoi introuvable.**] :

1) L'utilisateur doit être capable d'anticiper le comportement d'un élément de par ses propriétés visuelles. Si un bouton réagit à un clic de souris, tous les boutons du programme doivent réagir de la même façon pour garder une cohérence.

2) L'utilisateur doit être capable d'anticiper le comportement du programme en se servant de son expérience passée sur d'autres applications. Il est important de garder les habitudes de l'utilisateur.

3) Chaque alerte et chaque erreur que le programme génère doit représenter une opportunité d'amélioration.

4) Il est important que l'utilisateur puisse voir directement le résultat de son action.

5) Le programme doit être ouvert à l'exploration. Grâce à des boutons « action précédente » par exemple, l'utilisateur ne doit pas avoir peur de tester des manipulations et doit se sentir compétent.

6) Le programme doit être évident à utiliser tout seul. Si une aide est requise, c'est un signe d'amélioration.

7) L'utilisation de sons, d'animations et autres multimédias peuvent aider à comprendre le fonctionnement d'un programme. Cependant, ils ne doivent pas être

indispensables pour permettre à des personnes handicapées (sourds, muets, etc.) de l'utiliser également.

8) La personnalisation de l'application aide les utilisateurs à s'orienter dans le programme.

9) Les actions bloquées, comme l'interdiction de fermer une fenêtre avant la fin du processus, sont à éviter car pénibles pour les utilisateurs.

10) L'interface doit être pensée pour permettre aux utilisateurs de se concentrer sur leurs tâches et non pas sur l'outil qu'ils utilisent.

1.3 Structure du document

La première partie de ce document s'est concentrée sur le contexte technologique dans lequel s'inscrit ce projet. En effet, il est essentiel d'acquérir les bases techniques pour comprendre les enjeux et les difficultés de ce projet que l'étudiant a été amené à rencontrer.

La seconde partie couvre l'étape d'analyse que l'étudiant a dû effectuer en vue de préparer l'exécution de ce travail de Bachelor. Cette étape sert à étudier plusieurs solutions techniques que l'étudiant a à sa disposition. Le choix des librairies utilisées est expliqué à l'aide d'un comparatif basé sur plusieurs critères.

Enfin, la dernière partie retrace l'évolution du démonstrateur. Les besoins des utilisateurs, les difficultés techniques sous-jacentes, les solutions trouvées ainsi que les justifications des choix sont présentés. Le développement du projet est réalisé en se basant sur une méthodologie itérative afin de permettre une flexibilité des besoins de l'utilisateur final.

2 État de l'art

Avant de réfléchir à la manière d'atteindre les objectifs qui ont été fixés, il est important d'observer le travail d'autres équipes dans le même domaine technologique. Cela permet, premièrement, de mieux comprendre et anticiper les difficultés sous-jacentes à la réalisation d'un prototype et, deuxièmement, de donner de nouvelles idées grâce notamment à un point de vue différent et une approche de la problématique nouvelle.

2.1 Analyse de l'existant

Le terme « existant » signifie un outil qui répond entièrement ou partiellement à la problématique posée. Dans le cadre de cette analyse, il est fondamental de se voir en tant qu'utilisateur final. Aucune compétence technique ne doit être requise pour faire fonctionner les outils.

Après quelques recherches, voici une analyse succincte des outils sélectionnés.

2.1.1 SIG.MA

SIGMA est un service en ligne qui permet d'explorer les données contenues dans le web sémantique. [14] SIGMA utilise le moteur de recherche Sindice [15] spécialisé dans la recherche de données sémantiques. Sindice cherche des données sur le web de plusieurs façons et s'actualise chaque minute.

Le moteur fonctionne grâce à un mot-clé entré par l'utilisateur. Le moteur va ensuite chercher toutes les données ayant un rapport avec ce mot-clé sur plusieurs endpoints. Dès qu'une correspondance est trouvée, SIGMA affiche les informations dans un tableau. Les données peuvent être du texte, des nombres, des liens, des images et autres. L'utilisateur peut ensuite filtrer les concepts présentés pour affiner sa recherche [16].

Figure 7 Page de recherche de SIGMA

Mis dans le contexte de la problématique de ce travail, SIGMA bénéficie de points intéressants à s'inspirer et de faiblesses à améliorer. L'un des éléments les plus intéressants est le nombre impressionnant de sources différentes que le moteur rapporte. La contrepartie à cette abondance est le manque de structuration logique. Il est très difficile pour l'utilisateur de trier efficacement les données utiles des données anecdotiques ou hors contexte. En outre, SIGMA a prévu la possibilité d'inclure un lien des résultats des requêtes sur d'autres pages web. Cependant il est impossible d'exporter les données sous quelconque formats pour d'autres utilisations directement.

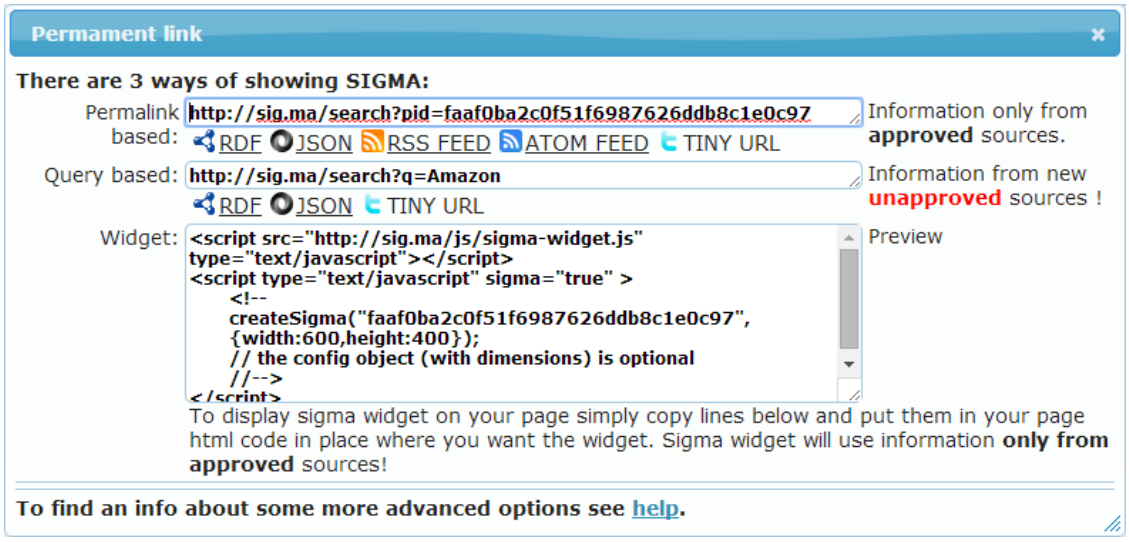


Figure 8 Différentes solutions pour exporter les résultats

Finalement, l'interface utilisateur reste très classique. L'utilisateur peut voir le type de données avec la valeur correspondante mais les relations entre les données sont inexistantes. Ce dernier point, qui représente l'un des principes fondamental du web sémantique, fait défaut à l'aspect innovateur de la problématique présentée dans ce document.

2.1.2 Visual SPARQL & Visual RDF

Visual SPARQL et Visual RDF sont deux projets créés par Alvaro Graves [17]. Les codes sources et des versions de démonstration sont disponibles sur son site internet.

Visual SPARQL est un programme en ligne qui permet de visualiser les résultats de requêtes SPARQL sous forme de graphes. Une fois la requête SPARQL entrée par l'utilisateur, l'application va dessiner des nœuds et des relations représentant des triplets.

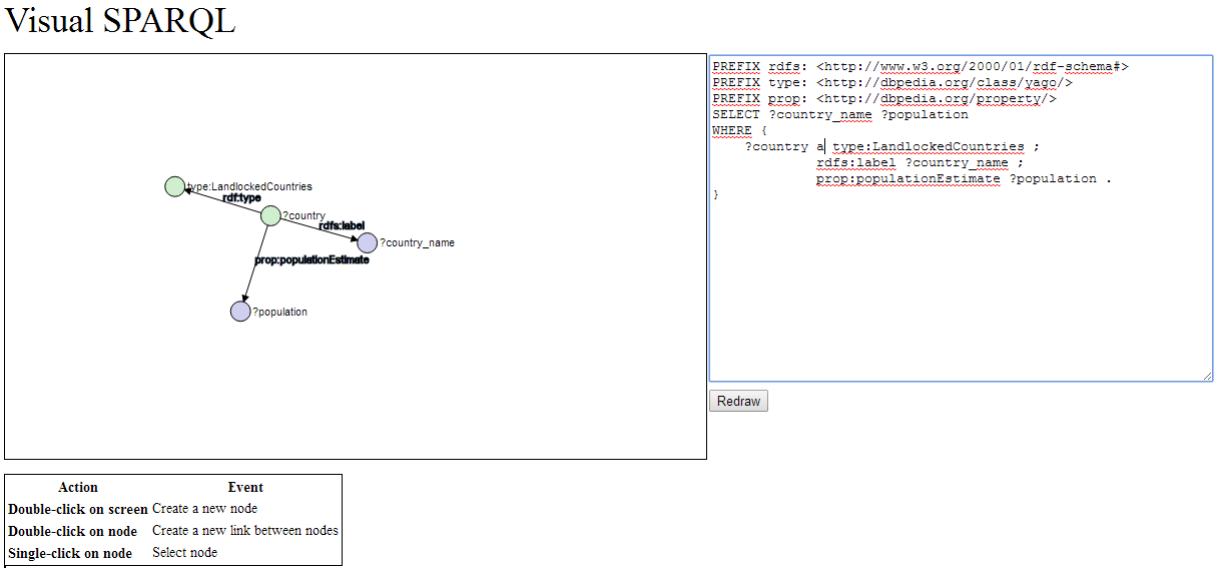


Figure 9 À partir de la requête à droite, le graphe à gauche se génère

La volonté derrière cette idée est de synthétiser une grosse quantité d'information sur une seule page. Pour explorer manuellement les données disponibles sur le web sémantique, l'utilisateur doit consulter un grand nombre de pages descriptives de ressources. C'est une tâche très longue et pénible car ces pages sont très denses et difficilement consultables. En créant une image regroupant toutes les informations

récupérées, l'utilisateur a désormais cette masse d'information à portée de main directement.

Bien que l'idée soit très intéressante, ce prototype contient des défauts importants. Le langage de requêtage SPARQL, qui n'est malheureusement pas totalement supporté, constitue le premier élément péjoratif. En effet, il est parfois impossible d'utiliser certaines fonctions SPARQL pour affiner sa requête. Par exemple, le mot-clé « FILTER », qui sert à filtrer les ressources qui ne satisfont pas une condition, empêche la requête d'être utilisée [18].

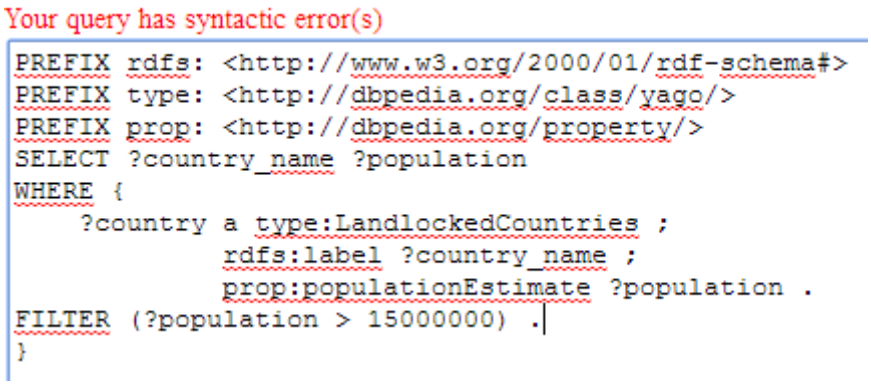


Figure 10 Le message d'erreur en rouge prouve que l'outil ne supporte pas complètement SPARQL

Le texte en rouge prévient que la requête ne peut être exécutée en l'état. Cela mène à un problème d'ergonomie évident. Le contrôle syntaxique s'active à chaque modification du champ et affiche à l'utilisateur le message précédemment cité lorsqu'un élément n'est pas supporté. Malheureusement, le texte en rouge ne cesse de déplacer le champ d'entrée vers le bas à chaque apparition. L'utilisateur est par conséquent contraint de suivre le champ avec le pointeur de sa souris pour se déplacer à l'intérieur du champ. Certes, utiliser les touches directionnelles du clavier évite le problème, mais le déplacement demeure alors très lent et peu de personnes s'en servent.

Enfin, le résultat final ne correspond pas aux données retournées par la requête. En réalité, l'exemple précédent retourne un grand nombre de données dont seuls les noms des variables et les prédicats entre les nœuds sont affichées à l'utilisateur.

country_name	population
"乌兹别克斯坦"@zh	29559100
"Usbekistan"@de	29559100
"Uzbekistan"@en	29559100
"Uzbekistan"@sv	29559100
"Ouzbékistan"@fr	29559100
"Uzbekistan"@it	29559100
"Uzbekistán"@es	29559100
"Узбекистан"@ru	29559100
"Oezbekistan"@nl	29559100
"Uzbequistão"@pt	29559100
"Uzbekistan"@pl	29559100
"ウズベキスタン"@ja	29559100
"埃塞俄比亚"@zh	91195675
"Äthiopien"@de	91195675
"Ethiopia"@en	91195675
"Etiopia"@pl	91195675
"Ethiopië"@nl	91195675
"Etiopía"@es	91195675
"Etiópia"@pt	91195675
"Éthiopie"@fr	91195675
"Etiopia"@it	91195675

Figure 11 Données réellement retournées par le endpoint

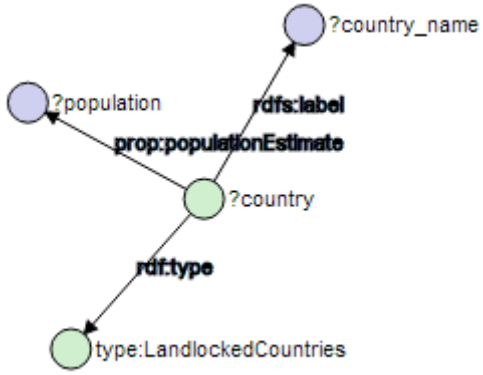


Figure 12 Graphe construit à partir des mêmes données

Visual RDF offre une représentation graphique d'une ressource RDF. L'utilisateur doit simplement entrer l'URI d'une ressource RDF et le programme va dessiner des nœuds représentant le concept central avec toutes ses relations.

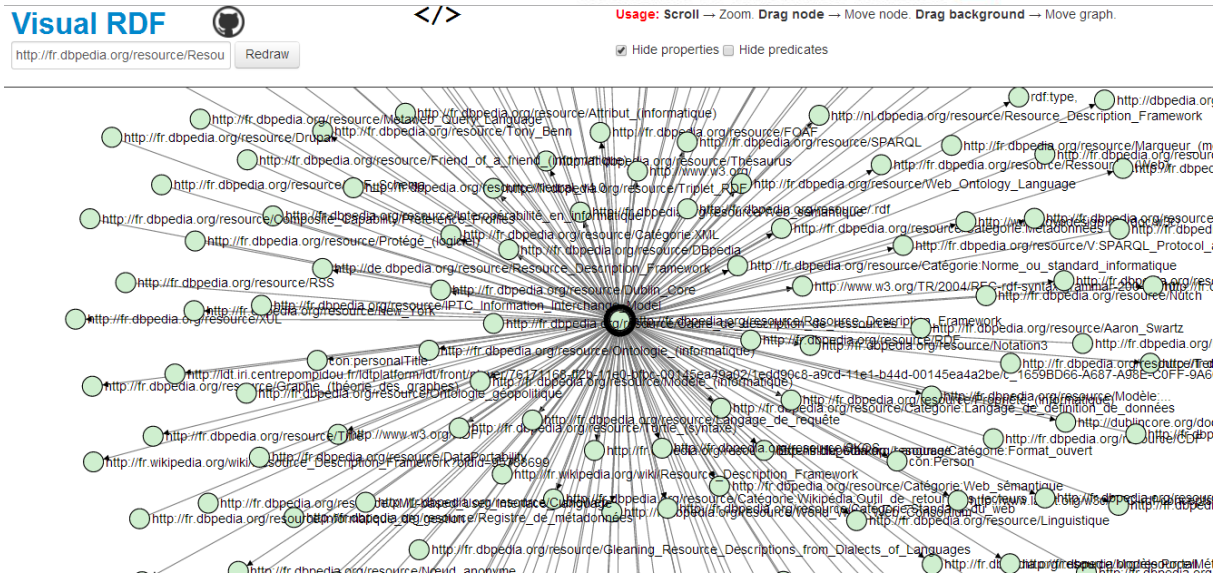


Figure 13 À partir d'une URI, Visual RDF affiche toutes les propriétés de la ressource

Contrairement à son prédécesseur, Visual RDF affiche les propriétés d'une ressource.

Le désavantage visuellement évident de ce programme est son affichage. Bien que l'interface permette de cacher certains textes et liens, le tout demeure très confus. Heureusement, l'utilisateur peut déplacer les nœuds comme il le souhaite et zoomer à volonté. Malgré tout, il est fastidieux de distinguer les informations.

Le fait que l'utilisateur ne puisse pas sélectionner les informations constitue le second problème. Bien que le texte puisse être clairement visible, il est impossible de copier les informations. Lorsque l'utilisateur essaie de sélectionner une information, le nœud se déplace avec la souris. Une boîte d'information apparaît en surimpression lorsque le pointeur de la souris reste sur un nœud, mais ce dernier disparaît lorsque l'utilisateur tente d'en copier le texte. Par conséquent, les informations affichées sont inutilisables.

Enfin, les informations sur la ressource sont incomplètes. Seules les références vers d'autres ressources ou d'autres ontologies sont affichées. Les informations brutes sont ignorées. Par exemple, la longueur du fleuve Amazone, qui est un nombre et qui ne fait référence à aucun concept, n'est pas prise en compte dans la recherche.

2.1.3 Lodlive

[19] Lodlive est un projet expérimental qui a pour but de promouvoir le Linked Data en démontrant la facilité d'accès des données sémantiques grâce à un outil de visualisation en ligne. Lodlive est également disponible en version téléchargeable. Lodlive se veut être le premier navigateur à utiliser des ressources RDF grâce à des endpoints SPARQL.

Lodlive propose plusieurs façons d'accéder aux données.

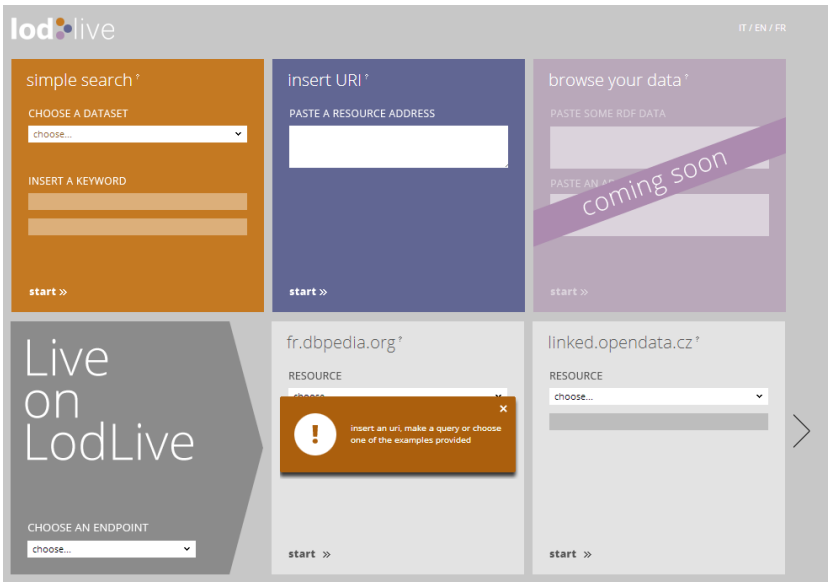


Figure 14 Page d'accueil de Lodlive

La première méthode consiste à effectuer une recherche directement depuis le site Lodlive. Après avoir spécifié le endpoint à interroger, l'utilisateur choisit les mots-clés qu'il recherche à l'aide d'un menu déroulant.

La deuxième manière d'utiliser Lodlive est de coller directement une URI d'une ressource RDF. Le moteur sémantique de Lodlive va ensuite aller chercher les informations à cette adresse.

Enfin, Lodlive propose une liste de endpoints que l'utilisateur peut interroger grâce à l'une des deux méthodes citées précédemment. À noter que chaque endpoint propose des URI exemples.

Prochainement, Lodlive sera capable de visualiser des données en les copiant directement sur un champ du site.

Une fois la ressource à visualiser définie, le moteur de Lodlive va extraire les données grâce à des requêtes SPARQL et dessiner la visualisation à l'écran.

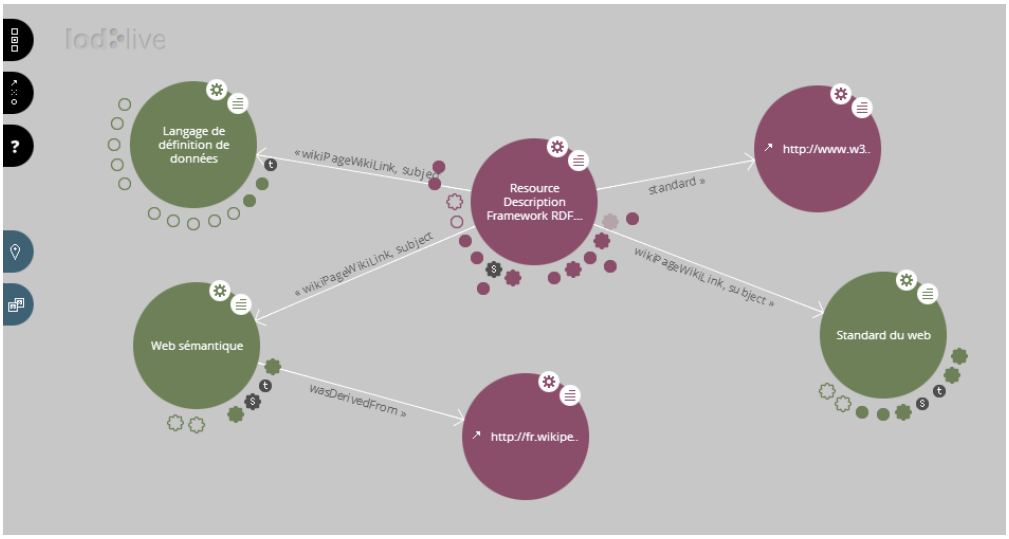


Figure 15 L'utilisateur peut contrôler l'affichage des graphes

La visualisation des triplets de données se fait sous forme de graphe. Le nœud central représente le sujet, les autres nœuds sont les objets et les liens entre les nœuds sont des prédicats (voir chapitre RDF). Chaque nœud peut être déplacé pour améliorer la disposition. Lorsque l'interface apparaît pour la première fois, seul le nœud de la ressource sélectionnée apparaît. Il suffit d'appuyer sur les petits ronds autour de la forme pour afficher les propriétés de la ressource. Les menus présents sur chaque nœud permettent entre autre de le faire disparaître, d'ouvrir la page web de la ressource, d'afficher toutes les propriétés ou de recommencer un affichage avec le nœud sélectionné comme sujet de départ.

Dans le cadre de ce travail de Bachelor, cet outil bénéficie de caractéristiques très intéressantes. Tout d'abord, la visualisation utilise des graphes pour représenter des triplets de données. Grâce aux labels présents sur chaque nœud et sur chaque relation, l'utilisateur comprend rapidement la façon dont les deux ressources sont reliées entre elles. Ensuite, pour améliorer la visibilité, l'application laisse le choix des ressources à afficher à l'utilisateur. Les ressources jugées inutiles restent de petits cercles et ne gênent pas le

confort de l'utilisateur. Enfin, le logiciel est rapide. À chaque fois qu'un lien est sollicité par l'utilisateur, le programme envoie qu'une seule requête SPARQL, plutôt que de chercher à charger le plus d'informations possible. Cela est justifié par le fait que le moteur de Lodlive ne saurait pas à quel moment s'arrêter et un grand nombre d'informations serait chargé inutilement.

En revanche, certains points empêchent Lodlive d'être directement utilisé pour le travail de Bachelor. Son principal défaut est directement lié à sa nature de navigateur. En effet, cet outil permet uniquement de voyager d'un concept à un autre. L'objectif du travail de Bachelor est d'utiliser des données du web sémantique pour réaliser une visualisation. Afin de créer un graphique, il est nécessaire de pouvoir extraire des données. Lodlive n'implémente pas cette fonctionnalité.

Pour terminer cette section, le tableau suivant récapitule les points forts et les points faibles de chaque solution.

Tableau 1 Récapitulatif de l'existant

Logiciel	Qualités	Défauts
Sigma	✓ Nombre de résultats ✓ Possibilité de créer un lien vers les résultats	× Structuration confuse × Liens entre les concepts inexistants × ***Impossible d'exporter les résultats***
Visual SPARQL & Visual RDF	✓ Innovant ✓ Nouvelle approche	× SPARQL pas supporté entièrement × Problèmes d'ergonomie × Résultats incomplets × ***Impossible d'exporter les résultats***
Lodlive	✓ Visualisation sous forme de graphe ✓ Ergonomie bien pensée ✓ Rapide	× ***Impossible d'exporter les résultats***

2.1.4 Conclusion

Chaque outil présenté dans ce chapitre a rempli son propre objectif. Bien qu'un bon nombre d'idées intéressantes puissent être réutilisé, les limitations fonctionnelles des programmes qui ont été présentés ne permettent pas de répondre complétement à la problématique de ce travail. C'est pourquoi, l'implémentation d'un prototype inspiré de ces travaux est nécessaire afin d'atteindre les objectifs de ce projet. En particulier, les fonctions sollicitées sont l'efficacité de visualisation, la sélection ainsi que l'extraction des données en vue de réaliser des analyses ultérieurement.

2.2 Méthodologies de travail utilisées

Comme expliqué plus tôt, l'absence d'outils capable de répondre directement à la problématique de ce travail de Bachelor force la conception d'une nouvelle application dont les fonctionnalités spécialement étudiées permettront d'apporter une solution de façon optimale au problème posé.

Dans le cadre de la démarche de développement d'un prototype, une méthodologie de recherche est nécessaire. Une telle méthodologie a pour but de fournir un cadre de travail structuré qui servira de référence de comparaison avec tous les autres travaux basés sur la même méthode. En effet, si tous les projets s'appuient sur une démarche similaire, il est ainsi plus aisé d'en comparer les problématiques, les difficultés et les résultats qui en découlent. En fin de compte, ces comparaisons serviront à identifier des corrélations entre les différents projets et permettront la découverte de démarches visant à éviter les difficultés expérimentées.

La méthodologie de recherche adoptée dans ce travail de Bachelor se nomme « Design Science Research Methodology » (DSRM) [20]. Cette démarche de travail a été spécialement conçue pour aider les recherches dans le secteur des systèmes d'information à trouver un cadre de référence.

La méthodologie se divise en six activités. Les activités se déroulent dans un ordre chronologique bien défini de façon à ne pas oublier un aspect important de la recherche.

2.2.1 Première activité : Identification du problème et motivation

Cette étape consiste à expliquer le domaine de recherche. Une description précise et détaillée de chaque point permet de bien comprendre la complexité que la solution va couvrir. De plus, le fait de justifier l'utilisation de la solution va motiver les développeurs et convaincre les lecteurs en plus de leur permettre de comprendre le raisonnement sous-jacent à chaque décision.

2.2.2 Deuxième activité : Définir les objectifs de la solution

Cette partie va fixer les conditions de réussite de la solution finale. Les objectifs peuvent être de deux natures différentes :

Quantitatif : mesurable avec des chiffres (par exemple un intervalle de temps)

Qualitatif : estimé grâce à des appréciations (par exemple degré de satisfaction des utilisateurs)

Ces objectifs doivent être choisis suite à la définition de la problématique. Tout en gardant un esprit réaliste et pratique, ils permettent d'apporter une solution adéquate aux problèmes posés.

2.2.3 Troisième activité : Modélisation et développement

Une fois les objectifs à atteindre fixés vient l'étape de la réflexion et de la création de la solution en elle-même. Ses fonctionnalités doivent être pensées pour résoudre tous les problèmes précédemment cités. Cette activité demande généralement des connaissances pointues dans le domaine d'activité en plus de compétences techniques à la réalisation de la solution.

2.2.4 Quatrième activité : Démonstration

La preuve de l'utilité du résultat des travaux de recherche est démontrée grâce à une application dans un domaine concret. En simulant la façon dont la solution parvient à surpasser la problématique expliquée au début du travail, cette activité montre la façon dont les objectifs ont été atteints.

2.2.5 Cinquième activité : Évaluation

L'évaluation de la solution sert à contrôler si cette dernière parvient à satisfaire tous les objectifs qui ont été fixés. Comme expliqué dans la deuxième activité, les conditions de satisfaction des objectifs peuvent être de différentes formes. Par conséquent, il n'existe pas qu'une seule manière d'évaluer les performances de la solution. Par exemple, il peut aussi bien s'agir de délai de réponse de l'application que d'avis général des utilisateurs sur l'interface. À la fin de cette phase, les responsables du projet peuvent décider s'il faut réitérer depuis la troisième activité si les exigences ne sont pas satisfaites afin d'améliorer les points faibles qui ont été ressortis lors de l'évaluation de la solution.

2.2.6 Sixième activité : Communication

Pour terminer cette méthodologie, le résultat du projet doit être communiqué aux autres équipes de recherche actives dans un domaine similaire pour qu'elles puissent s'informer des objectifs du projet, des difficultés rencontrées et les solutions qui ont été trouvées. Grâce à une approche admise communément grâce à la méthodologie DSRM, les lecteurs extérieurs n'auront pas de peine à comprendre la façon dont le projet a été guidé.

2.3 Analyse d'outils et de librairies

En appliquant la méthodologie DSRM, le processus de travail a été fixé. De l'analyse de l'existant est ressorti le besoin de développer un outil propre à la problématique qui touche ce travail de Bachelor. À ce stade, deux choix s'offrent à l'équipe de développement : créer un outil complétement ou réutiliser des applications existantes.

Un certain nombre de raisons peuvent justifier la première solution. En règle générale, le développement d'un outil du début à la fin est inévitable si aucune alternative n'existe ou si les versions existantes ne satisfont pas un objectif capital du projet.

Créer son outil propriétaire a des avantages non-négligeables. Grâce à ce choix, l'entreprise possède la maîtrise totale de l'application qu'elle utilise et peut en faire ce qu'elle veut : ajouter des fonctionnalités, la modifier, la vendre, etc. De plus, les développeurs à l'origine du projet connaissent les moindres spécificités. Ce dernier point est

intéressant afin que le produit réponde à chaque besoin de l'entreprise. Malheureusement, le développement logiciel coûte très cher en temps et en argent.

La deuxième possibilité pour l'équipe de développement est de faire appel à des librairies. Les librairies sont des outils de développement qui ont été créé pour répondre à un besoin très spécifique. À l'instar de l'architecte qui utilise chaque jour des conventions de construction qui ont été étudiées depuis de nombreuses années, l'informaticien réutilise des programmes éprouvés dont l'efficacité est assurée par une communauté suivant ses évolutions.

Malgré un investissement financier et chronophage moindre comparé à la première possibilité, l'utilisation de librairies comporte un certain nombre de revers.

Avant tout, une librairie est un produit dont l'emploi est soumis à un certain nombre de règles précisées dans la licence. La licence est employée pour protéger les droits de propriété de l'auteur sur le produit qu'il met à disposition. Concrètement, la licence précise ce que l'utilisateur a le droit de faire avec la librairie. Dans certain cas, ces limitations peuvent être contraignantes pour l'équipe de développement. Par exemple, elle peut ne pas avoir la possibilité d'adapter le code pour satisfaire un besoin très spécifique.

En outre, la qualité d'une librairie dépend essentiellement de la communauté qui gravite autour. En particulier, l'équipe de développement doit être attentive à ce que des tutoriels soient disponibles pour apprendre à utiliser la librairie pas à pas et à ce que la documentation, qui explique toutes les fonctionnalités de l'outil, soit complète et claire. Enfin, les nombreux supports d'aide sont généralement signe d'une importante communauté derrière la librairie qui découvre régulièrement de nouveaux problèmes et offre des solutions réalistes.

Avant de pouvoir choisir entre les deux possibilités, une étude sérieuse sur les librairies susceptibles d'être utiles doit être menée. Durant cette phase d'analyse, des librairies portant sur les visualisations de données, sur les animations, sur les conversions de données et sur les manipulations de requête SPARQL ont été étudiées. La recherche s'est restreinte au langage JavaScript car l'application sera utilisée sur des navigateurs internet. Ce choix est

justifié de par les données provenant du web sémantique ainsi que par la volonté d'accéder à l'application à distance.

Afin d'évaluer toutes les librairies sur la même base, la liste de critères suivante a été établie :

Complexité : Ce critère est évalué suite à un test de la librairie. Il doit donner une idée sur la facilité de prise en main de l'outil, sur ses limitations et les éventuels prérequis à son utilisation.

Documentation : La librairie doit disposer de documents complets et facilement compréhensibles capables de renseigner l'utilisateur sur les détails techniques. Lorsqu'un doute survient, l'utilisateur doit pouvoir se référer à la documentation de l'outil pour comprendre comment il doit procéder.

Tutoriels : Afin d'apprendre à se servir d'un logiciel, les tutoriels sont d'une aide très précieuses. Ces documents, qui reprennent pas à pas les étapes de conceptions d'un modèle basique, vont aider à la prise en main de l'outil. Comme il s'agit de la première approche concrète, ce critère est généralement déterminant dans le choix final de la librairie.

Communauté : La taille de la communauté est un indicateur indirect de la fiabilité de la librairie. Une communauté important témoigne d'une certaine confiance envers l'outil et son support. De plus, une masse volumineuse d'utilisateurs sera plus à même à répondre aux problèmes que les débutants rencontrent lors des premières utilisations.

Coût : Il représente un indicateur de base. L'équipe de développement doit réfléchir si les fonctionnalités du produit en justifient le coût. Dans ce travail, pour des raisons de budget limité, l'analyse porte essentiellement sur des librairies gratuites.

Licence : Un regard sur la licence est obligatoire avant d'utiliser le produit. Afin d'éviter des problèmes juridiques, l'équipe de développement doit savoir ce qu'elle a le droit de faire avec l'outil.

Les critères de documentation, de tutoriels et de communauté ont été évalués sur une échelle de trois points : 1 correspond à faible, 2 à moyen et 3 à important.

Parmi les librairies utilisées dans ce travail de Bachelor, lodash [21] et jQuery [22] ne sont pas analysées dans ce chapitre car elles sont utilisées pour des instructions basiques dans le code et sont totalement transparentes aux yeux de l'utilisateur.

2.3.1 RDF Translator

RDF Translator est un outil en ligne qui permet d'effectuer des conversions de toutes sortes. Il a été étudié pour sa fonction de transformation de JSON en JSON-LD. Les résultats des requêtes SPARQL sont manipulés en JSON dans le code JavaScript. Afin de garder les données et leurs ontologies ensemble, une conversion en JSON-LD peut être effectuée. RDF Translator peut être utilisé en tant que web service car il possède une REST API [23].

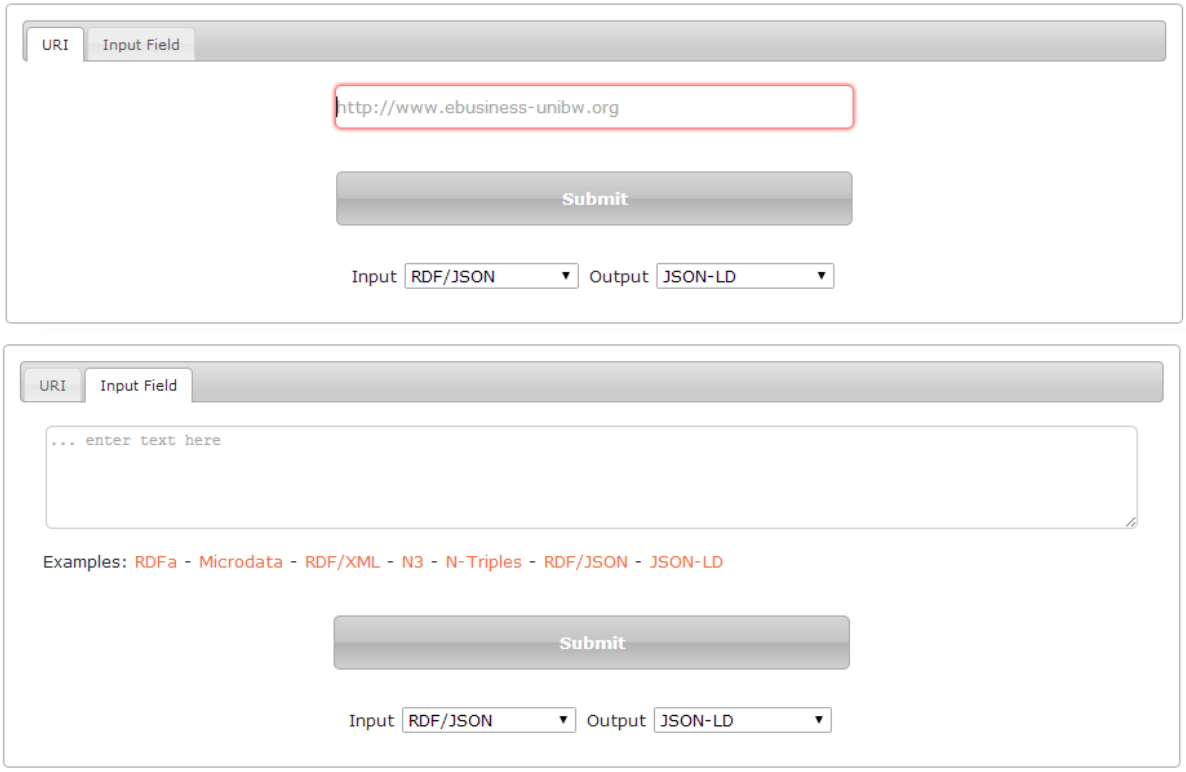


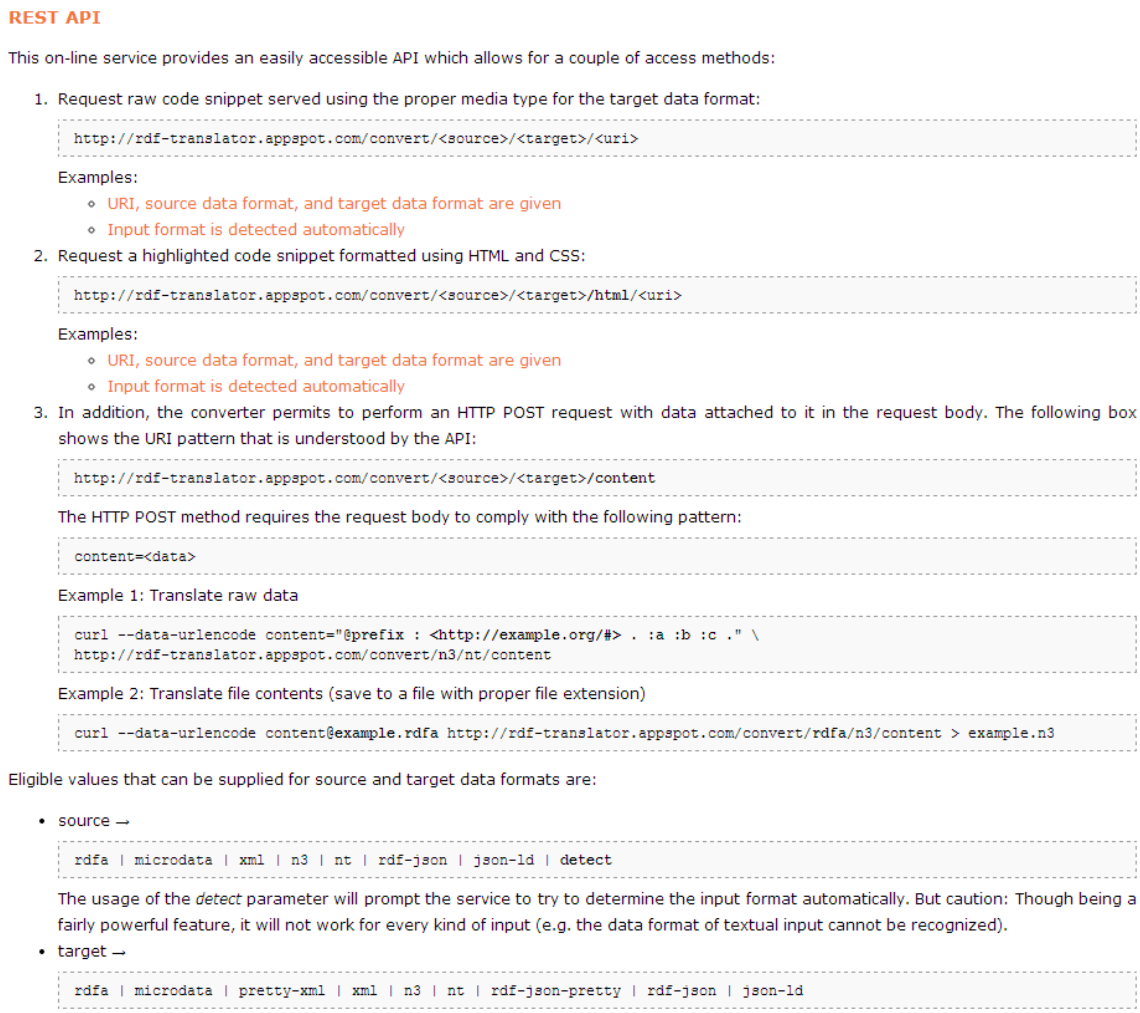
Figure 16 RDF Translator accepte les conversions d'URI et de texte directement

Complexité : Cet outil n'a pas été testé car son utilité dans le projet n'est pas encore déterminée. Toutefois, dans le cadre d'autres projets, l'étudiant a déjà fait appel à des web services. Par conséquent, l'utilisation de cet outil ne pose pas de problème particulier.

Documentation : La documentation de RDF Translator est évaluée à 1 (faible). L'outil est présenté sur une page uniquement. Un bref exemple démontre son utilisation mais il ne

peut pas être testé car il n'emploie pas de données réelles. Bien que ses fonctionnalités soient limitées, d'autres exemples plus détaillés auraient été les bienvenus.

Tutoriels : L'exemple précédent fait office de tutoriel. Ce critère est donc logiquement faible.



REST API

This on-line service provides an easily accessible API which allows for a couple of access methods:

1. Request raw code snippet served using the proper media type for the target data format:

```
http://rdf-translator.appspot.com/convert/<source>/<target>/<uri>
```

Examples:
- URI, source data format, and target data format are given
- Input format is detected automatically

2. Request a highlighted code snippet formatted using HTML and CSS:

```
http://rdf-translator.appspot.com/convert/<source>/<target>/html/<uri>
```

Examples:
- URI, source data format, and target data format are given
- Input format is detected automatically

3. In addition, the converter permits to perform an HTTP POST request with data attached to it in the request body. The following box shows the URI pattern that is understood by the API:

```
http://rdf-translator.appspot.com/convert/<source>/<target>/content
```

The HTTP POST method requires the request body to comply with the following pattern:

```
content=<data>
```

Example 1: Translate raw data

```
curl --data-urlencode content="@prefix : <http://example.org/#> . :a :b :c ." \
http://rdf-translator.appspot.com/convert/n3/nt/content
```

Example 2: Translate file contents (save to a file with proper file extension)

```
curl --data-urlencode content@example.rdfa http://rdf-translator.appspot.com/convert/rdfa/n3/content > example.n3
```

Eligible values that can be supplied for source and target data formats are:

- source →

```
rdfa | microdata | xml | n3 | nt | rdf-json | json-ld | detect
```

The usage of the *detect* parameter will prompt the service to try to determine the input format automatically. But caution: Though being a fairly powerful feature, it will not work for every kind of input (e.g. the data format of textual input cannot be recognized).

- target →

```
rdfa | microdata | pretty-xml | xml | n3 | nt | rdf-json-pretty | rdf-json | json-ld
```

Figure 4 Le tutoriel en entier de RDF Translator tient sur une image

Communauté : Une petite communauté d'utilisateurs est présente sur la toile. Elle apporte avec elle de petits tutoriels explicatifs et peut répondre aux questions sur des forums.

Coût : L'utilisation de RDF Translator est gratuite.

Licence : La licence LGPL a été appliquée à cet outil [25]. En bref, les utilisateurs ont le droit de copier le logiciel et de le distribuer librement à condition de ne pas le modifier.

En définitif, RDF Translator est un outil envisageable pour effectuer des conversions malgré ses quelques points faibles. Malheureusement, en raison de la complexité supplémentaire que représente la manipulation des données au format JSON-LD par rapport au format JSON, JSON-LD a été réservée pour une version ultérieure du programme. Elle sera réalisée sur la base du résultat de ce travail de Bachelor repris par les membres de l'équipe du projet OverLOD.

2.3.2 D3.js

[24] D3.js est une libraire graphique pour les applications JavaScript. Elle utilise les technologies HyperText Markup Language (HTML), Scalable Vector Graphics (SVG) et Cascading Style Sheets (CSS) pour donner un rendu agréable aux données.

HTML est un langage de publication web qui est interprété par un navigateur web chez le client qui reçoit les informations. Une page internet est construite en utilisant HTML.

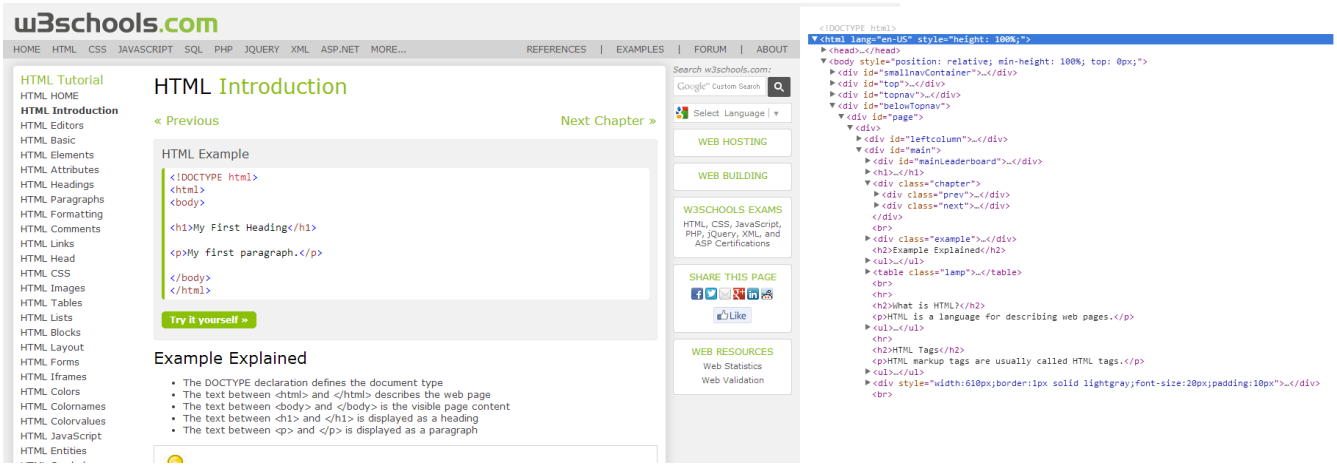


Figure 17 Une page internet et son code HTML[1]

SVG est un langage décrivant des illustrations en deux dimensions. Contrairement à une image matricielle, composée de pixels indépendants les uns des autres, une image vectorielle est formée de points reliés entre eux. L'avantage de cette technique, qui est

1 La page provient de http://www.w3schools.com/html/html_intro.asp

employé par SVG, est que l'image peut être modifiée et redimensionnée sans perte de qualité.

Figure 18 Même image en vectoriel et en matriciel[2]

Sur une page internet, les polices, les couleurs, la mise en forme et bien d'autres paramètres stylistiques sont décrits grâce à CSS. Une page HTML utilisant CSS peut être facilement sublimée.

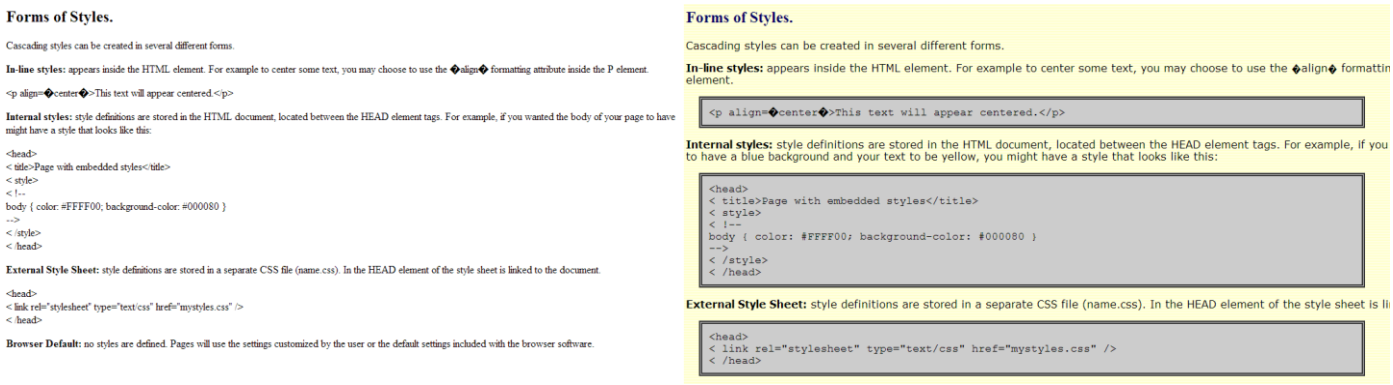


Figure 19 Un site internet sans et avec CSS[3]

Grâce à la combinaison de ces trois langages, D3.js permet de créer rapidement toutes sortes de visualisations. D3.js est rapide, supporte un nombre important de données et réagit dynamiquement aux modifications.

---

2 L'image provient de http://fr.openclassrooms.com/informatique/cours/debuter-dans-l-infographie-avec-gimp/les-images-numeriques

3 Les images proviennent de http://wac.osu.edu/workshops/css/default.htm et de http://wac.osu.edu/workshops/css/with-styles.htm

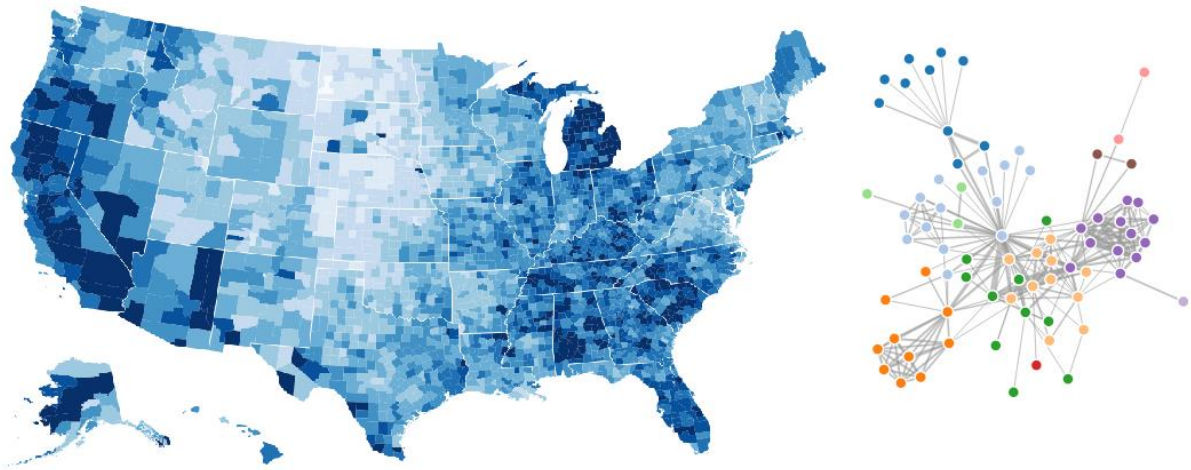


Figure 20 Exemples de visualisation avec D3.js[4]

Après un temps d'essai, voici les résultats de l'évaluation de la librairie.

Complexité : En raison d'objets exclusifs à cette librairie, certaines fonctionnalités sont difficilement assimilables au début. Malgré les aides à disposition en ligne, qui permettent en général d'atteindre son but, des doutes subsistent quant au fonctionnement interne. L'utilisation de sélections, de transitions et de bien d'autres fonctionnalités qui recèlent de mystères mathématiques est encore délicate.

Documentation : Heureusement, la librairie met à disposition de l'utilisateur la totalité de la documentation. L'intégralité du code source est non seulement ouvert à tout le monde, mais les auteurs de l'outil ont publié une pléthore d'exemple pour chaque fonctionnalité du logiciel. Cette abondance contrebalancera la complexité que l'utilisateur rencontrera lors de ses débuts.

Tutoriels : Dans le même ordre d'idées que la documentation, il existe beaucoup de tutoriels pour bien débuter la prise en main de la librairie. Chacune des techniques, basiques ou avancée, est illustrée pas à pas.

Communauté : Un grand nombre de développeurs font appel à D3.js pour construire des visualisations de données. C'est donc sans surprise que le support associé à cette librairie est important. Beaucoup de questions ayant été entretemps éclaircies ont été posées sur des

[4] Les images proviennent de http://bl.ocks.org/mbostock/4060606 et de http://bl.ocks.org/mbostock/4062045

forums d'aide au développement. Il y a beaucoup de chance pour qu'un développeur novice rencontre un problème et trouve une solution directement en cherchant sur les forums. Si par hasard le problème s'avéra inédit, une réponse dans la journée suivant la publication de la question est plus que probable.

Coût : D3.js est entièrement gratuit.

Licence : D3.js est protégé par une licence Berkeley Software Distribution (BSD) en trois clauses. La redistribution et l'utilisation du code source et binaire avec ou sans modification est permis si les conditions suivantes sont respectées :

Premièrement, la redistribution du code source doit contenir le texte déclaratif de la licence, cette liste de conditions et la déclaration suivante.

Ensuite, la redistribution sous forme binaire doit contenir le texte déclaratif de la licence, cette liste de conditions, la déclaration suivante et/ou autres clauses fournies avec la distribution.

Finalement, ni le nom de l'auteur, ni les noms des contributeurs ne peuvent être utilisés en vue de promouvoir les produits dérivés de ce logiciel sans une permission écrite [27].

Les limitations qu'impose la licence sont moindres, D3.js est par conséquent une librairie de choix dont l'utilisation demeure peu contraignante. Cette librairie sera utilisée pour visualiser les liens des données extraites du web sémantique.

2.3.3 Google Maps API

[26] Google Maps est une application en ligne qui permet de visualiser la planète Terre de différentes façons. Par exemple, l'utilisateur peut jongler entre une vue satellite et cartographique. En sélectionnant deux points sur la carte, Google Maps est également capable d'afficher un itinéraire et d'estimer le temps de trajet parmi plusieurs moyens de transport. En outre, l'utilisateur a la possibilité de regarder le paysage depuis les axes routiers importants à 360° grâce à une multitude d'images prises par un véhicule spécial.

Enfin, la fonction de zoom permet d'observer les cartes à l'échelle de la planète entière ou de la dimension d'un village.

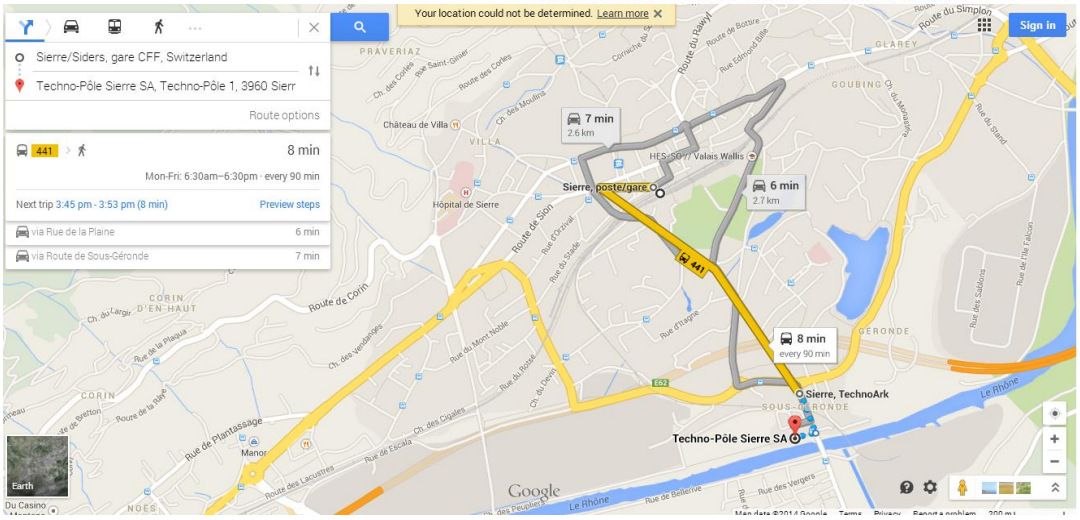


Figure 21 Carte avec trajets[5]

Figure 22 Vue satellite de la Terre[5]

L'API de Google Maps est utilisée pour bénéficier des fonctionnalités non-exhaustives citées précédemment sur n'importe quel site internet. Une API est un ensemble de fichiers contenant un code source fournissant l'accès à des fonctionnalités de l'outil à un logiciel extérieur [28]. Concrètement, le développeur donne des instructions à l'API, cette dernière les transforme et les donne à exécuter à Google Maps.

Dans le cadre de ce travail, les possibilités qu'offrent Google Maps sont très pertinentes. En effet, placer des points d'après des coordonnées géographiques, adapter le volume de

5 Les images proviennent de https://www.google.com/maps

cercles à une population et les faire évoluer dans le temps sont autant de points susceptibles d'aider un manageur à prendre une décision.

Ce travail utilise bien évidemment la version web de l'API car, comme expliqué précédemment, l'application utilise JavaScript.

Après une période d'entrainement, un avis plus éclairé, dont les détails sont listés ci-dessous, a pu être forgé.

Complexité : L'utilisation des objets, fonctions et variables compris dans l'API est relativement facile. Avec une base en programmation orientée objet et une compréhension de l'utilisation de Google Maps, la syntaxe du code fait naturellement sens. Aucune compétence particulière n'est exigée en plus.

Documentation : La documentation de l'API est très complète. Toutes les méthodes disponibles sont listées avec leurs paramètres. En plus d'une dénomination explicite, une courte description accompagne chaque fonction.

Tutoriels : Les tutoriels de la librairie débutent très simplement et aident le développeur jusqu'aux fonctionnalités de bases. Si ce dernier a besoin d'offrir un service plus spécifique, des exemples visuels avec le code source correspondant sont disponibles sur le site de référence.

Communauté : La communauté autour de Google Maps est vaste car cet outil est renommé. Dans le cas improbable où la solution à un problème ne se trouverait pas sur le site de l'API, les forums de discussion regorgeraient de réponses directes ou contournant la difficulté.

Coût : L'API de Google Maps est gratuite pour les organisations à but non-lucratif.

Licence : L'utilisation de la librairie est soumise à la licence Creative Commons 3.0 [29]. Cette version autorise la copie et la redistribution dans n'importe quel format et la modification du logiciel à n'importe quel but, même commercial. En revanche, la redistribution doit comprendre le nom des auteurs, un lien vers la licence et indiquer quels changements ont été effectués. L'auteur ne peut révoquer ces droits si la condition de la

licence est respectée. En plus de cette licence, l'API de Google Maps est soumise à des termes d'utilisation [30].

C'est en raison de toutes les fonctionnalités disponibles et des conditions d'utilisation raisonnablement souples que l'API de Google Maps sera utilisée dans ce travail.

D'autres librairies proposant des visualisations de cartes ont été étudiées. Elles n'ont pas été retenues pour des raisons de complexité ou de limitations. De plus, l'API de Google Maps a pu être testée par l'étudiant durant l'événement Sport Hackdays auquel une partie de l'équipe du projet OverLOD Surfer a assisté [31].

2.3.4 Sgvizler

[32] Sgvizler est une librairie JavaScript qui propose une visualisation des résultats d'une requête SPARQL. Une fois les résultats de la requête reçus, Sgvizler les convertit en format JSON et dessine le graphique spécifié auparavant. Sgvizler s'utilise à l'intérieur d'une page HTML en tant que conteneur [33].

Voici un prototype qui a été créé par l'étudiant pour tester le fonctionnement de la librairie.

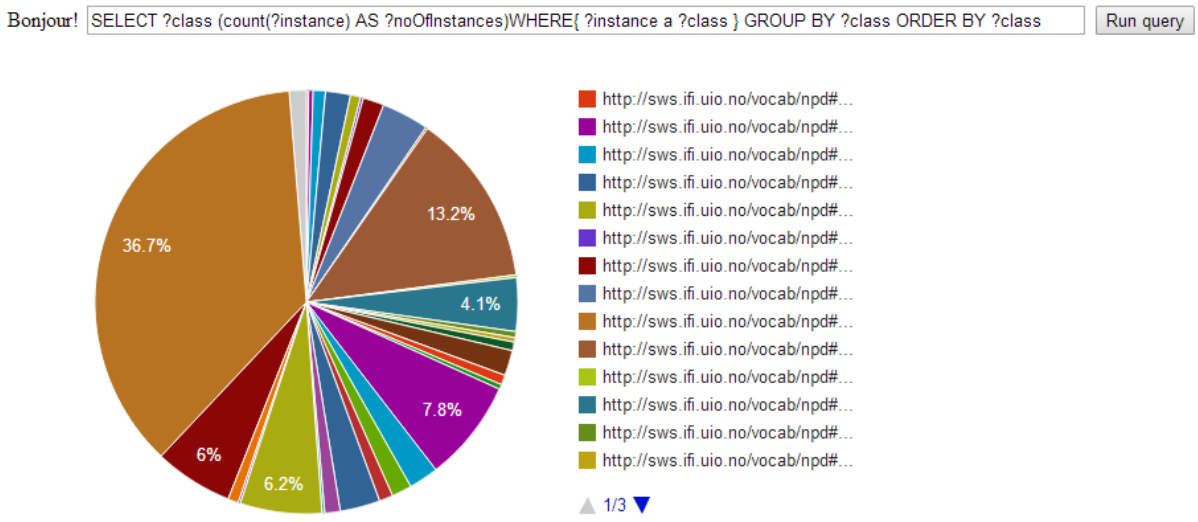


Figure 23 Prototype créé par l'étudiant afin de tester le fonctionnement de Sgvizler

Dans cet exemple, Sgvizler s'aide d'un graphique généré grâce à Google Charts [34]. Google Charts est une librairie proposant une multitude de visualisation de données et a été testé indépendamment de Sgvizler. Le prochain chapitre lui est dédié car elle est utilisée au sein du programme créé par l'étudiant.

Le point innovant qu'apporte Sgvizler est la coalition de deux étapes de processus. Tout d'abord, envoyer une requête sur un endpoint et récupérer les résultats puis les afficher à l'écran dans une forme graphique. Un bon nombre de librairie se concentrent sur l'un de ces deux points. Cependant, cette nouvelle approche, qui entre dans le cadre de la problématique de ce travail, en fait un outil unique actuellement.

C'est pour ces raisons qu'une étude plus approfondi a été portée à Sgvizler.

Complexité : Malgré la relative complexité des opérations effectuées par l'outil, le code reste compréhensible pour les utilisateurs possédant des connaissances en JavaScript et en SPARQL. En effet, après un rapide survol des instructions données en exemple dans le dossier contenant la librairie, les étapes sont clairement visibles bien que le fonctionnement interne soit masqué. Ce dernier élément s'avère être à double tranchant : le code simpliste permet de bien comprendre son utilisation au départ mais limite les possibilités de personnalisation par la suite. Il n'existe rien de plus pénible que de devoir déchiffrer le fonctionnement d'une librairie dans le détail.

Documentation : La documentation de Sgvizler est assez complète. L'utilisateur peut découvrir les rouages intrinsèques de l'outil à loisir. Malheureusement, mise à part les commentaires accompagnant le code source, aucune explication n'est donnée. L'utilisateur doit faire l'effort de chercher à comprendre comment les méthodes s'utilisent. Toutefois, les auteurs ont pensé à fournir quelques exemples dont les utilisateurs peuvent s'inspirer.

Tutoriels : En lieu et place d'un tutoriel au sens propre, les auteurs ont planché sur un wiki [35]. Un wiki est un ensemble de pages web modifiables de façon collaborative par le public dont le but est généralement de créer du contenu éducatif. Le wiki mis à disposition pour les utilisateurs de Sgvizler couvre les plus grandes thématiques. De l'installation aux problèmes de compatibilité en passant par toutes les visualisations, le wiki met les résolutions de problèmes courants à la portée de tous. Comme expliqué dans la partie

dédiée à la complexité, un tutoriel expliquant chaque étape n'est pas forcément pertinent vu le nombre réduit de lignes et la simplicité du code nécessaire à faire apparaître un graphique à l'écran.

Communauté : Le domaine d'application très spécifique est sans doute la raison pour laquelle Sgvizler est accompagné d'une communauté réduite. Heureusement, cet outil s'appuie sur d'autres librairies pour fonctionner qui possèdent, quant à elles, un support plus développé. En cas de problèmes techniques, l'utilisateur de Sgvizler se verra contraint d'utiliser ses compétences en recherche d'informations afin de trouver un forum susceptible d'apporter une réponse satisfaisante.

Coût : Sgvizler est un travail académique qui est mis à disposition de tous les curieux gratuitement.

Licence : La licence de Sgvizler spécifie que l'utilisateur a le droit de manipuler la librairie sans restriction, c'est-à-dire sans limitation quant à l'utilisation, la copie, la modification, la cohésion, la publication, la distribution, aux sous-licences et/ou à la vente de copies de l'application. Toutefois, les auteurs préviennent que le bon fonctionnement n'est pas garanti et que les utilisateurs ne peuvent pas les rendre responsables de dysfonctionnements [36].

En fin de compte, Sgvizler est une librairie innovante de par son approche du web sémantique. Elle combine la phase de récupération des données et leur affichage sous forme de graphiques. Elle n'est pas complexe et son utilisation est très vite acquise. Cependant, appréhender son fonctionnement interne est laborieux.

Les raisons de l'utilisation de Sgvizler sont décrites dans le chapitre 3 : Architecture de l'application.

2.3.5 Google Charts

[34] Google Charts est une librairie disposant d'un grand nombre de visualisations différentes. À partir d'un groupe de données, la librairie est capable de générer automatiquement un graphique. Ces graphiques disposent d'un système dynamique permettant à l'utilisateur d'obtenir le détail de chaque nœud en plaçant le pointeur de sa souris au-dessus. En plus des données et du conteneur sur lequel la librairie va dessiner

l'image dynamique, chaque graphique accepte un certain nombre d'options servant à en personnaliser l'aspect tel que la taille des points, la couleur, le titre, les légendes, etc.

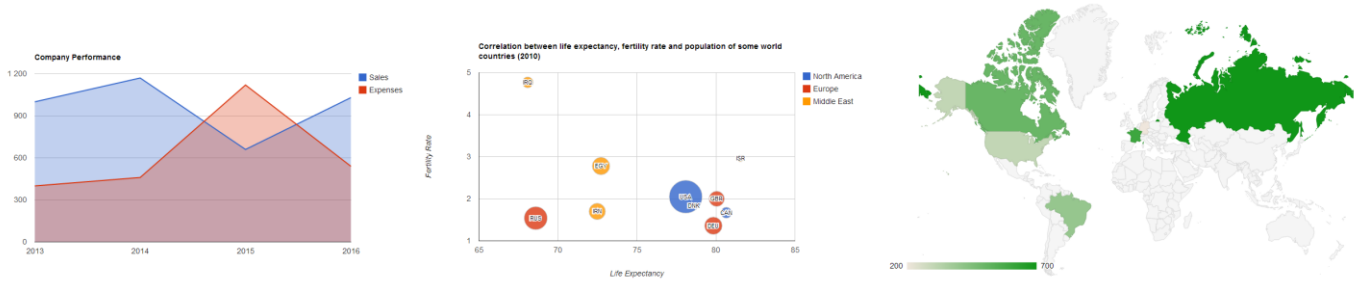


Figure 24 Exemples de graphique proposés par Google Charts

Complexité : La complexité de chaque graphique varie en fonction des données à fournir. Certaines visualisations demandent davantage d'informations qu'il faut structurer de manière très précise. En contrepartie, toutes les librairies s'appuient sur un schéma de fonctionnement très similaire ce qui en facilite grandement la compréhension.

Documentation : La documentation de Google Charts est fournie. L'utilisation de chaque classe est expliquée en détails et chaque méthode, avec tous ses paramètres, est décrite.

Tutoriels : Le site met à disposition un rapide tutoriel. Tout d'abord, le résultat final est illustré afin de comprendre l'utilité de l'outil. Le code source est affiché ensuite et chaque étape est expliquée. Itérativement, les possibilités de la librairie sont dévoilées pas-à-pas. Affichage, chargement des graphiques, préparation des données, personnalisation et interactivité seront maîtrisés par le néophyte après quelques minutes seulement.

Communauté : Google Charts dispose d'une grosse communauté. En plus des multitudes de questions répondues sur les sites de développement, le site internet de la librairie propose un forum, une liste de questions-réponses, une foire aux questions et une galerie présentant les meilleurs travaux réalisés à l'aide de la librairie.

Coût : Les graphiques proposés par la librairie sont gratuits. De plus, une rétrocompatibilité de trois années est garantie.

Licence : Google Charts est protégée par les conditions d'utilisation des API Google. Chaque page du site internet spécifie que le code source est régit par une licence Apache 2.0 [38].

En définitif, Google Charts est une librairie de visualisation de données comportant un nombre important de graphiques différents. Bien que sa documentation et sa communauté prouvent sa fiabilité, c'est la structure des données très semblable d'un graphique à l'autre qui a été déterminante dans ce choix de l'outil de visualisation des données.

3 Architecture de l'application

Ce chapitre explique la structure générale du programme développé par l'étudiant. Comme expliqué en conclusion du chapitre consacré à l'étude des applications existantes dans le domaine de la visualisation des données provenant du web sémantique, aucun outil n'est actuellement capable de répondre en l'état à la problématique posée par ce travail de Bachelor. Il en résulte la nécessité de développer intégralement une application qui répond à la problématique. Suite à l'étude et à la sélection des librairies qui fourniront les fonctionnalités de base, le développement à proprement parlé de l'application peut enfin débuter.

3.1 Étude des besoins

Avant d'écrire la moindre ligne de code, il convient de s'entendre sur les besoins de l'utilisateur. Les fonctionnalités du produit final doivent permettre de répondre à des besoins définis auparavant.

Pour commencer, les acteurs du système doivent être définis. En raison du projet OverLOD dans lequel ce travail de Bachelor s'inscrit, les utilisateurs ont été définis à une étape antérieure. Les utilisateurs de l'application sont scindés en trois groupes distincts :

Premièrement, des administrateurs de bases de données emploieront l'application. Les administrateurs sont des personnes qui savent extraire des données du web sémantique et les mettre à disposition des personnes de la deuxième catégorie. Les administrateurs qui utiliseront le programme sont capables de localiser les données dans le web sémantique et maîtrisent le langage de requêtage SPARQL.

La seconde catégorie est constituée de développeurs. Sur la base de données structurées, les développeurs construisent des applications servant à exploiter ces données. Le domaine de compétence des développeurs est divers car les données peuvent être utilisées dans des systèmes très différents. Une fois leur travail terminé, les développeurs mettent leur programme à disposition d'utilisateurs finaux.

Ces derniers forment la dernière catégorie d'utilisateurs. Ces utilisateurs ne possèdent pas de compétences techniques en informatique et ne font que consommer un produit terminé. En revanche, ils peuvent posséder des connaissances métiers, c'est pourquoi les données sélectionnées par les administrateurs doivent être pertinentes à ces utilisateurs.

Maintenant que les acteurs sont définis, il reste à préciser les fonctionnalités dont chaque catégorie a besoin.

3.1.1 Fonctionnalités souhaitées pour les administrateurs

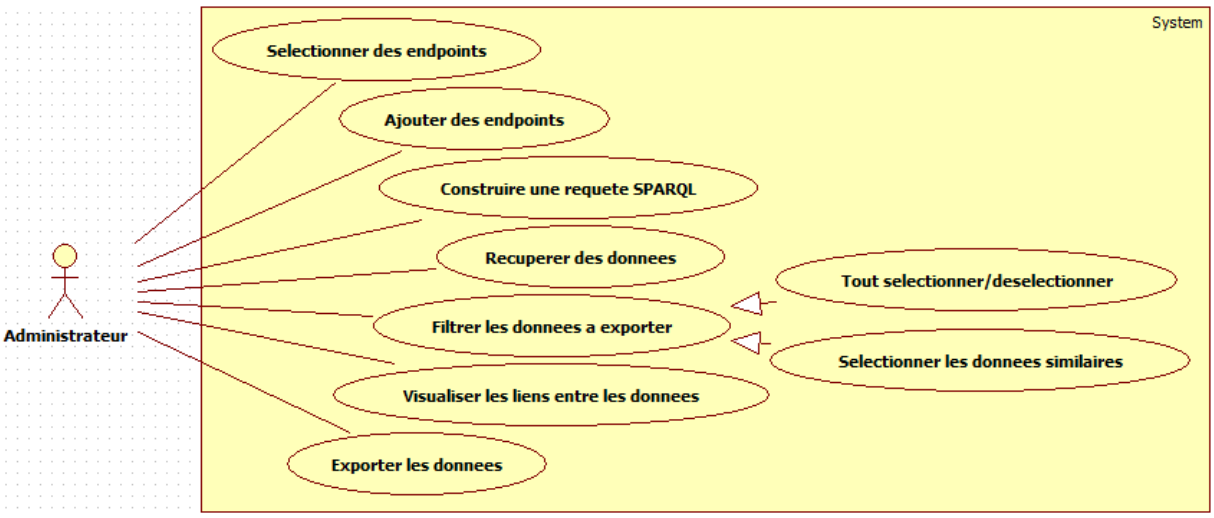


Figure 25 Diagramme de Use Case pour le rôle administrateur

Sélectionner des endpoints : Les administrateurs doivent être en mesure de préciser les destinataires des requêtes SPARQL.

Ajouter des endpoints : Les administrateurs peuvent ajouter un nouveau destinataire, comportant un nom et une adresse URL, et le sélectionner.

Construire une requête SPARQL : Afin de récupérer des données, une requête SPARQL doit être écrite. Afin de simplifier leur tâche, l'application doit aider les administrateurs dans la construction de la requête.

Récupérer des données : Une fois la requête terminée, les administrateurs peuvent l'envoyer aux endpoints sélectionnés et observer les résultats retournés.

Filtrer les données à exporter : Afin de préparer des données pour les utilisateurs finaux, l'administrateur est en mesure de sélectionner qu'une partie des données qu'il juge pertinente.

Tout sélectionner/désélectionner : Dans l'étape de filtrage des données, les administrateurs ont le choix de sélectionner l'intégralité des résultats ou de désélectionner leur choix actuel dans un souci de rapidité.

Sélectionner les données similaires : Pour optimiser la vitesse de sélection des données, l'administrateur peut choisir de sélectionner automatiquement tous les résultats possédant les mêmes propriétés.

Visualiser les liens entre les données : Les données, provenant du web sémantique, peuvent posséder des propriétés en commun. Les administrateurs doivent pouvoir étudier ces liens.

Exporter les données : Une fois les résultats filtrés, les administrateurs ont la possibilité de sauvegarder leur sélection au format JSON.

3.1.2 Fonctionnalités souhaitées pour les développeurs :

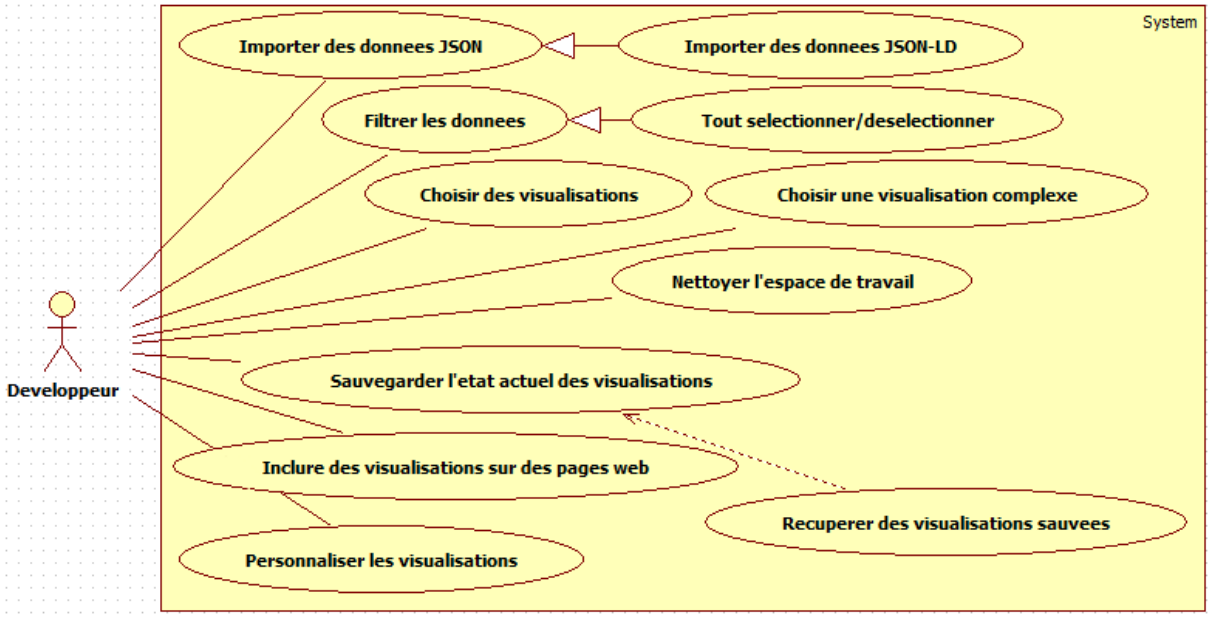


Figure 26 Diagramme de Use Case pour le rôle développeur

Importer des données JSON : Le développeur a besoin de disposer de données pour pouvoir créer des visualisations. Comme expliqué dans l'introduction de ce rapport, le format JSON possède une structure adaptée pour les données issues du web sémantique.

Importer des données JSON-LD : Le format JSON-LD permet de relier les données à leur ontologie afin de leur donner un sens. Pouvoir créer des visualisations à l'aide de ces ontologies permettrait de découvrir une nouvelle perspective dans la représentation des données du web sémantique.

Filtrer les données : Même si les données reçues ont déjà passé par un premier filtrage, le développeur a besoin d'affiner la sélection des données pour pouvoir générer des visualisations le plus précisément possible.

Tout sélectionner/désélectionner : À l'instar de l'administrateur, le développeur peut gagner du temps grâce à ces deux fonctionnalités. La première désigne la totalité des données à disposition et la seconde annule complétement la sélection.

Choisir des visualisations : Le développeur a la possibilité d'afficher différents éléments. Il peut s'agir de graphiques, d'images, de tableaux, etc.

Choisir une visualisation complexe : En plus des vues basiques, le développeur bénéficie d'un modèle plus abouti. Ce modèle doit exploiter plusieurs aspects relatifs aux données sélectionnées. Il peut s'agir par exemple de positions géographiques, de modifications de l'affichage en fonction de la sélection courante, d'évolutions temporelles, de facettes, etc.

Nettoyer l'espace de travail : Lorsque le développeur a besoin de recommencer du début, cette fonctionnalité lui permet de supprimer toutes les visualisations présentes à l'écran pour repartir sur de nouvelles bases.

Sauvegarder l'état actuel des visualisations : Si le développeur souhaite continuer son travail ultérieurement, il doit être capable d'enregistrer son avancement, c'est-à-dire garder une trace de toutes les visualisations, leur position et leurs données.

Récupérer des visualisations sauvées : Suite à la fonctionnalité précédente, le développeur est en mesure de recharger son travail dans l'application dans l'état où il l'a laissé.

Inclure des visualisations sur des pages web : Une fois les visualisations créées, le développeur doit être en mesure de placer simplement le résultat sur un site internet ouvert à la consultation. Il peut s'agir soit d'une image soit d'un graphique dynamique.

Personnaliser les visualisations : Un certain nombre d'options doit être disponible pour chaque visualisation. Le développeur peut en modifier le titre, la légende, les couleurs, etc.

3.1.3 Fonctionnalités souhaitées pour les utilisateurs finaux :

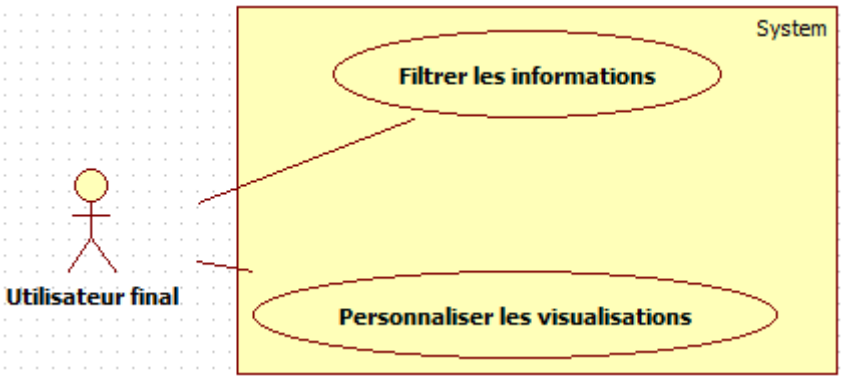


Figure 27 Diagramme de Use Case pour le rôle utilisateur final

Filtrer les informations : L'utilisateur final peut choisir quelles informations la visualisation doit afficher. Il peut décider d'afficher la totalité des données ou une partie seulement.

Personnaliser les visualisations : Dans une certaine mesure, l'utilisateur final peut modifier les options définies précédemment par le développeur. Le choix des couleurs ou la taille de l'affichage en sont des exemples.

3.2 Modélisation et création de l'application

À partir des besoins détaillés précédemment, un flux de données semble se préciser. Ainsi, si l'administrateur ne met pas de données à disposition du développeur, ce dernier n'a

pas la possibilité de construire des visualisations. De cette constatation se dégage un ordre à respecter dans les fonctionnalités susmentionnées.

Figure 28 Processus de création de visualisations

La figure ci-dessus retrace les principales étapes par lesquelles les données doivent passer afin d'être transformées puis mises à disposition des utilisateurs finaux. Les phases bleues sont maintenues par les administrateurs, les étapes vertes par les développeurs et la dernière action est effectuée par les utilisateurs finaux. La chronologie de la séquence ne peut être altérée et chaque fonctionnalité est requise dans le bon fonctionnement du système. Par exemple, il est impossible pour l'administrateur d'extraire des données s'il ne peut pas les explorer auparavant. De même, le développeur se verra contraint de créer des visualisations avant de pouvoir les publier.

Grâce au code couleur appliquée à l'illustration, trois paliers, qui correspondent à un acteur chacun, ressortent. Il est dès lors judicieux de regrouper ces fonctionnalités ensembles et proposer des applications différentes pour des acteurs différents. À l'exception de l'utilisateur final qui se sert d'un navigateur internet pour consommer les visualisations, l'administrateur ainsi que le développeur ont chacun besoin d'un programme dédié à leurs besoins. Ce travail de Bachelor se consacre par conséquent à l'implémentation d'une solution administrateur et d'une solution développeur.

3.3 Ressources nécessaires au fonctionnement de l'application

Avant d'utiliser l'application développée dans le cadre de ce travail de Bachelor, il est nécessaire de s'assurer que l'utilisateur dispose de tous les fichiers requis au bon fonctionnement du programme.

Une fois le disque inséré dans le lecteur de l'ordinateur, deux dossiers sont disponibles :

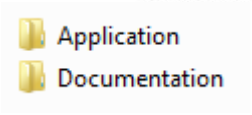


Figure 29 Dossiers à la racine du CD

Le dossier « Application » renferme la totalité des ressources que le programme utilise. Le dossier « Documentation » contient quant à lui tous les documents de ce travail de Bachelor.

À l'intérieur du dossier « Application » se trouvent les dossiers « administration » et « development ». Les deux contiennent les codes sources des programmes destinés aux administrateurs ainsi qu'aux développeurs. Les codes sources des librairies utilisées sont stockés dans le dernier dossier « library ». Les trois fichiers restants à ce niveau servent à la page d'accueil de ce travail de Bachelor. La page d'accueil est un moyen simple pour permettre aux utilisateurs de passer d'un programme à l'autre.

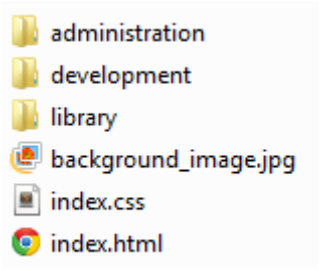


Figure 30 Dossiers à l'intérieur de la partie Application

Le dossier « administration » renferme les documents suivants :

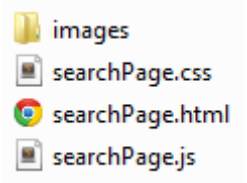


Figure 31 Contenu du dossier administration

Les images comprises dans le dossier « images » servent à améliorer l'apparence du programme.

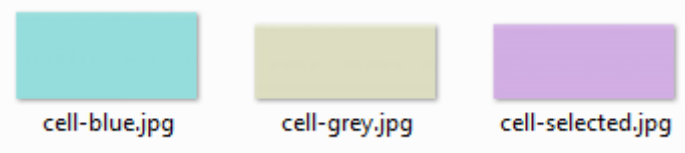


Figure 32 Contenu du dossier images de l'application d'administration

Le dossier « development », qui possède une structure similaire au dossier « administration », est synthétisé dans la figure suivante :

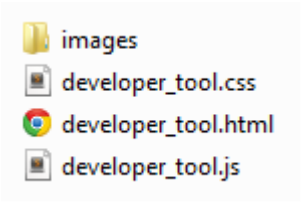


Figure 33 Contenu du dossier development

À l'instar du programme des administrateurs, les images qui embellissent l'application se trouvent dans le dossier du même nom :

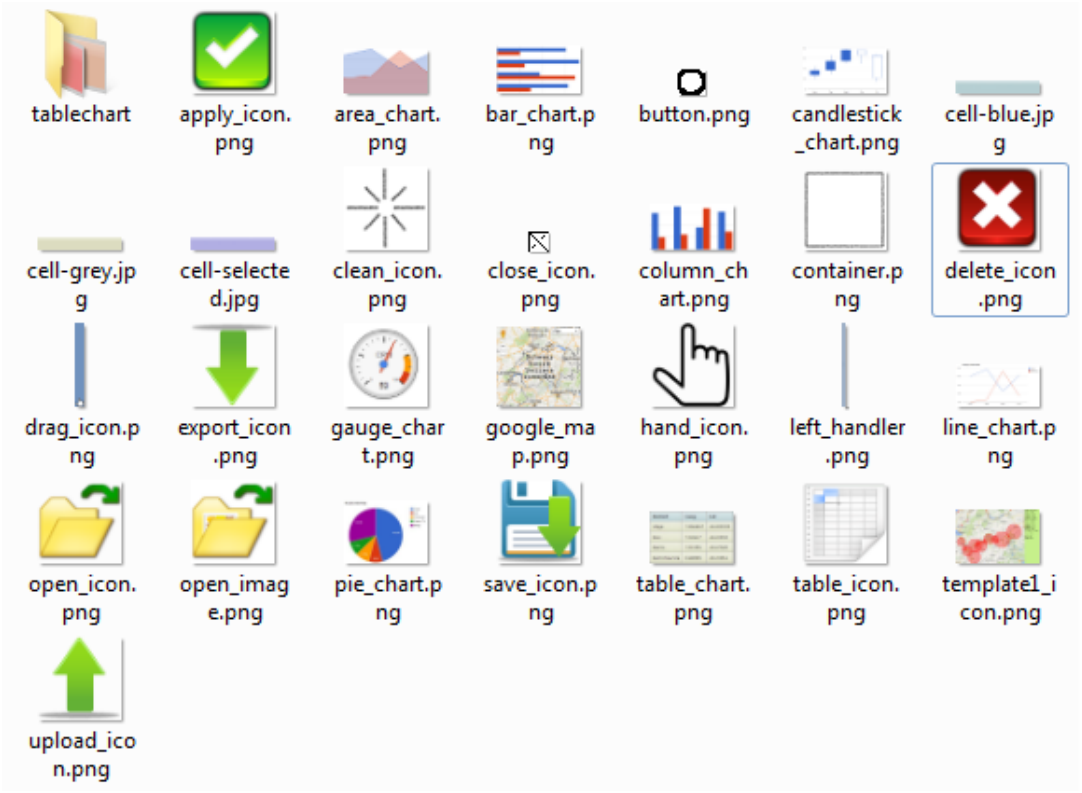


Figure 34 Contenu du dossier images de l'application de développement

Le sous-dossier « tablechart » contient les deux images suivantes :

Figure 35 Contenu du dossier tablechart

Finalement, le dossier « library », se trouvant dans le dossier-racine « Application », comprend les librairies utilisées dans les programmes d'administration et de développement :

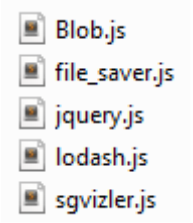


Figure 36 Contenu du dossier library

À noter que certaines librairies telle que Google Maps, Google Charts et D3.js sont chargées au lancement du programme depuis leur serveur, c'est pourquoi elles n'apparaissent pas dans les dossiers susnommés. Étant donné qu'une connexion internet est nécessaire pour atteindre les applications développées dans le cadre de ce travail de Bachelor, les librairies distantes ont été admises.

La figure suivante synthétise l'emplacement des dossiers :

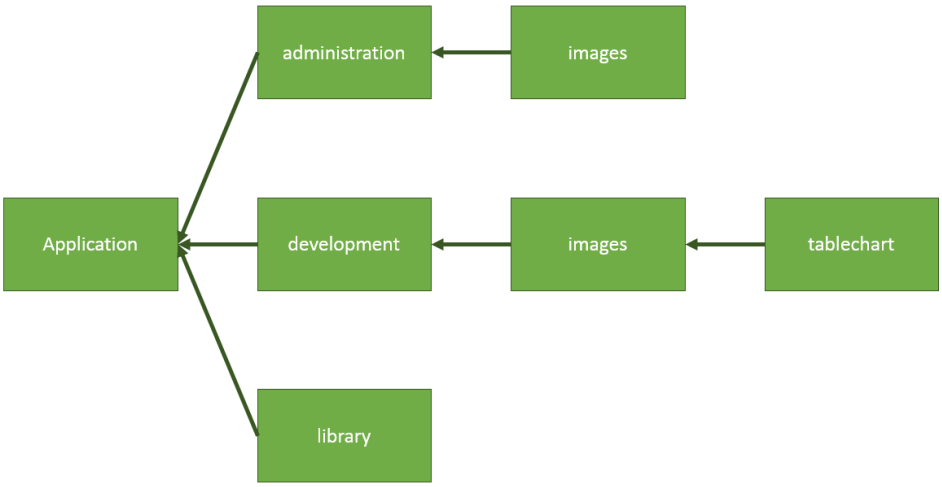


Figure 37 Structure du dossier Application du CD

Si chaque fichier listé dans cette section se trouve à sa place, l'application démarre sans aucun problème. L'absence d'une des ressources peut cependant entraîner des disfonctionnements dans l'exécution du programme. En cas de problème, l'utilisateur avisé peut ouvrir le menu de développement présent sur tous les navigateurs courants et se référer à la console listant les erreurs survenues lors du lancement du programme.

Les applications développées ont été optimisées pour le navigateur Google Chrome. L'utilisation d'un autre navigateur peut limiter les fonctionnalités disponibles.

3.4 Guide de l'utilisateur

L'objectif de cette section est de familiariser l'utilisateur aux deux applications résultant de ce travail de Bachelor. Dès que l'utilisateur a pris connaissance de ce guide introductif, il est en mesure d'exécuter toutes les fonctionnalités implémentées dans les deux logiciels.

3.4.1 Installation

Afin d'installer les applications, insérez le CD se trouvant à la fin de ce document dans l'ordinateur. Après quelques secondes, une fenêtre s'ouvre vous demandant quelle action exécuter. Choisissez « Ouvrir le dossier ». Si cette fenêtre de dialogue n'apparaît pas, ouvrez le poste de travail et sélectionnez la partition représentant le CD.

À la racine du CD, se trouvent les dossiers « Application » et « Documentation ».

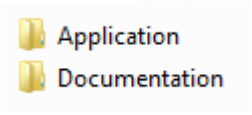


Figure 38 Dossiers à la racine du CD

Copiez le contenu du dossier « Application » et déposez-le à l'intérieur d'un serveur web. La configuration du serveur web n'est pas abordée dans ce document car les spécifications varient en fonction des fournisseurs. Au besoin, redémarrez le serveur.

À présent, ouvrez un navigateur internet[6] et rendez-vous sur la page d'accueil du programme.
Celle-ci se trouve à l'adresse suivante : *http://serveur_web/Application/index.html*

La page suivante s'affiche :

Figure 39 Page d'accueil du travail de Bachelor

3.4.2 Application d'administration

L'application d'administration est utilisée par les administrateurs pour rechercher des données disponibles sur le web sémantique, les trier et les exporter. Pour bénéficier au mieux des possibilités offertes par ce programme, des connaissances techniques en RDF, OWL et SPARQL sont requises.

Un accès direct à l'application d'administration est disponible en cliquant sur le bouton « ADMINISTRATION ».

[6] Les applications sont optimisées pour Google Chrome

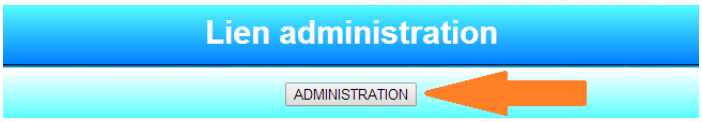


Figure 40 Lien vers l'application d'administration

L'écran suivant apparaît alors :

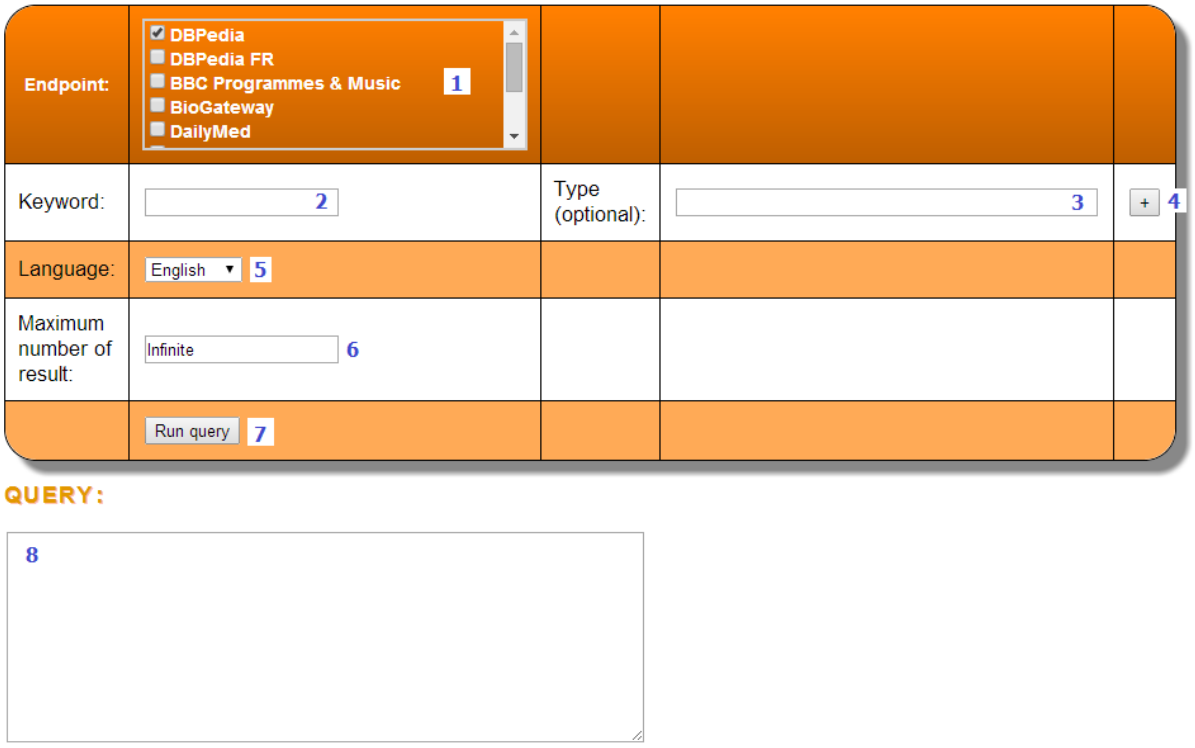


Figure 41 Interface utilisateur de l'application d'administration

1- Cette liste regroupe les endpoints que vous pouvez utiliser pour récupérer des données. Une sélection multiple est possible. En outre, vous avez la possibilité de rajouter vous-même des endpoints s'ils ne figurent pas dans la liste préétablie.

2- Vous spécifiez dans cette zone de texte un mot-clé à rechercher. À l'instar des moteurs de recherche web classique, l'application va rechercher les occurrences de ce mot à l'intérieur des endpoints sélectionnés. À chaque modification de champ dans le formulaire de recherche, la requête est reconstruite dynamiquement.

3- Cette zone de texte permet d'orienter les recherches du mot-clé adjacent. Par exemple, si le mot-clé est « Jupiter », le type peut être « Planet » ou « God » suivant le domaine de recherche. Ce critère de recherche est facultatif. Il correspond au

prédicat « rdf:type » d'un fichier RDF. À noter que les endpoints sont sensibles à la casse pour les mots-clés et les types. Par conséquent, « God » et « god » ne retourneront pas les mêmes occurrences.

4- Ce bouton ajoute une ligne de recherche de mot-clé à la ligne inférieure. Aucun bouton pour supprimer une ligne n'existe car si la ligne ne contient aucune information, elle est ignorée par l'application au lancement de la requête.

5- Cette liste contient quelques langues que vous pouvez choisir en fonction des mots-clés. Logiquement, si un mot-clé n'existe pas dans la langue spécifiée, les chances d'obtenir un résultat sont plus minces. Il est important de comprendre que la langue n'a pour unique but que de chercher des données. Si la page contenant les données rapportée par l'application est dans une langue différente, les données à disposition sur cette page peuvent être dans une autre langue également. L'application développée n'a aucune influence sur les résultats des requêtes.

6- Vous avez la possibilité de fixer une limite au nombre de concepts différents que la requête retournera. Par défaut, aucune limite de résultats n'est fixée. Cela signifie que chaque occurrence trouvée sur chacun des endpoints s'affichera dans la liste de résultats. Un délai d'attente important peut en résulter car beaucoup de réponses parviendront depuis les endpoints.

7- Le bouton « Run query » envoie la requête à tous les endpoints sélectionnés. Notez que la requête réelle est celle qui se trouve dans l'élément listé ci-dessous.

8- La requête mise à jour apparaît dans ce dernier champ. La requête décrite dans cet élément sera utilisée lorsque la demande de données sera envoyée aux endpoints. Si vous le désirez, vous pouvez écrire une requête entièrement vous-même sans passer par le formulaire.

Quand vous êtes satisfait de la requête et que les endpoints sont définis, appuyez sur « Run query » pour recevoir les résultats de la requête. Un délai plus ou moins long peut survenir en fonction de la rapidité du réseau, du taux d'occupation des endpoints et de la complexité de la requête.

Les résultats sont présentés sous deux formes différentes. Deux onglets sont disponibles pour passer d'une forme à l'autre.

Un graphe constitue la première manière d'afficher les résultats.

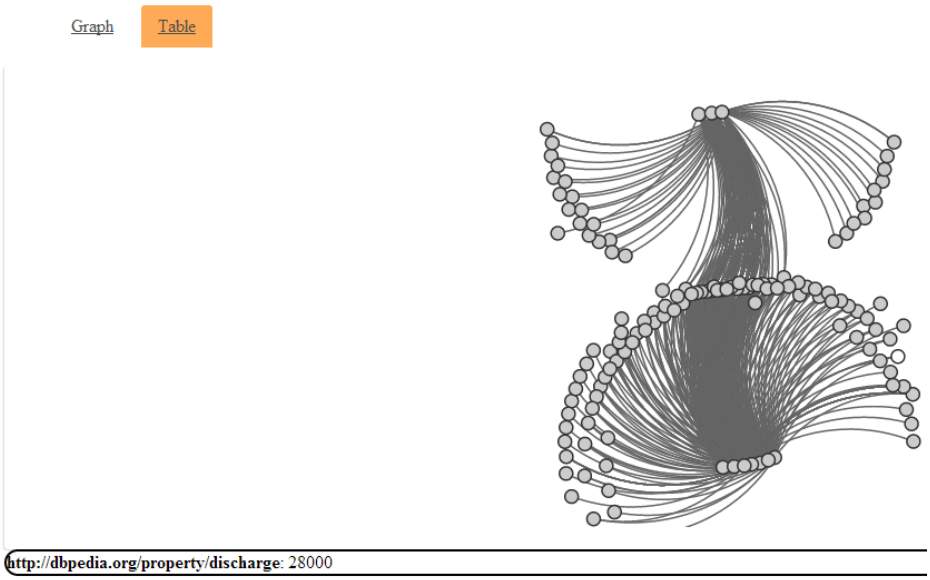


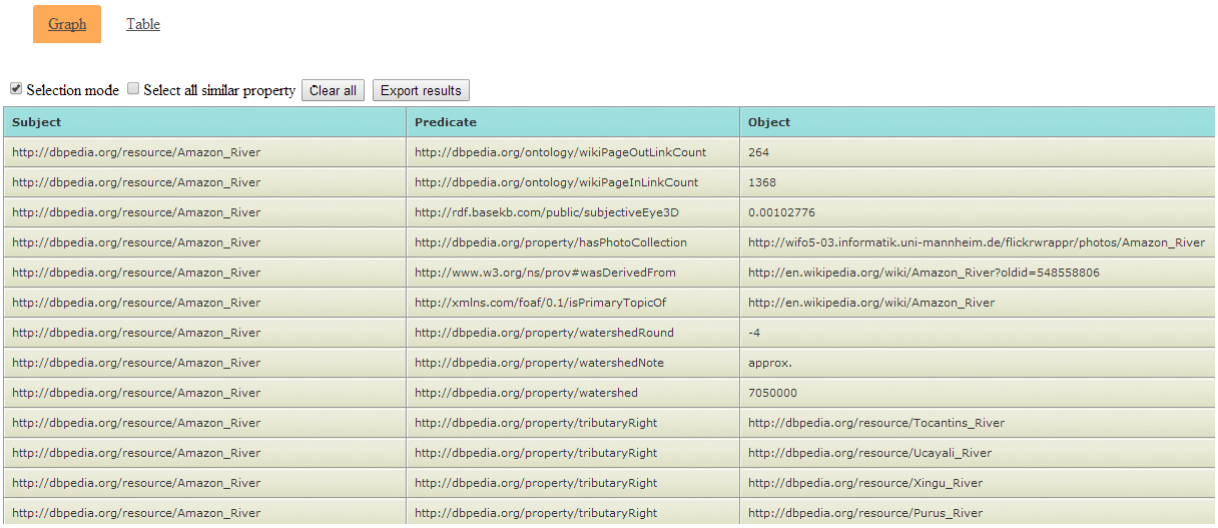
Figure 42 Vue en forme de graphe de l'application d'administration

L'objectif de cette vue est de montrer les liens qui existent entre les données. Les nœuds, qui représentent des objets dans la terminologie RDF, sont reliés entre eux via des prédicats.

L'application d'administration permet d'exporter des données. Un nœud gris signifie qu'il n'est pas sélectionné. Le nœud blanc représente la donnée sélectionnée actuellement. Lorsque vous cliquez sur un nœud blanc, la donnée est activée pour l'exportation et son nœud devient rouge.

Un petit encadré situé sous le graphe donne des informations sur les données que renferme le nœud sélectionné.

La seconde méthode d'affichage est construite sous la forme d'un tableau à trois colonnes.


Graph Table

Selection mode Select all similar property Clear all Export results

Subject	Predicate	Object
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/ontology/wikiPageOutLinkCount	264
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/ontology/wikiPageInLinkCount	1368
http://dbpedia.org/resource/Amazon_River	http://rdf.basekb.com/public/subjectiveEye3D	0.00102776
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/hasPhotoCollection	http://wifo5-03.informatik.uni-mannheim.de/flickrwrappr/photos/Amazon_River
http://dbpedia.org/resource/Amazon_River	http://www.w3.org/ns/prov#wasDerivedFrom	http://en.wikipedia.org/wiki/Amazon_River?oldid=548558806
http://dbpedia.org/resource/Amazon_River	http://xmlns.com/foaf/0.1/isPrimaryTopicOf	http://en.wikipedia.org/wiki/Amazon_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/watershedRound	-4
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/watershedNote	approx.
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/watershed	7050000
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Tocantins_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Ucayali_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Xingu_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Purus_River

Figure 43 Vue en forme de tableau de l'application d'administration

Chacune des colonnes correspond à l'un des triplets d'un fichier RDF. La première colonne contient le sujet, c'est-à-dire le concept retourné par les critères de recherche remplis dans le formulaire. Le prédicat du concept s'affiche dans la deuxième colonne. Finalement, l'objet apparaît dans la colonne restante.

Pour sélectionner une ligne, un simple clic sur l'une des cellules suffit. Rééditez le clic pour désélectionner la ligne. Pour sélectionner une plage de ligne, maintenez la touche « majuscule » enfoncée et effectuez un deuxième clic. Toutes les lignes contenues entre les deux sélections seront activées pour l'exportation. Si vous désirez désactiver une plage de lignes, assurez-vous que la case à cocher « Selection mode » ne soit pas active et effectuez deux clics de la même manière que pour la sélection de plage de lignes. L'option « Select all similar property » est un moyen simple et rapide d'exporter toutes les données ayant un élément en commun. Vérifiez que l'option soit activée, puis cliquez sur l'une des cellules. Toutes les lignes comportant la même propriété sont sélectionnées automatiquement.

Selection mode | Select all similar property | Clear all | Export results

Subject	Predicate	Object
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/ontology/wikiPageOutLinkCount	264
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/ontology/wikiPageInLinkCount	1368
http://dbpedia.org/resource/Amazon_River	http://rdf.basekb.com/public/subjectiveEye3D	0.00102776
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/hasPhotoCollection	http://wifo5-03.informatik.uni-mannheim.de/flickrwrappr/photos/Amazon_River
http://dbpedia.org/resource/Amazon_River	http://www.w3.org/ns/prov#wasDerivedFrom	http://en.wikipedia.org/wiki/Amazon_River?oldid=548558806
http://dbpedia.org/resource/Amazon_River	http://xmlns.com/foaf/0.1/isPrimaryTopicOf	http://en.wikipedia.org/wiki/Amazon_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/watershedRound	-4
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/watershedNote	approx.
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/watershed	7050000
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/watershedNote	approx.
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/watershed	7050000
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Tocantins_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Ucayali_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Xingu_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Purus_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryRight	http://dbpedia.org/resource/Madeira_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryLeft	http://dbpedia.org/resource/Putumayo_River
http://dbpedia.org/resource/Amazon_River	http://dbpedia.org/property/tributaryLeft	http://dbpedia.org/resource/Rio_Negro_(Amazon)

Figure 44 Exemples de sélection de propriétés similaires

Sur la première image, un clic avec l'option « Select all similar property » a été effectué sur la première cellule à gauche. Il en résulte une sélection de toutes les données liées au concept. Dans la seconde image, la première cellule « tributaryRight » a été activée suivi de toutes les cellules similaires dans la table.

Le bouton « Clear all » permet d'annuler toutes les sélections de la table.

Une fois la phase de sélection terminée, cliquez sur « Export results ». Une fenêtre de dialogue apparaît afin que vous puissiez spécifier l'endroit dans lequel vous souhaitez sauver vos données exportées au format JSON.

3.4.3 Application de développement

L'application de développement permet, sur la base de données au format JSON, de construire des visualisations qui serviront à être analysées par des utilisateurs finaux disposant de connaissances dans le domaine à l'étude. Des connaissances de base en statistiques sont requises afin de créer des visualisations correctes.

Depuis le menu principal, cliquez sur le lien « DEVELOPPEMENT » pour afficher l'interface de l'application de développement.

Figure 45 Lien vers l'application de développement

Si vous vous trouvez dans l'application d'administration, vous pouvez utiliser la fonction de retour en arrière du navigateur pour revenir à la page d'accueil.

Dans l'application de développement, le menu principal a été caché pour laisser plus de visibilité à l'espace de travail. Pour faire apparaître le menu, placez le curseur de la souris sur la zone à gauche de l'écran.

Figure 46 Bouton faisant apparaitre le menu principal

À ce moment, le menu principal apparaît.

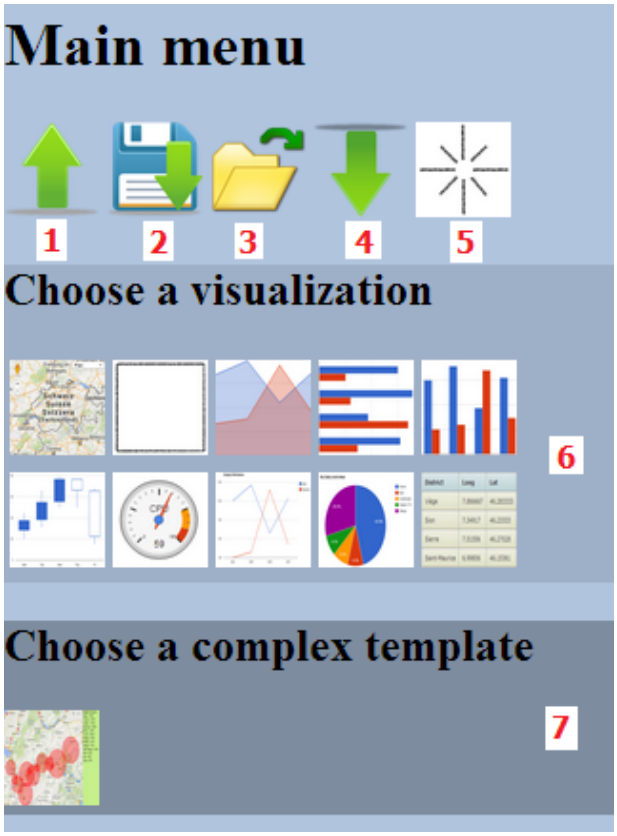


Figure 47 Menu principal de l'application de développement

1- Ce bouton est utilisé pour importer des données au format JSON dans l'application développeur. Ces données serviront à la construction de visualisations.

2- Vous pouvez vous servir de ce bouton pour sauvegarder l'état actuel de votre espace de travail. La position, la taille ainsi que les données de chaque visualisation sont prises en compte.

3- Cette fonctionnalité vous permet de recharger un espace de travail précédemment sauvé.

4- Lorsque le développeur souhaite publier les résultats de son travail, ce bouton lui sert à sauvegarder les visualisations présentes à l'écran à l'intérieur d'un fichier. Le contenu de ce dernier lui sera utile pour l'affichage des visualisations sur un site web.

5- Utilisez ce bouton si vous désirez nettoyer complétement l'espace de travail et repartir de zéro.

6- Plusieurs visualisations sont disponibles dans cette section. Vous pourrez placer des cartes Google Map, des images ou encore des graphiques.

7- Une visualisation au fonctionnement plus abouti est disponible dans cette dernière zone du menu principal.

Notez qu'une infobulle apparaît pour vous informer du rôle du bouton lorsque le curseur reste au-dessus plus d'une seconde.

Pour placer une visualisation sur l'espace de travail, cliquez sur l'icône du menu principal. Un carré bleu vous informant qu'aucune donnée n'est reliée à la visualisation pour l'instant s'affiche sur l'espace de travail.

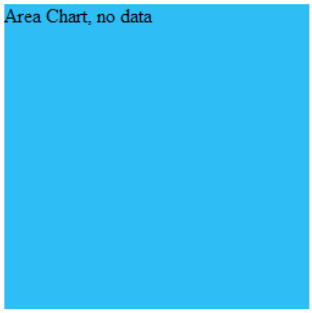


Figure 48 Conteneur sans données

Pour y attacher des données, commencez par importer un fichier JSON grâce au bouton d'import. Lorsque les données sont chargées dans le programme, une fenêtre de dialogue vous le signale. Effectuez ensuite un simple clic sur le carré bleu pour afficher le menu relatif à cette visualisation.

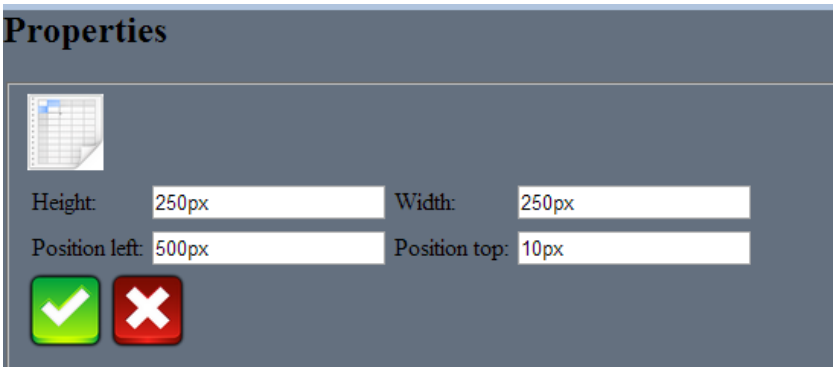
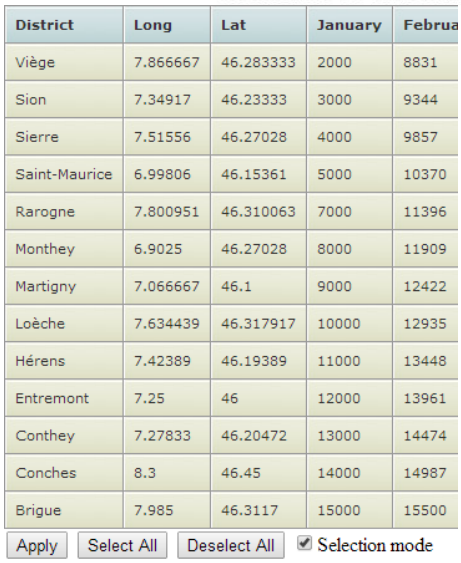


Figure 49 Menu de propriétés d'une visualisation

À l'intérieur de ce menu, vous pouvez spécifier les dimensions de la visualisation ainsi que sa position sur l'espace de travail. Utilisez le bouton « Valider » pour confirmer votre choix ou le bouton « Supprimer » pour effacer la visualisation de l'espace de travail. Le dernier bouton sert à lier des données à la visualisation.

Figure 50 Bouton servant à lier des données à une visualisation

Lorsque vous cliquez dessus, un tableau rempli avec les données du fichier JSON importé précédemment apparaît. Un message vous demande d'importer des données le cas échéant.



District	Long	Lat	January	Februa
Viège	7.866667	46.283333	2000	8831
Sion	7.34917	46.23333	3000	9344
Sierre	7.51556	46.27028	4000	9857
Saint-Maurice	6.99806	46.15361	5000	10370
Rarogne	7.800951	46.310063	7000	11396
Monthey	6.9025	46.27028	8000	11909
Martigny	7.066667	46.1	9000	12422
Loèche	7.634439	46.317917	10000	12935
Hérens	7.42389	46.19389	11000	13448
Entremont	7.25	46	12000	13961
Conthey	7.27833	46.20472	13000	14474
Conches	8.3	46.45	14000	14987
Brigue	7.985	46.3117	15000	15500

Apply | Select All | Deselect All | Selection mode

Figure 51 Exemple de tableau servant à lier des données à une visualisation

Le fonctionnement est similaire au tableau servant à l'administrateur. Cliquez simplement sur une cellule pour l'activer. Un deuxième clic désactive la cellule. Les boutons « Select All » et « Deselect All » active complétement la table ou la désélectionne respectivement. Pour sélectionner une plage de cellule, effectuez un premier clic sur une cellule, laissez la touche « majuscule » enfoncée puis sélectionnez une autre cellule. Toutes les cellules se trouvant dans l'intervalle s'activent. Utilisez le même mécanisme avec l'option « Selection mode » désactivée pour désélectionner une plage de cellules. Après chaque modification de sélection, un aperçu apparaît en dessous du tableau. Si les données ne permettent pas de créer une visualisation, un message d'avertissement rouge contenant un indice sur le problème s'affiche dans le bord supérieur de l'interface.

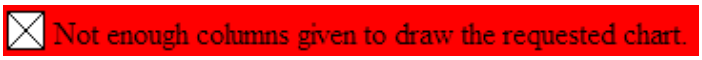


Figure 52 Message d'alerte

Utilisez le bouton blanc biffé pour enlever le message d'alerte.

Dès que l'aperçu vous semble adéquat, le bouton « Apply » est à votre disposition pour valider la visualisation. L'espace de travail réapparaît alors avec la visualisation générée.

3.4.4 Conclusion

Ce rapide survol permet d'avoir une vue d'ensemble des possibilités offertes par les applications développées dans ce travail de Bachelor. Il ne fait aucun doute qu'après quelques essais l'utilisation de ces deux programmes deviendra un automatisme.

4 Processus

L'objectif visé par ce chapitre est l'explication plus en détails du travail effectué dans l'implémentation de la solution. Il est intéressant, du point de vue d'un professionnel du développement informatique, de comprendre le fonctionnement interne de l'application d'administration et de développement. En se basant sur la figure présentée plus tôt, chaque étape est analysée dans l'ordre chronologique.

Figure 53 Processus de création de visualisations

4.1 Récupération des données

La toute première étape du processus consiste à construire une requête SPARQL. Cette dernière va ensuite être envoyée vers un ou plusieurs endpoints qui vont retourner des résultats.

Les administrateurs qui utilisent cet outil connaissent la syntaxe utilisée pour créer des requêtes SPARQL. Ils ont la possibilité d'utiliser le formulaire de l'application d'administration ou de créer une nouvelle requête de toute pièce.

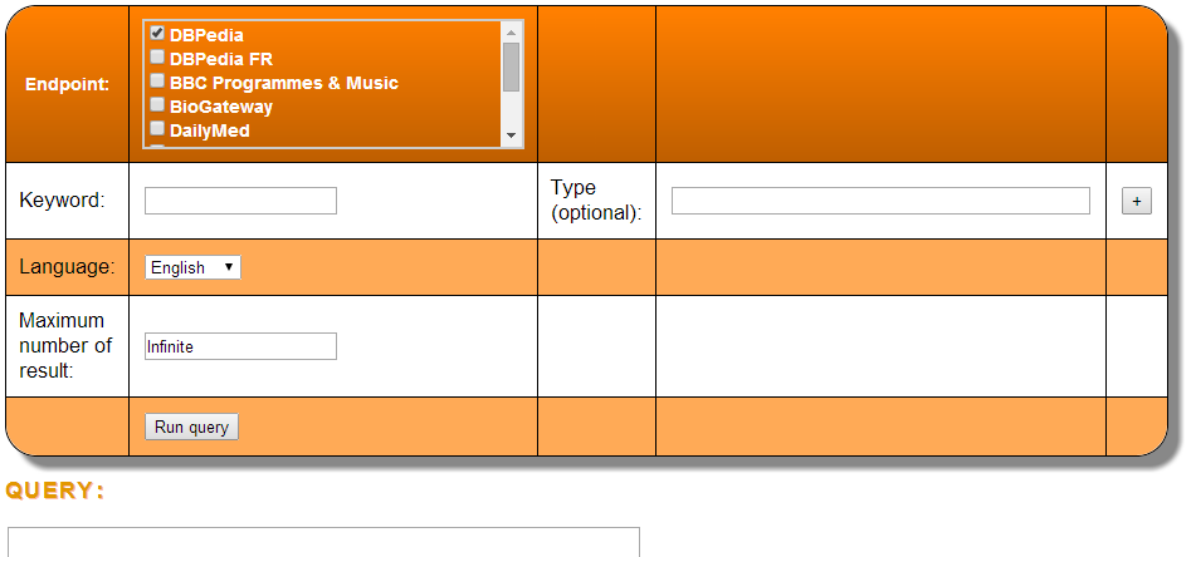


Figure 54 Formulaire de création de requêtes SPARQL

Lorsque le bouton d'envoi de requête est pressé, l'application récupère la requête ainsi que les endpoints sous forme de liste. La requête est ensuite envoyée vers chaque endpoint séparément. La communication avec les endpoints est dédiée à la librairie Sgvizler. Pour des questions de sécurité, la communication entre différents domaines est bloquée. Cependant, Sgvizler possède des fonctionnalités qui permettent d'effectuer des modifications au niveau des paquets qui partent sur le réseau et l'envoi et la récupération de données à travers internet. L'objectif de cette requête est de récupérer les URL des concepts qui satisfont aux conditions énoncées dans la requête.

À chaque résultat retourné, une seconde requête est construite. Ce mécanisme est caché à l'utilisateur. Dans ces requêtes secondaires, les endpoints vont retourner cette fois-ci la totalité des propriétés de chaque concept. Ces requêtes sont construites sous la forme suivante :

select distinct ?property ?obj
where
{
<URL du concept> ?property ?obj.
}

Une fois que toutes les réponses des requêtes secondaires d'un endpoint sont parvenues à l'application, les visualisations se construisent.

Tout d'abord, le tableau est créé puis les lignes s'ajoutent dynamiquement à l'intérieur. Le système de sélection du tableau est ensuite mis en place.

Par la suite, la vue en forme de graphe, dont la librairie D3.js a la responsabilité de la création, est générée. En se basant sur les triplets renvoyés par les endpoints, les nœuds sont créés. Au même moment, les liens entre les nœuds sont spécifiés. Dès que la librairie possède tous les nœuds, un conteneur est ajouté à l'écran. Grâce à quelques options données à D3.js telles que les dimensions, la physique et le texte, cette dernière est capable de gérer le graphe entièrement. Toutefois, la sélection des nœuds, n'étant pas implémentée à l'intérieur de la librairie, a fait l'objet d'une fonctionnalité ajoutée personnellement.

Arrivé à ce point, l'application réagit aux interactions avec l'utilisateur.

4.2 Exploration des données

Dans cette phase, l'administrateur manipule la sélection de données. Le but étant d'exporter à la fin les informations dont il a besoin.

Pour gérer l'extraction, deux vues sont disponibles : le graphe et le tableau. Au-delà de leur apparence très différente, le fonctionnement est identique. Lorsque qu'une donnée est sélectionnée, elle est copiée à l'intérieur d'un tableau qui sera transformé en fichier JSON lors de l'exportation finale.

La différence entre les deux vues étant la façon dont l'application perçoit la sélection. Dans le cadre du graphe et de la librairie D3.js, le clic sur un cercle est capté par la librairie. Afin de savoir quoi faire des données sélectionnées, la méthode va observer la couleur du cercle et réagir en conséquence. Par exemple, si le cercle est blanc, cela signifie qu'il a été précédemment sélectionné, ainsi, les données qu'il renferme sont envoyées dans le tableau d'exportation.

Le tableau de sélection est, quant à lui, intégralement construit par l'étudiant. À la création de chaque cellule, cette dernière reçoit la méthode qui va définir son

comportement lors d'un clic. Elle va vérifier si la touche « majuscule » est enfoncée et si l'option « Selection mode » est activé entre autre.

Figure 55 Les données peuvent être explorées de deux façons différentes

4.3 Extraction des données

Au moment où l'administrateur est satisfait des informations qu'il compte distribuer aux développeurs, il a la possibilité d'exporter sa sélection à l'intérieur d'un fichier JSON.

L'étape initiale consiste à rendre le tableau exportable, c'est-à-dire faire en sorte que sa structure soit supportée par l'outil responsable de la conversion du tableau en texte JSON.

Quand le texte JSON est disponible, les librairies Blob.js et file_saver.js sont en mesure d'ouvrir une fenêtre de dialogue demandant la location du futur fichier JSON et de l'écrire à l'emplacement spécifié par l'utilisateur.

Figure 56 Le bouton "Export results" récupère la sélection et les place dans un fichier

Cette action termine le processus du point de vue de l'administrateur.

4.4 Affinage des données

Cette étape constitue la première action du développeur en vue de créer une visualisation. Les données que le développeur importe dans son projet ont besoin d'être triées une fois encore pour permettre la génération de visualisations.

Pour disposer de données, une fonction d'importation de fichier JSON a été développée. Le fichier contenant les données au format JSON est dé-sérialisé avec la fonction inverse que l'administrateur a utilisée pour exporter les données. Il en résulte un tableau stocké dans la mémoire du navigateur avec lequel un tableau d'affichage va pouvoir être créé.

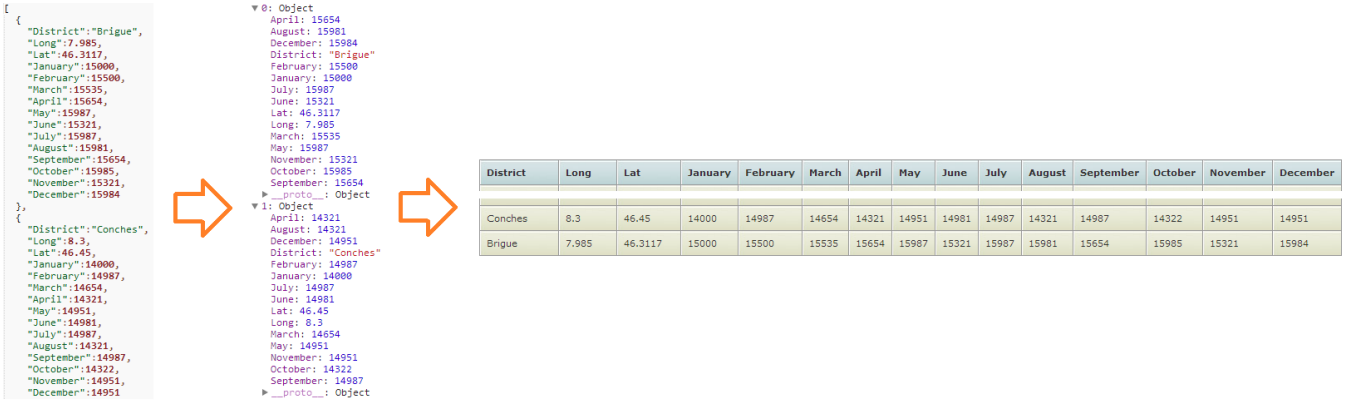


Figure 57 Transformation des données, de JSON, en mémoire et dans le tableau

La construction de ce tableau avec son système de sélection est sensiblement le même procédé que pour le tableau de l'application d'administration.

En fonction de la visualisation choisie, les prérequis à sa construction ne sont pas les mêmes. C'est pourquoi ce tableau permet de vérifier cellule après cellule si la visualisation fonctionne. Lorsqu'une cellule est activée, la donnée qu'elle renferme est placée à l'intérieur d'un tableau. Toutes les données contenues dans ce dernier serviront à dessiner un aperçu de la visualisation finale.

4.5 Création des visualisations

La création de l'aperçu de la visualisation et la version finale utilisent les mêmes fonctionnalités.

Pour commencer, la visualisation choisie par le développeur est identifiée. La méthode de construction est appelée en fonction de ce choix. Les méthodes de construction récupèrent les données, les formatent, récupèrent le conteneur à l'intérieur duquel sera dessinée la visualisation, spécifient les options de celle-ci, dessinent la visualisation sur l'espace de travail et finalement stockent les visualisations avec leurs données à l'intérieur d'un tableau qui servira à la sauvegarde et à la récupération de l'espace de travail.

Les données ont besoins d'être formatées correctement avant de servir dans une visualisation. L'une des librairies utilisées pour créer des graphiques est Google Charts. Cette dernière n'accepte qu'une structure très précise qui ne correspond pas forcément à la table affichée à l'utilisateur.

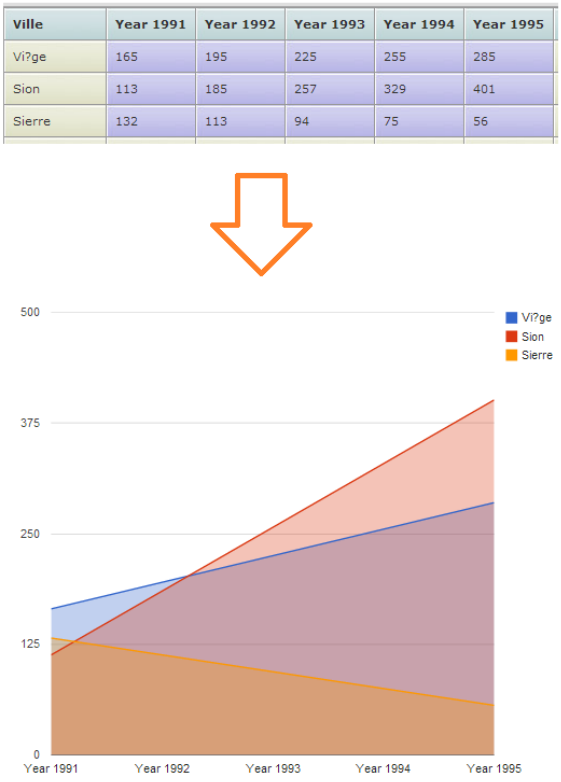


Figure 58 L'aperçu permet de se rendre compte du résultat final rapidement

4.6 Publication de visualisations

Afin de publier les visualisations finales, l'action est déclenchée par le bouton d'exportation du menu. Un tableau récupère tous les conteneurs présents sur l'espace de travail et supprime les conteneurs servant à redimensionner. Dès que les conteneurs sont « nettoyés », une fenêtre de dialogue demande à l'utilisateur l'emplacement dans lequel il souhaite sauvegarder ses visualisations à la suite de quoi un fichier contenant le code HTML de chaque visualisation est créé.

Figure 59 Bouton utilisé pour exporter les visualisations

Le responsable d'un site web n'a plus qu'à copier les visualisations sur une page pour les rendre accessibles à tous les utilisateurs.

4.7 Consultation de visualisations

Une fois une visualisation placée sur une page internet, tous les visiteurs peuvent la consulter. Comme tous les sites internet, les utilisateurs finaux doivent disposer d'un navigateur, d'une connexion internet et connaître l'adresse du site qu'ils souhaitent visiter.

4.8 Fonctionnalités manquantes

Parmi tous les cas d'utilisation énoncés plus tôt, quelques-uns n'ont malheureusement pas pu être implémentés. La raison à cela est une combinaison de la complexité technique de la fonctionnalité et du temps restant souvent très court. Ces fonctionnalités secondaires reportées sont listées ci-après.

Visualiser les liens entre les données : Tout d'abord, le graphe de l'application d'administration ne convainc pas. L'affichage est confus et l'utilisation n'est pas pratique. Quelques heures supplémentaires auraient été nécessaires à son amélioration. Il s'agit de l'une des premières fonctionnalités implémentées au début du projet qui a été laissée de côté par la suite pour que l'étudiant puisse se concentrer sur d'autres fonctionnalités

importantes. Ce graphe utilise la librairie D3.js que l'étudiant ne maîtrise pas encore. Le développement du graphe a nécessité par conséquent beaucoup de temps.

Exporter/Importer des données JSON-LD : Comme expliqué précédemment, JSON-LD est un format JSON spécialement étudié pour les données liées. Les applications actuelles se servent de JSON car la manipulation est plus simple. L'idée était de se baser sur l'implémentation de la partie JSON et l'adapter rapidement pour du JSON-LD. Au final, cette transition risque de ne pas être aussi simple car les structures, bien que proches, ont besoin d'un traitement spécifique chacune.

Personnaliser les visualisations : En l'état, l'utilisateur du logiciel de création de visualisations ainsi que l'utilisateur final n'ont pas la possibilité de customiser leurs visualisations à volonté. Bien que cette tâche ne soit pas particulièrement complexe au sens technique, d'un commun accord, les responsables et l'étudiant ont décidé de se consacrer sur les fonctionnalités importantes en premier lieu. L'échéance du travail se rapprochant très vite, la personnalisation a dû être laissée de côté.

Filtrer les informations : La visualisation complexe de l'application de développement permet de filtrer les informations qui apparaissent à l'écran. Ce n'est malheureusement pas le cas des visualisations publiées sur des sites web. Tout d'abord, le site en question doit posséder la librairie Google Charts pour pouvoir proposer des visualisations dynamiques. Dans le cas contraire, la visualisation n'est qu'une image vectorielle dessinée sur la page. Ensuite, le système de filtrage ne dépend pas de la visualisation mais d'un script qui l'accompagnerait. Cette fonctionnalité est par conséquent du ressort du site web recevant les visualisations.

4.9 Exemple appliqué

Finalement, après le survol détaillé de chaque phase visant à publier des visualisations, un exemple concret permet d'asseoir la compréhension de la solution et de son intérêt dans le domaine du web sémantique.

Premièrement, avant d'extraire des données, un objectif doit être fixé afin d'orienter les recherches de données. Dans cet exemple, le résultat doit montrer l'évolution du taux d'humidité dans l'air de la ville de Genève.

Dès le but à atteindre établi, l'administrateur se rend dans l'application permettant d'extraire des données. Après avoir sélectionné le endpoint « DbPedia », l'administrateur peut créer une requête SPARQL. Il peut s'aider du formulaire d'aide à la saisie ou écrire la requête de ses mains s'il le souhaite. Le bouton « Run query » envoie la requête

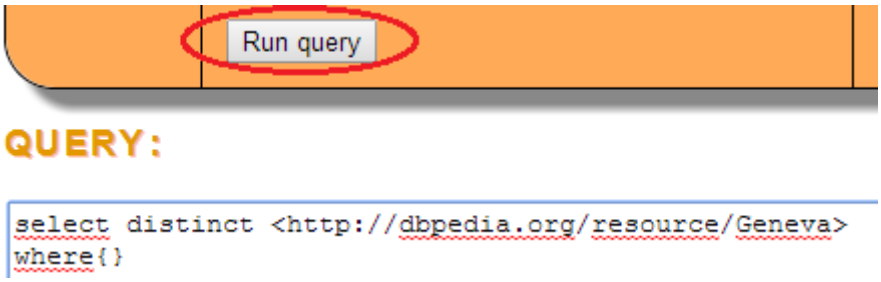
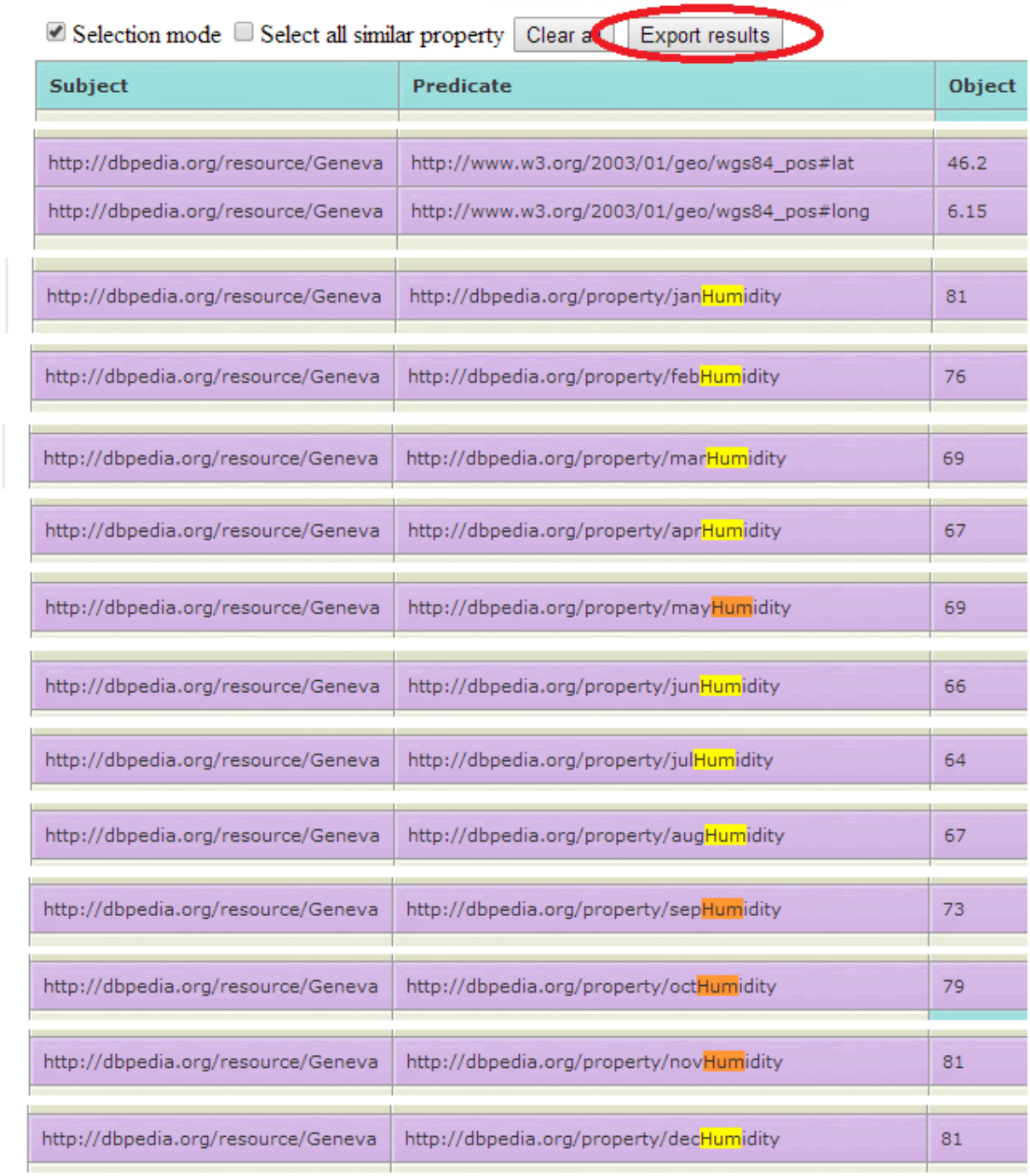


Figure 60 Requête SPARQL sélectionnant la ressource "Genève" directement

Après quelques instants, les vues listant les données apparaissent dans le bas de l'écran. Grâce au tableau, l'administrateur voit toutes les propriétés du concept « Genève ». Beaucoup d'éléments sont listés, aussi doit-il trier les propriétés utiles à la visualisation finale. Dans cet exemple, on souhaite récupérer les coordonnées géographiques de la ville pour la représenter sur une carte ainsi que les valeurs de l'humidité disponibles. Après une analyse rapide, l'administrateur constate que toutes les propriétés concernant l'humidité contiennent la chaine de caractère « hum ». Grâce à la fonction de recherche disponible sur les navigateurs (ctrl+f), l'administrateur trouve en quelques secondes tous les taux d'humidité de la ville de Genève sur une année. Une fois la sélection terminée, un simple clic sur le bouton d'exportation et le fichier JSON contenant ces données est instantanément créé.



Subject	Predicate	Object
http://dbpedia.org/resource/Geneva	http://www.w3.org/2003/01/geo/wgs84_pos#lat	46.2
http://dbpedia.org/resource/Geneva	http://www.w3.org/2003/01/geo/wgs84_pos#long	6.15
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/janHumidity	81
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/febHumidity	76
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/marHumidity	69
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/aprHumidity	67
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/mayHumidity	69
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/junHumidity	66
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/julHumidity	64
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/augHumidity	67
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/sepHumidity	73
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/octHumidity	79
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/novHumidity	81
http://dbpedia.org/resource/Geneva	http://dbpedia.org/property/decHumidity	81

Figure 61 Sélection des données sur le tableau

Malheureusement, l'application du développeur n'est pas en mesure de traiter le fichier JSON en l'état. Les librairies qu'utilise ce deuxième programme requièrent une structure de données très précise pour fonctionner. L'administrateur doit donc transformer le fichier JSON exporté avant de le communiquer aux développeurs. Cette étape, pour des raisons

techniques qui seront probablement corrigées dans une version ultérieure, est hélas nécessaire.

Figure 62 Conversion de la structure JSON pour l'importation dans l'application de développement

Après cette étape, le rôle de l'administrateur est terminé.

À présent, en tant que développeur, les données à utiliser sont disponibles. Grâce au bouton « Importer des données », le développeur charge le lot de données dans le programme. Il souhaite réaliser deux visualisations différentes : la première dessinera l'évolution de l'humidité de la ville de Genève sur une année tandis que la deuxième représentera sur la carte la ville avec la possibilité d'observer les données de chaque mois.

Dans le premier cas, une aire peut être placée sur l'espace de travail en cliquant sur le symbole « Area Chart ». Dans le menu de propriétés, le développeur sélectionne l'option servant à lier des données à une visualisation. Le tableau reprenant les données apparait et le développeur aguerri a tôt fait de parvenir au résultat escompté.

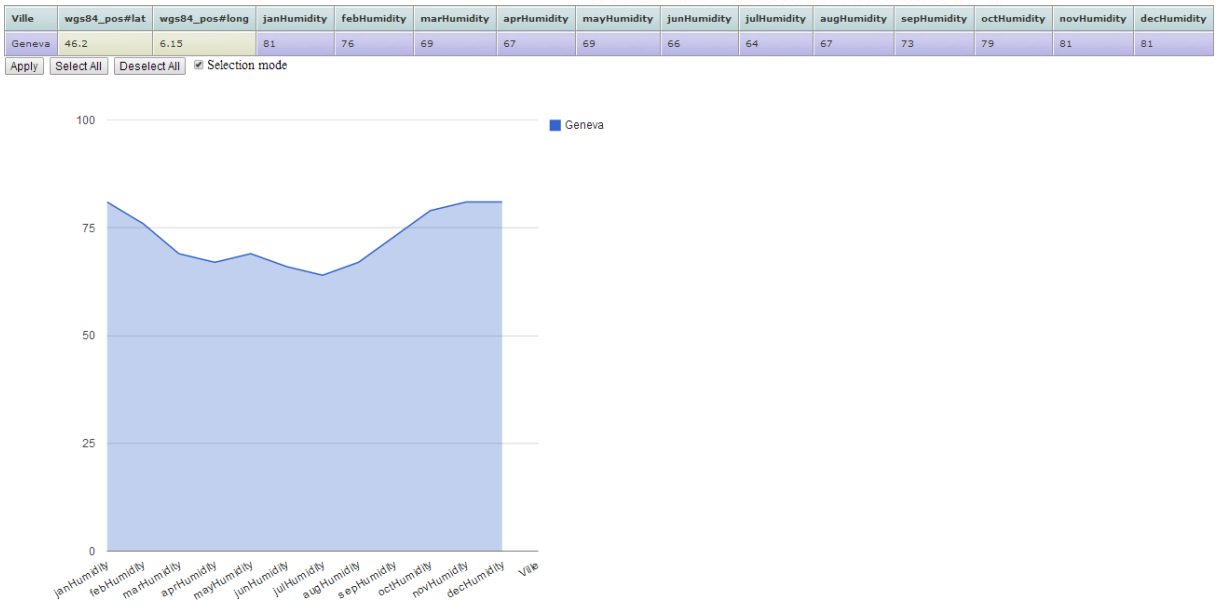


Figure 63 Grâce à l'aperçu, le développeur voit en temps réel la conséquence de ses actions

Dans le second cas, il s'agit d'une visualisation complexe. De la même manière, le développeur sélectionne les données nécessaires. Au moment de la validation des données sélectionnées, des fenêtres de dialogue demandent les colonnes comportant les noms des lieux et les données géographiques. La visualisation se construit ensuite et le développeur a loisir de changer les mois et constater les modifications à l'écran.

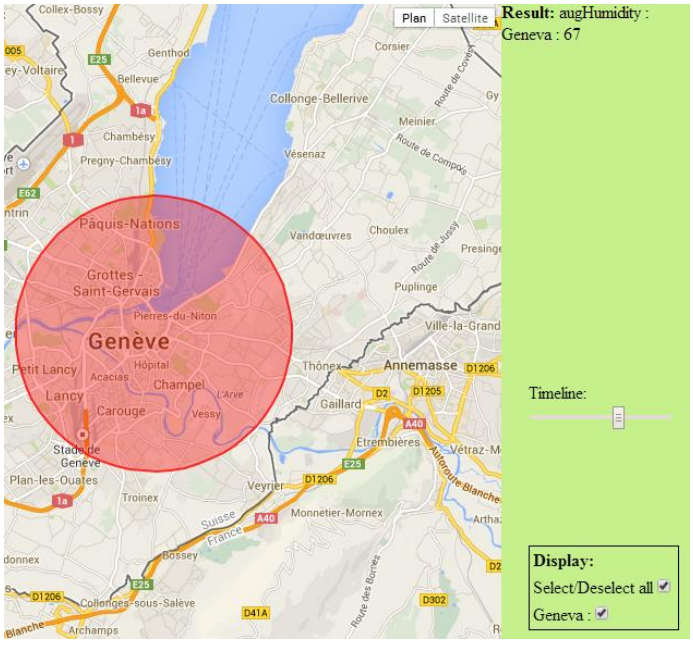


Figure 64 Visualisation complexe proposée par l'application

La dernière tâche du développeur consiste à exporter les visualisations en s'aidant du bouton prévu à cet effet sur le menu principal. Son travail est à présent fini.

Les web designers intéressés par la visualisation n'ont plus qu'à l'incorporer dans leurs pages web.

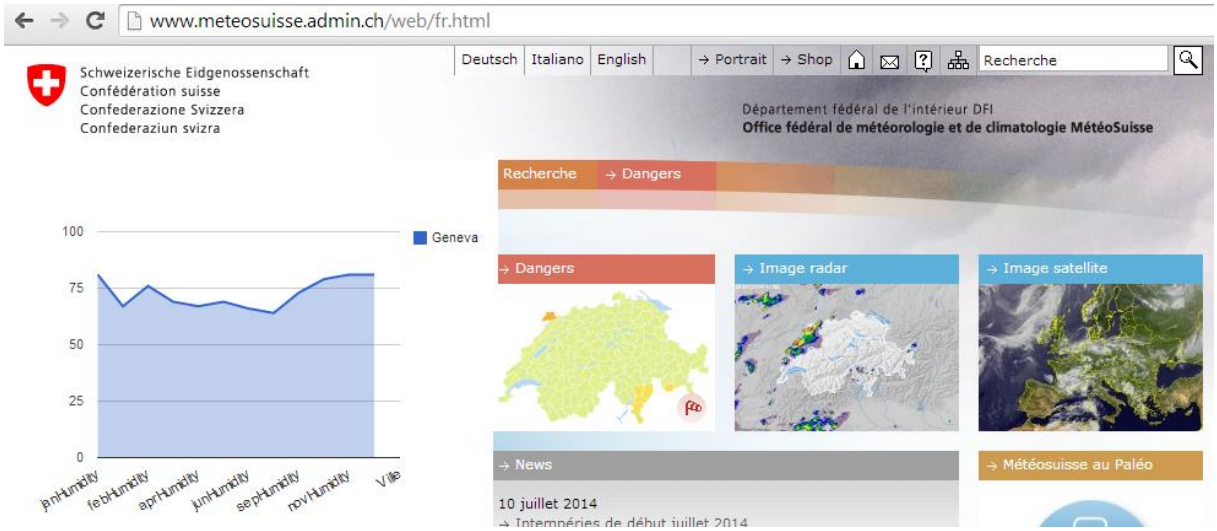


Figure 65 Exemple de publication de visualisation sur le site de Météo Suisse[7]

[7] L'apparence du site web n'a été modifiée qu'à l'intérieur du navigateur pour les besoins de l'illustration

5 Déroulement du projet

Cette partie relate de l'organisation de ce travail de Bachelor et rapporte des informations quant à la gestion du projet d'un point de vue administratif.

5.1 Planification

La planification du projet dans le temps imparti a été l'une des premières tâches de l'étudiant. Avec l'aide de l'équipe du projet OverLOD, une estimation des phases nécessaires à l'élaboration de la solution a été menée. La planification de ce travail de Bachelor s'est étendue du 5 mai au 28 juillet 2014, bien que la première entrevue avec l'équipe du projet OverLOD se soit faite le 16 avril 2014.

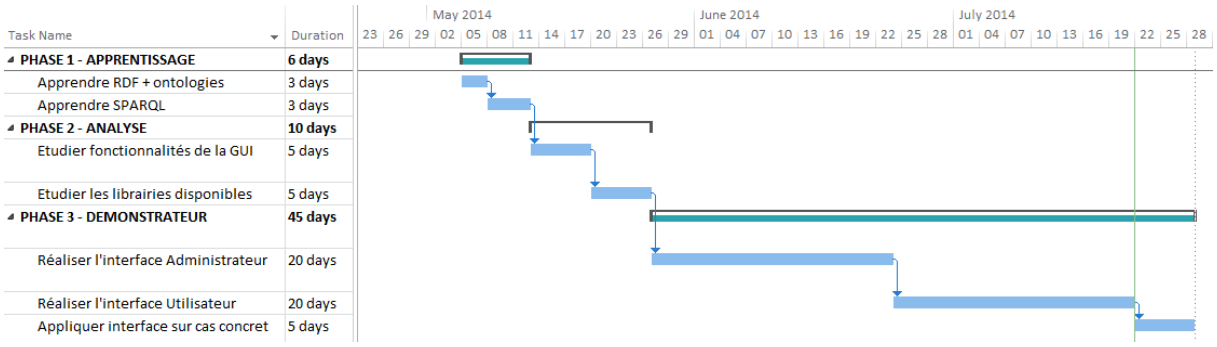


Figure 66 Planification réalisée au début du projet

Le projet a été séparé en trois phases distinctes. La durée de la première phase, qui concerne l'apprentissage des technologies utilisées dans le domaine du web sémantique, a été estimée à six jours. L'analyse des fonctionnalités de la solution ainsi que l'étude des librairies devaient durer dix jours. Finalement, l'élaboration du démonstrateur allait occuper le temps restant, soit 45 jours.

Dans la réalité, la planification a été légèrement altérée. La phase d'apprentissage a duré plus longtemps. La complexité du sujet a été sous-estimée au départ. L'étape d'analyse a, quant à elle, diminuée de quelques jours pour deux raisons. Premièrement, la participation à l'événement Make Open Data [31] a permis à l'étudiant de se familiariser avec quelques librairies. En outre, l'étude des fonctionnalités de la solution et des librairies se sont

déroulées en parallèle, contrairement au plan initial. La durée de la phase de développement n'a pas été altérée malgré ces modifications. Toutefois, l'application de développement a été pensée dès le départ à partir de données touristiques fictives.

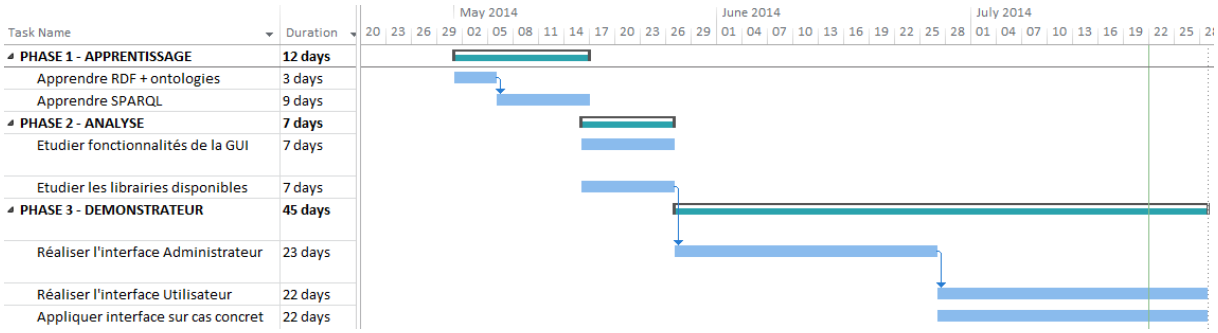


Figure 67 Planification réelle à l'issue du projet

En réalité, la dernière semaine, qui devait être consacrée à l'application d'un cas concret grâce à la solution développée, a servi à avancer les retards accumulés sur la rédaction du rapport final. Ce manquement d'écriture est dû à l'effort fourni à l'application de développement pour tenir les délais.

5.2 Réunions

Une réunion hebdomadaire a été décidée d'un commun accord entre la professeure responsable, un assistant et l'étudiant. La fréquence élevée de ces entrevues a pour but de suivre et anticiper l'avancement du projet. Pendant les réunions, l'étudiant fait part du travail accompli durant la semaine précédente. Il motive ses décisions et explique les problèmes qu'il a rencontrés. La responsable et son assistant font des remarques sur les éléments présentés par l'étudiant et donnent des pistes sur les tâches à accomplir pour la prochaine séance.

Vers la fin de ce travail de Bachelor, les réunions ont été rapprochées afin de surveiller l'avancement de la phase de développement plus précisément.

Étant donné que les objectifs de ce travail de Bachelor n'avaient pas été déterminés précisément à l'avance, un développement itératif a été privilégié, les réunions faisant office

de planification et de vérification à la fois. Le compte rendu de chaque réunion se trouve en annexe[8].

5.3 Décompte des heures

Le document remis en annexe[9] précise de quelle façon le temps a été géré par l'étudiant. Le décompte des heures précise la date, la durée, la tâche effectuée ainsi que les remarques relatives. Les durées sont estimées en minutes et la somme des heures figure à la fin du document (total de minutes / 60 = total d'heures).

Les tâches plus ou moins complexes s'alternent et, d'une manière générale, s'allongent sur la fin du projet.

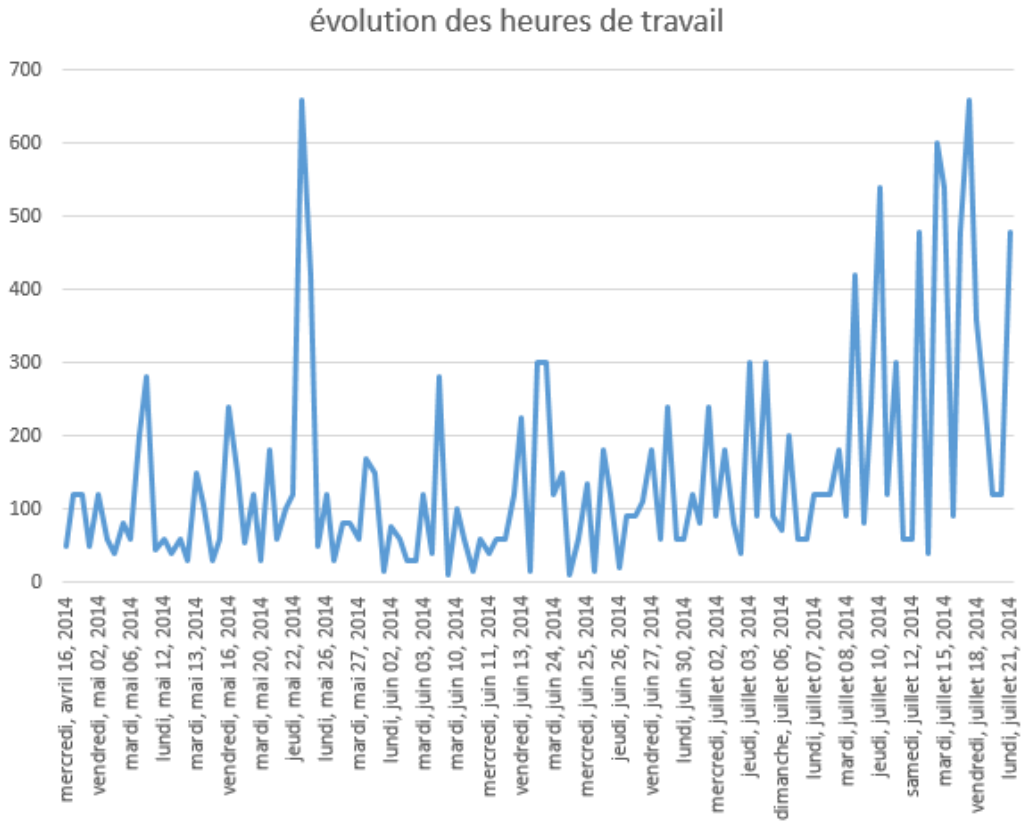


Figure 68 Évolution du nombre d'heures consacrées au travail de Bachelor

8 Voir Procès-verbaux des réunions
9 Voir Décompte des heures

5.4 Problèmes rencontrés

La première difficulté à faire face pour l'application d'administration a été la limitation du cross-site scripting (XSS). Le XSS est une vulnérabilité dans les réseaux informatiques qui permet aux personnes malintentionnées d'exécuter des scripts sur des sites distants. Afin d'empêcher ces problèmes, les sites internet et les navigateurs ignorent généralement les scripts non-identifiés. À cause de cette limitation, toutes les tentatives de l'étudiant d'envoyer une requête SPARQL à un endpoint ont échoué. Seule la librairie Sgvizler parvenait à récupérer des données, c'est pourquoi ses fonctionnalités de communication ont été exploitées dans l'application d'administration. Sgvizler se voit attribué l'adresse du endpoint ainsi que la requête à envoyer et est responsable de récupérer les données. Ce système fonctionne et est autorisé par la licence de la librairie.

La récupération de toutes les propriétés d'un concept a posé quelques problèmes plus tard. Les endpoints les plus sollicités ont été forcés d'imposer une limite à la complexité des requêtes SPARQL qu'ils recevaient pour éviter des temps de latence trop important. Récupérer toutes les propriétés d'une centaine de concepts est évidemment un processus très coûteux en performance pour le endpoint. Pour pallier à ce problème, l'idée a été de scinder la requête trop complexe en requêtes plus simples à traiter pour le endpoint. Cette solution a amené le principe de récupération de données à deux temps expliqué dans la section 4.1 Récupération des données. La première requête générale rapporte les URIs des concepts correspondant aux conditions de l'utilisateur et une nouvelle requête rapportant toutes les propriétés d'un concept est envoyée pour chaque URI. Cette solution a résolu le problème et est transparente aux yeux de l'utilisateur.

Dans le cadre de l'application de développement, plusieurs complications sont apparues.

Comme expliqué précédemment, la librairie choisie pour générer les graphiques a été Google Charts. Le point déterminant dans le choix de cette librairie a été la similarité des structures de données entre tous les graphiques. Ainsi, avec les mêmes données, Google Charts est capable de présenter plusieurs visualisations. La difficulté résidait dans le fait que la librairie a besoin d'une structure très précise en entrée et que les données reçues, visibles dans le tableau, sont présentées d'une façon différente. Il a par conséquent fallu développer

une fonctionnalité capable de transformer les données dans la forme adéquate. La particularité de cet accroc s'est avéré être le langage de programmation JavaScript. Comme l'explique le site w3schools [39] au sujet des tableaux en JavaScript : « Plusieurs langages de programmation supportent les tableaux indexés à l'aide de nom. Ces tableaux sont appelés tableaux associatifs. JavaScript ne supporte pas les tableaux associatifs.[10]» Cela signifie qu'un tableau indexé avec des noms existe en réalité sous la forme d'un tableau d'objets. Cette nuance empêche de transformer la structure d'un tableau aisément. Finalement, à l'aide de réflexion et de persévérance, cette spécificité a pu être outrepassée et la structure de données modifiée correctement.

Une fois la visualisation créée, l'utilisateur devrait être en mesure de redimensionner la visualisation. Grâce à une petite librairie appelée dragresize.js, les conteneurs peuvent être déplacés à la souris et les extrémités possèdent des attaches qui servent à étirer ou à réduire le cadre. Malheureusement, lorsque Google Charts crée une visualisation à l'intérieur d'un conteneur, cette dernière est constituée d'autres conteneurs qui ne bénéficient pas des propriétés offertes par dragresize.js. La difficulté se résume à rechercher tous les conteneurs et à leurs appliquer la fonction de redimensionnement. Malgré un bon nombre d'heures consacrées à la résolution de cette complication, aucun résultat satisfaisant n'est ressorti et il a été jugé plus sage de continuer les autres fonctionnalités.

Enfin, l'utilisateur de l'application de développement est en mesure de sauvegarder son espace de travail et de le récupérer. Cette fonctionnalité fait l'objet de réflexion depuis le début de l'implémentation de l'application. Il a tout d'abord fallut cibler les informations à enregistrer telles que la position de chaque visualisation, sa dimension et son contenu. En observant la structure HTML d'une page contenant des visualisations, l'étudiant a constaté que toutes ces informations étaient contenues à l'intérieur des balises représentant les conteneurs des visualisations. La première itération faisait donc le tri entre les visualisations et les conteneurs servant à l'application (comme le menu principal par exemple), récupérait les informations et les enregistrait au format JSON. À l'ouverture d'un espace de travail précédemment sauvé, la fonction de récupération recréait chaque conteneur et ajoutait les informations à l'intérieur. Dans la seconde itération, il a été possible de copier

[10] Traduit de l'anglais par Quentin Oberson, juillet 2014

complétement le contenu des balises et de les coller simplement à la récupération. Cependant, aucune donnée n'était liée à aucune des visualisations. Ainsi, lorsque l'utilisateur affichait le tableau de données d'une visualisation rechargée, aucune cellule n'était active. La résolution de ce problème a nécessité une reconfiguration complète du système de sauvegarde puisqu'il fallait également sauver toutes les données représentées dans chaque visualisation. C'est pourquoi, dans la dernière version, les données sont copiées en même temps que les balises et placées dans un tableau à dimension multiple. Lorsque l'espace de travail est ré-ouvert, les balises sont collées sur l'espace de travail et les données sous-jacentes sont utilisées pour activer les cellules du tableau au moment où l'utilisateur l'affiche. De cette façon, le tableau présenté à l'utilisateur reflète en tout temps l'état réel de la visualisation sélectionnée.

5.5 Tests

Les premiers tests auxquels ont été soumises les deux applications sont des tests fonctionnels par l'étudiant. Le premier critère de complétion est le comportement adéquat des fonctions. Grâce aux outils de développement disponibles dans les navigateurs, il est possible d'analyser précisément l'état de la page web et les valeurs des variables stockées en mémoire.

À plusieurs reprises, une modification du code a déclenché des comportements anormaux dans d'autres fonctionnalités. Les méthodes étant réutilisées maintes fois, leur implémentation doit prendre en considération tous les cas de figure. Si ces anomalies sont détectées tardivement, d'importantes modifications seront nécessaires. À leur tour, ces dernières pourront potentiellement découler sur des erreurs.

Pour pallier à ce problème, les tests unitaires sont la solution. Ils contrôlent chaque fonction et permettent de mettre en évidence les erreurs de programmation à chaque exécution. Cependant, la mise en place d'une telle démarche prend du temps. De plus, l'étudiant n'est pas familiarisé avec les tests unitaires d'application JavaScript. Après réflexion, il a été estimé que l'apprentissage et l'exécution de tests unitaires prendraient davantage de temps que la correction de chaque défaut trouvé. De plus, comme le code source a beaucoup changé au fil des itérations, réécrire les tests au fur et à mesure aurait

été une perte de temps supplémentaire. Ces modifications continues ne pouvaient pas être évitées car aucun objectif clairement défini n'a été posé dès le début de ce projet. Toutefois, à partir de la version se trouvant en annexe de ce document, l'élaboration de tests unitaires sur l'ensemble de l'application est vivement conseillée puisqu'il s'agit de la première version stable.

Enfin, une série de tests d'ergonomie ont été passée. Les sujets de ces tests sont répartis en deux groupes. Le premier groupe concerne des personnes sans connaissance pointues en informatique qui peuvent objectivement donner leur avis sans prendre en compte la complexité cachée. La deuxième catégorie de personnes est constituée de professionnel de l'informatique. Ils ont conscience des potentielles difficultés et peuvent donner leur avis sur les décisions qui ont été prises.

Voici les éléments qui sont ressortis de l'analyse de l'ergonomie :

5.5.1 Catégorie utilisateur sans compétence technique

Application d'administration

Le choix de la langue des résultats de recherche a été interprété comme étant le choix de la langue du programme. Il manque un bouton pour sélectionner tous les endpoints dans la liste. Un bouton pour interrompre l'exécution de la requête pourrait être prévu. Le bouton pour ajouter une ligne de mot-clé est intuitif. La vue en forme de graphe est impressionnante à première vue mais pas simple à utiliser. Pour exporter les données, le réflexe est d'effectuer un clic droit plutôt que de chercher un bouton. Pour sélectionner plusieurs lignes du tableau, les utilisateurs utilisent la touche « control » avant de s'apercevoir que les lignes ne se désactivent pas lorsque l'on clique à côté. Le bouton pour exporter les résultats est facilement identifiable. Le texte qui accompagne la case à cocher « Selection mode » pourrait être renommé en « Deselection mode » pour éviter les confusions.

Application de développement

Le menu principal signalé sur la gauche n'a pas été trouvé à plusieurs reprises. La manipulation des conteneurs sur l'espace de travail est très intuitive. Lorsque des données

ont été importées dans le projet, un logo permanent devrait le rappeler. Le filtrage à l'aide du tableau pose des problèmes car le fonctionnement s'éloigne un peu des autres standards. Le message d'erreur sur fond rouge est perturbant car il se place par-dessus les intitulés des colonnes. Le bouton de suppression des visualisations est intuitif. La zone bleue servant à déplacer les visualisations est intuitive car elle attire l'attention. Le bouton biffé utilisé pour supprimer les alertes est placé trop près du menu, ce dernier apparaît non-intentionnellement. L'utilisation du bouton coulissant « timeline » est intuitive grâce à l'infobulle qui signale la position actuelle. Certains utilisateurs ont le réflexe d'utiliser la touche « delete » pour supprimer une visualisation. Le bouton qui nettoie complètement l'espace de travail est intuitif car il rappelle un tableau blanc immaculé. Les utilisateurs ont de la peine à lier des données à une visualisation car le bouton est discret et n'évoque rien de particulier. Le nom des colonnes ne devrait pas être caché par une alerte rouge lorsque la fenêtre de dialogue demandant les indexes des colonnes apparaît. Les propriétés d'une Google Map ne devraient pas apparaître avec un clic car cette action est déjà utilisée par la librairie pour naviguer sur la carte.

5.5.2 Catégorie utilisateur avec compétence technique

Application d'administration

Un message devrait être affiché lorsque le tableau apparaît car il est placé en dehors de l'écran. La quantité trop importante de liens présentés sur la vue en forme de graphe nuit à la lisibilité. Le système de désélection de plusieurs lignes à la fois n'est pas intuitif et peut être simplifié. D'une manière générale, les personnes ayant l'habitude de travailler avec les ordinateurs utilisent fréquemment les touches de raccourcis telles que « delete », « control + z », « control + a », « shift », etc. Les prochaines versions devraient tenir compte de ces habitudes.

Application de développement

Les utilisateurs ont intuitivement envie de faire glisser les visualisations présentées sur le menu principal vers l'espace de travail. Le bouton pour supprimer un conteneur et le bouton qui les supprime tous sont trop éloignés l'un de l'autre et l'utilisateur ne cherche que dans la zone proche. Le bouton « valider » qui met à jour les nouvelles dimensions d'un conteneur

n'est pas utile, les modifications devraient être apportées dès que la valeur du champ change. Les exemples d'index apparaissant sur la fenêtre demandant le nom des colonnes sont bienvenus. De plus, si les champs se trouvant dans ces fenêtres ne sont pas renseignés, l'application ne devrait pas construire de visualisations. Un bouton permettant de revenir à l'espace de travail devrait être ajouté lors du filtrage des données à l'aide du tableau.

5.5.3 Conclusion des tests

L'objectif recherché dans la composition de l'interface de visualisation était d'être le plus intuitif possible. Les néophytes devaient être capables de comprendre les fonctionnalités avant même de les tester.

Toutefois, malgré le niveau de connaissance technique différent entre les deux groupes, la première approche d'un nouveau logiciel reste une découverte pour tout le monde. Si un bon nombre d'actions a été réalisé facilement dès la première tentative, beaucoup d'éléments sont encore à améliorer.

Cette série de tests a été révélatrice de la façon dont l'utilisateur appréhende un nouvel environnement. Beaucoup d'actions faisant sens chez l'étudiant qui connaissait exactement le fonctionnement de l'outil se sont avérés confuses pour un nouvel utilisateur.

De même, le comportement hésitant des utilisateurs a permis de déceler quelques erreurs cachées au sein des applications.

Toutes ces remarques peuvent être prises en compte afin d'améliorer la deuxième version du programme reprise par les membres du projet OverLOD.

6 Conclusion

6.1 Bilan technique

Au terme de ce travail de Bachelor, le constat est plutôt encourageant. Les objectifs principaux ont été atteints. Les applications développées permettent l'extraction de données du web sémantique et la création de visualisations. La majorité des défis techniques qui se sont présentés a été résolue.

Toutefois, quelques éléments restent à améliorer. Le graphe présentant les données issues des endpoints a besoin d'être plus clair et plus pratique. Bien qu'un effort particulier a été porté à l'ergonomie des applications, il subsiste des interactions qui ne sont pas intuitives et demandent un effort d'adaptation de la part de l'utilisateur. La transition des données de l'application d'administration à l'outil de création de visualisations a besoin de se soumettre à une structure standard de façon à ce que chaque fichier JSON puisse être récupéré dans le second programme sans aucune modification. Enfin, quelques heures seront nécessaires à la suppression des derniers défauts de programmation.

6.2 Avis personnel

Ce travail de Bachelor, qui demeure un projet expérimental, manquait d'objectifs clairs à atteindre dès le début. Je préfère savoir quels sont les buts visés de façon à pouvoir estimer l'ampleur des tâches restantes. Durant la phase de développement du projet, j'ai eu parfois l'impression d'être perdu, sans direction claire à suivre. Cependant, cette contrainte a été atténuée grâce à un suivi continu de la part de la professeure responsable qui a su m'aiguiller dans les instants de doute.

Le fonctionnement particulier du JavaScript, que je connaissais très peu, m'a posé des problèmes d'ordre technique. À plusieurs reprises, son comportement m'a surpris. Contrairement aux langages compilés avec lesquels j'ai l'habitude de travailler, le JavaScript est interprété par le navigateur de l'utilisateur. Cette particularité peut expliquer pourquoi un programme fonctionne avec le débuggeur enclenché et ne fonctionne plus sans par exemple. Malgré tout, je suis content d'avoir pu améliorer mes compétences en développement web qui me seront sûrement utiles dans un avenir tout proche.

Finalement, arrivé au terme de ce projet, je constate que je suis davantage attiré par le fonctionnement technique du web sémantique que par l'aspect graphique des représentations de données. Dans tous les cas, ce travail de Bachelor m'a énormément appris. Désormais, je connais mieux mes atouts, que je vais devoir développer, et mes faiblesses, que je vais devoir renforcer.

6.3 Idées d'amélioration

Le résultat de ce travail de Bachelor est loin d'être parfait. Voici quelques idées que pourra développer l'équipe du projet OverLOD dans la deuxième version des applications.

Premièrement, la vue en forme de graphe de l'application d'administration est brouillonne. Trop d'éléments apparaissent à l'écran, la structure est confuse et les animations ne sont pas pratiques.

Une structure standard entre les deux applications peut être apportée de façon à pouvoir récupérer directement les données sans devoir les adapter au préalable.

Des visualisations supplémentaires peuvent avoir leur place dans l'application de développement.

Malheureusement, quelques bugs subsistent dans les applications. Une suite de tests unitaires permettrait de les repérer rapidement et d'éviter des problèmes en cascade causés par les modifications du code.

Dans sa version actuelle, l'application de développement utilise des données au format JSON. Afin d'incorporer plus d'informations sur les visualisations, l'utilisation du format JSON-LD pourrait apporter une dimension supplémentaire. Les ontologies, qui relient les données entre elles, pourraient révéler de nouvelles connaissances.

Avec l'aide d'un expert en ergonomie, il est possible de rendre les applications davantage intuitives si bien que n'importe quelle personne pourrait se servir de ces outils pour bénéficier de la force du web sémantique à l'avenir.

7 Références

1. W3C page d'accueil, http://www.w3.org/, dernier accès le 24.07.2014
2. W3C page d'accueil du web sémantique, http://www.w3.org/standards/semanticweb/, dernier accès le 24.07.2014
3. W3C page d'accueil du linked data, http://www.w3.org/wiki/LinkedData, dernier accès le 24.07.2014

4. What is Open ?, https://okfn.org/opendata/, dernier accès le 24.07.2014
5. Exemple inspiré de http://www.w3schools.com/webservices/ws_rdf_rules.asp, dernier accès le 24.07.2014
6. W3C recommandation d'OWL 2, http://www.w3.org/TR/owl-ref/, dernier accès le 24.07.2014
7. W3C Page de référence de RDF, http://www.w3.org/RDF/, dernier accès le 24.07.2014
8. Nicola Guarino, Daniel Oberle and Steffen Staab, What is an Ontology ?, http://userpages.uni-koblenz.de/~staab/Research/Publications/2009/handbookEdition2/what-is-an-ontology.pdf, dernier accès le 24.07.2014
9. W3C recommandation de SPARQL, http://www.w3.org/TR/rdf-sparql-query/, dernier accès le 24.07.2014
10. W3C recommandation de JSON-LD, http://www.w3.org/TR/json-ld/, dernier accès le 24.07.2014
11. Ronald M. Baecker, Readings in human-computer interaction: toward the year 2000, Morgan Kaufmann – 1995
12. Comparaison entre JSON et XML, http://www.json.org/xml.html, dernier accès le 24.07.2014
13. 10 règles pour la création d'interface graphique, http://medicalcomputing.org/archives/0agui.php, dernier accès le 24.07.2014
14. Page d'accueil de Sigma, http://sig.ma/, dernier accès le 26.05.2014, adresse invalide le 24.07.2014, aucune alternative trouvée
15. Page d'accueil de Sindice, http://sindice.com/, dernier accès le 24.07.2014
16. Publication concernant Sigma, http://www.sciencedirect.com/science/article/pii/S1570826810000624, dernier accès le 24.07.2014

17. Page de projets d'Alvaro Graves, http://graves.cl/projects, dernier accès le 24.07.2014
18. L'exemple provient de http://www.w3.org/2009/Talks/0615-qbe/
19. Page d'accueil de Lodlive, http://en.lodlive.it/, dernier accès le 24.07.2014
20. Ken Peffers, Hiver 2007-8, voir annexe Design Science Research Methodology
21. Page d'accueil de Lodash, http://lodash.com/, dernier accès le 24.07.2014
22. Page d'accueil de jQuery, http://jquery.com/, dernier accès le 24.07.2014
23. Page d'accueil de RDF Translator, http://rdf-translator.appspot.com/, dernier accès le 24.07.2014
24. Page d'accueil de D3.js, http://d3js.org/, dernier accès le 24.07.2014
25. Page de présentation de LGPL, http://www.gnu.org/licenses/lgpl.html, dernier accès le 24.07.2014
26. Page d'accueil de Google Maps API, https://developers.google.com/maps/, dernier accès le 24.07.2014
27. Contrat traduit de l'anglais par Quentin Oberson, http://opensource.org/licenses/BSD-3-Clause
28. What is an API ?, http://money.howstuffworks.com/business-communications/how-to-leverage-an-api-for-conferencing1.htm, dernier accès le 24.07.2014
29. Traduit de l'anglais par Quentin Oberson : http://creativecommons.org/licenses/by/3.0/
30. Termes de la licence d'utilisation de Google Maps API, https://developers.google.com/maps/terms, dernier accès le 24.07.2014
31. Présentation du projet créé lors de la journée Make Open Data, http://make.opendata.ch/wiki/project:linkedbisses, dernier accès le 24.07.2014
32. Page d'accueil de Sgvizler, http://dev.data2000.no/sgvizler/, dernier accès le 24.07.2014

33. Martin G. Skjæveland. Sgvizler: A JavaScript Wrapper for Easy Visualization of SPARQL Result Sets. In: 9th Extended Semantic Web Conference (ESWC 2012), workshop and demo proceedings. Heraklion, Crete, Greece, 2012

34. Page d'accueil de Google Charts, https://developers.google.com/chart/interactive/docs/index, dernier accès le 24.07.2014

35. What in the world is a wiki ?, http://www.teachersfirst.com/content/wiki/, dernier accès le 24.07.2014

36. Traduit de l'anglais par Quentin Oberson, Licence de la version 0.6 : http://dev.data2000.no/sgvizler/browser/release/0.6/LICENSE, dernier accès le 24.07.2014

37. Pour davantage de détails sur la licence, http://www.apache.org/licenses/LICENSE-2.0, dernier accès le 24.07.2014

38. W3schools documentation sur les tableaux en JavaScript, http://www.w3schools.com/js/js_arrays.asp, dernier accès le 24.07.2014

39. Introduction aux triplestores, http://semanticweb.com/introduction-to-triplestores_b34996, dernier accès le 25.07.2014

40. Introduction à l'architecture orientée services, http://www.service-architecture.com/articles/web-services/service-oriented_architecture_soa_definition.html, dernier accès le 25.07.2014

41. Aperçu de SPARQL, http://www.w3.org/TR/sparql11-overview/, dernier accès le 25.07.2014

7.1 Liste des figures

Image de la page de garde :

http://en.wikipedia.org/wiki/Linked_data#mediaviewer/File:LOD_Cloud_Diagram_as_of_September_2011.png

7.2 Liste des tables

7.3 Liste des abréviations

Abréviation	Signification	Référence
API	Application Programming Interface	http://money.howstuffworks.com/business-communications/how-to-leverage-an-api-for-conferencing1.htm
BSD	Berkeley Software Distribution	http://opensource.org/licenses

CSS	Cascading Style Sheets	http://www.w3.org/Style/CSS/Overview.en.html
CSV	Comma-Separated Values	http://tools.ietf.org/html/rfc4180
DSRM	Design Science Research Methodology	"A Design Science Research Methodology for Information Systems Research", voir annexe correspondante
HTML	HyperText Markup Language	http://www.w3.org/html/
HTTP	HyperText Transfer Protocol	http://www.w3.org/Protocols/
JavaScript	Langage de programmation s'exécutant sur un navigateur web	https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference
JSON-LD	JavaScript Object Notation for Linked Data	http://www.w3.org/TR/json-ld/
OWL	Web Ontology Language	http://www.w3.org/TR/owl-ref/
RDF	Resource Description Framework	http://www.w3.org/RDF/
SVG	Scalable Vector Graphics	http://www.w3.org/Graphics/SVG/
SPARQL	SPARQL Protocol and RDF Query Language	http://www.w3.org/TR/rdf-sparql-query/
URI	Uniform Resource Identifier	http://www.w3.org/TR/uri-clarification/
W3C	World Wide Web Consortium	http://www.w3.org/
XML	Extensible Markup Language	http://www.w3.org/TR/REC-xml/
XSS	Cross-site scripting	http://www.houbysoft.com/v/en/papers/xss/

7.4 Déclaration de l'auteur

Je déclare, par ce document, que j'ai effectué le travail de Bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées. Je ne donnerai aucune copie de ce rapport à un tiers sans l'autorisation conjointe du RF et du professeur chargé du suivi du travail de Bachelor, y compris au partenaire de recherche appliqué avec lequel j'ai collaboré, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail.

8 Annexes

Liste des annexes dans l'ordre :

1. **Procès-verbaux des réunions** : Procès_verbaux.pdf
2. **Décompte des heures** : Décompte_heures.pdf
3. **Base technique d'OverLOD** : OverLOD Technical basis.pdf
4. **Méthodologie de travail** : Design Science Research Methodology 2008.pdf

Procès-verbal du 05.05.2014

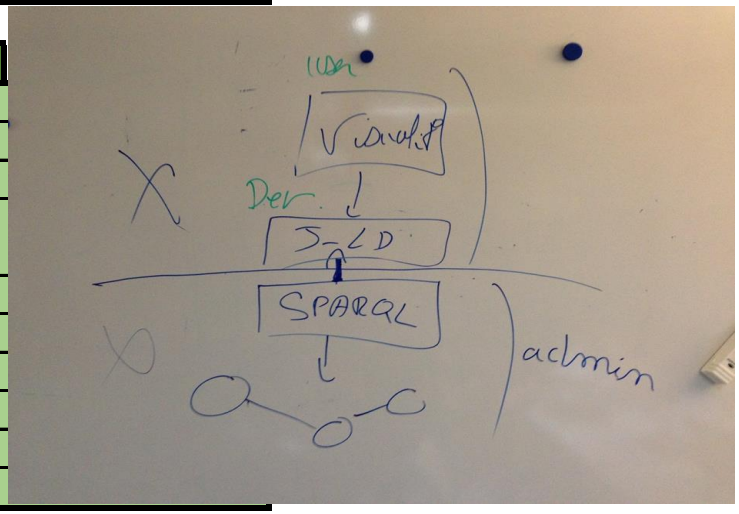
Discussions:
Le projet va comporter des phases itératives pour ajouter des fonctionnalités.
Il faut prévoir 1 semaine avant le 28 juillet pour commencer le rapport final et la présentation car c'est un travail très long.
Il est vivement conseillé de prendre des notes au fur et à mesure pour éviter de tout écrire à la fin. Le document « résumé » a été créé à cet effet.
La phase d'analyse va probablement être plus longue que la première estimation (prévoir une dizaine de jours).

Travail à effectuer:
Apprendre le format de requêtage SPARQL (trouver tutoriel sur le web, lire la théorie d'Anne)
Les technologies à regarder éventuellement :
-HTML5
-JavaScript et librairies graphiques
-RDF, RDFa, OWL, (savoir à quoi ressemble microdata et microformat)
-SPARQL
-JSON
-JSON-LD

Procès-verbal du	**12.05.2014**

Discussions:
On peut trouver à l'intérieur d'une ontologie (un fichier RDF) :
T-Box (ça correspond à un schéma pour les bases de données relationnelles)
A-Box (ça correspond aux données des bases de données relationnelles)
Nous avons dessiné l'architecture de base du travail à réaliser :
On trouve les ontologies tout en bas. SPARQL correspond à l'interface graphique de l'administrateur. Pour la première itération, elle doit ressembler à ça.
Ensuite, nous devons transformer le résultat de la requête en JSON-LD. Le développeur va créer une interface de visualisation pour l'utilisateur final avec ces données.
Nous avons parlé de l'éventualité de créer 2 parties à l'intérieur de l'interface de visualisation (1 pour le développeur/1 pour l'utilisateur).
Nous avons également soulevé l'éventualité de penser au projet en sens inverse, c'est-à-dire de partir de la visualisation finale et descendre jusqu'aux données.
Éviter de d'utiliser le pronom « on » dans le rapport final.
Ne pas hésiter à poser des questions génériques ou techniques par mail.

Travail à effectuer:
Finir le cahier des charges AVANT MERCREDI 14.05.2014.
Utiliser le nouveau template pour le cahier des charges.
Étudier un peu plus en profondeur JSON-LD
Faire des recherches sur les visualisations de données (trouver des idées + définir les critères d'une visualisation)



Procès-verbal du 19.05.2014

Discussions:
Quelques interfaces innovantes, impressionnantes
Le reste doit être facilement utilisable, compréhensible

Travail à effectuer:
Corriger le cahier des charges
Tester les librairies 3-4
Critères à trouver pour juger les librairies GRATUIT, LICENSES
Chercher les fonctionnalités de façon itérative
Méthodologie des choix, des recherches, JUSTIFIER
Pourquoi on a choisi cette librairie plutôt qu'une autre
PAS DE MEETING LA SEMAINE PROCHAINE => QUESTIONS PAR MAIL, si jamais mercredi après 16h
Voir étude bonne interface, google, google scholar

Procès-verbal du 02.06.2014

Discussions:

Travail à effectuer:
Voir librairie Flare pour visualisation
Faire le point avec Maria Sohkn, inviter pour la prochaine séance Mardi prochain.
Continuer sur interface SPARQL, voir photo du tableau blanc
Voir Google Web Designer
Approfondir les licenses qu'on utilise si besoin
Mettre les liens vers les licenses
Voir outils existants pour Admin!
URL utilisés avec Zhan:
https://dl.dropboxusercontent.com/s/kf87xf07h6fd4ur/test_cross_domain.html
http://153.109.124.88:8887/linkedbisses/quentin/test.html
Remote Desktop Connection - demander à Zhan pour accéder

Procès-verbal du 12.06.2014

Discussions:
Utiliser un zoom pour naviguer dans l'interface administrateur
Les deux interfaces sont indépendantes, car le rôle n'est pas le même. Prévoir un lien d'accès entre les deux.
Options pour cacher les noeuds inutiles, à voir selon le temps
Résultat sous forme de graphe, de cartes, ou autres => ABSTRAIT!!!
Etre plus professionnel: présentation soignée, entretien planifié, plus formel
Voir STRUCTURE OFFICIELLE pour le Rapport Final
Vérifier que les images couleurs passent en noir et blanc lors de l'impression
Voir livres (Demander à Maria, Google Scholar) et publications scientifiques pour bibliographie

Travail à effectuer:
Jolie cosmétique, exemples à chercher
Listes déroulantes, multi-choice (SERVICE), pour mots-clé aussi, ou ;
Le choix du type doit être optionnel
Maximum number of results, peut être vide
Demander confirmation à Maria pour la prochaine séance avant de passer le rendez-vous à tout le monde
Préparer des questions pour Maria à propos de l'interface
RENDEZ-VOUS POUR LE 23 JUIN A 10H00 DANS LA SALLE HABITUELLE, modifié au 26 juin à 13h30
Séance de la semaine prochaine avec Zhan seulement, confirmer le lundi pour le mercredi si besoin
Ecrire des PV plus formel

Procès-verbal du 26.06.2014

Discussions:
Modifier la position du bouton "+"
Créer un champ vide plutôt que 0 pour le maximum de résultats possibles
Placer un "s" à la fin du label "Number of result"
Conserver l'affichage des résultats sous forme de liste
Imaginer un Use Case de toursime en Valais, créer des données ou changer de portée (Suisse, monde, autres)
Voir les données open data de Crans-Montana
Possibilité d'incruster un graphique final sur une page web
Voir les données Linked Data de l'OFS

Travail à effectuer:
Expliquer en quoi l'interface du Travail de Bachelor est innovante
Finir le prototype administrateur
S'attaquer à la partie utilisateur final
Organiser la prochaine séance
Pour être professionnel: pas d'oubli, plus officiel (pas de notepad), pas de mélange, comme en entreprise, être plus formel
Envoyer le fichier word contenant les questions à Maria

Procès-verbal du 01.07.2014

Discussions:
L'idée est de créer une sorte de CMS pour les visualisations
ajouter outils
Il faudrait mettre l'accent sur les scénarios (scénarios = template)
Le développeur doit pouvoir éditer dynamiquement sa page
Il faut proposer des templates complexes (plus qu'un seul graphique)
Eventuellement tester un CMS pour voir comment ça fonctionne
Utiliser JSON-LD pour développement futur
Natel d'Anne Le Calvé: 078 800 88 47, envoyé des sms
Les 2 prochains samedi disponibles, Anne a les vacances à partir du 17.07.2014 jusqu'au 14.08.2014

Travail à effectuer:
Enlever explications évidentes dans le rapport
Référence rapidement, dans le sous-titre éventuellement
Mettre un exemple pour chaque protocole, approfondir les explications
Justifier l'utilisation de json-ld, pourquoi celui-là alors que d'autres existes?
Expliquer comment sérialiser des données RDF => json-ld, pourquoi?
L'introduction aux visualisations est trop centrée sur le projet, parler de manière plus général
Introduire les formats de données dans la partie introduction aux visualisations
Utiliser des numéros pour la hiérarchisation
Si je fais une critique, ça veut dire qu'elle ne sera pas dans le projet, ou sinon expliquer pourquoi on le fait pas
Tableau avec des points pour conclusion, faire un résumé plus visuel
Fixer prochain meeting: lundi 7 - 14h, Anne et Zhan
Parler de http://www.zingchart.com/

Procès-verbal du 07.07.2014

Discussions:
La défense orale aura lieu le 2 septembre à 11h. Elle durera 1heure (30 minutes de présentation, 15 min. questions, 15 min. délibérations)
Pour des conseils quant à la présentation, demander à Zhan
Spécifier dans le rapport que toutes les librairies étudiées ne sont pas exposées ("parmi les librairires…")

Travail à effectuer:
Dans la table, le croisement ligne-colonne doit rester sélectionné
Penser à fournir de la documentation technique si une équipe reprend mon projet
Changer le bouton "select data" en "upload", c'est ambigu autrement
Lorsque des données ont été importées dans le projet, afficher un message pour vérifier que des données existent à présent
Repenser le menu contextuel, il n'est pas intuitif
Créer une fonction "Tout sélectionner" et "tout désélectionner"
Ne pas se servir des lignes et des colonnes, mais l'intersection des lignes et des colonnes sur le tableau
Donner rendez-vous à Anne et Zhan pour le mercredi 9 juillet à 16h00
Rajouter la possibilité de créer des graphiques dans le projet
Prévoir des tests chez des utilisateurs externes pour avoir un feedback que je pourrai rajouter dans le rapport
Laisser le rapport de côté pour l'instant, se concentrer sur le prototype d'ici mercredi

Procès-verbal du 09.07.2014

Discussions:
Insister sur l'aspect sémantique dans la démonstration, les données sont prises depuis les endpoints donc impossibles à obtenir autrement
Chercher si il existe des widgets google chart utilisables
Faire le plus facile en premier, les choses difficiles en dernier
Envoyer meeting lundi ET mercredi à 11h
Dans la défense orale, faire une petite introduction sur le web sémantique de 4 minutes
Expliquer comment installer le programme (sur un serveur) dans le rapport
Pour faire des références, utiliser l'annotation [26]
Ajouter une section sur les acronymes et une section sur le glossaire (avec les liens de référence)
Rechercher des publications intéressantes dans google scholar

Travail à effectuer:
Changer le message d'erreur pour les graphiques
Créer une sélection plus efficace, utiliser shift, par colonnes, par lignes, sélection des croisements, tout sélectionner/désélectionner
Enlever les ? dans le texte
Améliorer le design du tableau
Créer une image de main pour déplacer les graphiques (plus intuitif)
Rajouter 5-6 styles de graphique en tout cas
Proposer 1 ou 2 templates complexes
Prévoir un autre scénario, un autre jeu de données (réel si possible)
Faire en sorte de choisir plusieurs endpoints pour l'administrateur
Système de sélection pour l'administrateur
Ajouter la possibilité de publier un template (au pire publier une image)
Ecrire une documentation technique pour les membres du projet OverLOD, expliquer la structure des dossiers, des fichiers, la structure du code

Procès-verbal du 14.07.2014

Discussions:

Travail à effectuer:
Déplacer le bouton d'exportation des données json sur la partie administrateur
Ajouter la possibilité d'ajouter des endpoints
Changer le 0 en infinite pour le maximum number of results
Afficher un message veuillez sélectionner un endpoint
Ajouter un template complexe, slider, plus compliqué, combinaison de librairies, facet
Coder le template complexe en dur et ajouter le dynamisme après
Ajouter des visualisations (3/5)
Ajouter une page de navigation entre les 2 pages (ajouter du style)

Procès-verbal du 16.07.2014

Discussions:
Il FAUT s'entrainer à la présentation
Parler de la finalité du produit dans la présentation, peu de la partie technique
Comprendre le problème, comprenre la solution apportée, être à l'aise avec la technique, prendre du recul par rapport à son travail et critiquer
Prévoir des scénarios concrets, raconter une histoire AVEC LE LINKED DATA, INFORMATIONS INEDITES!
Prévoir un endpoint RDF pour la défense orale, démontrer le prototype administateur
Répondre seul à mes questions concernant le rapport, demander à Anne ou à Zhan dans le pire des cas
Rendre la structure des dossiers plus claires (program, library, test, images, ...)

Travail à effectuer:
Rajouter le mot timeline dans le template complexe
Rendre le template complexe le plus dynamique possible, choisir tous les paramètres si possible
Si temps suffisant, faire une autre timeline, par années, par couleurs, quelques facettes seraient bienvenues
Créer un deuxième template avec facettes ou utiliser le même
Créer un système de publication de résultats
Enlever le message d'erreur rouge quand c'est ok
Finir le système de sauvegarde, de récupération
Ajouter le choix des propriétés des charts
Ecrire le manuel de l'utilisateur dans le rapport
Donner dernière version à Zhan le 28 juillet 2014, envoyer un mail pour prendre rendez-vous

Journal de travail

Date	Durée (minutes)	Tâche	Remarques
mercredi, avril 16, 2014	50	Réunion avec le team OverLOD	
Vacances	120	Chercher des renseigements sur le web à propos des termes utilisés	
Vacances	120	Lire les documents fournis par Fabian	
jeudi, mai 01, 2014	50	Réunion Kick-off avec Anne et Fabian	
vendredi, mai 02, 2014	120	Création des documents de travail (plan, journal, ...)	
vendredi, mai 02, 2014	60	Commencer à me renseigner sur RDF	
lundi, mai 05, 2014	40	Meeting	
lundi, mai 05, 2014	80	Lire les cours donnés par Anne	
mardi, mai 06, 2014	60	Premier jet du cahier des charges	
mardi, mai 06, 2014	200	Se renseigner sur le web sémantique, microformat, microdata, LNP, cambridgesemantics, …	
vendredi, mai 09, 2014	280	Tuto SPARQL cambridgesemantics	
lundi, mai 12, 2014	45	Meeting	
lundi, mai 12, 2014	60	Lire les tutoriels cambridgesemantics (patterns)	
lundi, mai 12, 2014	40	Se renseigner sur JSON-LD	
lundi, mai 12, 2014	60	Continuer le cahier des charges	
lundi, mai 12, 2014	30	Créer les procès-verbaux	
mardi, mai 13, 2014	150	Recherche visualisation, libraries javascript, client sparql, semantic search engines	Console SPARQL peut-etre existe déjà
mardi, mai 13, 2014	105	Se rappeler le JavaScript + essayer de lancer une requete SPARQL	Fonctionne pas, problème d'origine
vendredi, mai 16, 2014	30	Se renseigner sur JSON-LD	
vendredi, mai 16, 2014	60	Faire des recherches sur les visualisations, quels sont les étapes pour créer une visualisation?	
vendredi, mai 16, 2014	240	Découverte de gsvizler, création d'un prototype pour tester	
lundi, mai 19, 2014	150	Découverte de gsvizler, création d'un prototype pour tester	

lundi, mai 19, 2014	55	meeting	
lundi, mai 19, 2014	120	Recherche sur les graphiques, combien de colonnes/lignes pour quel graphique	
mardi, mai 20, 2014	30	Correction du cahier des charges	
mardi, mai 20, 2014	180	Comparatif de librairies	
mardi, mai 20, 2014	60	Recherche sur les GUI, conseils, critères	
jeudi, mai 22, 2014	100	Test de la librairie Processing JS pour faire des animations	
jeudi, mai 22, 2014	120	Test de la librairie RDF Store JS pour effectuer des requêtes SPARQL	Fonctionne pas, problème d'origine
vendredi, mai 23, 2014	660	Make Open Data	
samedi, mai 24, 2014	420	Make Open Data	
lundi, mai 26, 2014	50	Corriger le document de comparaison de librairie d'après l'avis de Zhan	
lundi, mai 26, 2014	120	Essayer la librairie de Google Charts	
lundi, mai 26, 2014	30	Essayer de faire fonctionner rdf store	Fonctionne pas, problème d'origine, j'ai demandé conseil à David Russo
lundi, mai 26, 2014	80	Comprendre comment sgvizler fait pour exécuter une requête SPARQL	
lundi, mai 26, 2014	80	Essayer d'éxecuter une requête SPARQL avec sgvizler et voir le résultat (Form)	
mardi, mai 27, 2014	60	Comprendre comment sgvizler fait pour exécuter une requête SPARQL	Inutile, autant utiliser sgvizler dans ce cas
mardi, mai 27, 2014	170	Créer un mockup, liste des fonctionnalités, templates de SPARQL	
mercredi, mai 28, 2014	150	Etudier comment CORS fonctionne	Voire résultat dans le résumé
vendredi, mai 30, 2014	15	Chercher un logiciel de web design	Google Web Designer semble bien pour commencer
lundi, juin 02, 2014	75	Meeting	
lundi, juin 02, 2014	60	Tester des outils de web design	
lundi, juin 02, 2014	30	Créer la première version de la page de recherche	
lundi, juin 02, 2014	30	Recherche d'exemples pour le moteur de recherche de l'administateur	Voir résumé Outils exemple

mardi, juin 03, 2014	120	Créer les fonctions javascript de la page de recherche	
mardi, juin 03, 2014	40	Etudier gfacet pour la présentation des résultats	
vendredi, juin 06, 2014	280	Créer l'affichage des résultats de la requête SPARQL	dbpedia plus disponible, maintenance des serveurs
mardi, juin 10, 2014	10	Contacter Maria Sokhn pour discuter de mon travail	Elle est très occupée pour les 2 semaines à venir
mardi, juin 10, 2014	100	Avancer le prototype d'interface administrateur	Prototype convaincant, avis personnel, voir résumé
mardi, juin 10, 2014	60	Commencer le rapport final	
mardi, juin 10, 2014	15	Regarder la librairie Flare	
mercredi, juin 11, 2014	60	Réfléchir à la structure du rapport final	
mercredi, juin 11, 2014	40	Tester le prototype interface administrateur	
jeudi, juin 12, 2014	60	Meeting	
jeudi, juin 12, 2014	60	Créer PV formel, contacter Maria, créer rendez-vous, observer les données à Stéphane Levet	
jeudi, juin 12, 2014	120	Rapport final, structure, introduction au web sémantique, RDF	
vendredi, juin 13, 2014	225	Rapport final, ontologies, OWL, SPARQL, JSON-LD, Interface de visualisation	
vendredi, juin 13, 2014	15	Ecrire des questions pour Maria au sujet des interfaces de visualisation	
lundi, juin 16, 2014	300	Améliorer le prototype interface administrateur (listes déroulantes, maximum number of results, éditer requête)	
lundi, juin 23, 2014	300	Enregistrer les résultats de la requête SPARQL (avec filtre en couleur) au format JSON (pas -LD) en local	
mardi, juin 24, 2014	120	Réfléchir depuis la visualisation finale jusqu'à la requête SPARQL (comment récupérer toutes les propriétés?)	Meeting avec Zhan le mercredi 25.06.2014
mardi, juin 24, 2014	150	Avancer le rapport final, analyse de l'existant	
mercredi, juin 25, 2014	10	Réparer prototype après maintenance DbPedia (variable zf -> tf)	

mercredi, juin 25, 2014	60	Meeting avec Zhan, comment récupérer toutes les propriétés, résolu	
mercredi, juin 25, 2014	135	Avancer le rapport final, analyse de l'existant	
mercredi, juin 25, 2014	15	Corriger les lignes qui dépassent sur l'interface	
jeudi, juin 26, 2014	180	Aller chercher toutes les propriétés de chaque ressource et les afficher	
jeudi, juin 26, 2014	120	Meeting	
jeudi, juin 26, 2014	20	Administratif	
jeudi, juin 26, 2014	90	Lire et écrire le chapitre sur la méthodologie de recherche	
vendredi, juin 27, 2014	90	Création d'un mockup pour l'interface utilisateur	
vendredi, juin 27, 2014	110	Recherche sur la façon de représenter les données du web sémantique	Un outil en ligne (http://www.zingchart.com/builder/) m'a fait remettre en question l'utilité du second prototype
vendredi, juin 27, 2014	180	Continuer le rapport final, analyse de librairies	
dimanche, juin 29, 2014	60	Rechercher un Use Case pour le prototype utilisateur, voir feuille de brouillon	
lundi, juin 30, 2014	240	Commencer un prototype utilisateur, voir feuille	
lundi, juin 30, 2014	60	Chercher un serveur web pour développer le prototype utilisateur, en cours de déployement	
lundi, juin 30, 2014	60	Relire les remarques d'Anne sur le rapport final et les corriger	
mardi, juillet 01, 2014	120	Créer mockup, avancer rapport final, analyse de librairies	
mardi, juillet 01, 2014	80	Meeting	
mardi, juillet 01, 2014	240	Créer prototype utilisateur sur la base du mockup	
mercredi, juillet 02, 2014	90	Optimiser le code du prototype v1	
mercredi, juillet 02, 2014	180	Tester comment déplacer des containers, les redimensionner et récupérer leurs positions	
mercredi, juillet 02, 2014	80	Configurer un serveur Tomcat pour vérifier les accès en local, créer un lien entre 2 répertoires (update auto)	CA FONCTIONNE!

mercredi, juillet 02, 2014	40	Résoudre conflits, travailler la mise en page du rapport (polices, hiérarchies, notes de bas de page, ...)	
jeudi, juillet 03, 2014	300	Sauver et charger des templates	
jeudi, juillet 03, 2014	90	Continuer le rapport final, analyse de librairies	
vendredi, juillet 04, 2014	300	Sélection des données, menu création de conteneurs, prototype google map, nouveau drag and drop, …	
samedi, juillet 05, 2014	90	Continuer le rapport final, analyse de librairies	
dimanche, juillet 06, 2014	70	Continuer le rapport final, analyse de librairies	
lundi, juillet 07, 2014	200	Ajouter position xy quand on édite un conteneur	
lundi, juillet 07, 2014	60	Continuer le raport final, corriger les remarques	
lundi, juillet 07, 2014	60	Meeting	
lundi, juillet 07, 2014	120	Ajouter des graphiques, améliorer la sélection des données	
mardi, juillet 08, 2014	120	Améliorer la sélection des données	
mardi, juillet 08, 2014	120	Afficher des google charts	
mardi, juillet 08, 2014	180	Améliorer le "user friendly"	
mardi, juillet 08, 2014	90	Améliorer le "user friendly"	
mercredi, juillet 09, 2014	420	Améliorer sélection, graphiques, …	
mercredi, juillet 09, 2014	80	Meeting	
mercredi, juillet 09, 2014	240	Moduler le code, nettoyer le code, améliorer tableau json	
jeudi, juillet 10, 2014	540	Améliorer les points discutés la veille	
vendredi, juillet 11, 2014	120	Améliorer les points discutés	
vendredi, juillet 11, 2014	300	Améliorer les points discutés	
samedi, juillet 12, 2014	60	Améliorer les points discutés	
samedi, juillet 12, 2014	60	Chercher un nouveau scénario	
dimanche, juillet 13, 2014	480	Importer des images, sauver et récupérer des visualisations	
lundi, juillet 14, 2014	40	Meeting	
lundi, juillet 14, 2014	600	Améliorer les points discutés	
mardi, juillet 15, 2014	540	Améliorer les points discutés	

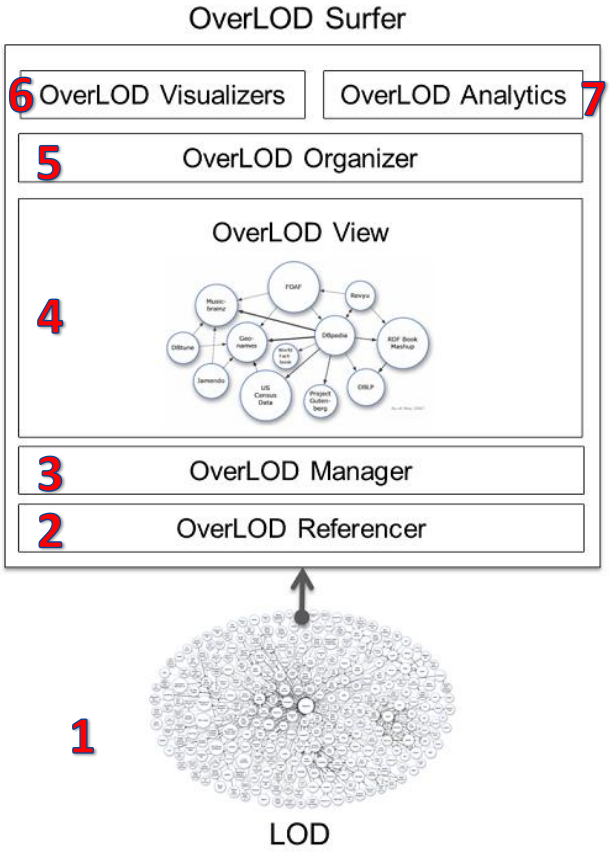
mercredi, juillet 16, 2014	90	Meeting	
mercredi, juillet 16, 2014	480	Améliorer les points discutés	
jeudi, juillet 17, 2014	660	Améliorer les points discutés	
vendredi, juillet 18, 2014	360	Rédaction du rapport final	
samedi, juillet 19, 2014	240	Rédaction du rapport final	
dimanche, juillet 20, 2014	120	Rédaction du rapport final	
dimanche, juillet 20, 2014	120	Tests auprès d'utilisateurs	
lundi, juillet 21, 2014	480	Rédaction du rapport final	
mardi, juillet 22, 2014	480	Rédaction du rapport final	
mercredi, juillet 23, 2014	540	Rédaction du rapport final	
jeudi, juillet 24, 2014	480	Rédaction du rapport final	
vendredi, juillet 25, 2014	480	Rédaction du rapport final	
samedi, juillet 26, 2014	360	Rédaction du rapport final	
dimanche, juillet 27, 2014	300	Rédaction du rapport final	
lundi, juillet 28, 2014		Rendu	
Nbre d'heures:	**327.5**		

OverLOD Surfer - Réflexion technique

But

Résumé de la dépose:

> L'outil OverLOD Surfer développera une architecture de composants logiciels et d'interfaces Web qui permettent à un acteur d'exploiter au mieux ses propres données et celles du LOD, en offrant d'une part une couche d'abstraction sur la technologie sous-jacente, complexe et méconnue, d'autre part la possibilité d'optimiser les données pour les acteurs d'un domaine plus spécifique.



The OverLOD platform facilitate and controls the consumption of existing structured data (mainly RDF but might also be Microdata or Microformats). One instance of OverLOD is setup for a specific use case. For example the Landkarte project for Softcom which will be used to illustrate the functionalities. In 1 different data publishers make their data available as RDF (this is not part of the OverLOD project). In 3, the platform's administrator configures the OverLOD Referencer (2) to define the data sources (1) that will be referenced/included in the platform. He will thus choose which data are part of the platform, also defining ways to validate the content if necessary (in 3), and make this content available as a single source of data (4) in a standard format well known by developers (non-RDF) for the applications of the platform (6-7).

Data manipulated from 2 to 7 are read-only. Only the data publishers (1) can change the data and eventually tell the OverLOD Referencer that a new update is available.

People handling 1 to 4 should be aware of what RDF is. But people handling 5 to 7 should not need to know anything about RDF, just consume regular data.

Softcom Landkarte example

In this project, an OverLOD instance would be very useful.

Here is a description of what has been done (a 15 days project for us).

Different data publishers make data available in RDF (in that project, our job was to choose the schemas – ontologies – and then help them to create RDF Data):

- The IT companies create different RDF files to describe their company, and then the Sofware solutions they propose for eGovernment actors (communes, etc.).
  TI was providing CSV file that we did transform to .ttl. iWeb did produce .ttl files by themselves. This publishing of RDF data was not automatic, but a one shot for the project.
- Some eCH standards are made available as RDF, for instance eCH 0070 which is published as a proof of concept. The ontology on http://logd.ch/voc/service, and on service, for instance on http://logd.ch/eCH-0070/id/10. There is so far no way to see a list of all services.
- The communes/districts/cantons are taken from data.admin.ch. We thus didn't need to re-create the ontology or the data, and also used that to show how linked data work: it is now possible to link the Landkarte data with statistic data (data.admin.ch) which are themselves linked to DBPedia.

Linked Data:

- The IT companies link their software solutions with eCH-0070, stating that this solution fulfills the needs of those eCH-0070.
- The communes (which URIs came from data.admin.ch) that use a software solution state that they use a software solution to fulfill a eCH-0070.

Softcom's repository

An instance of OWLIM was setup by Softcom, and the different files where simply loaded once and for all in that repository.

All data was thus made available in RDF files (mostly .ttl files) and sent to Softcom by emails. Softcom did load the files in their OWLIM. Those files thus include: the different IT companies files, the eCH-0070 .rdf file, and data from data.admin.ch that was extracted from the SPARQL end-point to .ttl files.

Once all data was loaded, SPARQL queries were issued to ensure that the data was coherent. This was done by us on our own repository, before sending the files to SoftCom by email. For instance how to be sure that a software's RDF file that has a link to a eCH-0070 does really link to a valid eCH-0070 ? -> a SPARQL query on the triple store that does include both, the software's file and the eCH-0070, is able to check that the object of the triple (give example) is a (give example)

With OverLOD, this data management would be more dynamic. The platform is configured to load data dynamically from the different sources:

- Rdf files from the IT companies are made available on the companies' servers. OverLOD upload those files

- eCH-0070 is already available from logd.ch, OverLOD upload those data
- data.admin.ch provides a SPARQL end-point, OverLOD is able to query that end-point.

For performance reason, we think so far that the original data should be copied as read-only inside the OverLOD platform (OWLIM repository). A mechanism should be setup in order to update the copy whenever original data changes.

Then, about the GUI, the landkarte is displaying data from the triple store, issuing SPARQL queries. With OverLOD, the SPARQL queries could be hidden to the data consumer. A data consumer could ask for specific data, OverLOD runs the SPARQL queries that were designed first, and return comprehensive data for the client.

More details

OverLOD users

Administrators

they can configure OverLOD and have to understand RDF. They configure the Referencer using the Manager, and prepare the Views for the developpers.

Developpers

they use data from OverLOD without knowing about RDF. They ask data from the Views to create Visualizers or Analytics apps.

Data sources - LOD (1)

[not created by OverLOD]

The platform access structured data as SPARQL end-points, RDF files (.rdf, .ttl, etc.), RDFa embedded in HTML pages, and maybe Microformat and Microdata.

We will first focus on RDF.

To be noticed that Microdata, through their promotion by schema.org and all the big search engines, will become a main source of structured data on the web. Schema.org does promote both, Microdata which was created as a simplified format because RDF was too complicated, but also RDF in the form of RDFa 1.1 Lite, which was designed by the W3C to find an agreement with schema.org and make RDF part of the game. The schema.org schema is thus also available as an OWL ontology.

If needed, see Transformation from HTML+Microdata to RDF:
http://www.w3.org/TR/microdata-rdf/

OverLOD Referencer (2)

[this is an API, the GUI being OverLOD Manager]

This part is configured by the OverLOD Manager (3), to consume data from the data sources (1)

For performance reason, we think so far that the original data should be copied as read-only inside the OverLOD platform (OWLIM repository). It is also the common current approach on the web

where data is often published in a distributed way, but consumed in a centralized way. That's also what Google does: to answer user queries very effectively, the all web is indexed and cached in Google's servers.

Do we need an option for each source to choose if a local-copy should be generated or not ?

With a local copy, a mechanism should be setup so that when original data changes, the copy is updated as soon as possible. The solution we think about sofar could be either:

- Pull: the referencer regularly check if data has been updated. Then how to detect the changes on a SPARQL end-point, RDFa in HTML, or on a RDF file -> one solution could be that the data sources have to implement a VoID description of their data, which does contain the date of the last update (`dcterms:modified`).
- Push: the data sources notify the reference when data changes, which is the solution used by Sindice (and its ping api) and PTSW (data publishers do 'ping' the server to tell that the data has changed and should be re-indexed, both of them also explained here). Or it might be a kind of RSS architecture ?
  The advantage here is that the reference knows right away if data changes and can ensure that data is always up-to-date.
  The disadvantage being that a data source will have to notify all the data consumers (the overLOD instance being one of them). But isn't it how RSS works ?

Handling a local copy of the data will allow two more important functionalities:

- data validation: check that the data coming from the sources is correct
- data 'chunks': it might be that we want only part of the data. If data comes from a SPARQL end-point, we can adapt the SPARQL CONSTRUCT query to take only the needed data. But if the data comes from a file, we should be able to load only part of it. For instance, in all the projects where we use data from geoNames, we did load the about.rdf about all concerned locations. But only a little amount of the triples from those about.rdf were really useful.

We will need here to handle both: T-Box and A-Box. Can we import only 'part' of a T-Box ? Might be interesting as well.

OverLOD Manager (3)

[this is a GUI that allows configuring the OverLOD Referencer, and maybe also the OverLOD Views]

The administrator uses the Manager, and he knows about RDF and ontologies.

Choosing data sources

The administrator looks for data sources by his own means, i.e. the Manager is not used for data 'discovery'.

Then the GUI allows him to add the data sources URL, as a SPARQL end-point, an HTML page with RDFa, RDF files, with maybe (to be decided) possibilities to visualize the content of those sources.

This first selection ensures that the data available in the platform is valid data and not any data found on the web.

Selecting more precisely the content of the data sources

The admin might create SPARQL queries to select the content of the source to be imported. For SPARQL end-point, it will be a SPARQL construct that returns the results. For an RDF File, he could either load the all file (no SPARQL), or part of the file. To load only part of the file, the tool could load it in a temporary repository and run the SPARQL CONSTRUCT to extract part of the content – otherwise see if there are APIs to run SPARQL over a file's content.

Validating the content

To ensure that data is valid, an OWL reasoner could be partly used. But because of the OWA, it might not be the best choice. SPARQL, on the other hand, allows for querying in a CWA and detect simple inconsistencies, as if a triple refers to an URI which doesn't exist in the store, etc.

A series of SPARQL queries could be defined for each data source or for a group of data sources. Once a file has to be uploaded, the SPARQL queries are run. In case of mistakes, a report is given to the admin, otherwise the file is automatically updated.

OverLOD View (4)

[this not a GUI but only data representation, the GUI for the admin to configure the Views is OverLOD Manager, and the GUI for the developers to consume the views is OverLOD Organizer]

The idea here is to have representation of data which is not RDF. But it is not clear yet what this could be.

How to put the RDF graph at disposal of the platform developers, without needing them to understand RDF or SPARQL ?

As proposed in the RCSO project, those views don't add any burden to the architecture's performance. They should be handled as DB views, enabling easier manipulation of the original data without requiring any data transformation.

We need to find a way which allows the developers to query any data they want without having to write SPARQL queries, and also without imposing them hard-coded web services where they can only query what the admin has decided they can query (give me the list of persons, give me the list of locations, etc.).

Now that JSON-DL has become a standard way that allows for enriching the very common JSON data representation with Linked Data, JSON-DL could play the role of 'views' and JSON-LD data could be returned to the developers ?

It is not clear yet what is the role of the admin to prepare the Views (using the Manager), and what is the role of the developers to consume the Views (using the Organizer).

Data/Ontology reconciliation (and ontology alignement)

This feature might not be implemented in a first version. But the goal would be to allow for a single view of data that are described using different ontologies/properties.

We could imagine that people are described using FOAF, but also using the SEMIC's core vocabulary for persons.

In the platform, there should be a way to deal with 'people' data, no matter on which ontology the data is based. This functionality has to know that to get the name of people, a certain property is used with FOAF and another property from another ontology.

OverLOD Organizer (5)

[this is a GUI for developers to interact with the views in order to create apps (visualization, analytics)]

For instance, the developers could access the classes/properties of the graph (without knowing they are owl:Class or owl:ObjectProperties/DatatypeProperties). They would say they want to retrieve locations with altitude/longitude, and the SPARQL queries would be generated behind the scene.

This could be the understanding of 'views'.

What should be the possibilities to search the data ? what about limiting the data (label that starts with "f", cities in this country, age > 40 ? also what about paging through the results ?

OverLOD Apps (visualisers (6), analytics(7))

Consume data to create different visualizations, for different media (web, tablet, smartphones)

The apps might consume JSON-LD data returned by the server.

Use Case de validation

La dépose parle du projet OVT, en soulignant que la publication des données OVT en RDF ne fait pas partie de ce RSCO.

Un cas d'application très concret est le projet "Landkarte/Daten Dreh Scheibe" réalisé pour Softcom en janvier (que j'ai finalement décrit en exemple ci-dessus). Maria les a relancés en mars mais nous n'avons pas de nouvelles à ce jour.

Premières idées techniques, à valider

Architecture

Réflexion générale

L'architecture doit être très souple, de la couche de données aux interfaces utilisateurs.

Contrairement à une architecture traditionnelle avec un schéma de données bien prédéfini, des DataObjects et BusinessObject prédéfinis, et des interfaces prédéfinies -> ici on aimerait exploiter la souplesse de RDF. Il doit être très facile de gérer de nouvelles données (par exemple une nouvelle ontologie, ou de nouvelles classes) ou de nouvelles propriétés des données (on avait la longitude/latitudes des lieux, maintenant on ajoute l'altitude, etc.).

L'idée exprimée au sujet des Linked Data Fragments ((Verborgh et al. 2014) référencé dans OverLOD article.docx), de passer une partie du traitement sur le client, me semble à exploiter. Donc le client demande des données, et c'est le client qui en fait qqch (contrairement aux solutions PHP où tout est fait côté serveur avant d'envoyer la page finalisée au client). Par rapport au code de data.admin.ch: eux ont carrément les requêtes SPARQL dans le client, et ils interprètent le JSON retourné. OverLOD peut s'inspirer de tout cela, sans reproduire exactement la même chose car le contexte est différent:

dans l'idéal le développeur du client n'a pas à connaître RDF/SPARQL, il connait par contre ce qui a été nommé les 'Views'. Il demande des infos au serveur, le serveur en génère le SPARQL nécessaire et retourne du JSON ou JSON-LD, et le client affiche cela. Si l'on part du principe que le développeur ne connait pas RDF, est-ce pertinent de lui retourner du JSON-LD ? ou alors on 'ouvre' la porte aux développeurs qui connaissent RDF mais cela n'est pas imposé ?

Avant LDFragment, il était question de Hydra (je dois parler de (Lanthaler & Gütl 2013)):

- basé sur JSON-LD + hydra ontology
- c'est une communauté qui inclut des gens de Google
- Hydra utilisé par LDFragment

Des infos sur Hydra: http://www.markus-lanthaler.com/hydra/

C'est Adrian Gschwend qui m'a parlé de LDFragment et Hydra, il me semble judicieux de tenir compte de son point de vue. Actuellement je me demande comment cette description des web services et des Web API correspond à la notion de 'views' d'OverLOD

Interface Web "sympa"

L'interface Web doit être à la pointe de ce qui se fait maintenant: HTML 5 ?
(Alain Duc, spécialisé dans les Rich User Interfaces m'a dit que les outils utilisés il y a quelques années sont en perte de vitesse et maintenant tout le monde passe à HTML 5)

L'interface de Sig.ma est très sympa et dynamique (peut-être un peu lourde car beaucoup d'options).

Les interfaces web sont donc les interfaces de configuration (Manager, Organizer), mais aussi des apps d'exemple.

Projets inspirants

- http://sig.ma
  La vidéo est intéressante pour montrer ce que le Linked Data apporte en matière de recherche de données en exploitant les liens entre les données, différentes sources, etc.
  Par contre OverLOD est un peu différent. Il ne fait pas de la découverte de données: c'est l'admin qui spécifie quelles sources référencer, comment les valider, etc.
  sig.ma étant open source, est-ce que cela ferait du sens de l'utiliser ? Est-ce que cela ne nous prendrait pas plus de temps d'installer/comprendre et ensuite développer sur sig.ma au lieu de créer notre propre solution ?
  D'autre part c'est open source pour la recherche, mais payant si commercial ? voir cette page http://sig.ma/?page=sigmaee
- Sindice/SIREn
  Sigma est en partie basé sur Sindice. Je ne pense pas que Sindice en soit, qui est une instance qui index tout le LOD, et qui n'est pas 'libre', soit intéressant pour notre projet. Par contre est-ce que au lieu d'avoir un triple store, on devrait se baser sur Siren ?
  http://rdelbru.github.io/SIREn/
  Une info sur la différence entre SIREn et Lucene ou ElasticSearch:
  https://groups.google.com/forum/#!topic/siren-user/wIBpM2HO6Ns

- http://linkeddatafragments.org/
  comme écrit ci-dessus, j'en ai parlé dans OverLOD article.docx
- W3C Linked Data platform
  Le descriptif d'un Linked Data platform contient toutes les opérations read/write nécessaires pour gérer les données. Dans le cadre d'OverLOD, cela me semble un peu différent vu que l'on ne fait que de la consommation de données. Mais à voir.
  Pour info Virtuoso se dit déjà conforme LDP:
  http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VirtLDP
- http://en.lodlive.it/ (juste trouvé, à voir)
- Voir http://redlink.co/apache-marmotta/, qui implémente le LDP, et qui pourrait servir de back-end ? Le triple store est KiWi, et intéressant car cela implémente du versionning . KiWi est un back-end pour Sesame!

Outils commerciaux, donc juste pour information:

- Commercial linked data platform: Callimachus http://3roundstones.com/index/index.html
  Avec une video d'intro intéressante, notamment pour recevoir des résultats utilisables dans Google Chart (?): http://callimachusproject.org/videos/0.17/epa-success-story.xhtml?view
- openAnzo:
  http://www.openanzo.org/projects/openanzo/wiki
  qui était openSource à la base, mais devenu commercial comme expliqué ici.

Sindice

(Sindice semble être en bas au printemps 2014, sera p-ê racheté par Yahoo ? info de Adrian)

Sauf erreur Sindice n'est pas un triple store. Cela index les documents RDF à l'aide d'un plugin de Lucene: Sirene.

Pour interroger Sindice, on ne part pas de SPARQL, mais de mots-clés ou d'URI
http://www.sindice.com/developers/queryLanguage

Donc il y a déjà un niveau d'abstraction intéressant

Le forum: https://groups.google.com/forum/#!forum/sindice-dev

L'api de recherche v.3: http://www.sindice.com/developers/searchapiv3

On appelle la requête REST et reçoit les résultats JSON.

Comment aussi utiliser des données pas forcément sur le LOD ? -> on peut avoir aussi un SPARQL end-point interne.

Sig.ma

De http://sig.ma/?page=sigmaee

What is Sig.ma EE?

Sig.ma EE is a standalone, deployable, customisable version of Sig.ma. Sig.ma EE is deployed as a web application and will perform on the fly data integration from both local data source and remote services (including Sindice.com). Sig.ma EE currently supports the following data providers:

- SPARQL endpoints (Virtuoso, 4store)
- YBoss + the Web (Uses YBoss to search then Any23 to get the data)
- Sindice

It is very easy to customise visually and to implement new custom data providers (e.g. for your relational or CMS data) by following documentation provided with Sig.ma EE.

Want to give it a quick try? Sig.ma EE now also powers the service on the http://sig.ma homepage so you can add a custom datasource (e.g. your publicly available SPARQL endpoint) directly from the "Options" menu you get after you search for something. It is very easy to implement new custom data source (e.g. for your relational or web CMS data) by implementing interfaces following the examples.

Sig.ma EE license.

Sig.ma EE is open source and available for download. The standard license under which Sig.ma EE is distributed is the GNU Affero General Public License, version 3. Sig.ma EE is also available under a commercial licence. Please contact us to discuss further.

Requêtes – aussi pour le Manager

Des classes par document: http://api.sindice.com/v3/search?&field=title&field=class&format=json

Des infos sur les datasets ? http://vocab.sindice.net/

Il faudrait pouvoir partir d'un mot-clé, et trouver des infos -> Sindice/Sig.ma

Pourrait-on donc faire des interfaces de customisation pour une instance locale de Sig.ma ?

Architecture envisagée

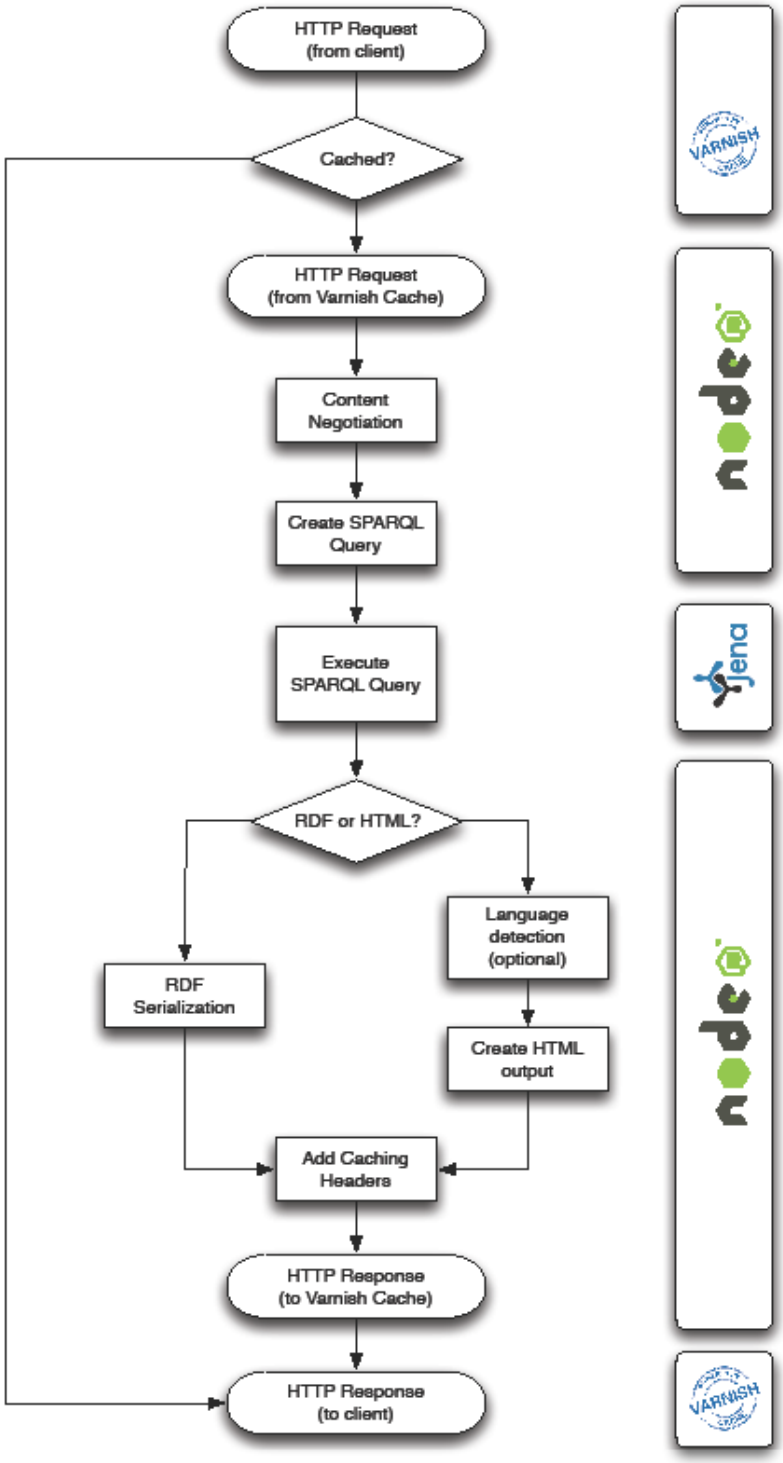
Un serveur node.js qui propose des web-services pour les clients.

Sur le même serveur, une instance de OWLIM. Le serveur node.js communique avec OWLIM soit par HTTP, soit par la librairie Sesame (java) - mais est-ce possible de l'appeler depuis node.js ?

Pour accéder aux sources RDF, soit directement avec des fonctionnalités simples HTTP, soit avec une librairie. Jena me semble plus évolué que Sesame pour faire des requêtes sur des SPARQL end-points ou autre, mais à voir. D'autre part avec Sesame j'ai eu des erreurs pour accéder à certains end-points, il faut donc que je trouve de quoi cela venait (documenté dans mon doc Musyop ou OLPC je pense).

Cette architecture permettra de gérer du cash, comme proposé par Adrian. On ajoutera alors une couche Varnish pour gérer cela. Le fait d'avoir des 'views' plutôt que directement n'importe quelle requête SPARQL qui arrive sur un end-point, permet justement de mettre en cache les résultats envoyés aux clients.

Architecture d'Adrian:



Côté client, du HTML 5 avec donc du javascript.

OverLOD Referencer + Manager

C'est ici qu'on fait un LOD pour des données spécifiques
L'admin détermine les sources en faisant donc une sélection précise.

Dépose: C'est une couche bas-niveau non visible à l'utilisateur de l'outil mais paramétrable. Il s'agit de parcourir les données du LOD en posant des requêtes précises sur des SPARQL end-points, ou en reposant sur des API tels ceux offerts par Sig.ma, Sindice (Tummarello & al., 2007), SQUIN (Hartig 2013), ou CKAN par exemple.

A ce jour je pense qu'on crée de zéro un serveur basé sur node.js, et le triple store étant OWLIM.

Data access/data chunks

Pour accéder aux différentes sources RDF, je pense donc utiliser Jena (ou Sesame si fonctionne bien), et définir des requêtes SPARQL sur le contenu (fichiers ou SPARQL end-points) pour en extraire la partie qui nous intéresse pour la copie locale.

Sur end-point, un sparql CONSTRUCT peut être exécuté. Si par contre on a un fichier, j'envisageais ci-dessus de le charger temporairement afin de pouvoir exécuter le CONSTRUCT. A voir: le projet RDFSlice semble exactement répondre à cette problématique d'extraction d'une partie des données, aussi en proposant d'extraire les données d'un fichier plutôt qu'un end-point -> ceci avec comme conséquence des limitations sur le SPARQL. J'ai référencé le papier dans OverLOD article.docx. Outils dispo ici: https://bitbucket.org/emarx/rdfslice.

RDFSlice mentionne LDSpider, un crawler open source et lightweight qui semble extraire des données de différents formats RDF (RDFa, etc.).

Il serait aussi possible de parcourir les informations liées à une ressource, comme exprimé par le LinkedData Sail (basé sur la librairie Gremlin). Dans un exemple de recommandation musicale: en partant d'une URI (un groupe DBPedia), ils ressortent les triples où il est sujet. Puis ils filtrent ces propriétés en déterminant 4-5 propriétés intéressantes, et finalement sortent les groupes liés par ces propriétés-là. Dans cette même idée, peut-être que le "Semantic Web Client Library" serait intéressant: http://wifo5-03.informatik.uni-mannheim.de/bizer/ng4j/semwebclient/

Voici d'autres idées, pas forcément intéressantes (voir description):

- SQUIN - Query the Web of Linked Data
  http://squin.sourceforge.net/index.shtml
  Ne semble plus actif, éventuellement lire l'article. Mais semble plus orienté sur le parcours de Linked Data sur le Web.
- http://pipes.deri.org/
  Pipe permet de faire en partie ce que OverLOD doit faire: définir des sources de données (fichiers, end-point), en extraire des informations par des requêtes CONSTRUCT. Les pipes sont configurés dans des fichiers XML.
  Malgré tout le projet semble dater un peu, et la documentation contient des informations "This section of the documentation requires updating" -> donc ne pas trop compter là-dessus (je l'ai aussi vu mentionné dans un forum et déconseillé car plus actif)
- Rhizomik http://rhizomik.net/html/rhizomer/
  Semble intéressant car on dirait qu'il permet d'explorer un dataset: les classes, les propriétés. Par contre l'interface est en dur ? est-ce que l'on pourrait ne reprendre que les fonctionnalités ? D'autre part il est dit: "A preliminary version of Rhizomer is available for download. Please, treat it with care because it is very preliminary ;-)"

- SPIN et SPARQLMotion ? (de TopBraid)
  mais TopBraid n'est pas gratuit pour les projets commerciaux
- et voir les autres bookmarks de FireFox

Des interfaces pour aider à l'écriture de requêtes SPARQL. Si on trouve des interfaces évoluées et libre, c'est bien. SPARQLed ne semble plus disponible (il est mentionné sous storage/querying de cette page). J'avais mis un POST ici à ce sujet, mais sans réponse. On pourrait contacter les auteurs si nécessaire. Sinon voir Flint que j'avais téléchargé, mais que je n'avais pas réussi à faire tourner sur OWLIM. Voir ma liste de bookmarks: "SPARQL and query interfaces"

Visualiser le contenu d'un fichier, choisir tout ou une partie.

Charger chaque fichier dans un named graph. Charger aussi la description des données actuelles (par exemple le VoID) afin de pouvoir l'indiquer aux consommateurs, mais aussi de gérer l'update (notamment si la solution 'pull' est choisie).

Validating the content

To ensure that data is valid, an OWL reasoner could be partly used. But because of the OWA, it might not be the best choice. SPARQL, on the other hand, allows for querying in a CWA and detect simple inconsistencies, as if a triple refers to an URI which doesn't exist in the store, etc.

Here it might not be that simple. If a property as a defined Range, but the given URI in a triple does not exist, the reasoned will infer that this URI is an instance of that class. But there is the possibility to SPARQL over the non-inferred statements…to be checked.

A series of SPARQL queries could be defined for each data source or for a group of data sources. Once a file has to be uploaded, the SPARQL queries are run. In case of mistakes, a report is given to the admin, otherwise the file is automatically updated.

It seems thus mandatory to handle those SPARQL queries for groups of data sources instead of specifically for each data source. For instance, in the case of Softcom, many IT companies will give their RDF files about company descriptions, and RDF files about products description. The admin needs to define a bunch of SPARQL queries to validate RDF files of companies descriptions (any company description) and a bunch of SPARQL queries to validate RDF files of products descriptions.

Should this functionality combine with OWL reasoning ? for instance, do we need to limit values of certain data (the age of a person is positive, etc.) ?

Pour les règles de contrôle SPARQL, voir SPIN ? ou encore an OWL reasoner

Local copy management

See pull/push envisagée ci-dessus: *OverLOD Referencer (2)*

Voir l'outil sparqlPuSH mentionné par RDFSlice.

OverLOD View

Que seraient les vues ?

On a décidé des classes qui nous intéressent, des propriétés ?

Les vues pourraient être des ensembles de requêtes ? des requêtes REST pour Sindice, SPARQL pour d'autres ?

Comment gérer la réconciliation ? à voir si implémenté ou non…

Elles ont un nom bien parlant, et on peut visualiser les données qu'elles retournent

➔ Il faut donc un visualiseur qui est le: Organizer

The idea here is to have representation of data which is not RDF. But it is not clear yet what this could be.

How to put the RDF graph at disposal of the platform users, without needing them to understand RDF or SPARQL.

We need to find a way which allows the users to query any data they want without having to write SPARQL queries, and also without imposing them hard coded web services where they can only query what the admin has decided they can query.

Now that JSON-DL has become a standard way that allows for enriching the very common JSON data representation with Linked Data, JSON-DL data returned by

OverLOD Organizer

Ici on parcourt les données spécifique à cette instance (décidée par le Manager), et on réduit encore la sélection à des données spécifiques pour une représentations ou une analyze.

On exécute les requêtes sauvées sous formes de vues -> on visualise alors les DATA un peu comme dans Sig.ma mais avec une interface plus user-friendly et qui cache la technologie sous-jacente.

Il nous faut déjà ici un Viewer de data.

Est-ce que le résultat de l'Organizer n'est pas une vue plus spécifique ? donc des filtres sur les requêtes de base ? etc ?

OverLOD Visualizers

Juste un exemple de visualisation, peut-être celles utiles à Maria

Ce qui nécessite peut-être l'API pour accéder aux vues génériques ou spécifiques.

Voir http://dev.data2000.no/sgvizler/, utilisé par Adrian. Et ma liste de bookmarks "Visualisation".

Un TB sera fait d'ici l'été 2014.

Voir aussi la présentation faite par Michael Lugen à Bern en janvier (de l'équipe d'Adrian), qui parle de sgvizler, mais aussi:

- Sgvizler – http://dev.data2000.no/sgvizler/
- rdfstore-js – https://github.com/antoniogarrote/rdfstore-js
- rdfQuery – https://code.google.com/p/rdfquery/
- D3 – http://d3js.org/

OverLOD Analyzer

Partie qui implique Riccardo (lui a donc déjà 10 jour financés ?)

Des services de statistique de Sindice:

https://groups.google.com/forum/#!topic/sindice-dev/9Xhd8QjjSCI

A Design Science Research Methodology for Information Systems Research

Ken Peffers[1,2]
University of Nevada, Las Vegas, College of Business Administration
4505 Maryland Parkway Las Vegas NV 89154-6034 USA
Tel +1-702-895-2676, Fax +1-702-446-8370 k@peffers.com

Tuure Tuunanen[2]
The University of Auckland Business School, The Dept. of ISOM
Symonds Street 7, Private Bag 92019 Auckland, New Zealand
Tel +64-9-373-7599 ext. 84622, Fax: +64-9-373-7430 tuure@tuunanen.fi

Marcus A. Rothenberger
University of Nevada, Las Vegas, College of Business Administration
4505 Maryland Parkway Las Vegas NV 89154-6034 USA
Tel +1-702 895 2890, Fax +1-702-895-0802 marcus.rothenberger@unlv.edu

Samir Chatterjee
Claremont Graduate University School of Information
Systems & Technology Claremont, CA 91711
Tel +1-909-607-4651 samir.chatterjee@cgu.edu

Published in Journal of Management Information Systems,

Volume 24 Issue 3, Winter 2007-8, pp. 45-78.

[1] Corresponding Author [2] The first two authors made substantially similar contributions to this paper. First authorship was determined by rotation among papers.

KEN PEFFERS is an Associate Professor of MIS at the University of Nevada Las Vegas. He received his Ph.D. in Management Information Systems from Purdue University. His current research focuses on making the right IS investments for the firm, on IS planning, and on requirements determination for new information systems. His research articles have been published in such journals as *Communications of the ACM*, *Journal of Management Information Systems*, *Information Systems Research*, *IEEE Transactions on Engineering Management*, *Organization Science*, *JITTA*, and *Information & Management*. Dr. Peffers is a member of the Sault Ste. Marie Tribe of Chippewa Indians of Michigan.

TUURE TUUNANEN is a Senior Lecturer in The Department of Information Systems and Operations Management at The University of Auckland. He holds a D.Sc. (Econ) from the Helsinki School of Economics. His current research interests lie in the areas of IS development methods and processes, requirements engineering, risk management and convergence of IS and marketing disciplines, specifically in design of interactive consumer services and products. His research has been published in *Information & Management*, *Journal of Database Management*, *Journal of Information Technology Theory and Application*, *Journal of Management Information Systems, and Technology Analysis and Strategic Management*. In addition, his work has appeared in a variation of conference proceedings within his research interest areas, such as AMCIS, ECIS, eCOMO, DESRIS, HICSS, IRIS, ISD, Mobility Roundtable, PACIS, RE, WeB, and WITS. Dr. Tuunanen is a member of ACM, AIS and IEEE. More up-to-date information about his research is available at http://www.tuunanen.fi.

MARCUS A. ROTHENBERGER is an Associate Professor of MIS at the University of Nevada Las Vegas. He holds a Ph.D. in Information Systems from Arizona State University. Dr. Rothenberger's work includes theory testing, theory development, and design science research in the areas of software process improvement, software reusability, performance measurement, and the adoption of Enterprise Resource Planning systems. His work has appeared in major academic journals, such as the *Decision Sciences Journal, IEEE Transactions on Software Engineering*, *Communications of the ACM*, and *Information & Management*. Dr. Rothenberger is regularly involved in major academic conferences, including the *International Conference on Information Systems* and the *Americas Conference on Information Systems*. He is a Member of the *Association for Information Systems* (AIS) and the *Decision Sciences Institute* (DSI).

SAMIR CHATTERJEE is an Associate Professor in the School of Information Science and Director of the Network Convergence Laboratory at Claremont Graduate University. Prior to that, he taught at the J Mack Robinson College of Business, Georgia State University, in Atlanta. He holds a Ph.D. from the School of Computer Science, University of Central Florida. In the past, his research work has been on ATM scheduling systems, efficient routing protocols, TCP/IP performance over HFC cable networks, QOS and all-optical networking. Currently he is exploring fundamental challenges in Voice/Video over IP, real-time protocols and secured PKI infrastructures. He is a member of ACM, IEEE, and IEEE Communications Society. He has published widely in

respected scholarly journals such as *Communication of the ACM*, *Computer Networks & ISDN Systems*, *Computer Communications*, *Communications of the AIS*, *Information System Frontiers*, *ACM Computer Communication Review* and others. He has authored more than 34-refereed ACM and IEEE conference papers. He is a core member of Internet2 Middleware working group on videoconferencing and has served as an expert on the Computer Science and Technology Board panel under National Research Council. He has received several NSF grants and funding from private corporations such as BellSouth, Northrop-Grumman, Bank of America, GCATT, Georgia Research Alliance, Hitachi Inc, Fore Systems for his research. He is the secretary of EntNet Technical Committee for IEEE Communications Society. He also co-founded a startup company VoiceCore Technologies Inc in 2000.

A Design Science Research Methodology for Information Systems Research

ABSTRACT: The paper motivates, presents, demonstrates in use, and evaluates a methodology for conducting design science (DS) research in information systems. DS is of importance in a discipline oriented to the creation of successful artifacts. Several IS researchers have pioneered DS research in IS, yet over the last 15 years little DS research has been done within the discipline. The lack of a methodology to serve as a commonly accepted framework for DS research and of a template for its presentation may have contributed to its slow adoption. The design science research methodology (DSRM) presented here incorporates principles, practices, and procedures required to carry out such research and meets three objectives: it is consistent with prior literature, it provides a nominal process model for doing DS research, and it provides a mental model for presenting and evaluating DS research in IS. The DS process includes six steps: problem identification and motivation, definition of the objectives for a solution, design and development, demonstration, evaluation, and communication. We demonstrate and evaluate the methodology by presenting four case studies in terms of the DSRM, including cases that present the design of a database to support health assessment methods, a software reuse measure, an Internet video telephony application, and an IS planning method. The designed methodology effectively satisfies the three objectives and has the potential to help aid the acceptance of DS research in the IS discipline.

KEYWORDS AND PHRASES: Design science, design science research, design theory, methodology, mental model, process model, case study.

Introduction

Information systems is an applied research discipline, in the sense that we frequently apply theory from other disciplines, such as economics, computer science, and the social sciences, to solve problems at the intersection of IT and organizations. However, the dominant research paradigms that we use to produce and publish research for our most respected research outlets largely continue to be those of traditional descriptive research borrowed from the social and natural sciences. We have recently accepted the use of interpretive research paradigms, but the resulting research output is still mostly explanatory and, it could be argued, not often applicable to the solution of problems encountered in research and practice. While design, the act of creating an explicitly applicable solution to a problem, is an accepted research paradigm in other disciplines, such as engineering, it has been employed in just a small minority of research papers published in our own best journals to produce artifacts that are applicable to research or practice.

Without a strong component that produces explicitly applicable research solutions, IS research faces the potential of losing influence over research streams for which such applicability is an important value. For example, we wonder whether the preference for theory building and testing research may help to explain why the center of gravity for research in systems analysis and design, arguably IS research's *raison d'être,* appears to have moved to engineering, dominated by such research streams as requirements engineering and software engineering. Engineering disciplines accept design as a valid and valuable research methodology because the engineering research culture places explicit value on incrementally effective applicable problem solutions. Given the explicitly applied character of IS practice and the implicitly applied character of IS research, as part of the business academe, we should do so as well.

In recent years several researchers have succeeded in bringing design research into the IS research community, successfully making the case for the validity and value of design science (DS) as an IS research paradigm [20, 31, 55] and actually integrating design as a major component of research [33]. In spite of these successful efforts to define DS as a legitimate research paradigm, DS research has been slow to diffuse into the mainstream of IS research in the past 15 years [56] and much of it has been published in engineering journals.

An accepted common framework is necessary for DS research in IS and a mental model [18, 45, 54] or template for readers and reviewers to recognize and evaluate the results of such research. Every researcher trained in the culture of social science research has mental models for empirical and theory building research that allow the researcher to recognize and evaluate such work and perhaps one for interpretive research as well. Even if all of these mental models are not exactly the same, they provide contexts in which researchers can understand and evaluate the work of others. For example, if a researcher reviewed an empirical paper that failed to describe how the data was gathered, he or she would probably always regard that as an omission that required notice and correction. Since DS research is not part of the dominant IS research culture, no such commonly understood mental model exists. Without one, it may be difficult for researchers to evaluate it or even to distinguish it from practice activities, such as consulting.

A number of researchers, both in and outside of the IS discipline, have sought to provide some guidance to define design science research [20]. Work in engineering [2, 14, 16, 38], computer science [37, 46], and IS [1, 10, 20, 31, 33, 40, 55, 56] has sought to collect and disseminate the appropriate reference literature [51], characterize its purposes, differentiate it from theory building and testing research, in particular, and from other research paradigms, explicate its essential elements, and to claim its legitimacy. However, so far this literature has not

explicitly focused on the development of a methodology for carrying out DS research and presenting it.

We propose and develop a design science research methodology (DSRM) for the production and presentation of DS research in IS. This effort contributes to IS research by providing a commonly accepted framework for successfully carrying out DS research and a mental model for its presentation. It may also help with the recognition and legitimization of DS research and its objectives, processes, and outputs and it should help researchers to present research with reference to a commonly understood framework, rather than justifying the research paradigm on an *ad hoc* basis with each new paper.

The remainder of the study is structured as follows. In the next section, we review literature to understand the state of the problem. There we also define DS research for the purpose of this methodology and recognize existing practice rules. In the following section, we identify the objectives of a solution in a methodology. Then we explicate the design of a DSRM as a research procedure. Next, we demonstrate the successful use of the DSRM by reviewing four IS research studies, in which the methodology in use was consistent with the DSRM, in order to provide a *proof of concept*, to show that the designed methodology is consistent with successfully undertaken design science research. Finally, we evaluate the process in terms of the objectives of our study, discuss research issues surrounding the methodology, and conclude.

Problem Identification: Completing a Design Science Research Methodology for IS Research

When IS researchers started to develop an interest in DS research in the early 1990s, there already was agreement in prior research about the basic difference between DS and other

paradigms, such as theory building and testing, and interpretive research: “Whereas natural sciences and social sciences try to understand reality, design science attempts to create things that serve human purposes [43, p. 55].” Three papers from the early 1990s [31, 33, 55] introduced DS research to the IS community. Nunamaker et al. [33] advocated the integration of system development into the research process, by proposing a multimethodological approach that would include 1) theory building, 2) systems development, 3) experimentation and 4) observations. Walls et al. [55] defined information systems design theory as a class of research that would stand as an equal with traditional social science based theory building and testing. March and Smith [31] pointed out that design research could contribute to the applicability of IS research by facilitating its application to better address the kinds of problems faced by IS practitioners.

Once this literature provided a conceptual and paradigmatic basis for DS research, Walls et al. [56] expected its widespread adoption within IS, believing that this would lead to IS research having more impact on practice through close ties between DS research and practical applications. Despite the precedents of these early papers, Walls et al. [56] observed that this rush to publish DS research did not occur and that the DS research paradigm had only occasionally been used explicitly in the past ten years. Given that many papers in reference disciplines, such as engineering and computer science, use design science as a research approach and, in doing so, realize benefits from the practical applicability of research outcomes, e.g., [3, 19, 27-29], it would seem reasonable that it could also happen in IS.

Toward a DS Research Methodology

Some engineering literature, e.g., [14], has pointed to a need for a common DS research methodology. Archer's [2] methodology focuses on one kind of DS research, which resulted in building system instantiations as the research outcome, or "the purposeful seeking of a solution" to a problem formulated from those desires [32]. Archer believed that design could be codified, even the creative part of it [32]. Archer's own industrial engineering research outcomes reflect his views on research methodology. His work included purpose-oriented designs for hospital beds and for mechanisms that prevented fire doors from being propped open. Through this work he defined six steps of DS research: programming (to establish project objectives), data collection and analysis, synthesis of the objectives and analysis results, development (to produce better design proposals), prototyping, and documentation (to communicate the results). With these steps he asserted that designers can approach design problems "systematically," by looking at functional level problems like goals, requirements etc., and by progressing towards more specific solutions [22].

A methodology is "a system of principles, practices, and procedures applied to a specific branch of knowledge [13]." Such a methodology might help IS researchers to produce and present high quality design science research in IS that is accepted as valuable, rigorous, and publishable in IS research outlets. For DS research, a methodology would include three elements: conceptual principles to define what is meant by design science research, practice rules, and a process for carrying out and presenting the research.

Principles: DS Research Defined

With just a decade and a half of history, DS research in IS may still be evolving; however, we now have a reasonably sound idea about what it is. "Design science…creates and evaluates IT artifacts intended to solve identified organizational problems [20, p. 77]." It involves a rigorous process to design artifacts to solve observed problems, to make research contributions, to evaluate the designs, and to communicate the results to appropriate audiences [20]. Such artifacts may include constructs, models, methods, and instantiations [20]. They might also include social innovations [52] or new properties of technical, social, and/or informational resources [24]; in short, this definition includes any designed object with an embedded solution to an understood research problem.

Practice Rules for DS Research

Hevner et al. [20] provided us with practice rules for conducting DS research in the IS discipline in the form of seven guidelines that describe characteristics of well carried out research. The most important of these is that the research must produce an "artifact created to address a problem [20]." Further, the artifact should be relevant to the solution of an "heretofore unsolved and important business problem [20]." Its "utility, quality, and efficacy [20]" must be rigorously evaluated. The research should represent a verifiable contribution and rigor must be applied in both the development of the artifact and its evaluation. The development of the artifact should be a search process that draws from existing theories and knowledge to come up with a solution to a defined problem. Finally, the research must be effectively communicated to appropriate audiences [20].

Procedures: a Process Model and Mental Model for Research Outputs

Prior research has introduced principles that define what DS research is [20], and what goals it should pursue [16, 20], as well as practice rules that provide guidance for conducting [2, 16, 20, 38] and justifying it [1, 33, 55]. Nevertheless, principles and practice rules are only two out of the three characteristics of a methodology [13]. The missing part is a procedure that provides a generally accepted process for carrying it out.

Hitherto, IS researchers have not focused on the development of a consensus process and mental model for DS research, such as that called for in engineering literature [16, 38] and required by the IS research discipline. This lack of a consensus-based DS research process model might help to explain why, despite many citations, the message of DS research has not resulted in more research in IS that makes explicit use of the paradigm [56]. Instead, much of the DS research published by IS researchers has been published in engineering journals, where DS behaviors are more the norm. Some of that published in IS journals has required *ad hoc* arguments to support its validity [7, 9, 34, 35, 42]. For example, in [35], the authors use information theory to justify the use of an IS planning method, which, in reality, was a designed method. In [34], the researchers justify their work as a practical extension of another methodology, rather than making explicit design claims. In [42], the authors describe the development of a software reuse measure in the context of a field study and evaluate the artifact using one project of the field company that is treated as a case study.

Defining Objectives of a Solution: Process and Mental Model Consistent With Prior Research

Our overall objective for the paper is the development of a methodology for DS research in IS. We do this by introducing a DS process model, which together with prior research on DS provides DS research with a complete methodology. The design of this conceptual process will seek to meet three objectives; it will (1) provide a nominal process for the conduct of design science research; (2) build upon prior literature about design science in IS and reference disciplines; and (3) provide researchers with a mental model or template for a structure for research outputs.

A Nominal Process

Such a process could accomplish two things for DS research in IS. It would help provide a roadmap for researchers who want to use design as a research mechanism for IS research. Such a process would not be the only way that DS research could be done, but it would suggest a good way to do it. It could also help researchers by legitimatizing such research, just as researchers understand the essential elements of empirical IS research and accept research that is well done using understood and accepted processes.

Building on Prior Research

There is a substantial body of research, both within the IS literature and in reference disciplines, that provides us with a tradition to support such a process. A process for design science research should build on this work while integrating its principles into a comprehensive methodology for conducting DS research. There are two sets of applicable literature. One revolves around issues

of actually doing academic design work, i.e., *design research*. The second set addresses the meta level of conducting research at a higher level of abstraction: it is *research about design research*. Below, we discuss the differences and how both contribute to meeting this objective.

The *design research* literature contains a large number of references to processes that are described incidentally to the production of research-based designs. Many of these descriptions are specific to research contexts and to the practical needs of design practitioners. In engineering, for example, there have been a number of design research efforts in which the focus has been on processes targeting the production of artifacts [53]. Evbuonwan et al. [15] mention fourteen such process models. Many, such as Cooper's StageGate [11, 12], are clearly intended as design or development methodologies, rather than research methodologies or processes, such as the one we are seeking to develop for DS research in IS. Likewise, in computer science, Maguire's [30] human-centered design cycle addresses the specific problems of requirements engineering methods for different situations and, in information systems, Hickey and Davis [21] addressed the issue from a functional view. Iivari et al. [23] considered the differences between IS development methods and methodologies and the needs that arise for method development. Processes described in this literature are of interest, but, since they vary widely and are generally context specific, they cannot necessarily be directly applied to the development of a general process for design science research.

The *research about design research* literature is rich with ideas about how to conduct research. This literature, while not providing process models that can be applied directly to the problem of DS research, does provide concepts from which we can infer processes. In IS, Nunamaker et al. [33] provide an abstract model connecting aspects of design research, but leave the actual process for conducting it to the researcher's inference. Walls et al.'s [55, 56]

information system design theory provides theory at a high level of abstraction from which we can infer a process. March and Smith's [31] and Hevner et al.'s [20] guidelines for design science research influence methodological choices within the DS research process. In the computer science domain, Preston and Mehandjiev [37] and Takeda et al. [46] proposed a "design cycle" for intelligent design systems.

In engineering, Archer [2] and Eekels and Roozenburg [14] presented design process models that can be incorporated into a consensus process. Adams and Courtney [1] proposed an extension of Nunamaker et al.'s [33] system development research methodology via inclusion of action research or grounded theory approach as ways to conduct research. Cole et al. [10] and Rossi and Sein [40] proposed basic steps to integrate design science research and action research. So far, no complete, generalizable process model exists for DS research in IS, however, if we develop such a process model, it should build upon the strengths of these prior efforts.

A Mental Model

The final objective of a DSRM process is to provide a mental model for the characteristics of research outputs. A mental model is a "small-scale [model]" of reality …[that] can be constructed from perception, imagination, or the comprehension of discourse. [Mental models] are akin to architects' models or to physicists' diagrams in that their structure is analogous to the structure of the situation that they represent, unlike, say, the structure of logical forms used in formal rule theories [25]." Outcomes from DS research are clearly expected to differ from those of theory testing or interpretative research and a process model should provide us with some guidance, as reviewers, editors, and consumers, about what to expect from DS research outputs. March and Smith [31] contributed to this expectation with their ideas about research outputs.

Hevner et al. [20] further elaborated on this expectation by describing DS research's essential elements. A mental model for the conduct and presentation of DS research will help researchers to conduct it effectively. In the next section we use prior research about design and design science research to design a DSRM process.

Design: Development of the Methodology

Development of the methodology required the design of a DSRM process. To accomplish this, we looked to influential prior research and current thought to determine the appropriate elements, seeking to build upon what researchers have said in key prior literature about what DS researchers did or should do. Our aim here was to design a methodology that would serve as a commonly accepted framework for carrying out research based on design science research principles outlined above. Rather than focusing on nuanced differences in views about DS among various researchers, we sought to use a consensus building approach to produce the design. Consensus building was important to ensure that we based the DSRM on well-accepted elements.

A number of researchers in IS and other disciplines have contributed ideas for process elements. Table 1 presents process elements, stated or implied, from seven representative papers and presentations and our synthesis: the components of the DSRM process. The authors agree substantially on common elements. The result of our synthesis is a process model consisting of six activities in a nominal sequence, which we justify and describe here and graphically in Figure 1.

<<<< INSERT TABLE 1 HERE >>>>

All seven papers include some component in the initial stages of research to define a research problem. Nunamaker et al. [33] and Walls et al. [55] emphasize theoretical bases, while engineering researchers [2, 14], focused more on applied problems. Takeda et al. [46] suggest the need for problem enumeration, while Rossi and Sein [40] advocate need identification. Hevner et al. [20] asserted that DS research should address important and relevant problems.

> *Activity 1. Problem identification and motivation.* Define the specific research problem and justify the value of a solution. Since the problem definition will be used to develop an artifact that can effectively provide a solution, it may be useful to atomize the problem conceptually so that the solution can capture its complexity. Justifying the value of a solution accomplishes two things: it motivates the researcher and the audience of the research to pursue the solution and to accept the results and it helps to understand the reasoning associated with the researcher's understanding of the problem. Resources required for this activity include knowledge of the state of the problem and the importance of its solution.

Some of the researchers explicitly incorporate efforts to transform the problem into system objectives*,* also called meta-requirements [55] or requirements [14], while for the others this effort is implicit, e.g., part of programming and data collection [2] or implicit in the search for a relevant and important problem. Identified problems do not necessarily translate directly into objectives for the artifact because the process of design is necessarily one of partial and incremental solutions. Consequently, after the problem is identified, there remains the step of determining the performance objectives for a solution.

> *Activity 2. Define the objectives for a solution.* Infer the objectives of a solution from the problem definition and knowledge of what is possible and feasible. The objectives can be quantitative, e.g., terms in which a desirable solution would be better than current ones, or qualitative, e.g., a description of how a new artifact is expected to support solutions to problems not hitherto addressed. The objectives should be inferred rationally from the problem specification. Resources required for this include knowledge of the state of problems and current solutions, if any, and their efficacy.

All of the researchers focus on the core of design science across disciplines: *design and development*. In some of the research, e.g., [14, 33], the design and development activities are further subdivided into more discrete activities whereas other researchers focus more on the nature of the iterative search process [20].

> *Activity 3. Design and development.* Create the artifact. Such artifacts are potentially constructs, models, methods, or instantiations (each defined broadly) [20] or "new properties of technical, social, and/or informational resources [24]". Conceptually, a design research artifact can be any designed object in which a research contribution is embedded in the design. This activity includes determining the artifact's desired functionality and its architecture and then creating the actual artifact. Resources required moving from objectives to design and development include knowledge of theory that can be brought to bear in a solution.

Next, the solutions vary from a single act of *demonstration* [55] to prove that the idea works, to a more formal *evaluation* [14, 20, 33, 40, 51] of the developed artifact. Eekels and Roozenburg[14] and Nunamaker et al. [33] include both of these phases.

> *Activity 4. Demonstration.* Demonstrate the use of the artifact to solve one or more instances of the problem. This could involve its use in experimentation, simulation, case study, proof, or other appropriate activity. Resources required for the demonstration include effective knowledge of how to use the artifact to solve the problem.

> *Activity 5. Evaluation.* Observe and measure how well the artifact supports a solution to the problem. This activity involves comparing the objectives of a solution to actual observed results from use of the artifact in the demonstration. It requires knowledge of relevant metrics and analysis techniques. Depending on the nature of the problem venue and the artifact, evaluation could take many forms. It could include such items as a comparison of the artifact's functionality with the solution objectives from activity two above, objective quantitative performance measures, such as budgets or items produced, the results of satisfaction surveys, client feedback, or simulations. It could include quantifiable measures of system performance, such as response time or availability. Conceptually, such evaluation could include any appropriate empirical evidence or logical proof. At the end of this activity the researchers can decide whether to iterate back to step three to try to improve the effectiveness of the artifact or to continue on to communication

and leave further improvement to subsequent projects. The nature of the research venue may dictate whether such iteration is feasible or not.

Finally, Archer [2] and Hevner et al. [20] propose the need for *communication* to diffuse the resulting knowledge.

> *Activity 6. Communication.* Communicate the problem and its importance, the artifact, its utility and novelty, the rigor of its design, and its effectiveness to researchers and other relevant audiences, such as practicing professionals, when appropriate. In scholarly research publications, researchers might use the structure of this process to structure the paper, just as the nominal structure of an empirical research process (problem definition, literature review, hypothesis development, data collection, analysis, results, discussion, and conclusion) is a common structure for empirical research papers. Communication requires knowledge of the disciplinary culture.

<<<<< INSERT FIGURE 1 HERE >>>>>

This process is structured in a nominally sequential order; however there is no expectation that researchers would always proceed in sequential order from activity one through activity six. In reality, they may actually start at almost any step and move outward. A problem-centered approach is the basis of the nominal sequence, starting with activity one. Researchers might proceed in this sequence if the idea for the research resulted from observation of the problem or from suggested future research in a paper from a prior project. An objective-centered solution, starting with activity two, could be triggered by an industry or research need that can be addressed by developing an artifact. A design and development-centered approach would start with activity three. It would result from the existence of an artifact that has not yet been formally thought through as a solution for the explicit problem domain in which it will be used. Such an artifact might have come from another research domain, it might have already been used to solve

a different problem, or it might have appeared as an analogical idea. Finally, a client/context initiated solution may be based on observing a practical solution that worked; it starts with activity four, resulting in a DS solution if researchers work backwards to apply rigor to the process retroactively. This could be the by-product of a consulting experience.

Demonstration in Four Case Studies

To demonstrate the use of the DSRM, we apply it retroactively to four already published IS research projects. In the first, researchers design and develop a data warehousing solution to support data gathering and analysis necessary for public health policy [6, 7]. The second explicates the design of a software reuse measure that was used in subsequent case study research [41, 42]. The third reports on the design of an application and middleware for the Internet2 environment that provides telephony and video functionalities [9, 17]. Finally, the fourth depicts the development of a method, critical success chains (CSC) [34, 35], for use in generating a portfolio of new ideas for mobile financial services applications.

We show in each case, how the process of motivating, developing, designing, demonstrating, evaluating, and communicating the artifact is consistent with the DSRM. In none of the cases were the publication outputs explicitly described and presented as using a DS research process, because a designed methodology had not been hitherto available. In the summaries that follow, we used the language of the DSRM to interpret the research processes actually used by the researchers to determine how well the DSRM fits with the research processes used.

Case 1: The CATCH Data Warehouse for Health Status Assessments

The Comprehensive Assessment for Tracking Community Health (CATCH) methods had been published [44] and successfully used in multiple counties in the United States. The methodology requires data to be gathered from multiple sources including hospitals, health agencies, and healthcare groups, and surveys. CATCH organizes over 250 healthcare indicators into 10 categories that represent a variety of healthcare issues. The output of the CATCH methodology is a prioritized listing of community healthcare challenges. In this work Berndt, Hevner, Studnicki, and Fisher automated the use of CATCH by developing a data warehouse that implements the methodology. Figure 2 summarizes how the DSRM applies to the steps undertaken as part of this design science research effort [5-7].

<<<<< INSERT FIGURE 2 HERE >>>>>

Problem-centered Approach

The lack of automated support made the data gathering for the CATCH methodology labor-intensive and slow; thus extended trend analyses were cost prohibitive for most communities. The need for a more efficient automated data access for CATCH health assessments triggered the development of the CATCH data warehouse.

Problem Identification and Motivation

The United States has the highest health care spending of any nation in the world, both as a percentage of GDP and per capita. Nevertheless, the US does not rank among the countries with the healthiest populations. Thus, there was a need to assess the country's health status in order to assist communities to develop comprehensive health strategies, leading to better resource allocation for prevention and treatment. The formulation of such strategy had to be based on

local health data. The availability and quality of health data was low, which is why health data rarely was the basis for decision making on health policies. Although CATCH was an available assessment method at the time, the labor-intensity of non-automated data gathering limited its adoption.

Objective of the Solution

The objective was to develop a data warehouse solution for the automated support of the CATCH methods that enables users to run cost-effective analyses. The major challenges included the diversity of the data sources, the diversity of target groups for which reports were generated, and the need to conform to the public policy formulation process. The data warehouse was to provide a rich environment that would enable an improvement of research capabilities on critical healthcare issues with the long-term goal of centering the role of public health agencies around monitoring and improving the health status of the population using this technology.

Design and Development

The artifact is the data warehouse that supports and automates CATCH. The researchers drew from data warehousing research to develop the CATCH data warehouse with data arranged in a star schema. The design includes three levels of granularity: the report structures, aggregate dimensional structures, and fine-grained and transaction oriented dimensional structures. Staging and quality assurance methods were established to enable a successful use of the data warehouse and performance issues were addressed. The design and related methods have been and continue to be refined based on emerging performance needs.

Demonstration

After developing proof-of-concept level prototypes, the artifact was extensively adapted to production use by user organizations. The researchers point to the application of the CATCH data warehouse in multiple counties and provide screenshots of several output screens in their publications. In related research, it was also demonstrated that the CATCH data warehouse could be used to conduct bioterrorism surveillance: a similar data warehouse approach was used in a demonstration surveillance system in the State of Florida.

Evaluation

The original CATCH methods have been used and refined for over more than ten years in more than 20 US counties. The researchers implemented the CATCH data warehouse as a fully functional version in Florida's Miami-Dade county. The verification of the accuracy of the automated generation of the report through a comprehensive manual check identified only minor problems in use. The data warehouse was found to be flexible and effective in this field application.

Communication

Manuscripts relating to the CATCH data warehouse have been published in academic journals, academic conference proceedings, and professional outlets. The development of the health care data warehouse was presented in *Decision Support Systems* [7] and *Upgrade*, a professional online magazine [6]. The challenges of quality assurance in the CATCH data warehouse was discussed in *IEEE Computer* [5]. Further, the use of data warehousing technology and CATCH in the context of bioterrorism has been explored in proceedings of the *IEEE International Conference on Intelligence and Security Informatics* [4, 8]. In addition, this research effort received attention from various newspapers in Florida.

Contribution

The CATCH data warehouse research resulted in architecture and applications. This artifact was used effectively to collect data in a consistent and automated fashion from disparate local health care organizations, which, among themselves, had no consistent information systems or data collection infrastructure. The immediate contribution of this research to public health policy was the ability to collect data that could be effectively used to formulate such policy within Florida, where it was implemented. In a broader context, the artifact could serve as a template for the implementation of such systems elsewhere. Furthermore, the architecture and applications could serve as a model for the development of similar systems, such as one that was developed for bioterrorism surveillance, to serve other public or business needs.

Case 2: A Software Reuse Measure Developed at MBA Technologies

MBA Technologies was a medium-size Phoenix based software developer that specialized in the development of business process and accounting systems; the company obtained high reuse in its software development by leveraging of existing components that were mapped to an enterprise-level model. The model and its components represented generic business solutions that can be customized to a specific set of requirements. The objective of this work by Rothenberger and Hershauer [42] was to develop a generic reuse metric for such enterprise-level model-based software development environment and to apply the generic measure to the specifics of the organization. Figure 3 provides a summary of the research steps discussed below.

<<<<< INSERT FIGURE 3 HERE >>>>>

Objective-centered Solution

In spring 1997 the researchers wanted to conduct an in-depth case study on the reuse efforts at MBA Technologies that required the assessment of the reuse rates the company obtained in its projects. Existing reuse measures available in the information systems and computer science literatures were not suitable to assess the reuse rate in the enterprise-level model context; existing measures were only defined on a high level and did not define specifics required for an application to actual projects. To use the underlying principles of a high-level measure in the field setting, decisions had to be made how to assess and count modified component reuse, partial component reuse, generated code, and multiple layers of abstraction.

Problem Identification and Motivation

Most software development companies do not assess their success at reuse, even if they are actively pursuing an increase in the reuse of software artifacts through a formal reuse program. Thus, many software developers invest in corporate reuse programs without being able to evaluate whether their programs lead to an increase of reuse. Also, without a formal reuse measure they are not able to identify differences in reuse success among projects. The development and subsequent dissemination of a reuse measure that can be applied to enterprise-level model-based reuse efforts would enable the researchers to conduct an in-depth analysis of MBA Technologies' reuse success across multiple completed projects. Further, a measure would provide the means for continued monitoring of reuse success in software projects.

Objective of the Solution

The objective was to develop a reuse rate measure that allowed the researchers to assess the reuse rate, or reuse percentage, of the participating organization for subsequent case study research. Such a measure would represent the development effort that was reused from existing code as a percentage of the total project development effort. The measure was to be developed in

a generic fashion that would ensure its applicability to settings other than the participating organization as well.

Design and Development

The software measurement literature was used to evaluate the suitability of potential size or complexity measures. The concept of the reuse rate was obtained from software reuse literature, which served as the theoretical foundation for the development of the reuse metric. The result of the design effort was a generic reuse measure that could be applied to any enterprise-level model-based reuse setting and that was customized to the specific organizational setting at MBA Technologies. The reuse rate was defined as the reused development effort divided by the total development effort of the project. The metric artifact operationalized this high-level definition by formalizing how to count reused development effort and total development effort in the context of an enterprise-level model-based reuse setting. This operationalization required making decisions on how to count duplicate use of code stubs, modified reused components, and other special cases. These decisions and assessments were made based on prior findings in the software reuse literature.

Demonstration

Assessing and reporting the reuse rate for a project in the participating organization demonstrated the measure's feasibility and efficacy. Details about the company's development environment, including a classification of code into three levels of abstraction, the use of generated code, specifics about the component design, and the classification of certain code stubs, were obtained through structured interviews. Size measures in thousands of lines of code (KLOC) and the classification of code stubs at the lowest level of abstraction were obtained directly from source code. The measure yielded separate reuse percentages for code on three

layers of abstraction, according to the organization's classification, as well as a weighted total reuse percentage. Further, reused generated code was reported separately.

Evaluation

In the subsequent case study, the measure was used to assess the reuse rates of five projects at MBA Technologies, with sizes varying from 57 KLOC to 143 KLOC. The assessed total project reuse rate for non-generated code ranged from 50.5% to 76.0%. In structured interviews, developers were asked to assess the projects' reuse rates without prior knowledge of the measured results. The relative assessments were consistent with the actual measurements.

Communication

The contributions of this effort were disseminated in peer reviewed scholarly publications. The development of the reuse rate measure was published in *Information & Management* [42]. Further, the measure was used to assess the projects of the software development organizations in a subsequent case study which appeared in the *Decision Sciences Journal* [41].

Contribution

The research artifacts resulting from this study included a designed and evaluated formal measure and metric for software reuse rates. These artifacts provide a valid and effective measure for use in development practice at the organizational and project level for evaluation and assessment of the effectiveness and performance of software reuse efforts. They could be valuable measures for use in research where measures of software reuse are required.

Case 3: SIP-based Voice and Video-over-IP Software

The Session Initiation Protocol (SIP) is an Internet Engineering Task Force (IETF) standard for IP Telephony that was developed for voice-over-internet communication. Researchers at the Network Convergence Lab (NCL) at Claremont Graduate University had been involved since early 2000 in the standardization process for SIP-based voice communication. In early 2001, the Internet2 Consortium wanted to explore video-over IP applications as an emerging architecture for IP networks in cooperation with the NCL. This led to a research effort by Chatterjee, Tulu, Abhichandani, Li, Laxminarayan, Gemmill, Srinvasan, and Lynn focusing on the extension of the SIP standard [9, 17, 47, 48], which is summarized in Figure 4.

<<<<< INSERT FIGURE 4 HERE >>>>>

A Design and Development-centered Approach

Building on the SIP-based voice communication standard, researchers at NCL aimed to design and deploy a voice and video-conferencing over IP application that enhances the SIP-based voice communication standard. This design science research artifact was to be deployed across 202 universities.

Problem Identification and Motivation

There were three particular technical problems that emerged in discussions within the IETF and the Internet2 consortium. First, while SIP standards were emerging, there were no actual SIP-based software artifacts that would provide telephony and video functionalities and features. Secondly, since universities and companies use a variety of vendor products, technologies, and standards, there was a need to develop middleware that provided a uniform way for storing and finding information related to video and voice users, as well as devices and technologies in

enterprise directories. This problem was particularly relevant to Internet2 because universities were implementing diverse technology solutions. Thirdly, there was a need to solve the security problem: some applications including SIP cannot traverse firewalls and fail to work when private IP addresses are used behind network address translators (NATs).

Objectives of the Solution

Several requirements drove the research effort. First, researchers needed to follow SIP technical standards closely. Secondly, the performance of the artifact could not be allowed to overwhelm the capabilities of a typical desktop computer of the time. Also, there were functions and features necessary to meet the requirements of the end-users, including point-to-point calls, instant messaging, and video conferencing. Furthermore, the middleware software for storing user and device information had to be compatible with existing directory services within participating campuses. Finally, a security solution was required that would be implemented within the application in such a way that no external measures were required within firewalls and routers.

Design and Development

The design and development process followed that of an IS development research project. It started with a requirements gathering process, in which a diverse set of potential end-users participated, resulting in requirements documentation, which was later used for designing a detailed technical architecture through Internet2 member meetings and mailing list discussions. In particular, the middleware that was developed was standardized through the International Telecommunications Union's standardization section (ITU-T), which required participation from several European and other international participants. The software was developed, based on

computer science and networking literature, to provide a proof-of-concept and a fully working client application.

Demonstration

The implemented artifact includes the SIP application and its directory middleware. Implementation details serve as a demonstration of the approach. CGUsipClientv1.1.x is a java-based application implemented on a commercial SIP stack. It uses Java Media Framework (JMF) application programming interfaces (API) for voice and video operations. It provides point-to-point telephony, video calls, directory service lookup, click-to-call, and secured authentication. It uses technologies to solve the security problems mentioned above and utilizes a Lightweight Directory Access Protocol (LDAP)-based solution for providing directory information. It uses an H.350 directory to offer *"White Page"*, *"Click-to-Call",* and *"Single Sign-On"* facilities. *"White Page"* displays user information. *"Click-to-Call"* enables a user to call another user by clicking on the other user's SIP Uniform Resource Identifier (URI). *"Single Sign-On"* provides an authenticating facility with a SIP-based proxy or a registrar based on the credentials fetched from the LDAP structure instead of explicitly providing the username and the password for registration.

Evaluation

Once the software was developed, researchers started a thorough testing process. First, the artifact was extensively tested for debugging purposes within a closed group. Next, the application was shared with the entire Internet2 community via a web portal, where users were able to download the software after providing information about themselves. The information provided was automatically linked to the middleware directory. More than 250 institutions downloaded the software artifact. The researchers found that 30% of those who downloaded the

software used it for one or more hours daily. In addition, the CGUsipClient was tested for performance, usability, and usefulness. The researchers measured the call setup time, CPU usage, end-to-end delay; all results were satisfactory. The test of the H.350 middleware standard implementation showed that the directory service performed well and lookup time was satisfactory. Finally, the security mechanism that was developed to open and close pinholes in the firewall/NAT for active SIP sessions was successfully tested, indicating only minimal and acceptable delays. The Internet2 working group was pleased with the efforts, judging the design process successful.

Communication

Preliminary results of this project were reported in refereed conferences, e.g., [47, 48] and detailed results appeared in the *IEEE Journal on Selected Areas in Communications* [9] and the *Journal of Internet Technology* [17]. In addition, the middleware work received recognition in Internet2 and NSF press releases. Trade magazines, such as *Network World* and corporations, such as *Packetizer* maintain the H.350 middleware standard information and details.

Contribution

This research enhanced and further developed the existing SIP VoIP standard into an SIP-based video-over-IP standard. The enhanced standard was successfully evaluated and made available to the Internet2 working group, which may result in the commercial use of this new standard. Further, the existence of a Video-over-IP standard may serve as a foundation for future research aimed at enhancements of this technology.

Case 4: Developing a Method at Digia to Generate Ideas for New Applications that Customers Value

Digia Ltd. was a Helsinki-based research and development firm specializing in innovative software applications for wireless communication that focused on the creation of personal communication technologies and applications for wireless information devices. This case reports the efforts of Peffers, Gengler, and Tuunanen at Digia, also illustrated in Figure 5 and reported in detail in [34, 35], to develop a better IS planning method.

<<<<< INSERT FIGURE 5 HERE >>>>>

A Client-initiated Project

In fall 2000, Digia chairman, Pekka Sivonen, approached one of the authors with a request to help define a portfolio of potential applications for Digia to develop to meet the need for financial services delivered by the next generation wireless devices [34]. The researcher accepted this invitation because it fit with his current research objective and that of colleagues: to develop a method to support the generation of ideas for IS projects that would provide the greatest impact on achieving a firm's strategic goals. Since few applications for providing financial services using mobile devices were in operation at the time, this looked like a good opportunity to use a new conceptual method for IS planning that the authors earlier trialed in a business case environment. Because the client's objective was a portfolio of applications and the research objective was the development of a requirements engineering methodology for determining this portfolio, this meant that the initiative for the project came from a proposed demonstration of the new methodology.

Problem Identification and Motivation

Literature had shown that in most firms there was no shortage of ideas for new IS projects, but most tended to be suboptimal [34]. The problem was to design a method for managers to make use of the ideas of many people within and around the organization, while keeping the focus on what is important and valuable for the firm. Bottom-up planning generates so many ideas that it may be impossible to sort out the few that have the potential to have high impact on the firm, because most are self serving, narrowly focused, and of little potential impact. Top down planning has the benefit of strategic perspective and better alignment with the interests of owners, but its weakness is an inability to take advantage of knowledge from around the organization and beyond the organization about ideas that may be important to the firm. It generally ignores the ideas of all except those that originate in the executive suites.

Objectives of the Solution

The researchers' objective was to demonstrate a new IS planning method in an industry setting. This allowed the researchers to study how well in a non-controlled test environment the method would meet the proposed objectives: 1) allow them to make use of the ideas of many from in and around the organization, 2) include experts outside the firm and potential users, but to keep the focus on ideas with high strategic value to the firm, and 3) to transform the resulting data into forms that can be used for IS planning and development.

Design and Development

The Digia project researchers made use of a pilot study, conducted at Rutgers University [34], as the basic template for the new method. They used personal construct theory (PCT) [26] and critical success factors [39] as theoretical bases for the method development. For the data collection they borrowed "laddering," a PCT based technique, developed for use in marketing research for structured interviewing to collect rich data on subject reasoning and preferences. For

the analysis they adapted hierarchical value maps, which had been used in marketing to display aggregated laddering data graphically. They incorporated an ideation workshop, where business and technical expertise was brought to bear on the task of developing feasible ideas for new business applications from the graphical presented preferences and reasoning of the subjects. The result of the design effort was the critical success chain (CSC) method for using the ideas of many people in and around the organization to develop portfolios of feasible application ideas that are highly valuable to the organization. In the Digia case these concepts were applied to a real industry setting, which in turn allowed the researchers to extend the CSC method with concepts relevant to the case organization.

Demonstration

The researchers used the opportunity at Digia to demonstrate CSC's feasibility and efficacy [34, 35]. They started by recruiting and interviewing 32 participants, approximately evenly divided between experts and potential end users. They conducted individual structured interviews, using stimuli collected from the subjects ahead of time. The interview method was intended to encourage participants to focus on the value of ideas.

The laddering interviews provided rich data about applications the participants wanted and why. Using qualitative clustering, data was used to create five graphical maps, containing 114 preference and reasoning constructs. The next step was to conduct an ideation workshop with six business and engineering experts and managers from the firm to convert the participant preferences to feasible business project ideas at a "back-of-the-envelop" level. In the workshop, conducted in isolation in a single five-hour stretch, the participants developed three business ideas, with application descriptions, business models, and interaction tables. These were further

developed by analysts in post workshop work to be integrated into the firm's strategic planning effort.

Evaluation

The CSC method met the project's objectives. It enabled the researchers to use rich data collected by a widely representative sample of experts and potential lead users from outside the firm, to keep the focus on ideas of potential strategic importance to the firm, and to analyze the data in such a way so as to make it useful for IS planning in the firm. Digia representatives were very enthusiastic about the results of the workshop [34]. This feedback and the successful implementation of the method in practice enabled the project researchers to present initial "proof-of-concept" level validation of the new method [34, 35]. The firm intended to use the resulting applications to plan its continued product development efforts.

Communication

The case study was reported in the *Journal of Management Information Systems* [34] and *Information & Management* [35]. The structure of these papers closely follows the nominal sequence of activities presented in the DSRP model. In addition, the findings were presented in several practitioner oriented outlets, including a book chapter, [36], technical reports, e.g., [49], and trade magazine articles, e.g., [50].

Contribution

The artifact developed as a result of this research is a method for IS planning that can be used to make use of the knowledge of many people from in and around the organization, maintain focus on potential systems and applications of strategic value to the firm, and produce outputs of use to designers and managers in the IS development process. Consequently, it may be

valuable for use in IS planning practice. In research the method may be extended to enable planning efforts that focus on the development of requirements at the feature level, particularly to develop requirements engineering methods for such contexts as the development of cross-cultural feature sets and for special populations, such as for disabled persons.

Evaluation of the DSRM Process

We evaluate the DSRM process in terms of the three objectives for the DSRM described above. First, it should be consistent with prior DS research theory and practice, as it has been represented in the IS literature, and with design and design science research, as it has been conveyed in representative literature in reference disciplines. Secondly, it should provide a nominal process for conducting DS research in IS. Thirdly, it should provide a mental model for the characteristics of research outputs. We will address each objective below.

First, the DSRM process is consistent with concepts in prior literature about design science in IS. Because we used a consensus method to design the DSRM, this consistency is an inherent outcome of the process. For example, Nunamaker et al.'s [33] five-step methodology can be mapped roughly to the DSRM process. Likewise Walls et al.'s [55, 56] "components of an information system design theory," Takeda et al.'s [46] "design cycle" solution for intelligent computer aided design systems, Rossi and Sein's [40] steps, Archer's [2] process for industrial design, Eekels and Roozenburg's [14] process for engineering design, and Hevner et al.'s [20] guidelines for the required elements of design research, are all consistent with the DSRM.

Secondly, the DSRM provides a nominal process for conducting DS research. In addition, in the demonstration of four cases we showed how each of the four DS research projects described in the cases followed a process consistent with the DSRM. In addition, the cases

demonstrate each of the four research entry points described in the DSRM, including a problem-centered initiation, an objective-centered initiation, a design and development-centered initiation, and a client/context-centered initiation. In each case the process worked well, and it was effective for its intended purposes.

Thirdly, The DSRM provides a mental model for the presentation of research outcomes from DS research. The explication of the CATCH data warehouse, reported in [7], incorporated all of the elements of the DSRM process, although it did not use the DSRM terminology. Rothenberger and Hershauer [42] followed the general outlines of the process in the structure of the paper, including a statement of the problem in the introduction, an explicit "purpose" section to outline the objectives of a solution, a design section called "creating the measure," demonstration sections called "application of the measure to a specific problem," and "example case data." Peffers et al. used a structure based on [20] and consistent with the DSRM to report the Digia case [34]. Chatterjee et al. [9], incorporated elements of the DSRM in presenting the research. The paper identified the problem and defined the potential objectives or "benefits" of a solution in the introduction; it also incorporated the other elements of the DSRM in "Design, Implementation, and Performance of CGUsipClient."

Discussion

We defined design science earlier in this paper. Recently, however, researchers [9] have raised questions about similarities between DS and action research (AR). Both Cole et al. [10] and Järvinen [24] conclude that the similarities between these research approaches are substantial. Cole et al. [10] argued that the approaches share important assumptions regarding ontology, epistemology, and axiology. Järvinen [24] pointed to many similarities, although they

employ different terminology, and went so far as to suggest that we cannot clearly differentiate between them. Perhaps the clearest distinction between them is found in their conceptual origins. DS research comes from a history of design as a component of engineering and computer science research, while action research originates from the concept of the researcher as an "active participant" in solving practical problems in the course of studying them in organizational contexts. In DS research, design and the proof of its usefulness is the central component, while in action research, the focus of interest is the organizational context and the active search for problem solutions therein. Resolution of this point will have to remain outside the scope of this paper, but it does present an interesting and perhaps fruitful area for further thought. It would appear that the DSRM could be used as a structure to present AR. Likewise, the search for a designed artifact could be presented as AR. Clearly the side-by-side existence of the two methodologies presents the researcher with choices for the structure of the research process and the presentation of the resulting solution. This discussion also raises an interesting question about whether the DSRM could be used in an action research study, whether researchers could use it to design new innovations based on technical, social and/or informational resources or their combinations [24], and whether action research and design science research could be conceptually and methodologically integrated.

The DSRM is intended as a methodology for research, however, one might wonder whether it might also be used as a methodology for design in practice. There would appear to be no reason it could not be so used, however, there are elements of the DSRM that are intended to support essential DS research characteristics that might not always apply well to design in practice.

A design artifact, such as a curved wooden staircase, a kitchen appliance, or a surgical knife, is not necessarily required to embody new knowledge that would be conveyed to an audience through a scientific publication outlet. Consequently there is no inherent requirement that a designer employ any rigorous process to create it. There may, on the other hand, be organizational, evidentiary, regulatory, or other reasons why some level of process rigor may be required. The designer of the curved staircase might be free to work from a simple sketch with a few measurements, while the designer of the surgical knife might be required to proceed through a very careful process of data collection, consultation, documentation and testing. Thus, for design in practice, the DSRM may contain unnecessary elements for some contexts, while being much too general to support design in others.

An important step in the evaluation of the DSRM was its application to four cases of previously published DS research. The four studies were chosen in part because that they represent examples of DS research with four different entry points as specified in the DSRM. None of the articles in which these studies were reported uses the language of the DSRM to explain its research approach. Instead they each use ad hoc arguments to support the validity of the research. We found this to be common in prior DS research in the IS field, because hitherto no generally accepted framework for conducting and presenting DS research existed, at least not until Hevner et al.'s [20] guidelines provided characteristics of good DS research outcomes. Like Hevner et al. [20], we have used secondary data, in the form of four cases, to demonstrate the application of the DSRM. Results of the analysis of the four cases show that they are all instances of DS research that can be well framed in terms of the DSRM. Thus, we have used the case discussions as a vehicle, not only to evaluate the DSRM, but also to transfer established DS research into a formal research framework and to illustrate its applicability. We expect that the

case studies will provide useful templates for researchers who want to apply DSRM to their efforts. The development and evaluation of the DSRM was heavily influenced by design research, thus DSRM concepts have guided us in the conduct and presentation of this work and this is reflected in the structure of this paper. Clearly, the next step would be to directly adopt the proposed methodology in new DS research. This is something that we are currently working on in our ongoing research.

The ad hoc justification of prior DS research suggests the difficulty that authors faced, for lack of reference to a commonly accepted DS methodology, in supporting the validity of DS research in IS. Without a framework that is shared by authors, reviewers, and editors, DS research runs the danger of being mistaken for poor quality empirical research or for practice case study. The DSRM completes a DS research paradigm with a methodology that is consistent with the DS research processes employed in the IS discipline, in this way establishing a common framework for future researchers to validate DS research, without making ad hoc arguments for its validity.

Conclusion

In this paper we sought to develop a methodology for DS research in IS. We wanted this methodology to be well grounded in existing literature about design science in IS and related disciplines. In addition, we wanted a methodology that would provide guidance for researchers who work on DS research and provide a mental model for the presentation of its outcomes.

Interestingly, other research paradigms have been adapted for use in our discipline without such a formal definition. This is hardly surprising because IS research, if one counts from the tenure of our senior journals, is only about one-third of a century old. As a result, it was handed

behavioral and natural science traditions from much older research disciplines in the business academe and adopted them without much adaptation. Consequently, this paper represents a unique effort to formally define a research methodology for use in IS.

We should emphasize that this paper represents one general methodological guideline for effective DS research. Researchers should by no means draw any inference that the DSRM is *the* only appropriate methodology with which to conduct such research. We can imagine that the efforts of others could result in at least five other types of DS research methodologies:

1. A methodology to support curiously motivated DS research, although such research is not common in business disciplines, might look quite different than the DSRM. Some research in the social and natural sciences is driven primarily by curiosity and may therefore lack explicit outcome objectives.

2. A methodology to support research within a specific stream in IS might incorporate elements specific to the context of that research. For example, a methodology to support the design of methods for requirements analysis might provide guidelines for specific expected elements of requirements analysis, including organizational context, data gathering, modeling, and the form of the requirements specification. We observed a number of context specific design research methodologies in engineering.

3. Where the motivation for the research is to solve problems in a specific organizational context, action research, as suggested in preceding paragraphs, might be an alternative or complementary paradigm through which to design IS research artifacts.

4. With respect to specific activities in the research process, future researchers may enhance the DSRM, for example, by developing subsidiary processes.

5. Finally, circumstances, such as context-specific constraints, may motivate researchers to develop and implement ad hoc processes that, while inconsistent with this DSRM, may,

nonetheless, be well justified and produce valid results.

While these five examples come readily to mind, it seems likely that there are other ways that DS research could be well done. We present these alternatives here, without recommendation and without knowledge of their prior use, in speculation about what valid alternatives to the DSRM might be subsequently developed and used. In doing so, we are suggesting that the DSRM should not be used as a rigid orthodoxy to criticize work that does not follow it explicitly.

The case studies we have provided with this paper demonstrate its use within the scope of four research problems. Further use will tell us whether there are problem domains where it requires extension or where it does not work well. Another interesting problem is that of the research entry point. We demonstrated that there are multiple possible entry points for design science research. Of course, this issue is not unique to design science research. We do not recall reading a theory testing paper where the authors say that they decided on the research questions after they collected the data or even after they did the analysis, but we have all observed that this happens with no ill effects. The "scientific method" is an espoused theory, approximated but not always matched by theory in use. We think that a research methodology should account, as far as it is practical, for the research process in use.

References

1. Adams, L., and Courtney, J. Achieving Relevance in IS Research via the DAGS Framework. *37th Hawaii International Conference on System Sciences*, Big Island, Hawaii: IEEE, 2004, 10.

2. Archer, L.B. Systematic method for designers. In, Cross, N., (ed.), *Developments in design methodology*, London: John Wiley, 1984, 57-82.

3. Aumann, H.H., Chahine, M.T., Gautier, C., Goldberg, M.D., Kalnay, E., McMillin, L.M., Revercomb, H., Rosenkranz, P.W., Smith, W.L., Staelin, D.H., Strow, L.L., and Susskind, J. AIRS/AMSU/HSB on the Aqua Mission: Design, Science Objectives, Data Products, and Processing Systems. *IEEE Transactions on Geoscience and Remote Sensing*, 41, 2 (2003), 253-264.

4. Berndt, D.J., Bhat, S., Fisher, J.W., Hevner, A.R., and Studnicki, J. Data Analytics for Bioterrorism Surveillance. In, Chen, H.E.A., (ed.), *IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2004, 17-27.

5. Berndt, D.J., Fisher, J.W., Hevner, A.R., and Studnicki, J. Healthcare Data Warehousing and Quality Assurance. *IEEE Computer* (2001), 56-63.

6. Berndt, D.J., Hevner, A.R., and Studnicki, J. Data Warehouse Dissemination Strategies for Community Health Assessments. *Upgrade*, 2, 1 (2001), 48-54.

7. Berndt, D.J., Hevner, A.R., and Studnicki, J. The Catch data warehouse: support for community health care decision-making. *Decision Support Systems*, 35 (2002), 367-384.

8. Berndt, D.J., Hevner, A.R., and Studnicki, J. Bioterrorism Surveillance with Real-Time Data Warehousing. In, Chen, H.E.A., (ed.), *IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2003, 322-335.

9. Chatterjee, S., Tulu, B., Abhichandani, T., and Li, H. SIP-based Enterprise Converged Network for Voice/Video over IP: Implementation and Evaluation of Components. *IEEE Journal on Selected Areas in Communications - Recent Advances in Managing Enterprise Network Services*, 23, 10 (2005).

10. Cole, R., Purao, S., Rossi, M., and Sein, M.K. Being Proactive: Where Action Research Meets Design Research. *Twenty-Sixth International Conference on Information Systems*, Las Vegas, USA: AIS, 2005, 325-336.

11. Cooper, R. Winning with New Products Doing it Right. *IVEY Business Journal*, July-August 2000 (2000), 54-60.

12. Cooper, R.G. Stage-Gate Systems - a New Tool for Managing New Products. *Business Horizons*, 33, 3 (1990), 44-54.

13. DMReview. *Glossary*. Brookfield WI USA: SourceMedia, 2007 (available at http://www.dmreview.com/rg/resources/glossary.cfm) unpaginated.

14. Eekels, J., and Roozenburg, N.F.M. A methodological comparison of the structures of scientific research and engineering design: their similarities and differences. *Design Studies*, 12, 4 (1991), 197-203.

15. Evbuonwan, N.F.O., Sivaloganathan, S., and Jebb, A. A survey of design philosophies, models, methods and systems. *Proceedings Institute of Mechanical Engineers*, 210 (1996), 301-320.

16. Fulcher, A.J., and Hills, P. Towards a strategic framework for design research. *Journal of Engineering Design*, 7, 1 (1996), 183-193.

17. Gemmill, J., Srinivasan, A., Lynn, J., Chatterjee, S., Tulu, B., and Abhichandani, T. Middleware for Scalable Real-time Multimedia Communications Cyberinfrastructure. *Journal of Internet Technology*, 5, 4 (2004), 405-420.

18. Gentner, D., and Stevens, A.S. *Mental Models*. Hillsdale, New Jersey: Erlbaum Publishers, 1983.

19. Haran, M., Karr, A., Last, M., Orso, A., Porter, A., Sanil, A., and Fouche´, S. Techniques for Classifying Executions of Deployed Software to Support Software Engineering Tasks. *IEEE Transactions on Software Engineering*, 33, 5, (2007), 287-304.

20. Hevner, A.R., March, S.T., and Park, J. Design Research in Information Systems Research. *MIS Quarterly*, 28, 1 (2004), 75-105.

21. Hickey, A.M., and Davis, A.M. A unified model of requirements elicitation. *Journal of Management Information Systems*, 20, 4 (2004), 65-84.

22. Hoffman, R.R., Roesler, A., and Moon, B.M. What Is Design in the Context of Human-Centered Computing? *IEEE Intelligent Systems*, 19, 4 (2004), 89-95.

23. Iivari, J., Hirschheim, R., and Klein, H.K. A paradigmatic analysis contrasting information systems development approaches and methodologies. *Information Systems Research*, 9, 2 (1998), 164-193.

24. Järvinen, P. Action Research is Similar to Design Science. *Quality & Quantity*, 41, 1 (2007), 37-54.

25. Johnson-Laird, P., and Byrne, R. Mental Models Website, 2000 (available at http://www.tcd.ie/Psychology/Ruth_Byrne/mental_models/).

26. Kelly, G.A. *The Psychology of Personal Constructs*. New York: W W Norton & Company, 1955.

27. Klassi, J. Environmental enhancement of the oceans by increased solar radiation from space. *Oceans*, 17 (1985), 1290-1295.

28. Krishnamurthy, D., Rolia, J.A., and Majumdar, S. A Synthetic Workload Generation Technique for Stress Testing Session-Based Systems. *IEEE Transactions on Software Engineering*, 32, 11 (2006), 868-882.

29. Lisetti, C., and LeRouge, C. Affective computing in tele-home health. *Proceedings of the 37th Hawaii International Conference on System Sciences*, Island of Hawaii: IEEE, 2004, 8.

30. Maguire, M. Methods to support human-centred design. *International Journal of Human-Computer Studies*, 55, 4 (2001), 587-634.

31. March, S., and Smith, G. Design and Natural Science Research on Information Technology. *Decision Support Systems*, 15 (1995), 251-266.

32. McPhee, K. Design Theory and Software Design. *Technical Report*: University of Alberta, The Department of Computing Science, 1996, 75.

33. Nunamaker, J.F., Chen, M., and Purdin, T.D.M. Systems Development in Information Systems Research. *Journal of Management Information Systems*, 7, 3 (1991), 89-106.

34. Peffers, K., Gengler, C., and Tuunanen, T. Extending Critical Success Factors Methodology to Facilitate Broadly Participative Information Systems Planning. *Journal of Management Information Systems*, 20, 1 (2003), 51-85.

35. Peffers, K., and Tuunanen, T. Planning for IS applications: a practical, information theoretical method and case study in mobile financial services. *Information & Management*, 42, 3 (2005), 483-501.

36. Peffers, K., and Tuunanen, T. The Process of Developing New Services. In, Saarinen, T., Tinnilä, M., and Tseng, A., (eds.), *Managing Business in a Multi-Channel World: Success Factors for E-Business*, Pennsylvania: Idea Group Inc, 2005, 281-294.

37. Preston, M., and Mehandjiev, N. A Framework for Classifying Intelligent Design Theories. *The 2004 ACM workshop on Interdisciplinary software engineering research*, Newport Beach, California: ACM, 2004, 49-54.

38. Reich, Y. The Study of Design Methodology. *Journal of Mechanical Design*, 117, 2 (1994), 211-214.

39. Rockart, J.F. Chief executives define their own data needs. *Harvard Business Review*, 57, 2 (1979), 81-93.

40. Rossi, M., and Sein, M.K. Design research workshop: a proactive research approach. *26th Information Systems Research Seminar in Scandinavia*, Haikko Finland: The IRIS Association, 2003.

41. Rothenberger, M.A. Project-Level Reuse Factors: Drivers for Variation within Software Development Environments. *Decision Sciences Journal*, 34, 1 (2003), 83-106.

42. Rothenberger, M.A., and Hershauer, J.C. A Software Reuse Measure: Monitoring an Enterprise-Level Model Driven Development Process. *Information & Management*, 35, 5 (1999), 283-293.

43. Simon, H. *The sciences of the artificial*. Cambridge MA: MIT Press, 1969.

44. Studnicki, J., Steverson, B., Myers, B., Hevner, A., and Berndt, D. Comprehensive assessment for tracking community health (CATCH). *Best Practices and Benchmarking in Healthcare*, 2, 5 (1997), 196-207.

45. Swaab, R.I., Postmes, T., Neijens, P., Kiers, M.H., and Dumay, A.C.M. Multi-party negotiation support: Visualization and the development of shared mental models. *Journal of Management Information Systems*, 19, 1 (2002), 129-150.

46. Takeda, H., Veerkamp, P., Tomiyama, T., and Yoshikawam, H. Modeling Design Processes. *AI Magazine*, 1990, 37-48.

47. Tulu, B., Abhichandani, T., Chatterjee, S., and Li, H. Design and Development of a SIP-Based Video Conferencing Application. *Lecture Notes in Computer Science - High Speed Networks and Multimedia Communications: 6th IEEE International Conference, HSNMC 2003, Estoril, Portugal, July 23-25, Proceedings*: Springer-Verlag Heidelberg, 2003, 503-512.

48. Tulu, B., Chatterjee, S., and Laxminarayan, S. A Taxonomy of Telemedicine Efforts with respect to Applications, Infrastructure, Delivery Tools, Type of Setting and Purpose. *38th Annual Hawaii International Conference on System Sciences (HICSS- 38)*, Hawaii, USA: IEEE, 2005, 147b.

49. Tuunanen, T. Critical Success Chains Method. Helsinki, Finland: LTT - Tutkimus Oy, Elektronisen Kaupan Instituutti, 2001, 13.

50. Tuunanen, T. Mitä käyttäjät todella haluavat. *Tietoviikko*, Helsinki, Finland, 2002, 1.

51. Vaishnavi, V., and Kuechler, B. Design Research in Information Systems. Association for Information Systems, 2005 (available at http://www.isworld.org).

52. van Aken, J.E. Management research based on the paradigm of the design sciences: The quest for field-tested and grounded technological rules. *Journal of Management Studies*, 41, 2 (2004), 219-246.

53. van Aken, J.E. Valid knowledge for the professional design of large and complex design processes. *Design Studies*, 26, 4 (2005), 379-404.

54. Vandenbosch, B., and Higgins, C.A. Executive support systems and learning: A model and empirical test. *Journal of Management Information Systems*, 12, 2 (1995), 99-130.

55. Walls, J., Widmeyer, G., and El Sawy, O. Building an Information System Design Theory for Vigilant EIS. *Information Systems Research*, 3, 1 (1992), 36-59.

56. Walls, J., Widmeyer, G., and El Sawy, O. Assessing Information System Design Theory in Perspective: How Useful was our 1992 Initial Rendition? *Journal of Information Technology Theory & Application (JITTA)*, 6, 2 (2004), 43-58.

Table 1. Design and Design Science Process Elements from IS and Other Disciplines and Synthesis Elements for a DS Research Methodology in IS

Common design process elements	**Archer [2]**	**Takeda, Veerkamp, Tomiyama, and Yoshikawam [46]**	**Eekels and Roozenburg [14]**	**Nunamaker, Chen, and Purdin [33]**	**Walls, Widmeyer, and El Sawy [55]**	**Rossi and Sein; Cole, Purao, Rossi, and Sein [10, 40]**	**Hevner, March, and Park [20]**
1. Problem identification and motivation	Programming Data collection	Problem enumeration	Analysis	Construct a conceptual framework	Meta-requirements Kernel theories	Identify a need	Important and relevant problems
2. Objectives of a solution			Requirements				Implicit in "relevance"
3. Design and development	Analysis Synthesis Development	Suggestion Development	Synthesis, Tentative design proposals	Develop a system architecture Analyze and design the system. Build the system	Design method Meta design	Build	Iterative search process Artifact
4. Demonstration			Simulation, Conditional prediction	Experiment, observe, and evaluate the system			
5. Evaluation		Confirmatory evaluation	Evaluation, Decision, Definite design		Testable design process/product hypotheses	Evaluate	Evaluate
6. Communication	Communication						Communication

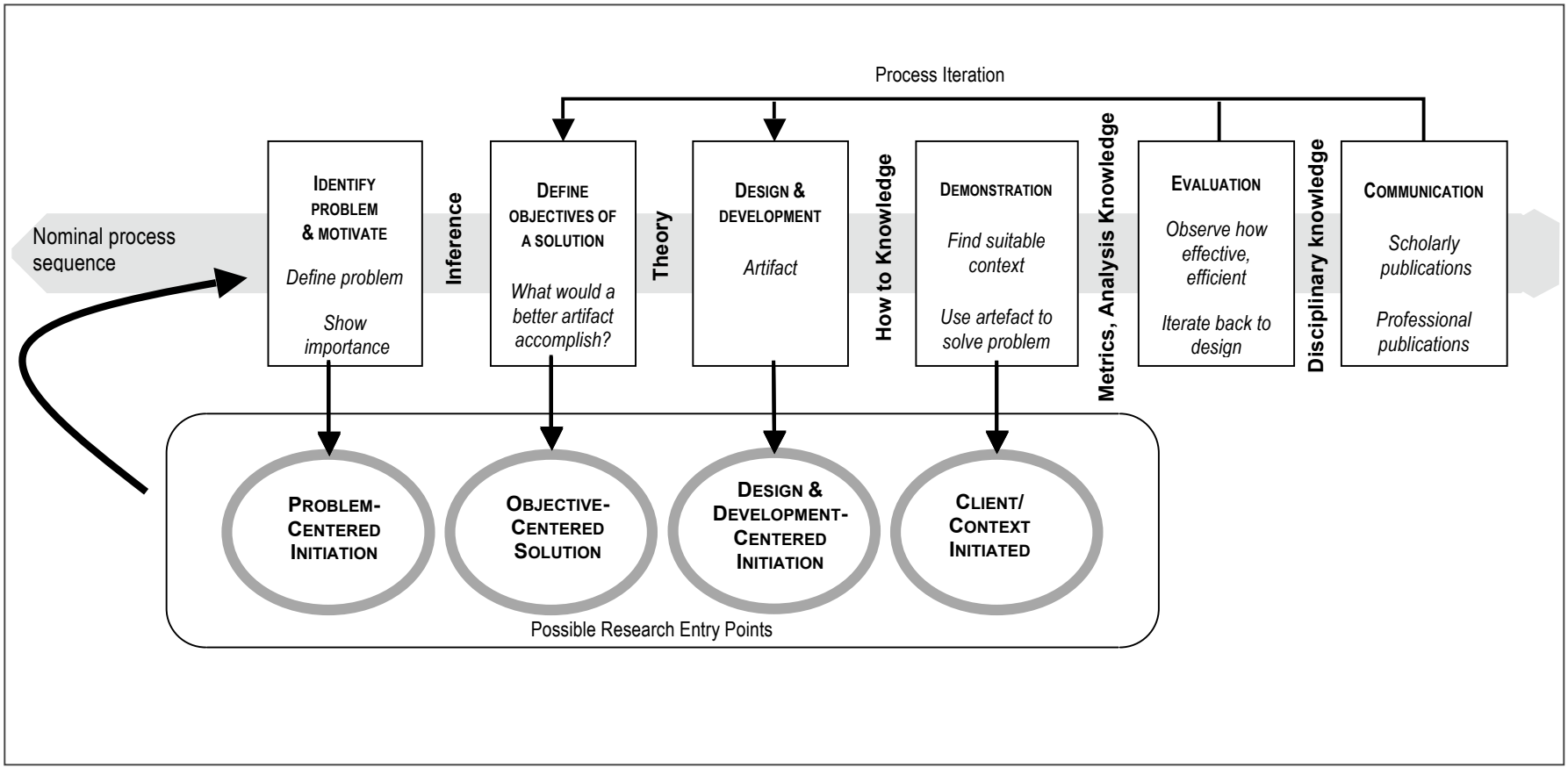


Figure 1. **Design Science Research Methodology (DSRM) Process Model**

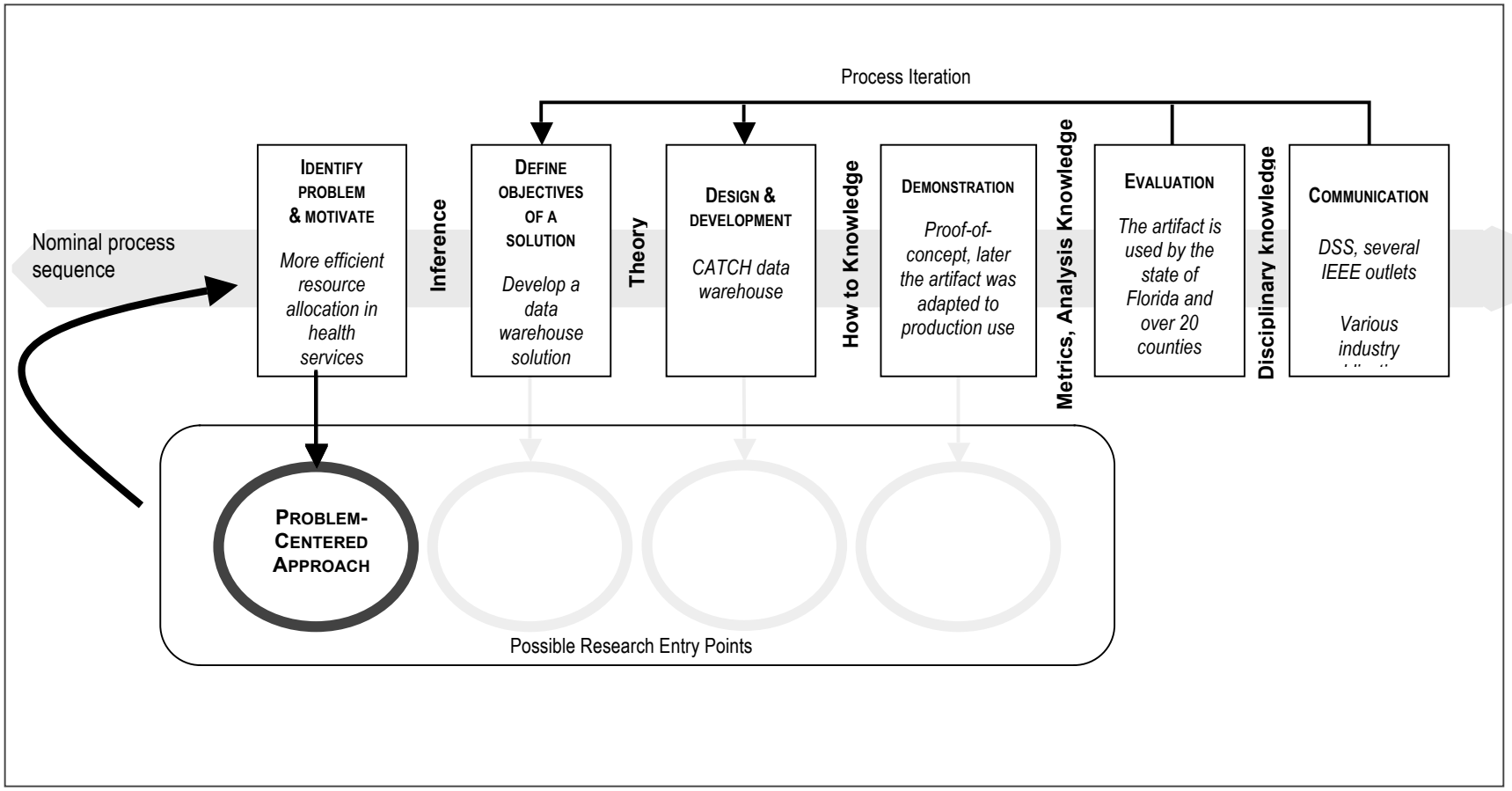


Figure 2. **DSRM Process for the CATCH Project**

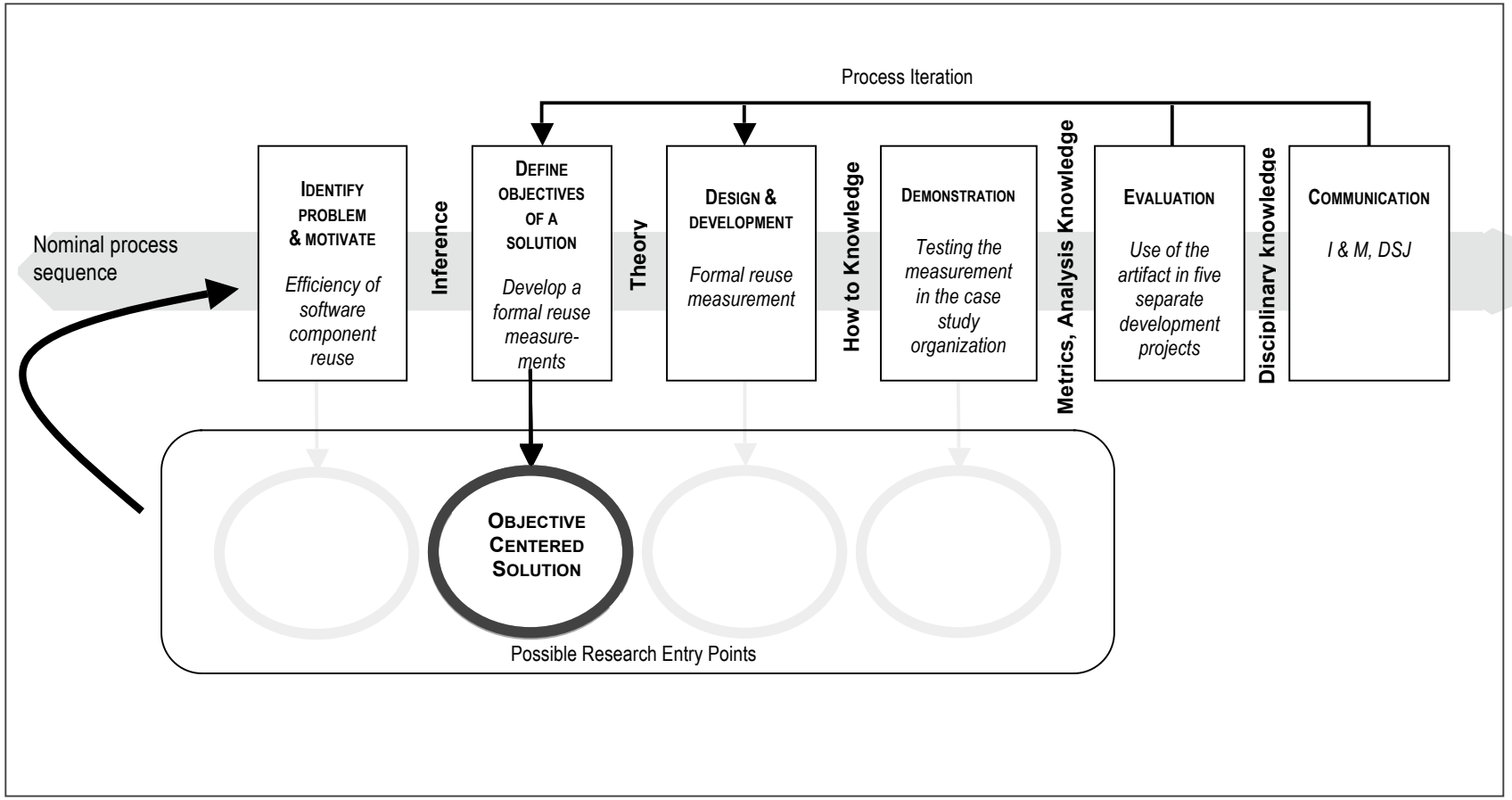


Figure 3. **DSRM Process for the MBA Technologies Study**

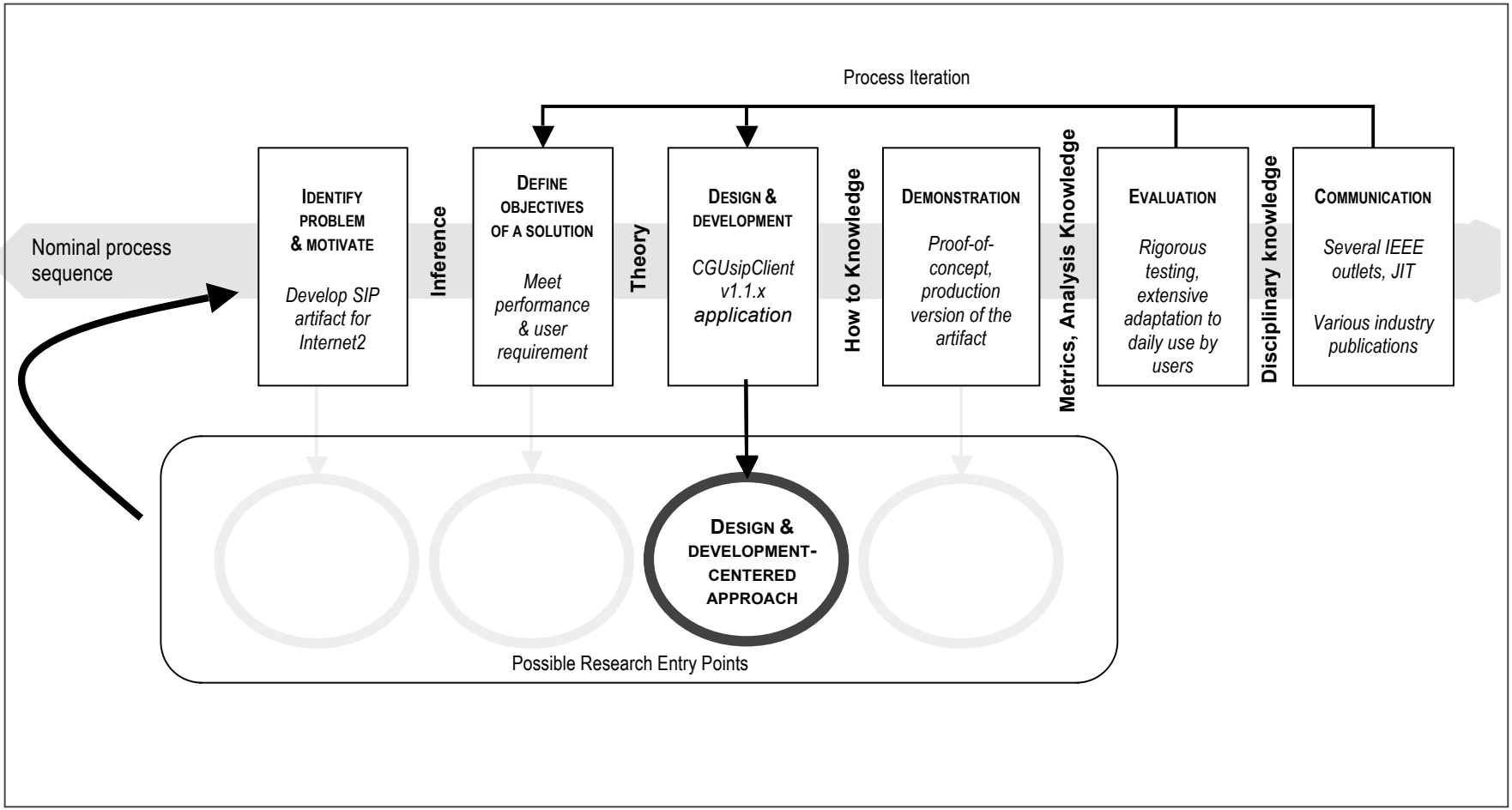


Figure 4. **DSRM Process for the CGUsipClient v1.1.x Project**

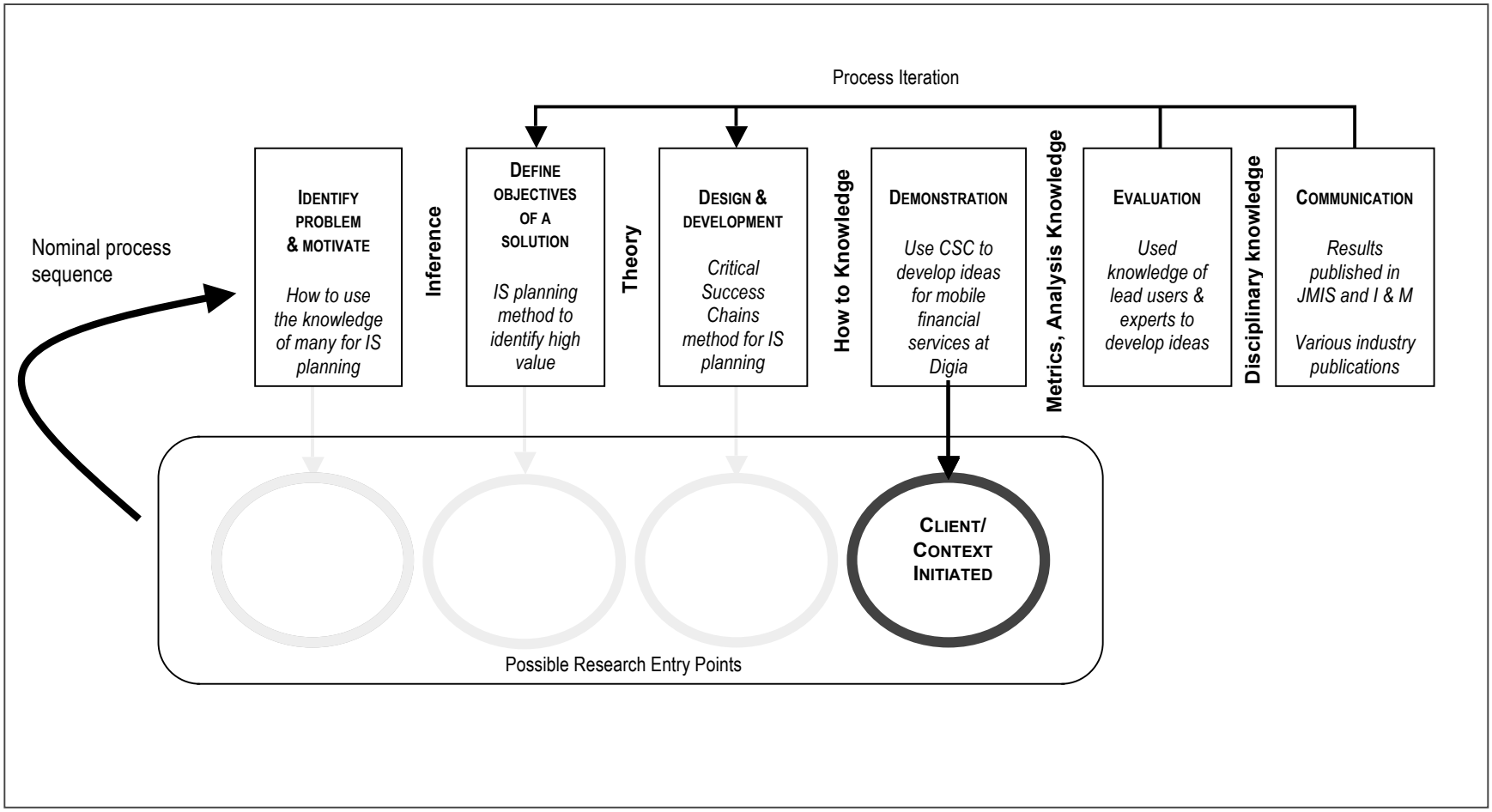


Figure 5. **DSRM Process for the Digia Study**