Accès Unique à des Ressources Numériques Distribuées.

Saïd Oulahal
INRIA | MSH-Alpes
Zirst – 655 avenue de l'Europe | BP47
38334 Saint Ismier Cedex | 38040 Grenoble cedex
said.oulahal@inrialpes.fr | said.oulahal@msh-alpes.prd.fr

en collaboration avec Laurence Raphaël
MSH-Alpes
BP 47
38040 Grenoble cedex
laurence.raphael@msh-alpes.prd.fr

Résumé

L'un des enjeux principaux aujourd'hui en informatique documentaire est l'interopérabilité. En effet l'accès aux données numériques se heurte à deux difficultés majeures : la multi-localisation des données, leur hétérogénéité. L'objectif est de permettre aux utilisateurs un accès unifié et simple à des services distants sans en connaître forcément tous les détails techniques.
Les protocoles Z39.50 et OAI-PMH (Open Archives Initiative) tentent de répondre à cette problématique en permettant l'accès à des données réparties via une interrogation simultanée. Ces deux solutions sont actuellement mises en œuvre au sein du réseau des Maisons des Sciences de l'Homme pour interconnecter leurs ressources documentaires numériques.

Mots clefs

Z39.50, ASN.1, BER, OAI-PMH, XML, HTTP, Dublin Core, CQL, SOAP, SRW, ez3950, Zee-rex, ZOOM, XER, WSDL

1 Introduction

Le Deep Web ou Web invisible est la "partie immergée du Web". L'information "invisible" (cad non-indexée par les moteurs de recherche classiques) serait 300 fois plus importante en volume que le WEB "visible". La qualité des résultats serait 1000 fois supérieure à une recherche sur le WEB avec des moteurs généralistes. Font ainsi partie du WEB invisible les notices bibliographiques indexées dans les bases bibliographiques (celle de la BNF et de bien d'autres), des textes (rapports, articles) disponibles en lignes dans le cadre des "Open Archives" (cf. infra).
Le Deep Web représenterait 550 milliards de pages dont 500 milliards échapperaient totalement au travail d'indexation des moteurs de recherche (d'après Transfert, n°17, septembre 2001, p. 33) [1]. Il y aurait 100 000 bases de données accessibles, dont 95 % accessibles gratuitement. Ces bases spécialisées représenteraient 54 % du WEB invisible. Les 60 plus grosses bases de données accessibles représenteraient à elles seules 40 fois plus d'informations que la totalité du WEB indexé par les moteurs de recherche [2], avec très souvent des interfaces ou méthodes d'interrogation disparates. Ces chiffres à eux seuls nous montrent l'importance de la mise en place de protocoles pour l'interrogation simultanée et unifiée de cette masse de données cachées et réparties entre des services distants.
L'objectif de cette présentation est d'offrir une vue d'ensemble des protocoles Z39.50 et OAI-PMH qui tentent de répondre à cette problématique.

2 Z39.50

2.1 Définition

Le protocole Z39.50 est une norme d'interrogation issue du domaine de la documentation bibliographique. A la différence des moteurs de recherches sur le WEB qui n'offrent qu'une recherche sur un ensemble de mots, le protocole Z39.50 permet une spécification des champs tel que le nom d'auteur, ISBN,...
Le but de cette norme est de permettre une interrogation simultanée de bases de données hétérogènes et réparties, sans connaître la structure des données dans les bases interrogées et d'effectuer une interrogation simultanée de plusieurs bases

de données par une requête unique, avec la récupération de données sur sa machine dans un format normalisé pour une ré-exploitation.

2.2 Historique

Le protocole Z39.50 est une norme (mise au point en 1981) destinée au repérage de l'information. On la connaît officiellement sous la dénomination ANSI/NISO Z39.50 -- Information Retrieval (Z39.50) : Application Service Definition and Protocol Specification [3] depuis 1992.
Ce document [4, 5] précise un jeu de règles et de procédures concernant deux systèmes qui communiquent en vue de la recherche et du repérage de l'information dans une base de données. En tant que norme d'application de réseau, Z39.50 est une norme ouverte qui permet la communication entre des systèmes qui tournent sur des plates-formes hétérogènes et utilisent des logiciels différents. Elle est gérée par la « Z39-50 maintenance Agency » hébergée à la Bibliothèque du Congrès [6] et développée par le ZIG (Z39-50 Implementors Group) [7].
La version 3 a été approuvée en 1995 par le National Information Standards Organization (NISO), la seule organisation accréditée par l'American Standards Institute (ANSI) pour approuver et tenir à jour cette norme. Actuellement, une nouvelle révision est en cours d'étude [6]. La norme Z39.50 est aussi reconnue mondialement depuis 1997 sous la dénomination ISO 23950 de l'Organisation internationale de normalisation (ISO).
Cette norme était aussi connue sous le nom de WAIS (basé sur la version 1, Z39.50-1988). Le protocole WAIS n'est plus maintenu aujourd'hui et il est incompatible avec le protocole Z39.50 depuis la version 2 de ce dernier (1995).

2.3 Architecture

Le système repose sur une architecture client-serveur.
Pour la norme Z39.50, le client, appelé «origin», constitue la partie du système local qui exécute toutes les fonctions de communication intervenant dans le lancement d'une recherche, la transmission de l'interrogation et la demande de notices en réponse à cette interrogation. Le serveur Z39.50 est la partie appelée «target». Il assure l'interface avec la base de données du système éloigné et répond aux messages qu'il reçoit du système d'origine, en transmettant par exemple des notices répondant à l'interrogation posée.

2.4 Communication

Le protocole Z39.50 se situe au niveau de la couche application suivant le modèle ISO. Le protocole de communication de données (Protocol Data Unit) est spécifié selon la notation de syntaxe abstraite ASN.1 (ISO8824) [9 et 10]. L'ensemble des données sont échangées après une sérialisation sous forme binaire suivant la norme Basic Encoding Rules (ISO8825).
L'utilisation d'ASN.1 permet de porter le protocole Z39.50 sur différentes plates-formes et sous différents langages (en fonction de la disponibilité du compilateur adéquat).
La communication entre le client et le serveur s'effectue via une Z39.50-Association (Z-association) suivant une session de type A-association. Ces associations sont elles-mêmes définies suivant une ACSE (Association Control Service Element) (ISO 8649) [8].
La Z-association détermine un certain nombre de règles pour le bon déroulement de la communication :

- Une communication est explicitement établie par le client et explicitement terminée par le client ou le serveur.
- On peut établir plusieurs Z-associations pendant une A-association.
- Une Z-association ne peut pas être relancée.
- Z-association ne permet pas une inversion de rôle entre le client et le serveur.
- Pendant une Z-association, il peut être créé autant de fois de « ResultSet » (nom d'ensemble de résultats sur le serveur).
- En fin de session (Z-association) le serveur détruit toute information relative au client (ResultSet).

2.5 Commande

On parle plutôt de service dans le cas de la norme Z39.50.
Les informations listées se réfèrent à la version 3 de la norme, la version 4 étant en cours de finalisation
La norme Z39.50 possède 14 services, dont trois services de base.
Services de base :

Init — Service d'initialisation lancé par le client. Il permet au client de prendre connaissance du profil du serveur et de lui communiquer ses propres paramètres. Pendant cette opération le client et le serveur négocient le niveau de communication. Pour garder une compatibilité ascendante, il y a échange de niveau de version. En cas de différence de version, les systèmes basculent sur la version la plus faible (exemple serveur en version 3 et client en version 2 : basculement du serveur en version 2).
Tous les services ne sont pas forcément disponibles et ne sont pas tous obligatoires (sauf les services

de base). C'est donc aussi pendant cette phase que le serveur indique au client la liste des services qu'il peut rendre.

Search Ce service permet au client d'interroger une ou des bases de données du serveur et de recevoir des informations concernant le résultat de sa question. Le résultat peut être de forme bref (**B**ref) ou complet (**F**ull). Cette option permet de réduire le flot de données en phase de recherche.

Present Ce service permet au client de demander les enregistrements correspondants à sa question sous la forme désirée (présentation, nombre, format).

Services secondaires :

Segment	Ce service est utilisé si les enregistrements retournés doivent être segmentés. Cette fonction est utile dans le cas de réseau à faible débit.
Delete	Permet au client de demander au serveur de détruire un ou plusieurs jeu de résultats (ResultSet). C'est ce qui permet d'établir plusieurs Z-association.
Access control	Permet au serveur d'effectuer à tout moment des contrôles d'accès sur le client.
Scan	Permet au client de balayer un index à partir d'une clé de recherche, sur une ou plusieurs bases de données (sélection des bases, identique au service Search).
Sort	Tri et fusion d'un ou plusieurs résultats de recherches précédentes.
Extended services	Sauvegarder un ensemble de résultats. Sauvegarder une recherche. Définir une recherche périodique. Commander un item. Mettre à jour une base de données. Créer une spécification d'importation.
Duplicate detection	Demande de dédoublonnage
Resource control	Contrôle des ressources effectué par le serveur (crédit du client, comptage des requêtes, etc…)
Trigger resource control	Client peut déclencher une demande de "Resource control"
Resource report	Etablissement par le serveur d'un rapport sur les ressources (suite du "resource control")
Close	Fin de la session.

2.6 Objet de communication

Le protocole est structuré en différent objets constitués eux-mêmes de classe et sous-classe.
Objet Attribut Set : ATR
La norme Z39.50 définit un ensemble de classe d'attribut (Z-39-50-attributSet x). Les attributs sont des éléments de relations entre une valeur littérale et une valeur numérique (EX : titre=1003). Ils sont utilisés pour établir les nom de champs de recherche (champ textuel). C'est par ce mécanisme que le serveur établit une relation entre un champ d'indexation (Nom d'auteur, Titre,...) et un champ au sein de la base de données ou du fichier contenant l'information.
Il y a 6 classes d'attribut :

Bib-1	Bibliographic-1 attribut set. A été développé pour les ressources de type bibliographiques. C'est un des jeux d'attributs le plus riche en terme de sous-classes (forte spécification des recherches), puisqu'il compte 6 autres sous-classes, qui vont permettre d'affiner la requête : attribut d'usage. attribut de relation. attribut de position. attribut de structure. attribut de troncature. attribut de complétude.
Exp-1	Explain attribut set
Ext-1	Extended services attribut set
CCL-1	Common Commande Language attribut set
GILS	Government information locator service
STAS	Scientific and Technical

Objet Gestion des erreurs :
Pour assurer une parfaite communication entre le client et le serveur, la norme définit 2 classes de gestion d'erreur (Z39-50-diagnostic x).

bib-1
diag-1

Objet Notice :
Le format de retour des notices (résultat des recherches) est défini suivant 21 classes (Z39-50-recordSyntax x). Les 15 premières classes sont issues de la norme MARC et adaptées à un pays (ex : Unimarc pour la France). Elles ne sont pas décrites en ASN.1 à la différence des 6 dernières classes. Cette différence fait que dans le premier cas, on reçoit une notice sous forme de données ISO2709 [11] et, dans le dernier cas, sous forme variable (spécifiée au niveau du client, par exemple sous forme de donnée XML).

Pour les autres objets, voir les appendices de la norme.

2.7 Requête

Le protocole propose 6 formats de requête :

type 0	Format libre, propre au système.
type 1	Z39.50, notation polonaise inversée (RPN). C'est le mode par défaut, tout serveur Z39.50 doit supporter ce type. Format RPN : (term1, term2, opérateur). Exemple de requête : Find @attrset bib-1 @attr 1=1003 martin @attr1=4 @attr 5=1 computer @and. Cela donne une recherche sur l'auteur (1=1003) « martin » **et** le titre (1=4) « computer* », la troncature à droite (*) est réalisée par la condition @attr 5=1, tout cela dans attribut de type bib-1.
type 2	ISO 8777, Common Command Language (CCL), interactive text searching.
type 100	ANSI Z39.58, CCL for online interactive information retrieval. Le format CCL permet de spécifier directement dans la requête l'attribut de recherche. AU=dupond -> recherche avec dupond comme nom d'auteur. C'est un format de requête plus naturel.
type 101	notation polonaise inversée étendue (ERPN).
type 102	Ranked list Query.

2.8 Solution de mise en œuvre

Le site d'indexdata [12 , 13] propose un ensemble d'outils pour mettre en place des serveurs et clients Z39.50.
Pour la partie serveur, le logiciel ZEBRA (*à ne pas confondre avec le routeur de même nom*) est un des plus performants. Il est composé d'une partie indexation et d'un serveur. Depuis la version 1.3.1, il est complètement sous licence GPL. Il ne comporte plus de restriction au niveau du nombre d'enregistrements inclus dans une base. Il peut indexer autant des bases sous forme ISO2709 que des bases sous forme de fichiers XML ou de texte taggé. Dans le cadre du Réseau des MSH (Maison des Sciences de l'Homme), il est utilisé pour 14 bases de données contenant environ 520 300 notices. L'ensemble des outils (client et serveur) repose sur l'utilisation d'une librairie YAZ [12]. Cette librairie gère la partie communication. Elle est issue de la compilation des spécifications écrites en ASN.1.

La mise en œuvre peut être réalisée de deux manières distinctes au niveau utilisateur.

1. Utilisation d'un client Z39-50 autonome.
   Le client effectue directement une connexion sur le serveur (Figure 1).
   Dans ce mode d'utilisation, on trouve essentiellement des outils de commerce tels que EndNote [14].

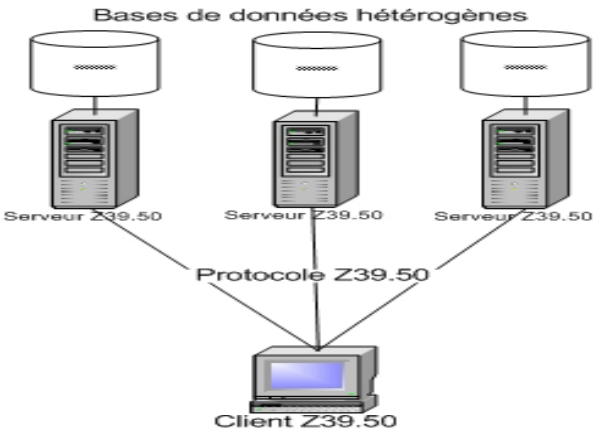


Figure 1 – *Client autonome.*

2. Utilisation d'une passerelle

Dans ce mode d'utilisation, l'utilisateur passe par une interface WEB. La gestion des connexions (mode connecté) est réalisée par le serveur de la passerelle (Figure 2).
Il existe plusieurs outils sous licence GPL.
ZAP ! [12] est un module pour le serveur APACHE. Il permet de réaliser rapidement des interfaces d'interrogation, avec une interrogation multi-serveurs.
PHP/YAZ [12] comporte un ensemble de fonctions pour réaliser des requêtes sur des serveurs Z39.50. Il ne sait pas réaliser d'interrogation multi serveurs.

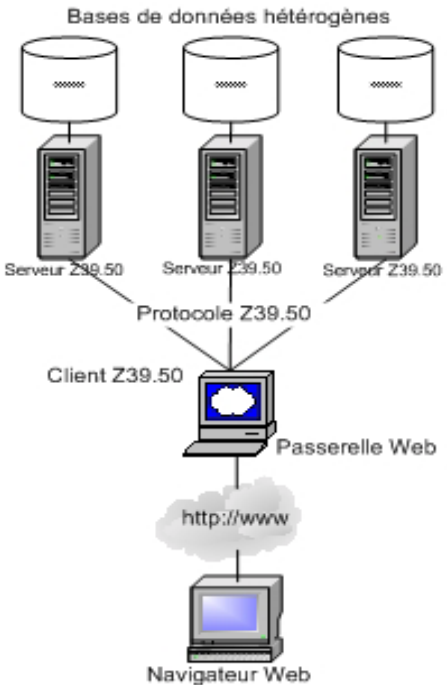


Figure 2 – *Passerelle.*

Dans le cas du réseau des MSH [15] (Figure 3), nous avons réalisé une interface d'interrogation multi-bases. Cette interface repose sur un module développé en java (et JSP) et sur ZAP !.
Le module ZAP ! assure la connexion avec les serveurs Z39.50. Cette solution offre l'avantage de ne pas avoir à réécrire le protocole de communication. Il transmet les requêtes (formatées par la partie en java) de l'utilisateur et renvoie les résultats sous forme de fichier XML. La réception en mode XML permet de construire un ensemble d'objet de type « notice », après parsing du fichier XML. Ces objets sont par la suite traités en fonction de certains champs et conservés à la demande de l'utilisateur dans un panier pour une importation ultérieure.

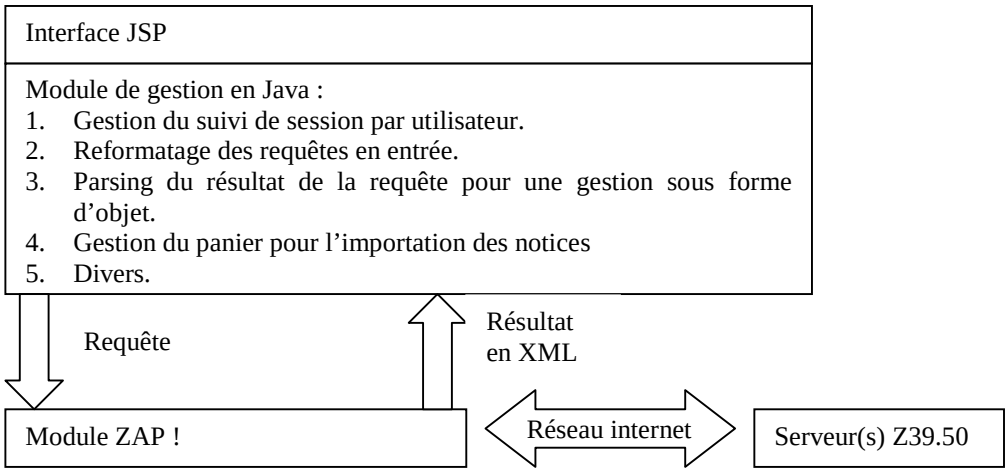


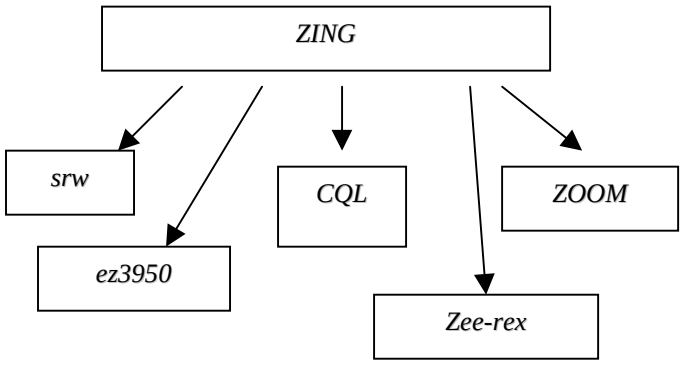
Figure 3 – *JSP/ZAP*

2.9 Evolution de la norme.

Aujourd'hui le protocole Z39.50 se heurte à quelques difficultés de développement :

- Il ne fonctionne qu'en mode connecté, d'où un conflit lors de l'utilisation via un serveur WEB (mode non connecté).
- Il ne dispose pas d'outil de localisation des serveurs actifs.
- Il ne supporte pas l'UNICODE.

Le groupe ZING (Z39.50-international: Next Generation) [16] a été crée pour rendre la norme conforme au contexte technologique actuel (WEB, XML, Web Services). Cette évolution doit aussi faciliter le travail des développeurs par une réduction des coûts et passer d'une logique "bibliothèque à bibliothèque" (BtoB) à une logique "utilisateur à bibliothèque" (CtoB). Il est en charge pour cela de cinq axes de recherche.



2.9.1 CQL

CQL (Common Query Language) est un langage formel de représentation de questions aux systèmes de recherche.
Le but de CQL est de combiner la puissance des langues expressives telles que SQL à l'intuitivité et la simplicité des langues comme CCL ou celle utilisée par les moteurs de recherche du WEB (google, …). Et ceci pour avoir un langage qui soit simple tout en fournissant les moyens d'exprimer des concepts plus complexes [17].

2.9.2 SRW/U

SRW (Search and Retrieve Web Service) est un protocole servant à intégrer l'accès à diverses bases de données. Il utilise les mécanismes SOAP et le langage CQL pour la formulation des requêtes. Il fournit la sémantique pour rechercher des bases de données contenant des métadonnées, c'est à ce niveau que le SRU (Search and Retrieve URL Service) entre en jeu. Les échanges sont en XML et codés en XER [33]. Les services du protocole Z39.50 sont repris sous forme de méthode (dans le sens Web Service).

2.9.3 Ez3950

L'intérêt principal de Ez3950 est de permettre l'intégration de XML dans le protocole Z39.50 au niveau du transport des données, et ceci sans modifier le protocole. Ainsi toutes les spécifications ASN.1 qui font actuellement Z39.50 peuvent être passées en WSDL sans grande difficulté.
Une description WSDL complète doit comporter cinq parties :

- **Types** : spécifient les structures de données utilisées pour l'échange entre le client et le serveur. WSDL supporte XSD comme langage de description de type. Une version XSD de ASN.1 peut être générée automatiquement en utilisant ASN.1 avec XER (XML Encoding Rules) plutôt que BER (actuellement utilisé).
- **Messages :** En termes de Z39.50 ce serait une spécification des messages de requête et de réponse.
- **Operations:** WSDL supporte quatre types d'opérations : One-way Operation, Request-response Operation, Solicit-response Operation et Notification Operation. Trois d'entre elles sont utilisées par Z39.50 : Request-response, Solicit-reponse et Notification.
- **Bindings :** définit quels ports le serveur écoute, et pour chacun le protocole de transport utilisé (HTTP par défaut, SOAP, etc.).
- **Services :** décrit l'implémentation d'un port en écoute, fournit des informations sur le serveur, etc.

2.9.4 Zee-rex

ZeeRex est un projet d'évolution du protocole Z39.50 utilisant XML et qui tente de palier certains problèmes de cette norme :

- Localisation : le but est de trouver les ressources Z39.50 (Serveurs et bases de données) pour un client.
- Profile du serveur et de ces bases : une fois la ressource connue, il est nécessaire de connaître ce dont elle est capable.

Fondamentalement, Zeerex se base sur le principe de l'échange d'enregistrements concernant les bases de données qui sont généralement consultées par l'intermédiaire de Z39.50.
Actuellement, la recherche et l'indexation des serveurs disponibles sont réalisées manuellement. Cette méthode doit faire face à deux grandes difficultés :

- Il est difficile de savoir combien de serveurs se trouvent sur Internet
- Une telle liste de serveurs est difficile à maintenir à jour. Il est en effet possible d'automatiser le processus de recherche des serveurs morts de la liste et de les ôter, mais il est plus difficile d'automatiser le processus d'ajout de nouveaux serveurs puisqu'il n'y a aucune façon de récupérer automatiquement les renseignements à leur sujet.

En 2000, le ZIG a proposé un annuaire distribué des serveurs Z3950 d'appellation 'Friends and Neighbours' (amis et voisins).
C'est ce principe qu'utilise Zeerex pour maintenir à jour sa liste de serveurs. Le principe, simple, fonctionne comme Google.com (principe de la chenille). Chaque serveur participant contient un enregistrement listant tous les serveurs surs qu'il connaît. Zeerex prend connaissance de cette liste et peut aller visiter chacun de ces serveurs pour les référencer.

2.9.5 ZOOM

Une des complexités de la norme Z39-50 est la phase de développement d'outils. Elle spécifie les services à rendre, mais ne propose aucune API standard. Le but du projet ZOOM est d'offrir une API abstraite disponible en de multiples langages (C, C++, Java, Python, …). Cette API ne spécifie toutefois que les services de base.

3 OAI

3.1 Définition

Une " archive ouverte " est la traduction littérale d'Open Archive. Si le terme " archive " est communément employé pour désigner la préservation de documents sur une longue durée et par extension la conservation de documents "anciens", ici il désigne des entrepôts d'informations contenant des documents numériques, quelle que soit leur antériorité. Le terme " open " a pour sens, dans ce contexte, l'ouverture de l'architecture technique de ces entrepôts d'informations par la mise en place de protocoles communs qui facilitent l'accès au contenu par un ensemble de services extérieurs. Une archive ouverte peut donc se définir simplement comme un réservoir d'objets numériques. Apparue pour répondre aux besoins particuliers des chercheurs, l'OAI introduit aussi la question de la gestion des étapes de révision d'un document et donc du maintien sur un même serveur de plusieurs documents retraçant les différentes étapes d'élaboration d'un document scientifique [18].

3.2 Historique

L'Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) [19] est un protocole qui a été lancé en 1999 lors de la Convention de Santa-Fé qui rassemblait plusieurs coalitions de bibliothèques et institutions académiques américaines. Ce protocole est géré et développé par l'organisation du même nom. L'OAI-PMH rend toutes les archives qui sont « OAIcompliant » (cad conforme à l'OAI) interopérables entre elles. Il est né d'un désir de la communauté scientifique de se doter d'un outil d'auto-archivage permettant une meilleure accessibilité aux publications scientifiques et techniques [20].

3.3 Architecture

Le protocole OAI-PMH se base sur 2 étapes, une phase de dépôt et une phase de collecte de métadonnées[1].
L'enregistrement du document est réalisé par le dépositaire (individu ou organisme), il s'appuie sur la constitution de métadonnées (au minimum de type Dublin Core, mais aussi Marc21, DTD spécifique, …) pour l'indexation du document et son identification. La constitution des bases d'archives se fait sans outil spécifique mais simplement via une interface WEB, par laquelle le document est déposé dans un entrepôt (data provider). Un fournisseur de services (service provider) permet à l'utilisateur, via une interface de consultation des entrepôts, l'accès à différentes bases d'archives en une seule interrogation. Cette consultation est soit réalisée à la volée en direction des différents entrepôts soit après une phase de collecte des métadonnées grâce à un "moissonneur" (Harvester). Le moissonneur est un outil de collecte de métadonnées. Il met à jour une base de donnée locale, à intervalle régulier, en allant collecter ces métadonnées depuis un ensemble d'entrepôt (serveur OAI). C'est en somme un cache WEB (serveur proxy).
L'ensemble du protocole OAI (dépôt et consultation) repose sur le protocole HTTP 1.1 [21] (mode non-connecté). Cette norme autorise une interrogation sous forme de mots-clés avec un retour de données structurées au format XML.
L'OAI s'applique à tous types de documents et supports (articles, thèses, rapports de stage, vidéos, images, sons, …).

1 Les métadonnées sont des éléments d'information sur d'autres éléments d'information. Le préfixe « méta », qui vient du grec, signifie littéralement « avec » ou « après »; utilisé dans d'autres termes comme « métaphysique » ou « métalangue », il signifie « ce qui dépasse, englobe ». Dans les normes Z39.50 et OAI, les métadonnées désignent expressément des éléments d'information à propos d'une ressource, par exemple le nom du créateur et la date de création.

La norme minimale d'échange exigée par le protocole OAI pour spécifier les métadonnées est la norme Dublin Core (sans qualificatifs) [22] (schéma XML pour la version simple de DC [23]). Cependant, pour fournir une description plus détaillée, on peut employer tout autre ensemble de métadonnées décrit par une DTD [24] de XML.
L'OAI-PMH ne spécifie aucun service (ou outil) à la différence du protocole Z39.50. Il n'offre que 6 types de requête pour effectuer l'ensemble des traitements relatifs au dialogue entre « data providers » et « services providers », avec pour chaque réponse une validation du résultat par un schéma XML. Cette architecture lui permet d'être facilement intégrable au sein de service spécifique.

3.4 Entrepôt OAI

Les entrepôts OAI sont constitués de fiches d'informations (principe des notices du protocole Z39.50) et des documents que l'on désire archiver.
Une fiche au format XML est constituée elle-même de trois parties [25] :

Entête (header)	Constituée de trois parties
	Identifiant unique (identifier) de référencement de cette ressource.
	Date (datestamp) indique la date de création, de modification ou d'effacement de la ressource.
	SetSpec indique la spécialisation du document. Ce champ peut avoir plusieurs valeurs.

Métadata	Structurée selon le format Dublin Core (DC est le format minimal) ou tout autre format défini par une DTD. Par principe le nom du format débute par « oai_ ». Un ensemble de namespace réalise un lien vers la spécification des métadonnées. Dans la description des métadonnées, on retrouve aussi la localisation du document original (ex : <dc :identifier>).

A propos (about)	Autres éléments d'informations.

Exemple de fiche :

```
<header>
        <identifier>oai:arXiv.org:cs/0112017</identifier>
        <datestamp>2002-02-28</datestamp>
        <setSpec>cs</setSpec>
        <setSpec>math</setSpec>
</header>
<metadata>
        <oai_dc:dc
                xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
                xmlns:dc="http://purl.org/dc/elements/1.1/"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
                http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
                <dc:title>Using Structural Metadata to Localize Experience of Digital Content</dc:title>
                <dc:creator>Dushay, Naomi</dc:creator>
                <dc:subject>Digital Libraries</dc:subject>
                <dc:description>With the increasing technical ..... </dc:description>
                <dc:description>Comment: 23 pages including, 8 figures</dc:description>
                <dc:date>2001-12-14</dc:date>
                <dc:type>e-print</dc:type>
                <dc:identifier>http://arXiv.org/abs/cs/0112017</dc:identifier>
        </oai_dc:dc>
</metadata>
<about>
        <provenance
```

```
                xmlns="http://www.openarchives.org/OAI/2.0/provenance"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/provenance
                http://www.openarchives.org/OAI/2.0/provenance.xsd">
                <originDescription harvestDate="2002-02-02T14:10:02Z" altered="true">
                        <baseURL>http://the.oa.org</baseURL>
                        <identifier>oai:r2.org:klik001</identifier>
                        <datestamp>2002-01-01</datestamp>
                        <metadataNamespace>http://www.openarchives.org/OAI/2.0/oai_dc/</metadataNamespace>
                </originDescription>
        </provenance>
</about>
```

3.5 Dublin Core

3.5.1 La norme

La norme de métadonnées du Dublin Core représente un ensemble d'éléments simples mais efficaces pour décrire une grande variété de ressources en réseau. Elle comprend quinze éléments dont la sémantique a été établie par un consensus international de professionnels provenant de diverses disciplines. Conçu pour l'ensemble des ressources électroniques, le Dublin Core est très général, tous les éléments sont facultatifs et peuvent être répétés sans limitation. Afin de décrire des ressources et d'effectuer des recherches sur des éléments plus précis, les métadonnées initiales peuvent être subdivisées en sous-éléments plus détaillés par l'utilisation de "Qualificatifs" (on parle alors de Dublin Core qualifié).

Exemple:

dc.Coverage.informatique.coordinates
dc.Coverage.date

Largement préconisé pour retrouver les ressources disponibles sur l'Internet, dans les balises méta (ex : <META NAME= "DC.Creator" CONTENT="Martin">), le Dublin Core sert aussi de plus grand dénominateur commun pour connecter des ressources hétérogènes.

```
<html> <head>
                <title>DC in HTML</title>
                <link rel="schema.DC" href="http://purl.org/dc/elements/1.0/">
                <meta name="DC.Title" content="Recording qualified Dublin Core metadata in HTML">
                <meta name="DC.Description" content="Présentation de la norme  ... ">
                <meta name="DC.Creator" content="Martin Dupond">
                <meta name="DC.Contributor" content="… ">
                <meta name="DC.Subject" content="metadata, Dublin Core, OAI, Z39.50"><meta
                name="DC.Date" content="2003-09-09">
                [...]
        </head> …
```

3.5.2 Liste des éléments

Titre : TITLE
Créateur : CREATOR
Sujet : SUBJECT
Description : DESCRIPTION
Editeur : PUBLISHER
Contributeur : CONTRIBUTOR
Date : DATE
Type : TYPE
Format : FORMAT
Identifiant : IDENTIFIER
Source : SOURCE
Langue : LANGUAGE
Relation : RELATION
Couverture : COVERAGE
Droits : RIGHTS

3.6 Protocole de communication

Il est basé sur le protocole HTTP1.1. Les actions et leurs paramètres sont spécifiés par des paramètres ordinaires associés aux échanges HTTP, comme pour tout site WEB dynamique. Cette simplicité est l'une des explications du succès de l'OAI.

3.6.1 Requête de commande

Les requêtes utilisent les méthodes GET ou PUT. En règle générale la commande PUT est plus adaptée pour l'échange de longs messages. Une requête est constituée d'un verbe et d'un ensemble de paramètres (fonction du verbe). La requête est traitée par un programme côté serveur (sous forme de CGI-BIN en PERL ou tout autre langage, voir sous forme de servlet).
Format des requêtes: URL?verb=Commande&liste de mot clés.
Exemple: http://arXiv.org/oai2?verb=GetRecord&identifier=oai:arXiv.org:cs/0112017&metadataPrefix=oai_dc

3.6.2 Réponse

La réponse à une requête OAI est une réponse HTTP sous forme text/xml (Content-Type : text/xml).
Les données XML pour toutes les réponses ont la forme suivante :
1. la déclaration du fichier XML, avec comme information la version XML et l'encodage des caractères.
   < ?xml version="1.0" encoding="UTF-8" ?>
2. le reste du contenu est englobé dans un élément racine portant le même nom que la nature de la requête et un ensemble de namespace pour la spécification des tags.
   xmlns
   xmlns:xsi
   xsi:schemaLocation
3. pour chaque réponse, les deux premiers éléments de la racine XML sont :
   responseDate (« date locale de la réponse »)
   requestURL (« URL de la requête »)

Par exemple, ci-dessous, la réponse à la requête GetRecord:
Requête:
verb=GetRecord&identifier=oai:arXiv.org:quant-ph/02131001&metadataPrefix=oai_dc Réponse:

```
< ?xml version="1.0" encoding="UTF-8" ?>
<OAI-PMH
        xmlns="http://www.openarchives.org/OAI/2.0/"
        xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
        xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
                http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
                <responseDate>2002-02-08T08:55:46Z</responseDate>
                <request verb="GetRecord" identifier="oai:arXiv.org:quant-ph/0213001"
                        metadataPrefix="oai_dc">http://arXiv.org/oai2</request>
                <error code="idDoesNotExist">No matching identifier in arXiv</error>
</OAI-PMH>
```

3.7 Gestion d'erreurs

La gestion des erreurs de communication est issue elle aussi du protocole HTTP (Status-Code) [21].
Une réponse normale est signalée par un type d'erreur : HTTP 200 Status-Code.
Lors d'une requête illégale (nature, argument incorrects), le type d'erreur retournée est par exemple : HTTP 400 Status-Code.

3.8 Gestion des flots de données

Deux requêtes OAI (ListRecords et ListIdentifiers) peuvent retourner une liste de données très longue. Dans ce cas, la réponse est scindée en une série de requêtes et réponses OAI. En pratique, cette division est effectuée par le serveur qui répond à une requête avec une liste incomplète et un jeton (resumptionToken). Afin d'obtenir une liste complète, le client devra renouveler une ou plusieurs requêtes avec en argument le jeton. La réponse complète est ainsi la combinaison des listes incomplètes retournées par le serveur. Les jetons ont une durée de vie limitée, qui est définie au niveau du serveur (principe de suivi de session ou du timeout d'une connexion Z39.50).

3.9 Paramètres

Un certain nombre (3) d'éléments permettent de passer des paramètres aux actions.

Datestamp :

correspond à la date d'insertion (création) du document dans la base des publications. Il peut être utilisé comme argument from ou until dans les requêtes ListRecords ou ListIdentifiers.

Set (« type de document ») :

permet de regrouper les documents par type dans la base des publications. Les types sont définis par le gestionnaire de la base. Les différents types de documents sont accessibles depuis le protocole OAI via la requête ListSets. Il peut être utilisé comme argument dans les requêtes ListRecords ou ListIdentifiers.

metadataPrefix:

permet de sélectionner une structure (utilisé comme argument dans les requêtes).

3.10 Actions (ou commandes)

Le protocole OAI [26] supporte 6 actions [27] qui sont exprimées sous la forme de verbes. Ces actions sont celles qui sont initiées par un moissonneur «service provider» ou tout autre application désireuse d'effectuer des requêtes en direction d'un entrepôt «data provider». Ce dernier doit réagir en envoyant une réponse à son interlocuteur.
Cette section liste les différentes requêtes définies dans le protocole OAI. Chaque sous-section correspond à la valeur possible du mot clé "verb", y sont également indiqués les différents arguments additionnels de la requête OAI. Ces arguments peuvent être : obligatoire, optionnel ou exclusif.

ListMetadataFormats

Action	Retourne la liste des formats de métadonnées disponibles pour un entrepôt.
Argument(s)	Identifier (« identifiant ») : OPTIONNEL.

ListIdentifiers

Action	Retourne la liste des identifiant de la base des publications, mais seulement avec quelques informations de gestion pour chacun : identifiant, date de dernière modification, ensembles auquel il appartient, etc.
Argument(s)	Until : OPTIONNEL, format année ('YYYY'). From : OPTIONNEL, format année ('YYYY'). set (« type de document ») : OPTIONNEL. ResumptionToken (« jeton ») : EXCLUSIF.

ListSets

Action	Liste les différents types de structures de données disponibles sur le serveur.
Argument(s)	Aucun.

Identify

Action	Retourne les informations de l'organisme de dépôt (serveur OAI). Un certain nombre d'informations doivent être retournées par l'entrepôt, dans un format précis, comme par exemple un nom, une URL de base, et certaines informations de gestion. L'entrepôt peut également y ajouter d'autres informations, à son gré.
Argument(s)	Aucun.

ListRecords

Action	Cette commande est semblable à ListIdentifiers, mais cette fois les métadonnées seront également retournées, pas seulement les informations de gestion. Il s'agit donc d'un moyen de récupérer tout ou partie d'un entrepôt OAI.
Argument(s)	until : OPTIONNEL, format année ('YYYY'). from : OPTIONNEL, format année ('YYYY'). set (« type de document ») : OPTIONNEL. ResumptionToken (« jeton ») : EXCLUSIF. MetadataPrefix (« format des données ») : OBLIGATOIRE.

GetRecord

Action	Retourne un enregistrement précis de la base des publications, à partir de son identifiant. Le format de données doit être également spécifié. Si l'entrepôt ne connaît pas le format, alors il retourne une erreur.
Argument(s)	Identifier : OBLIGATOIRE. MetadataPrefix : OBLIGATOIRE.

3.11 Solution de mise en œuvre

Il existe différents outils de mise en œuvre. Un des plus connus est le logiciel Eprints [28] pour la partie entrepôt. Le CCSD [29] qui est en charge du dépôt en ligne et de l'archivage pour le CNRS utilise Eprints lui aussi. Une description de l'installation d'Eprints se trouve sur le lien [37]. L'OAI propose aussi une interface de test d'un certain nombre de sites [30], ceux-ci s'étant 'auto-déclarés auprès de l'OAI.
D'autre outils existent et sont référencés sur le site de l'OAI [31]. En particulier, la MSH-Alpes, dans le cadre du Portail du réseau des MSH, a choisi le logiciel Dspace [34] développé par HP (Hewlett-Packard) et le MIT (Massachussett Institute of Technology) comme entrepôt de données. Celui-ci, outre sa conformité au protocole OAI-PMH, a pour objectif principal d'assurer la pérennité et la conservation des archives ainsi que la gestion du processus d'édition (workflow) des documents. Ce logiciel a été également retenu par l'INSERM et plusieurs grandes universités américaines. Une version française est en cours de réalisation à la MSH-Alpes.

3.11.1 Test Manuel

On peut réaliser un test manuel sur un serveur [32]. Dès la première connexion, le serveur renvoie un message d'erreur pour indiquer qu'il ne connaît pas la commande (action sur le serveur sans paramètre).

```
<OAI-PMH xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
    <responseDate>2003-09-23T09:31:55Z</responseDate>
    <request>http://diglib.lib.utk.edu/cgi/b/broker20/broker20</request>
    <error code="badVerb">Illegal OAI verb</error></OAI-PMH>
```

Puis on peut exécuter des commandes en construisant les requêtes à la main.
Exemple : Commande : http://diglib.lib.utk.edu/cgi/b/broker20/broker20?verb=ListMetadataFormats
Résultat :

```
<OAI-PMH
xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/http://www.openarchives.org/OAI/2.0/OAI-
PMH.xsd">
    <responseDate>2003-09-23T09:41:46Z</responseDate>
    <request verb="ListMetadataFormats">http://diglib.lib.utk.edu/cgi/b/broker20/broker20</request>
    <ListMetadataFormats>
        <metadataFormat>
            <metadataPrefix>oai_dc</metadataPrefix>
            <schema>http://www.openarchives.org/OAI/2.0/oai_dc.xsd</schema>
            <metadataNamespace>http://www.openarchives.org/OAI/2.0/oai_dc/</metadataNames
            pace>
        </metadataFormat>
    </ListMetadataFormats></OAI-PMH>
```

Commande : http://diglib.lib.utk.edu/cgi/b/broker20/broker20?verb=ListSets
Résultat :

```
<OAI-PMH
xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/http://www.openarchives.org/OAI/2.0/OAI-
PMH.xsd">
    <responseDate>2003-09-23T10:46:08Z</responseDate>
    <request verb="ListSets">http://diglib.lib.utk.edu/cgi/b/broker20/broker20</request>
    <ListSets>
        <set><setSpec>UTBib:etdbib</setSpec>
            <setName>Electronic Theses and Dissertations</setName></set>
        <set><setSpec>UTBib:fabib</setSpec>
            <setName>Finding Aids</setName></set>
        <set><setSpec>UTBib:jeibib</setSpec>
            <setName>Journal of Economic Issues</setName></set>
        <set><setSpec>UTBib:mpmbib</setSpec>
            <setName>Miscellaneous Published Materials</setName></set>
        <set><setSpec>UTBib:rthbib</setSpec>
            <setName>Roth Photograph Collection</setName></set>
        <set><setSpec>UTBib:tdhbib</setSpec>
            <setName>Tennessee Documentary History</setName></set>
    </ListSets></OAI-PMH>
```

4 Conclusion

Les divers groupes de réflexion autour de Z39.50 travaillent aujourd'hui sur la "modernisation" et la simplification de ce protocole à partir de ceux utilisés sur Internet (HTTP, XML). Comme il est déjà très intégré au fonctionnement des catalogues de bibliothèques, l'évolution de ce protocole lui permettra d'être encore plus en adéquation avec les besoins des chercheurs.

L'OAI-PMH est une réponse technique à un problème touchant particulièrement les institutions scientifiques . En effet, ce protocole essaie de répondre à l'amélioration de l'accès à l'information scientifique et technique (IST) et à sa diffusion. La première initiative autour de l'OAI est apparue autour du serveur de "pré-prints" de Los Alamos [35]. La plupart des chercheurs souhaitaient alors offrir une diffusion rapide et ouverte à tous de leurs résultats de recherche, la centralisation de ces documents sur un seul serveur permettait une recherche facilitée. La réponse proposée par OAI-PMH permet de garder une administration des documents sur les sites propres des institutions, tout en conservant la facilité de la recherche par l'intermédiaire des fournisseurs de services. L'OAI est aussi une tentative de réponse à une réflexion nécessaire sur les modes de diffusion de l' information scientifique et technique (IST). La prépondérance des éditeurs de revues scientifiques pose en effet questions aux institutions de recherche et une nouvelle forme d'économie liée à l'IST semble devoir être proposée. Actuellement, les institutions financent les recherches de leur collaborateurs, elles financent l'édition des résultats mais aussi la diffusion (via l'abonnement aux revues) de ces résultats auprès de leurs propres chercheurs. Un accès ouvert aux résultats de la recherche doit permettre une plus grande diffusion de ceux-ci principalement vers les pays qui n'ont pas actuellement les moyens financiers pour les recevoir. Cela ne veut pas dire que l'IST peut être totalement gratuite. Une première esquisse de réponse est tentée par l'expérimentation de nouvelles formes de revues électroniques proposée par le serveur BioMed Central [36] : les chercheurs proposent leurs articles pour être publiés dans une revue avec Comité de Lecture. Lorsqu'un article est accepté, le chercheur doit payer pour son édition mais la lecture de son article est gratuite . Comme le coût pour l'éditeur est le même pour chaque article qu'il soit publié ou non, la facture pour le chercheur donc l'article est accepté sera directement proportionnelle au nombre d'articles refusés. Par exemple, dans le cas d'une revue comme Nature où le nombre de refus serait de l'ordre de 95%, le coût de l'acceptation d'un article serait de l'ordre de 3000 €. Compte tenu de l'état de développement des dispositifs techniques d'édition électronique, et compte tenu du développement des pratiques dans les communautés scientifiques des sciences "exactes", il apparaît important de s'interroger sur ce que peuvent apporter ces développements techniques pour les communautés des sciences et de considérer quels sont les modèles éditoriaux (voire économiques) qui peuvent s'appliquer.

Références

[1] http://www.invisible-web.net/
[2] http://sicd1.ujf-grenoble.fr/Reseau_Documentaire/Bibliotheques/BUS/Services/Formations/doc_formation/_vti_cnf/ress_internet_doct.ppt
[3] http://www.niso.org/
[4] http://lcweb.loc.gov/z3950/agency/document.html
[5] AFNOR, ISO 23950 : Information et documentation. Recherche d'information (Z39.50) - Définition du service de l'application et spécification du protocole. 1998.
[6] http://lcweb.loc.gov/z3950/agency
[7] http://www.loc.gov/z3950/agency/zig/zig.html
[8] http://www.iso.ch/iso/fr/CatalogueDetailPage.CatalogueDetail?CSNUMBER=8649
[9] Olivier Dubuisson. ASN.1 – Communication entre systèmes hétérogènes. Springer Verlag, 1999.
[10] http://asn1.elibel.tm.fr
[11] http://bibliotheque.bgp-fr.com/Unimarc_decriptage-ISO2709.pdf
[12] http://www.indexdata.dk/
[13] http://lcweb.loc.gov/z3950/agency/resources/software.html
[14] http://www.msh-reseau.prd.fr/EspaceTravail/endnote.jsp
[15] http://www.msh-reseau.prd.fr/RessourcesDoc/ClientZap/index.jsp
[16] http://lcweb.loc.gov/z3950/agency/zing/
[17] http://zing.z3950.org/cql/intro.html
[18] http://www.urfist.jussieu.fr/urfist/archives-ouvertes.htm
[19] http://www.openarchives.org/
[20] http://www.openarchives.org/meetings/SantaFe1999/sfc_entry.htm
[21] http://www.w3.org/Protocols/rfc2616/rfc2616.html
[22] http://www.rcip.gc.ca/Francais/Normes/metadonnees_recherche.html#dublin_core
[23] http://dublincore.org/schemas/xmls/simpledc20020312.xsd
[24] http://www.rcip.gc.ca/Francais/Normes/glossaire.html#Document_Type_Definition
[25] http://sophia.univ-lyon2.fr/doc/eprints.org/
[26] http://www.openarchives.org/OAI/openarchivesprotocol.html
[27] http://documentation.in2p3.fr/oai/OAI-publi_in2p3.pdf
[28] http://software.eprints.org/
[29] http://ccsd.cnrs.fr/
[30] http://oai.dlib.vt.edu/cgi-bin/Explorer/2.0-1.45/testoai
[31] http://www.openarchives.org/tools/tools.html
[32] http://diglib.lib.utk.edu/cgi/b/broker20/broker20
[33] http://asf.gils.net/xer
[34] http://www.dspace.org/
[35] http://www.arXiv.org/
[36] http://www.biomedcentral.com/
[37] http://www.inist.fr/rencontresIST/interventions/tanguy3.pdf