Tutorium

Die Open Archive Initiative

Dr. Heinrich Stamerjohanns
Institute for Science Networking, Universität Oldenburg
stamer@uni-oldenburg.de
Dr. Bruno Klotz-Berendes
Universitätsbibliothek Dortmund
bruno.klotz-berendes@ub.uni-dortmund.de

Gliederung des Tutoriums

- Teil I - Geschichte und Überblick
- Teil II - Technische Einführung
- Teil III - Implementation von Data Provider und Service Provider
- Kaffeepause 15:45 - 16:15
- Teil IV - OAI Serviceprovider - Beispiele
- Teil V - Metadaten
- Teil VI - Sets
- Teil VII - Realisierung auf Verbundebene

Danksagung

- Einige Folien sind von uns,
- die meisten haben wir aus anderen Vorträgen übernommen:

  Uwe Müller

  Andy Powell

  Herbert Van de Sompel

  Carl Lagoze

  Hussein Suleman

  Michael Nelson

  Simeon Warner

  (und wahrscheinlich anderen...)

Tutorium
zur Open Archive Initiative

Teil I: Eine Einführung in die Open Archive Initiative und in das Protokol für Metadaten Harvesting

Dr. Bruno Klotz-Berendes
Universitätsbibliothek Dortmund
bruno.klotz-berendes@ub.uni-dortmund.de

Die Entstehung der Initiative

- Die Wurzeln der OAI sind in den Entwicklungen der Eprint Archive zu suchen.
  - arXiv, CogPrints, NACA (NASA), RePEc, NDLTD, NCSTRL
- Jeder dieser Dokumentenserver bot eine eigene Suchoberfläche an, die sich natürlich in der Bedienung unterschieden.
- Die Konsequenz daraus ist, dass der Endnutzer verschiedene Suchschnittstellen lernen musste.
- Schlussfolgerung: Ein Suchinterface über alle Archive
  - Universal Pre-print Service (UPS)

Crosssuche oder Harvesting

- Zwei Möglichkeiten um einen UPS zu verwirklichen:
- CrossSuche über viele Archive auf der Basis des Z39.50 Protokolls
- Harvesting von Metadaten - Einspielen der Daten in mehrere zentrale Server mit einem Suchinterface
- US digital library Untersuchungen in diesem Bereich (e.g. NCSTRL) zeigten, dass die Crosssuche nur bei einer kleinen Anzahl erfolgreich ist.

NCSTRL: N > 100; schlecht

Probleme bei der Crosssuche

- Inhaltlicher Aufbau des Archivs
  - Wie erfahre ich, was in welchem Feld gespeichert ist?
- Suchsprache - Suchsyntax
  - Die Suchsprache und die Syntax variieren und verändern sich
- Problem des Rankings
  - Wie bewerte ich die Ergebnisse von verschiedenen Archiven - Einheitliche Ergebnispräsentation
- Performance
  - Das langsamste Archiv bestimmt die Antwortzeit
- Browsingfunktion - sehr schwierig

Erfolgreiches Beispiel - KVK

- Der KVK ist ein erfolgreiches Beispiel für eine Crosssuche über verschiedene Archive
- gemeinsame Metadatenbasis (RAK - MAB)
- verlässliche Metadaten - hoher Qualitätsstandard
- wenige Archive
- verlässliche Archive
- keine einheitliche Ergebnispräsentation
- kein Browsen

Universal Preprint Service

- Ein Meta-Archiv auf der Basis von Metadaten, welche mittels Harvesting von vielen Archíven eingesammelt wurden.
- Gezeigt in Santa Fe NM, October 21-22, 1999

  http://ups.cs.odu.edu/

  D-Lib Magazine, 6(2) 2000 (2 articles)

  http://www.dlib.org/dlib/february00/02contents.html
- UPS wurde kurz darauf in Open Archives Initiative umbenannt. (OAI) http://www.openarchives.org/

RDN experience

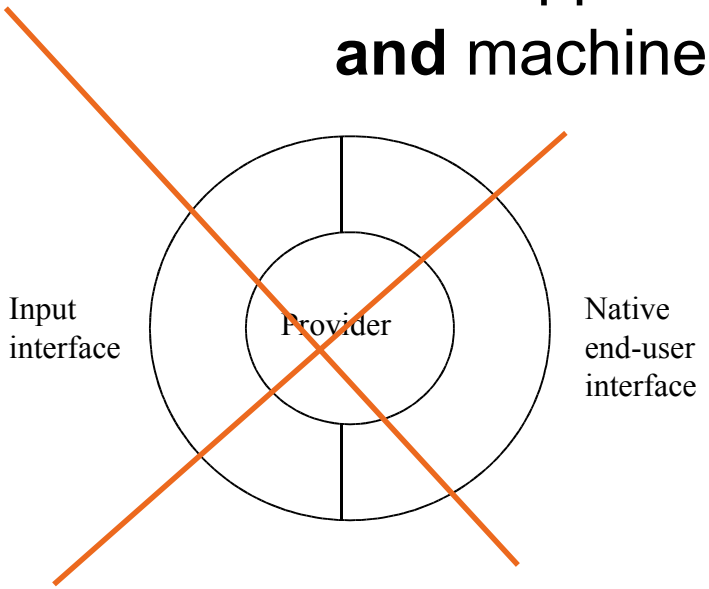
- similar experience within the UK Resource Discovery Network (RDN)
- cross-searching of only 5 subject gateways
- problems with cross-searching approach
  - performance
  - central browse interface
- looking for metadata harvesting solution

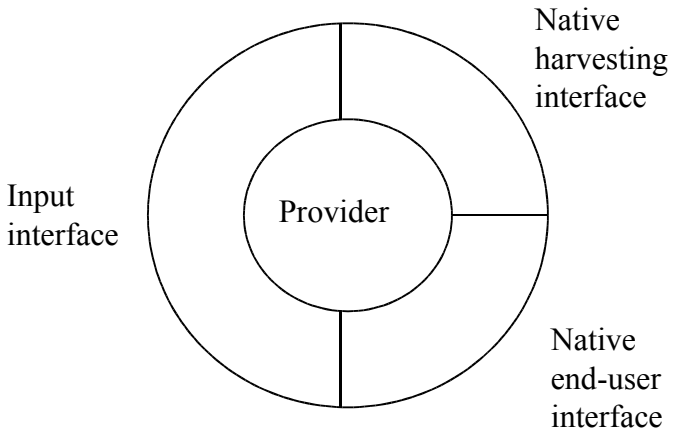
Daten- und Serviceprovider

- UPS definierte zwei Dienstleistungsprofile
- Datenprovider
  - Publiziert Inhalte der Community
  - Bietet Metadaten zu den Objekten des Archivs an - Schnittstelle
- Serviceprovider
  - Sammelt die Metadaten von den Datenprovidern ein
  - Ein Suchinterface für alle Archive, von denen Metadaten eingesammelt wurden.
- Anmerkung:
  - Der Datenprovider kann immer noch eine Endnutzersuchinterface anbieten.

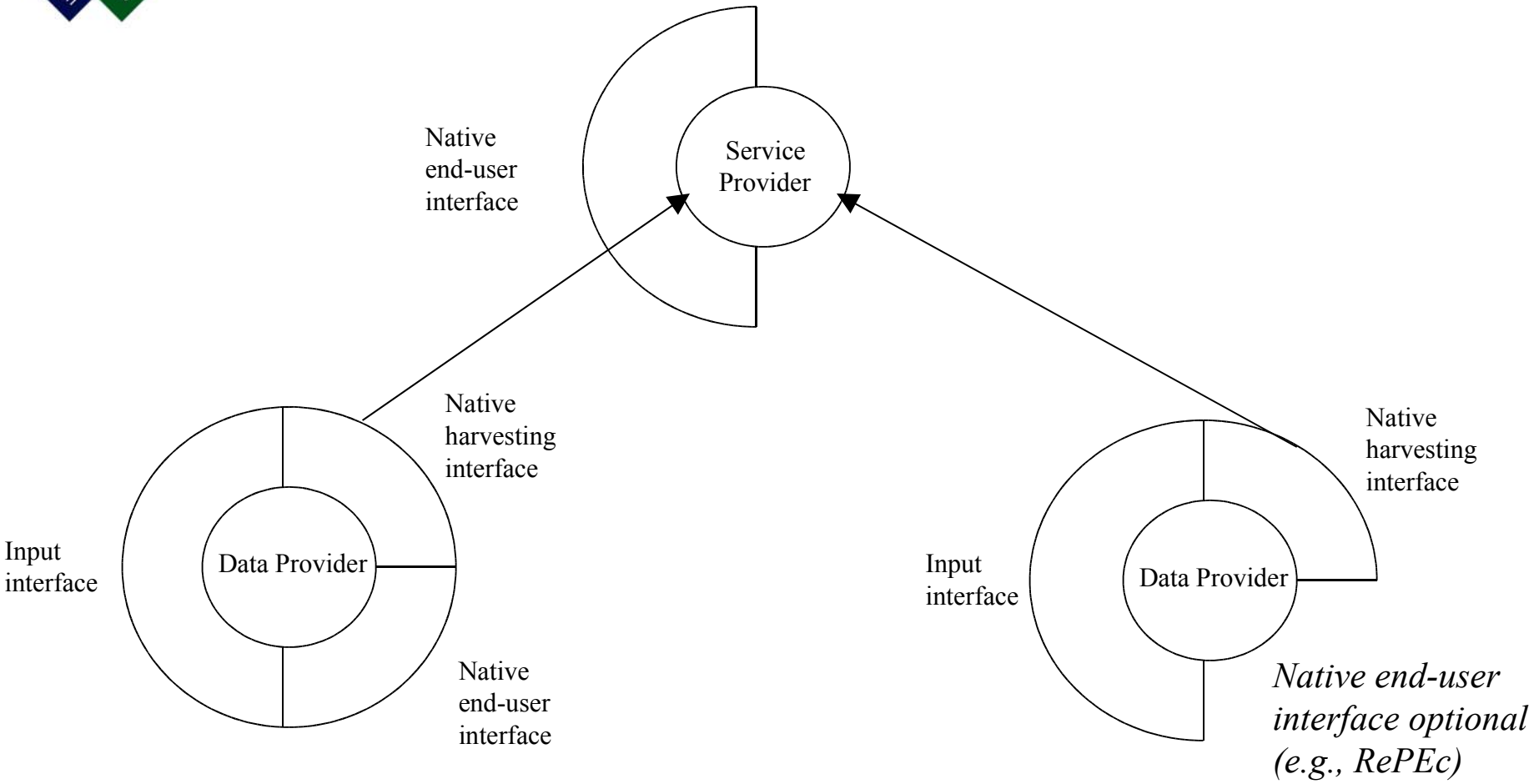
Human vs. machine interfaces

- move away from only supporting human end-user interfaces for each archive…
- …to supporting both human end-user interface **and** machine interfaces for harvesting





Service provider harvesting



Anforderungen an das Harvesting

- Damit das Einsammeln der Metadaten funktioniert, müssen Absprachen in den folgenden Bereichen erfolgen:
- Transportprotokoll – HTTP vs. FTP vs. …
- Metadatenformat – DC vs. MARC vs. …
- Qualitätskriterien – vertrauenswürdiges Archiv
- Urheberrecht und Verwertungsrechte – Wer darf was mit dem Objekt machen?
- Ergebnisse waren die “Santa Fe Convention”

Santa Fe Convention [02/2000]

- goal: optimize discovery of e-prints

- inputs…
  1. UPS prototype
  2. RePEc/SODA “data provider / service provider” model
  3. Dienst protocol
  4. deliberations at Santa Fe meeting [10/1999]

OAI-PMH v 1.0 [01/2001]

- goal: optimise discovery of document-like objects

- inputs…
  1. Santa Fe Convention
  2. various DLF meetings on metadata harvesting
  3. deliberations at Cornell
  4. alpha-testers of OAI-PMH v 1.0
  5. recognition of DC as ‘best’ core metadata format for interoperability across multiple archives

OAI-PMH Version 1.0 veröffentlicht Jan. 2001

- low-barrier interoperability specification
- Harvesting Model mit Datenprovider u. Serviceprovider
- Schwerpunkt bilden Dokumente und verwandte Publikationsformen
- Unabhängiges Protocol
- HTTP basiert
- Antworten in XML
- Dublin Core, ohne weitere Qualifier
- Experimentierstatus: 12-18 Monate

Organisationsstruktur von OAI

- **Steering Committee** - besteht aus 12 Vertretern aus verschiedenen wissenschaftlichen Institutionen

  politische Weiterentwicklung, richtungsweisende Diskussion und Promotion

- **Executive Committee** - C. Lagoze u.
  H. Van de Sompel

  Koordination der Aktivitäten

- **Technical Committee** - Evaluierung und Weiterentwicklung der OAI - Architektur, basierend auf Erfahrungen der Anwender

What's in a name?

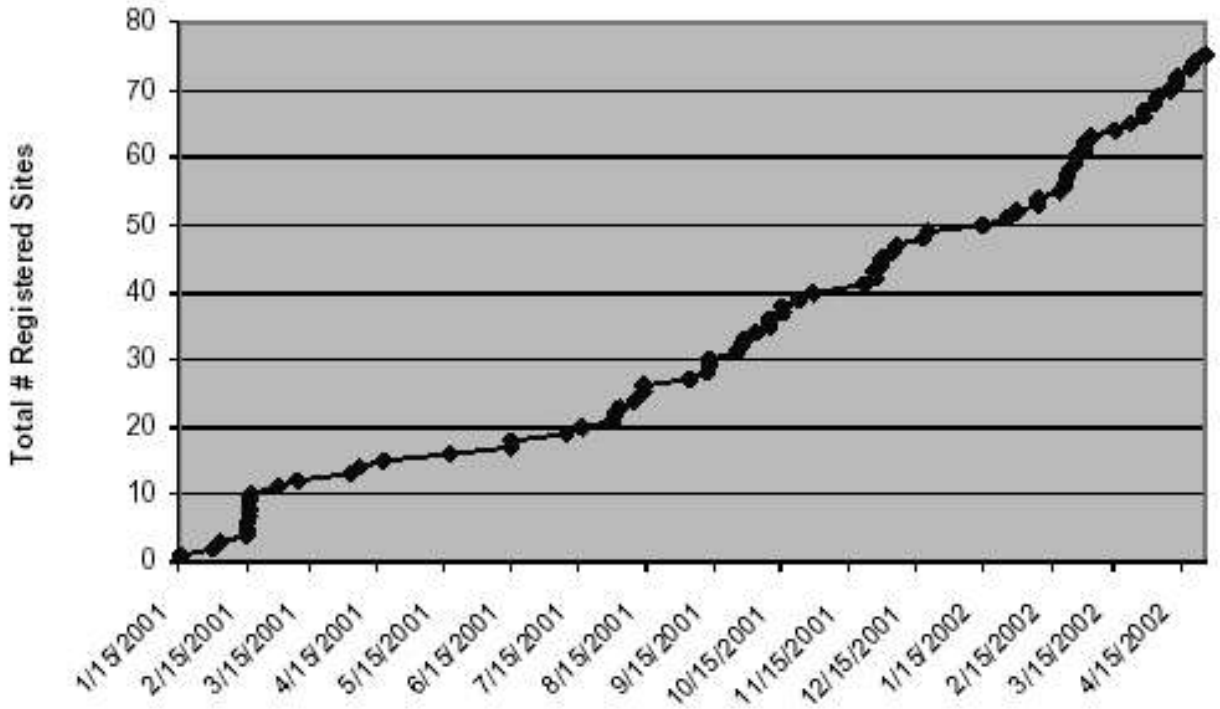
Open Archives Initiative

the protocol is openly documented, and metadata is "exposed" to at least some peer group (note: rights management can still apply!)

archive defined as a "collection of stuff" -- not the archivist's definition of "archive". "Repository" used in most OAI documents.

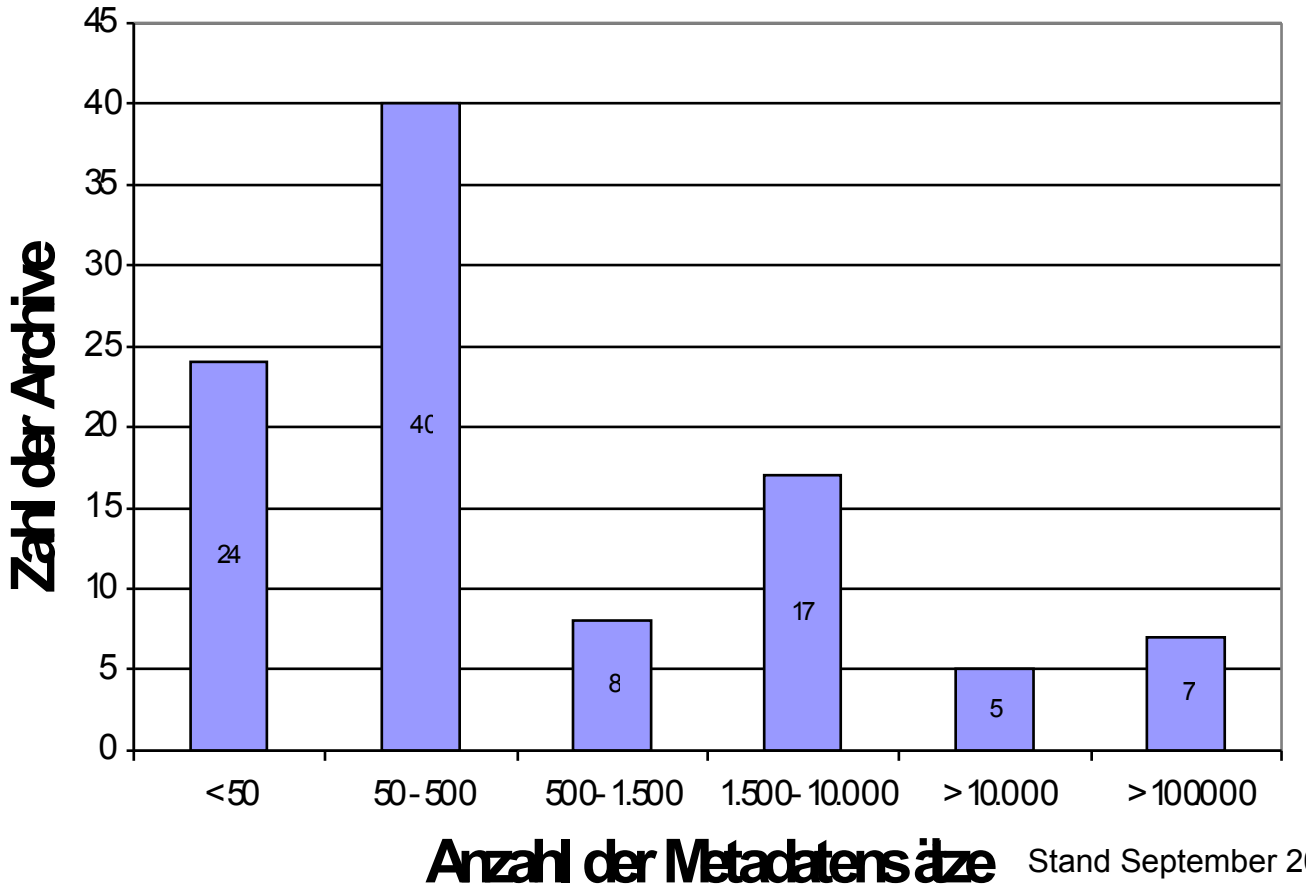
OAI is happening at break-neck speed...

Zunahme der Datenprovider



Quelle: H. Van de Sompel, C. Lagoze, Notes from the Interoperability Front: A Progress Report on the Open Archives Initiative, ECDL 2002, Rom

OAI - Archive



Stand September 2002

Wichtige Stationen von OAI 1.0 bis zur Version 2.0

- October 21-22, 1999 - initial UPS meeting
- February 15, 2000 - Santa Fe Convention published in D-Lib Magazine

  precursor to the OAI metadata harvesting protocol
- June 3, 2000 - workshop at ACM DL 2000 (Texas)
- August 25, 2000 - OAI steering committee formed, DLF/CNI support
- September 7-8, 2000 - technical meeting at Cornell University

  defined the core of the current OAI metadata harvesting protocol
- September 21, 2000 - workshop at ECDL 2000 (Portugal)
- November 1, 2000 - Alpha test group announced (~15 organizations)
- Dezember 2000 Dini Jahrestagung in Dortmund

Wichtige Stationen von OAI 1.0 bis zur Version 2.0

- January 23, 2001 - OAI protocol 1.0 announced, OAI Open Day in the U.S. (Washington DC)

  purpose: freeze protocol for 12-16 months, generate critical mass
- February 26, 2001 - OAI Open Day in Europe (Berlin)
- Workshops in Berlin und Stuttgart
- July 3, 2001 - OAI protocol 1.1 announced

  to reflect changes in the W3C's XML latest schema recommendation
- September 8, 2001 - workshop at ECDL 2001 (Darmstadt)
- Bibliothekartag Augsburg 2002 – Empfehlungen zu Setbenennung

OAI-PMH Version 2.0 veröffentlicht Juni 2002

- Ziel: Dauerhafter Austausch der Metadaten zwischen Datenprovidern und Serviceprovidern

- inputs:
  1. OAI-PMH v.1.0
  2. Feedback der OAI-implementers
  3. Überlegungen der OAI-tech [09/01 - 06/02]

  - alpha test group of OAI-PMH v.2.0 [03/02 - 06/02]

  - officially released June 14, 2002

OAI-PMH v.2.0 [06/2002]

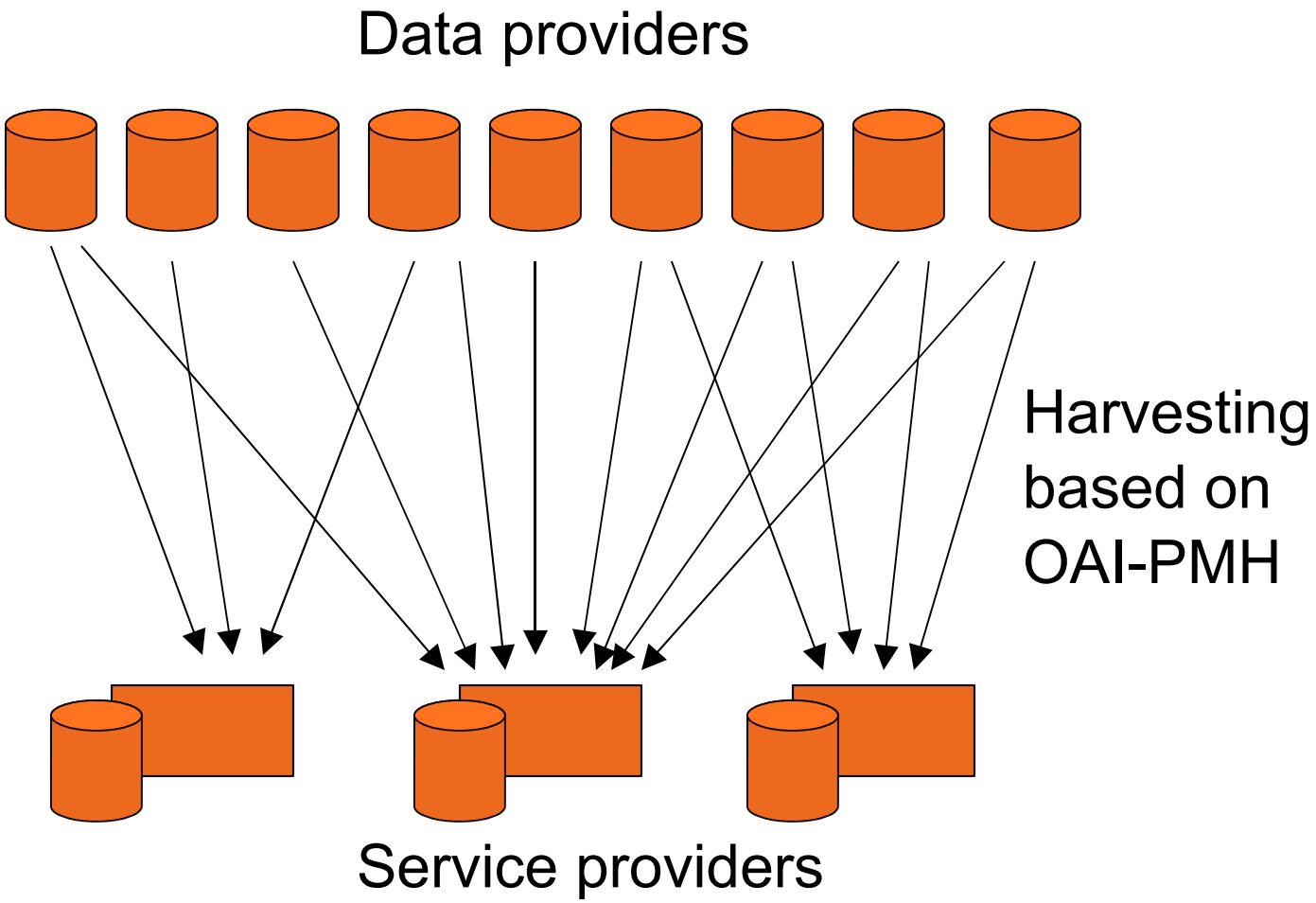
- low-barrier interoperability specification
- Metadaten Harvesting model: Datenprovider / Serviceprovider
- Metadaten der digitalen Objekte (Quellen)
- Unabhängiges Protokoll
- HTTP basiert
- XML basiert
- mindestens Dublin Core, ohne Qualifier
- Stabil - Produktionsbasis

	Santa Fe convention	OAI-PMH v.1.0/1.1	OAI-PMH v.2.0
nature	experimental	experimental	stable
verbs	Dienst	OAI-PMH	OAI-PMH
requests	HTTP GET/POST	HTTP GET/POST	HTTP GET/POST
responses	XML	XML	XML
transport	HTTP	HTTP	HTTP
metadata	OAMS	unqualified Dublin Core	unqualified Dublin Core
about	eprints	document like objects	resources
model	metadata harvesting	metadata harvesting	metadata harvesting

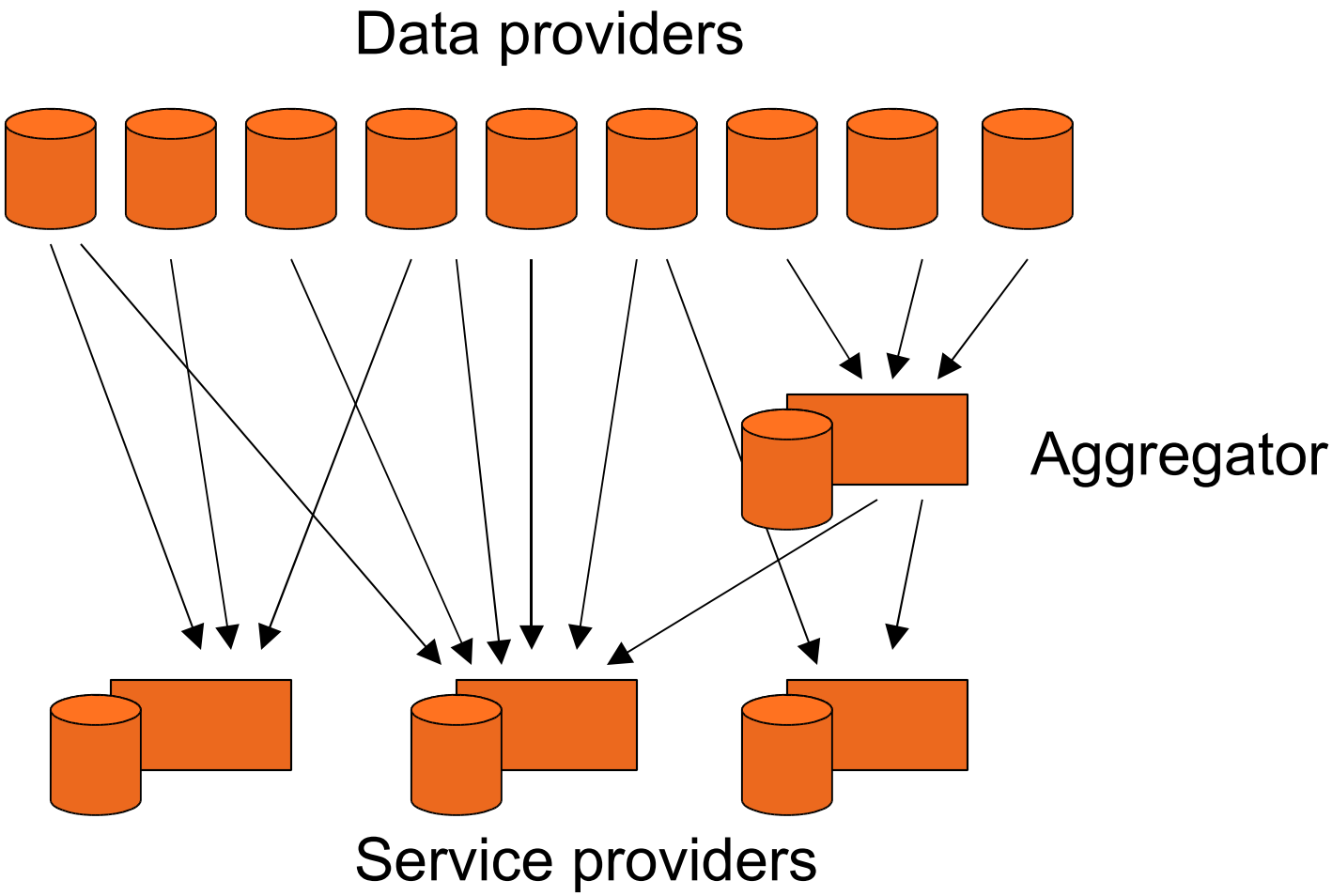
Flexible Weiterentwicklung

- Einfaches Protokoll basierend auf HTTP and XML erlaubt eine zügige Weiterentwicklung
- Eine Vielzahl an Implementierungshilfen
- Unabhängig von der eingesetzten Dokumentenserversoftware
- Verschiedene Serviceprovider können verschiedene Datenprovider harvesten
- Aggregierende Datenprovider dienen als Sammelstelle für kleine Datenprovider
- Serviceprovider können ihr Suchinterface mit weiteren Suchen über Z 39.50 ausstatten.

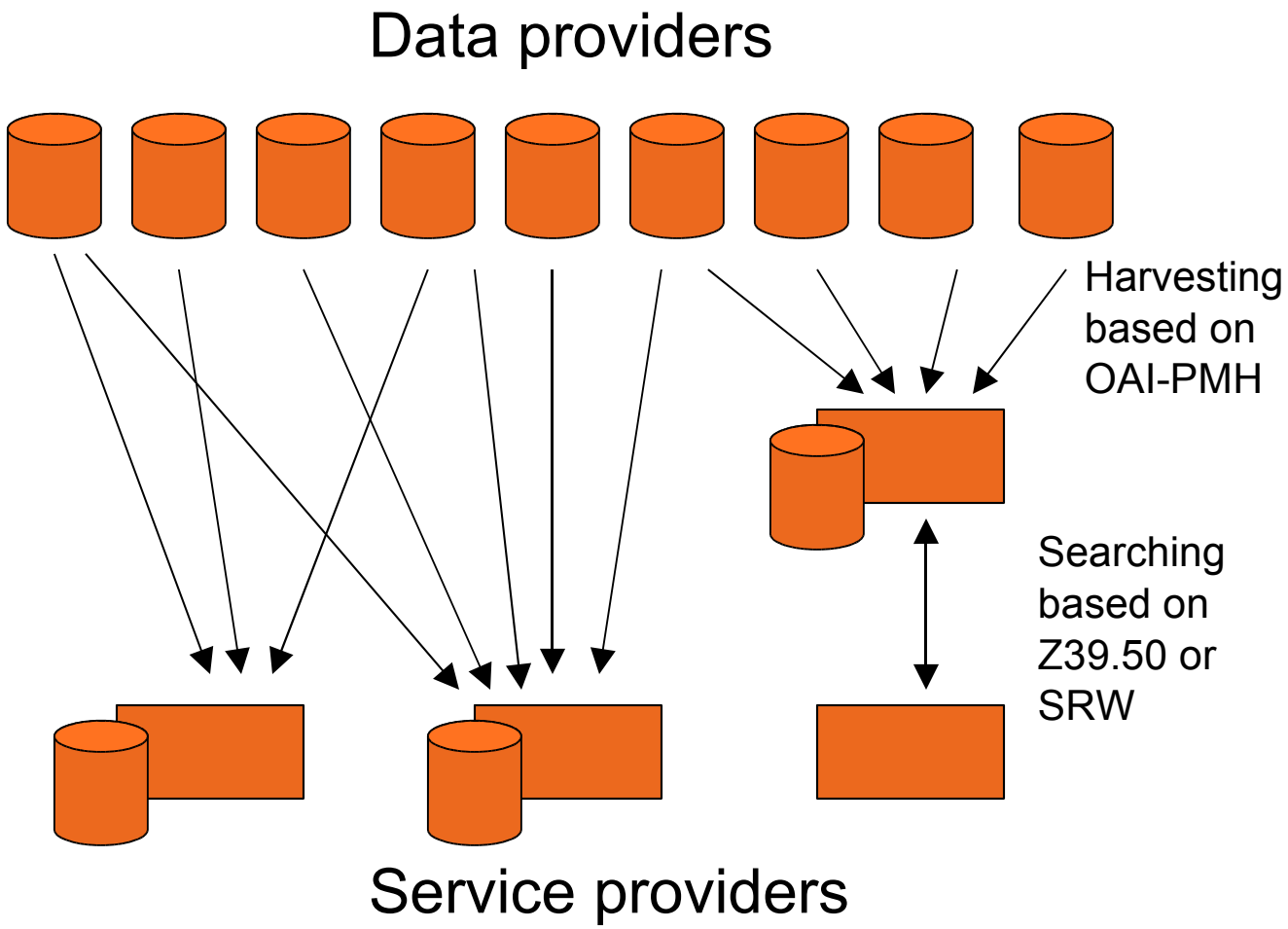
Multiple data and service p's



Aggregators



Can be mixed with x-searching



Zusammenfassung

- OAI-PMH – OAI Protocol for Metadata Harvesting
- Kostengünstiger Metadatentransfer vom Datenprovider zum Serviceprovider
- In der Entwicklung der letzten 2-3 Jahre hat ein Wechsel der Ausrichtung vom spezifischen Eprint Archiv hin zu allgemeinen digitalen Quellen stattgefunden.
- basiert auf HTTP und XML – Web – friendly
- Steuerung des Datenflusses zwischen DP u. SP (Tokenfunktion)

Zusammenfassung (2)

- Mindestens DC simpel als Metadatenformat, aber offen für alle anderen Formate, die in XML encoded sind.
- OAI-PMH ist kein Endnutzersuchprotokoll
- Metadaten und Volltext sind üblicherweise frei zugänglich – Volltexte müssen aber nicht.
  - OAI-PMH kann auch innerhalb geschlossener Gruppen benutzt werden.
- Zugriffskontrolle basiert auf dem zugrunde liegenden HTTP Protokoll

Wichtige Ressourcen

- OAI Web site:
  http://www.openarchives.org/
- OAI-PMH specification:
  http://www.openarchives.org/OAI/openarchivesprotocol.html
- Implementation guidelines:
  http://www.openarchives.org/OAI/2.0/guidelines.htm
- Discussion lists:
  http://www.openarchives.org/mailman/listinfo/oai-general
  http://oaisrv.nsdl.cornell.edu/mailman/listinfo/oai-implementers
- Repository explorer:
  http://oai.dlib.vt.edu/cgi-bin/Explorer/oai2.0/testoai
- Tools: http://oai.dlib.vt.edu/cgi-bin/Explorer/oai2.0/testoai

Gliederung des Tutoriums

- Teil I - Geschichte und Überblick
- Teil II - Technische Einführung
- Teil III - Implementation von Data Provider und Service Provider
- Kaffeepause 15:45 - 16:15
- Teil IV - OAI Serviceprovider - Beispiele
- Teil V - Metadaten
- Teil VI - Sets
- Teil VII - Realisierung auf Verbundebene

Tutorium
zur Open Archive Initiative

Teil II
Technische Einführung

Dr. Heinrich Stamerjohanns
Institute for Science Networking, Uni Oldenburg
stamer@uni-oldenburg.de

Was ist ein „Open Archive“

- Jedes WWW-basierende System, auf welches durch eine genau definierte Schnittstelle des Open Archives Protocol for Metadata Harvesting (OAI-PMH) zugegriffen werden kann.
- wird auch als ein „OAI-compliant Repository“ bezeichnet
- Keine Auswirkung auf:
  - physikalische Speicherung der Daten
  - Kosten der Daten
  - Metadaten und Datenformate
  - Zugriffskontrolle auf den Server

Harvesting vs. Federation

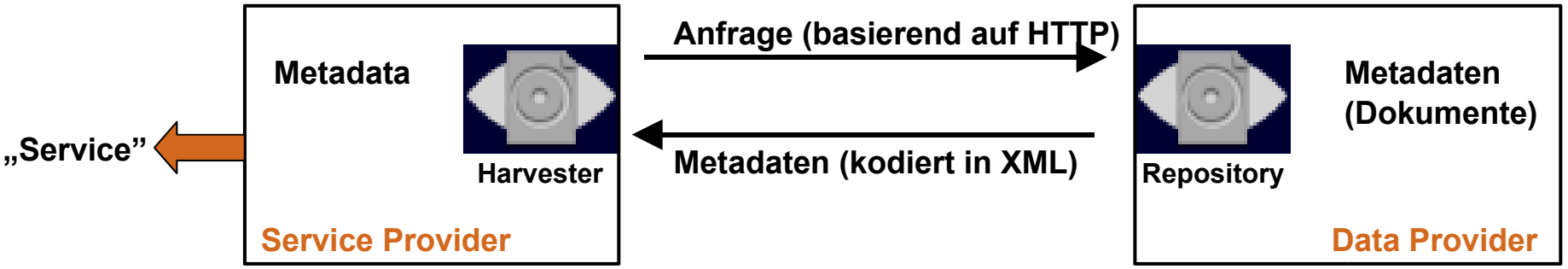
- Unterschiedliche Strategien, um Interoperabilität zu erreichen
  - Federation: Dienste laufen verteilt auf verteilten Daten (z.B. Federated searching)
  - Harvesting: Daten/Metadaten werden von verteilten Quellen zum Ziel, wo Dienste angeboten werden, transferiert (e.g. Union catalogues)
- Federation benötigt mehr Aufwand an jeder verteilten Quelle, ist aber für das lokale System einfacher
- gilt umgekehrt für Harvesting
- OAI konzentriert sich auf Harvesting

Metadata vs. Data

- Daten beziehen sich auf digitale Objekte oder deren digitale Repräsentation
- Metadaten ist Information über solche Objekte (z.B. Titel, Autor, usw.)
- OAI konzentriert sich auf Metadaten, mit der impliziten Annahme, dass Metadaten normalerweise brauchbare Links zu dem digitalen Objekt selbst enthalten

The Open Archives Initiative (OAI)

- Hauptideen
  - weltweite Konsolidierung von Archiven
  - freier Zugang zu den Archiven (zumindest: Metadaten)
  - konsistente Interfaces für Archive und Service Provider
  - low barrier protocol / einfache Implementation
- Einfacher Aufbau:



Anforderungen an das Protokoll

- sollte
- in einem maschinenlesbarem Format sein
- in einem strikten Format kodiert werden, welches validiert werden kann
  - character-encoding
  - metadata-encoding
- verschiedene Content-Modelle unterstützen
  - metadata formats
- existierende Technologie nutzen (HTTP, XML, DC)
  - einfach zu implementieren
  - einfach anzupassen

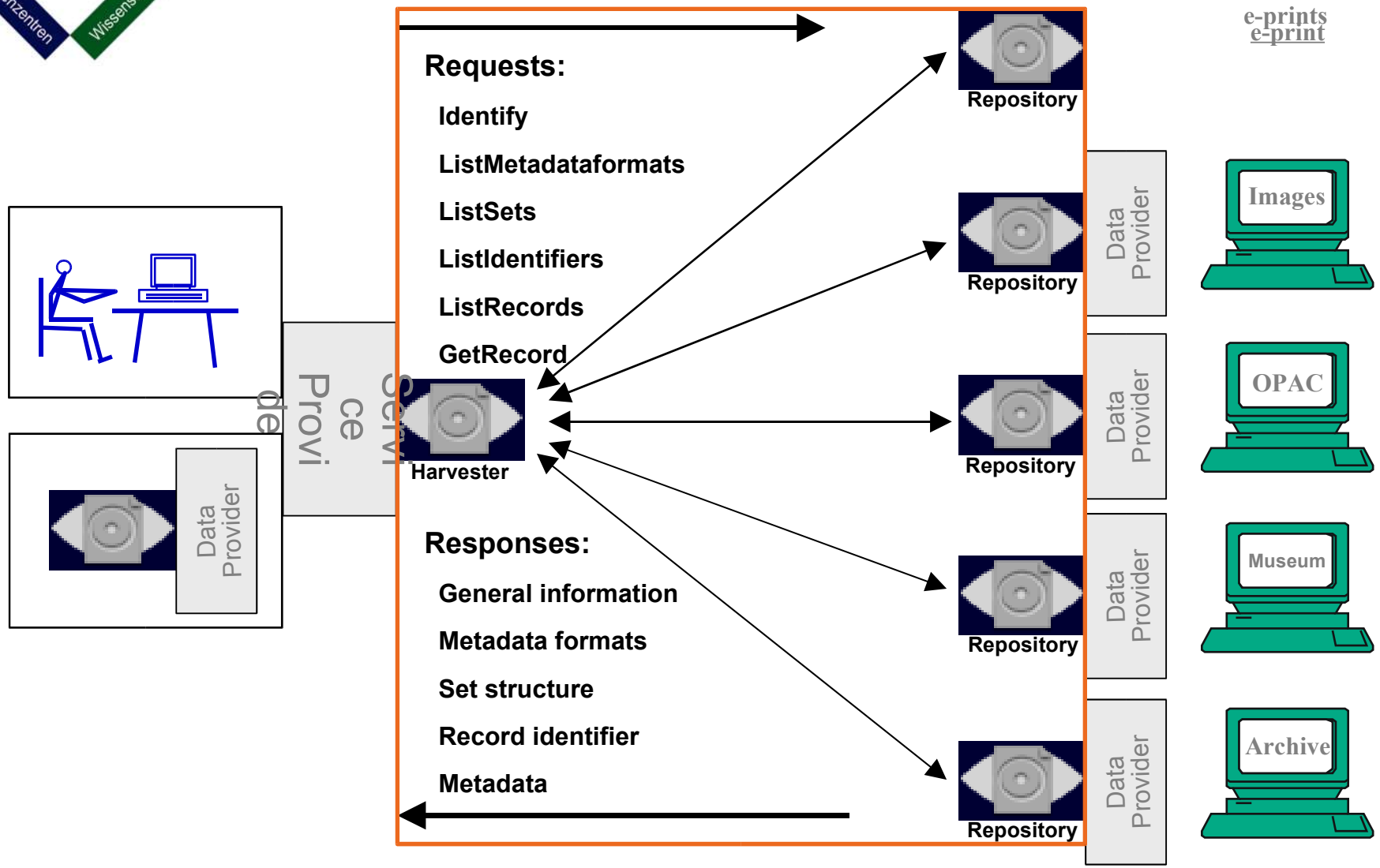
Data und Service Provider

- Data Provider bezeichnen Entitäten (z.B. Datenbanken), die Daten / Metadaten besitzen und diese über wohldefinierte OAI-Protokolle mit anderen teilen wollen (intern oder extern)
- Service Provider sind Entitäten, die Daten von Data Providern einsammeln um höhere Dienste (z.B. Suchmaschinen) anzubieten
- OAI benutzt diese Bezeichnungen für sein Client / Server Modell (Data = Server, Service = Client)

OAI: Allgemeine Annahmen

- zwei Gruppen von “Teilnehmern”
- *Data Provider* (Open Archives, Repositories)
  - freier Zugang zu Metadaten
  - nicht notwendigerweise: freier Zugang zu den Volltexten
  - einfache Implementation / niedrige Barrieren
- *Service Provider*
  - nutzen OAI interface der *Data Provider*
  - sammeln und speichern Metadaten asynchron (keine synchrone verteilte Anfrage)
  - können einige Untermengen von *Data Provider* einsammeln (Mengen (set) Hierarchie, Datestamp)
  - können die Metadaten anreichern
  - bieten (value-added) Dienste, die auf den Metadaten basieren

OAI-PMH: Struktur Modell



OAI-PMH: Überblick über das Protokoll

- Protokoll basiert auf HTTP
  - Anfrageargumente als GET oder POST Parameter
  - sechs Anfragetypen
  - z.B. http://archive.org?
    verb=ListRecords&from=2002-11-01
  - Antworten werden in XML kodiert
  - unterstützt jedes Metadatenformat

    zumindest: Dublin Core als kleinster gemeinsamer Nenner
  - logische Mengenhierarchie, die von den Data Providern definiert werden
  - Zeitmarken (datestamps, letzte Änderung der Metadaten)
  - Fehlermeldungen (auch in XML)
  - Flusskontrolle

Protokolldetails: Definitionen

Harvester (Ernter, Einsammler)

- Client Anwendung, die OAI-PMH Anfragen stellt

Repository

- netzwerkfähiger Server, welcher OAI-PMH Anfragen beantworten kann

Resource

- Das Objekt, welches die Metadaten beschreiben, die Art der Resourcen werden durch OAI-PMH nicht definiert

Item

- Komponente eines Repository, von welchem Metadaten über eine Resource zur Verfügung gestellt werden kann

  hat einen eindeutigen Identifier

Record (Datensatz)

- Metadaten in einem spezifischen Metadatenformat

Identifier

- eindeutiger Schlüssel key für einen Item in einem Repository

Set (Menge)

- optionales Konstrukt um Items in einem Repository anzuordnen

Protokolldetails: Definitionen (2)

resource

item = identifier

Metadaten über *David*

item

Dublin Core metadaten

MARC metadaten

SPECTRUM metadaten

record

Was ist ein „Record“ ?

- Ein Record bezieht sich auf eine unabhängige XML Struktur, die mit einem digitalen oder physikalischen Objekt assoziiert werden kann
- Records sind normalerweise mit Metadaten und nicht mit Daten assoziiert
- OAI propagiert das Einsammeln (Harvesting) von Records, die Metadaten enthalten, sowie weitere Felder, die das Harvesting unterstützen

Eindeutigkeit und Persistenz

- Jeder Datensatz (Record) muss eindeutig durch unterschiedliche Identifier adressierbar sein
- Jede Metadatenentität sollte idealerweise persistent sein, so dass Service Provider sich auch später auf diese Quelle beziehen können

Protokolldetails: Records

- Metadaten einer Resource in einem spezifischen Format
  1. header (Kopf) (notwendig)
     - Identifier (1)
     - Datestamp (1)
     - setSpec Elemente (*)
     - Statusattribut für gelöschte Items (?)
  2. metadata (notwendig)
     - XML kodierte Metadaten mit “root tag”, Namespace
     - Repositories müssen zumindest Dublin Core unterstützen
  3. about (über) (optional)
     - Rights Angaben
     - Provenance Angaben

Beispiel: OAI Record

(beachte: Schema und Namensräume sind der Einfachheit halber weggelassen worden)

```
<record>
  <header>
    <identifier>oai:physnet.de:tut1</identifier>
    <datestamp>2003-03-27</datestamp>
    <setSpec>tut</setSpec>
  </header>
  <metadata>
    <oai_dc>
      <title>OAI Tutorial at OA Forum III</title>
      <creator>Heinrich Stamerjohanns</creator>
      <language>ger</language>
    </oai_dc>
  </metadata>
  <about>
    <metadataID>oai:physnet.de:tut1md</metadataID>
  </about>
</record>
```

Zeitstempel (Datestamps) & Harvesting

- Jeder Record braucht ein Datestamp, welches das Datum der Erstellung oder Modifikation der Metadaten beschreibt
- zwei mögliche Granularitäten: YYYY-MM-DD, YYYY-MM-DDThh:mm:ssZ
- Funktion: Information über Metadaten, “selective harvesting” (**`from`** und **`until`** Argumente)
- erlaubt inkrementelle und kontinuierliche Metadatenübertragung von einem Data Provider zu einem Service Provider
- Löschung (deletion): drei Level
  - no, persistent, transient

Protokolldetails: Metadaten Schema

- OAI-PMH unterstützt Verteilung von mehreren Metadatenformaten von einem Repository
- Eigenschaften von Metadatenformaten
  - id, um das Format anzugeben (**`metadataPrefix`**)
  - Metadaten Schema URL (XML Schema, um XML zu validieren)
  - XML Namespace URI (Globaler Identifier für Metadatenformat)
- Repositories müssen unqualified Dublin Core unterstützen

Multiplizität von Metadaten

- beliebige Metadatenformate können definiert werden und mittels OAI-PMH übertragen werden
- Mehrfache Metadatenformate für eine Resource sind erlaubt
- Metadaten müssen mit XML Namespace Spezifikation übereinstimmen
- Jedes andere Format ist erlaubt, solange es ein XML encoding besitzt
- Z.B. MARC (Bibliotheken), IMS (Lehre), ETDMS (Diplomarbiten/Dissertationen), RFC1807 (Bibliographien)

XML Namensräume (Namespaces) und Schema

- Konsistenz und Datenqualität werden durch den Gebrauch von XML Schema Beschreibungen für jede mögliche Antwort sichergestellt
- XML Namespaces werden, wenn notwendig, benutzt, um eindeutig zu definieren, welche Teile der Antwort wirklich Metadaten sind, und welche Teile das Metadata Harvesting Protokoll unterstützen

Mengen (Sets)

- Das Protokoll unterstützt einen Mechanismus um Teilkollektionen (sub-collections) einzusammeln
- Keine wohldefinierte Semantik – diese hängt völlig von den lokalen Data Providern ab
- Semantik kann jedoch durch Absprache zwischen Data und Service Providern erreicht werden
- optional – Archive müssen keine Sets definieren
- Anwendungen:
  subject gateways, dissertation search engine, …
- Beispiele (Deutschland, siehe http://www.dini.de)
  - Publikationstypen (thesis, article, …)
  - Dokumenttypen (text, audio, image, …)
  - inhaltliche Sets, nach DNB (Medizin, Biologie, …)

Protokolldetails: Anfrageformat

- Anfragen werden durch den Gebrauch der **`GET`** oder **`POST`** Methoden von HTTP gestellt
- Repositories müssen beide Methoden unterstützen
- mindestens ein key=value Paar: verb= [RequestType]
- weitere key=value Paare hängen von der Anfrage ab
- Beispiel für eine **`GET`** Anfrage: http://archive.org/oai?verb=ListRecords&metadataPrefix=oai_dc
- Besondere Zeichen
  z.B. "**:**" (host port separator) wird "**%3A**"

Protokolldetails: Antwort

- XML eingebettet in HTTP
- content type muss “text/xml“ sein
- Status Codes (unterschiedlich von OAI-PMH Fehlern)
  e.g. 302 (redirect), 503 (service not available)
- Kompression: optional bei OAI-PMH,
  nur identity encoding is notwendig
- Antwortformat: wohlgeformetes XML mit Markup:
  1. XML declaration
     (**`<?xml version="1.0" encoding="UTF-8" ?>`**)
  2. Root Element namens **`OAI-PMH`** mit drei Attributen
     (**`xmlns`**, **`xmlns:xsi`**, **`xsi:schemaLocation`**)
  3. drei Child Elemente
     - **`responseDate`** (UTC datetime)
     - **`request`** (Anfrage, die diese Antwort generiert)
     - a) **`error`** (im Fehlerfalle)
       b) Element mit dem Namen der OAI-PMH Anfrage

Flusskontrolle (Flow Control)

- Flusskontrolle auf zwei Protokollebenen
  - HTTP (503, retry-after)
  - OAI-PMH, Resumption-Token
- HTTP “retry-after” Mechanismus kann eingesetzt werden, um Anfragen eines Clients zurückzustellen
- Resumption Tokens werden benutzt, um nur Teilantworten zurückzugeben
- Der Client bekommt einen Token, den er für eine neue Anfrage am Server benutzen kann, um weitere Antworten zu bekommen

Protokolldetails: Flusskontrolle

- vier Anfragetypen geben eine Liste zurück
- drei können “große” Listen zurückgeben
- OAI-PMH unterstützt Teilantworten
- die Entscheidung trifft der Data Provider
- die Antwort auf eine Anfrage enthält
  - eine unvollständige Liste
  - Resumption Token
    + mit Verfallsdatum, Größe der kompletten Liste, Cursor (optional)
- neue Anfrage desselben Typs
  - Resumption Token als Parameter
  - alle anderen Parameter werden weggelassen!
- die Antwort enthält
  - nächste (vielleicht letzten) Teil der Liste
  - Resumption Token (leer, wenn der letzte Teil der Liste geschickt wird)

Protokolldetails: Flusskontrolle (2)

Beispiel

Protokolldetails: Fehler und Ausnahmen

- Repositories müssen OAI-PMH Fehler angeben, die als ein oder mehrere **`error`** Elemente zurückgegeben werden
- definierte Fehleridentifier
  - **`badArgument`**
  - **`badResumptionToken`**
  - **`badVerb`**
  - **`cannotDisseminateFormat`**
  - **`idDoesNotExist`**
  - **`noRecordsMatch`**
  - **`noMetaDataFormats`**
  - **`noSetHierarchy`**

Anfragetypen

- sechs verschiedene Anfragetypen
  1. Identify
  2. ListMetadataFormats
  3. ListSets
  4. ListIdentifiers
  5. ListRecords
  6. GetRecord
- Der Harvester muss nicht alle Typen benutzen
- Das Repository muss alle Typen implementieren
- Notwendige und optionale Argumente, die von den Anfragetypen abhängen

Metadata Harvesting Protokoll

- Metadata Multiplizität
- Datumszeiträume (Date Ranges)
- Resumption Tokens
- Fehler und Ausnahmen

Identify

- Funktion
  - allgemeine Information über das Archiv
- Parameter
  - keine
- Beispiel URL
  - http://physnet.de/oai/oai2.php?verb=Identify
- Fehler/Ausnahmen
  - **`badArgument`**
    z.B. physnet.de/oai/oai2.php?verb=Identify**&set=biology**

Anfragetypen: Identify (2)

Antwortformat

Element	*Example*	#
repositoryName	My Archive	1
baseURL	http://archive.org/oai	1
protocolVersion	2.0	1
earliestDatestamp	1999-01-01	1
deleteRecords	no, transient, persistent	1
granularity	YYYY-MM-DD, YYYY-MM-DDThh:mm:ssZ	1
adminEmail	oai-admin@archive.org	+
compression	deflate, compress, …	*
description	oai-identifier, eprints, friends, …	*

Identify – Antwort

```
- <OAI-PMH xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
  http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
    <responseDate>2003-03-30T14:46:08Z</responseDate>
    <request verb="Identify">http://physnet.uni-oldenburg.de/oai/oai2.php</request>
   - <Identify>
      <repositoryName>PhysNet, Oldenburg, Germany, Document Server</repositoryName>
      <baseURL>http://physnet.uni-oldenburg.de/oai/oai2.php</baseURL>
      <protocolVersion>2.0</protocolVersion>
      <adminEmail>mailto:stamer@uni-oldenburg.de</adminEmail>
      <earliestDatestamp>2000-01-01T00:00:00Z</earliestDatestamp>
      <deletedRecord>no</deletedRecord>
      <granularity>YYYY-MM-DDThh:mm:ssZ</granularity>
      <compression>gzip</compression>
     - <description>
       - <friends xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/friends/
         http://www.openarchives.org/OAI/2.0/friends.xsd">
           <baseURL>http://naca.larc.nasa.gov/oai2.0/</baseURL>
           <baseURL>http://techreports.larc.nasa.gov/ltrs/oai2.0/</baseURL>
           <baseURL>http://physnet.uni-oldenburg.de/oai/oai.php</baseURL>
           <baseURL>http://cogprints.soton.ac.uk/perl/oai</baseURL>
           <baseURL>http://ub.uni-duisburg.de:8080/cgi-oai/oai.pl</baseURL>
          - <baseURL>
              http://rocky.dlib.vt.edu/~jcdlpix/cgi-bin/OAI1.1/jcdlpix.pl
            </baseURL>
         </friends>
      </description>
   </Identify>
</OAI-PMH>
```

ListMetadataFormats

- Funktion
  - liste sowohl die Metadatenformate, die vom Archiv unterstützt werden, als auch deren Schema Locations und Namespaces
- Parameter
  - identifier – für einen spezifischen Record (O)
- Beispiel URL
  - http://physnet.de/oai/oai2.php?verb=ListMetadataFormats
- Fehler/Ausnahmen
  - **`badArgument`**
  - **`idDoesNotExist`**

    archive.org/oai-script?verb=ListMetadataFormats**&**
    identifier=really-wrong-identifier
  - **`noMetadataFormats`**

ListMetadataFormats Antwort

```
- <OAI-PMH xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
  http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
    <responseDate>2003-03-30T14:56:43Z</responseDate>
    <request verb="ListMetadataFormats">http://physnet.uni-oldenburg.de/oai/oai2.php</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/</metadataNamespace>
       </metadataFormat>
    </ListMetadataFormats>
  </OAI-PMH>
```

ListSets

- Funktion
  - hierarchisches Listing der Mengen in welchem die Records organisiert sind
- Parameter
  - keine
- Beispiel URL
  - http://physnet.de/oai/oai2.php?verb=ListSets
- Fehler/Ausnahmen
  - **`badArgument`**
  - **`badResumptionToken`**
    archive.org/oai-script?verb=ListSets**&**
    resumptionToken=any-wrong-token
  - **`noSetHierarchy`**

ListIdentifiers

- Funktion
  - Liste die Header für alle Records, die den entsprechenden Parametern entsprechen
- Parameter
  - **`from`** – Startdatum (O)
  - **`until`** – Enddatum (O)
  - **`set`** – Menge von welcher eingesammelt werden soll (O)
  - **`metadataPrefix`** – Metadatenformat, für welches die Identifier gelistet werden sollen (R)
  - **`resumptionToken`** – Flusskontrolle (X)
- Beispiel URL
  - http://physnet.de/oai/oai2.php?verb=ListIdentifiers&metadataPrefix=oai_dc

ListIdentifiers

- Fehler/Ausnahmen
  - **badArgument**, z.B.. …**&from=2002-12-01-13:45:00**
  - **badResumptionToken**
  - **cannotDisseminateFormat**
  - **noRecordsMatch**
  - **noSetHierarchy**

ListRecords

- Funktion
  - Metadaten für mehrere Records
- Parameter
  - **`from`** – Startdatum (O)
  - **`until`** – Enddatum (O)
  - **`set`** – Menge von der eingesammelt werden soll (O)
  - **`metadataPrefix`** – Metadatenformat (R)
  - **`resumptionToken`** – Flusskontrolle (X)
- Beispiel UR
  - http://physnet.de/oai/oai2.php?verb=ListRecord&metadataprefix=oai_dc&from=2001-01-01

ListRecords

- Fehler/Ausnahmen
  - **badArgument**
  - **badResumptionToken**
  - **cannotDisseminateFormat**
  - **noRecordsMatch**
  - **noSetHierarchy**

ListRecords – Antwort

```
<?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>2003-03-28T14:59:21Z</responseDate>
 <request verb="ListRecords" metadataPrefix="oai_dc">
          http://physnet.uni-oldenburg.de/oai/oai2.php</request>
 <ListRecords>
  <record>
   <header>
    <identifier>oai:physdoc:http://www.ensta.fr</identifier>
    <datestamp>2002-01-25T00:00:00Z</datestamp>
   </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>Pole de Calcul Parallele,</dc:title>
      <dc:date>2000-01-05</dc:date>
      <dc:identifier>http://www.ensta.fr</dc:identifier>
      <dc:language>eng</dc:language>
     </oai_dc:dc>
   </metadata>
  </record>
  <record>
   <header>
    <identifier>oai:physdoc:ftp://ftp.esi.ac.at/pub/Preprints/esi955.ps</id
    <datestamp>2002-01-25T00:00:00Z</datestamp>
   </header>
   <metadata>
     <oai_dc:dc
```

GetRecord

- Funktion
  - gib die Metadaten für einen einzelnen Record in der Form eines OAI Records zurück
- Parameter
  - **`identifier`** – eindeutige Id für den Record (R)
  - **`metadataPrefix`** – Metadatenformat (R)
- Beispiel URL
  - http://physnet.de/oai/oai2.php?verb=GetRecord &identifier=oai:test:123&metadataPrefix=oai_dc
- Fehler/Ausnahmen
  - **`badArgument`**
  - **`cannotDisseminateFormat`**
  - **`idDoesNotExist`**

Zeiträume (Date Ranges)

Address dl/oai.pl?verb=ListIdentifiers&metadataPrefix=oai_dc&from=2001-06-26&until=2001-06-26 Go

```
<?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>2002-05-26T19:41:16Z</responseDate>
    <request verb="ListIdentifiers" metadataPrefix="oai_dc" from="2001-06-
      26" until="2001-06-26">http://rocky.dlib.vt.edu/~jcdlpix/cgi-
      bin/OAI2.0/beta2/jcdl/oai.pl</request>
  - <ListIdentifiers>
    - <header>
        <identifier>oai:JCDLPICS:200102dlb1</identifier>
        <datestamp>2001-06-26</datestamp>
        <setSpec>200102dlb</setSpec>
      </header>
    - <header>
        <identifier>oai:JCDLPICS:200102dlb2</identifier>
        <datestamp>2001-06-26</datestamp>
```

Gliederung des Tutoriums

- Teil I - Geschichte und Überblick
- Teil II - Technische Einführung
- Teil III - Implementation von Data Provider und Service Provider
- Kaffeepause 15:45 - 16:15
- Teil IV - OAI Serviceprovider - Beispiele
- Teil V - Metadaten
- Teil VI - Sets
- Teil VII - Realisierung auf Verbundebene

Tutorium
zur Open Archive Initiative

Teil III

Implementation

Data und Service Provider

- Data Provider
- Service Provider
- Communities

Dr. Heinrich Stamerjohanns
Institute for Science Networking, Uni Oldenburg
stamer@uni-oldenburg.de

Data- und Service Provider

- Erste Fragen
- Metadaten
- Organisation
- Voraussetzungen Data-Provider
- Architektur
- Einige Besonderheiten
- Übliche Probleme
- Details für Implementation
- Werkzeuge zum Testen

Allgemein: Erste Fragen

Data Provider

Welche Daten möchte ich anbieten?

(Welchen Service Providern biete ich diese Daten an?)

Service Provider

Welchen Dienst möchte ich anbieten?

Von welchen Data Providern werde ich die Daten einsammeln?

Welche Metadatenformate soll ich unterstützen?

Data Provider & Service Provider

Auf welche Aspekte muss ich mich einigen?

Metadatenformate...

Abbildungen von Metadaten (Metadata Mappings)

- Data Provider muss seine Metadaten auf das Format, welches er durch das OAI Interface anbietet, abbilden (map).
- Unqualified Dublin Core ist als kleinster gemeinsamer Nenner notwendig
  - http://dublincore.org/
  - Dublin Core Metadata Element Set enthält 15 Elemente
  - Elemente sind optional, können sich jedoch auch wiederholen
  - Üblicherweise wird ein Link in dem <identifier> Tag zur Resource oder zumindest zu einer lesbaren Webseite angeboten
- Ursprungsformate werden empfohlen
- Metadatenformate der eigenen Community werden empfohlen

Organisation

- Besondere Fächer/Themengebiete/Communities: andere Metadatenbeschreibungen sind vielleicht notwendig
  - beschreibe Resourcen in einer besonderen Weise
  - Definition eines eigenen XML Schemas (welches für Validierung durch andere öffentlich zugänglich sein sollte)
- definiere eine Mengen Hierarchie
  - um die eigenen Metadaten für “selective harvesting” aufzuteilen
  - Einigung zwischen Data Provider und Data und Service Provider
- zusammengefasste Data Provider
  - wenn ein Service Provider einsammelt, sollte er nicht auch die “sub data providers” befragen (Dubletten)
- Subject Gateways
  - sind bei Einigung auf bestimmte Sets auf einfache Weise möglich

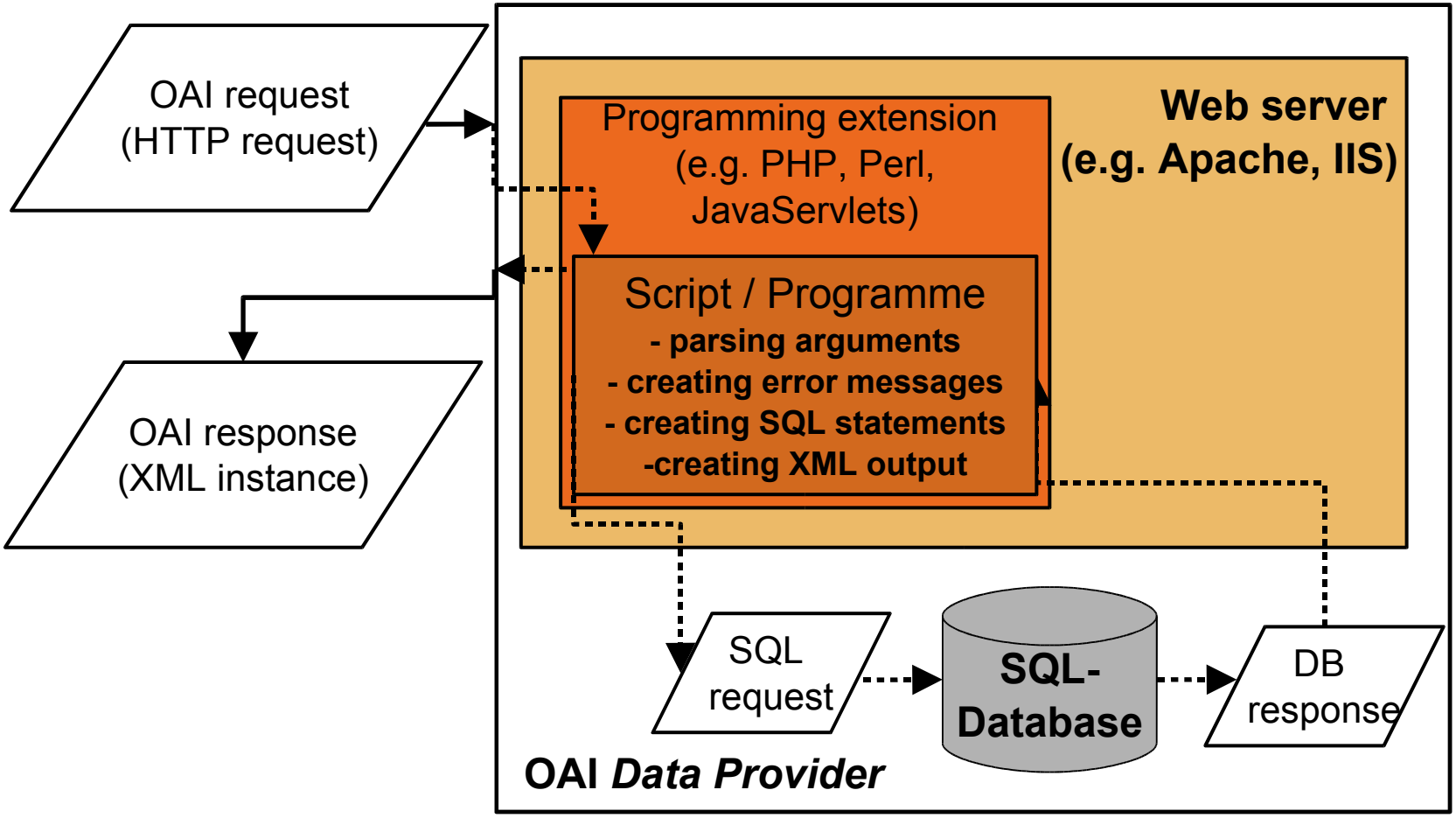
Server Technologie

- WWW Server
- Protokoll kann in beliebiger Form implementiert werden
  - CGI Skript (Perl, C++, Java)
  - Java Servlet
  - PHP
- Metadaten (z.B. Datenbank) Zugriff ist notwendig
- Siehe www.openarchives.org für eine Liste von öffentlich erhältlicher Software

Metadatenquelle

- Datenbank in proprietärem Format
- Metadatensammlungen in wohldefinierten Format (en)
  - Dateien auf der Festplatte
- Metadaten können dynamisch oder statisch aus den Daten extrahiert werden
- Synthetitische Sammlung

Data Provider: Architektur



Zeitstempel (Datestamps)

- werden für jedes Record benötigt, im inkrementelles Harvesting zu unterstützen
- muss bei jeder Hinzufügung/Modifikation/Löschung aktualisiert werden, so dass Änderungen sicher weitergegeben werden
- unterscheidet sich von den Daten innerhalb der Metadaten – dieses Datum wird ausschliesslich für Harvesting benötigt
- Kann entweder YYYY-MM-DD oder YYYY-MM-DDThh:mm:ssZ (muss als GMT Zeitzone angegeben werden)

Eindeutige (unique) Identifier

- Jeder Record muss einen eindeutigen (unique) Identifier besitzen
- Identifier müssen gültige URIs sein
- Beispiel
  - oai:<archiveId>:<recordId>
  - oai:physnet.de:etd-1234567890
- Jeder Identifier muss zu einem einzigen und immer dem gleichen Record aufgelöst werden (für ein vorgegebenes Metadatenformat)

Löschungen (Deletions)

- Archive können (sollten) gelöschte Records mitführen und Identifier und Datestamp bewahren
- Alle Protokoll Ergebnismengen (result sets) können auf gelöschte Records hinweisen
- Wenn Löschungen mitgeführt werden, muss die Information für immer gespeichert werden, so dass dies an Service Provider mit unterschiedlichen Harvesting Zeitplänen korrekt weitergegeben werden kann.

Details für die Implementation

- benötigte Werkzeuge
- einfacher Programmablauf
- Allgemeine Struktur
- Erweiterbare Metadatenerstellung
- Kodierung in XML
- Cachen der Antworten
- Fehlerbehandlung
- Verhinderung von DOS (Denial-of-service)
- Erstellung von Resumption Tokens

Benötigte Werkzeuge

- Code templates wenn erhältlich (für viele Sprachen erhältlich)
- Einfache Programmierumgebung
- XML Generatoren (nur für nicht-triviale Dinge..)
- Datenbank Bibliotheken / Treiber (e.g. DBI, ODBC, JDBC, PEAR)

Einfacher Programmablauf

```
parse WWW request to extract parameters
if (verb='Identify')
   ProcessIdentify;
else if (verb='ListMetadataFormats')
   ProcessListMetadataFormats;
else if (verb='ListSets')
   ProcessListSets;
else if (verb='GetRecord')
   ProcessGetRecord;
else if (verb='ListIdentifiers')
   ProcessListIdentifiers;
else if (verb='ListRecords')
   ProcessListRecords;
else
   ReportError ('badVerb');
```

Data Provider: Allgemeine Struktur

Argument Parser

- validiert OAI Anfragen

Error Generator

- erzeugt XML Antworten mit kodierten Fehlermeldungen

Database Query / Local Metadata Extraction

- holt Metadaten aus dem Repository
- bildet diese entsprechend dem angefragten Metadatenformat ab

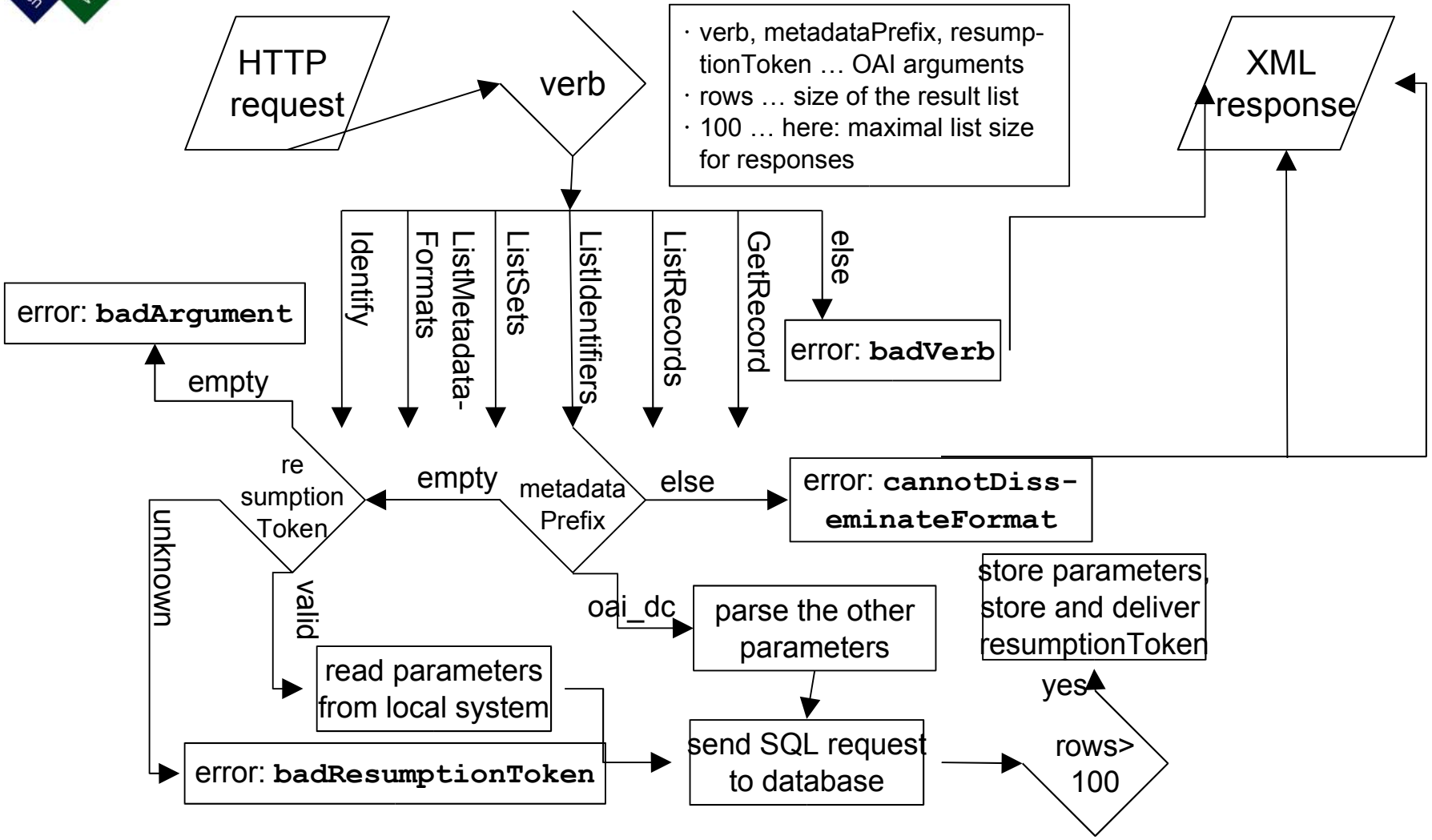
XML Generator / Response Creation

- erzeugt XML Antworten mit kodierter Metadaten information

Flow Control

- sorgt für unvollständige Listen bei “grossen” Repositories
- benutzt Resumption Tokens als Mechanismus

Data Provider: Flow Chart

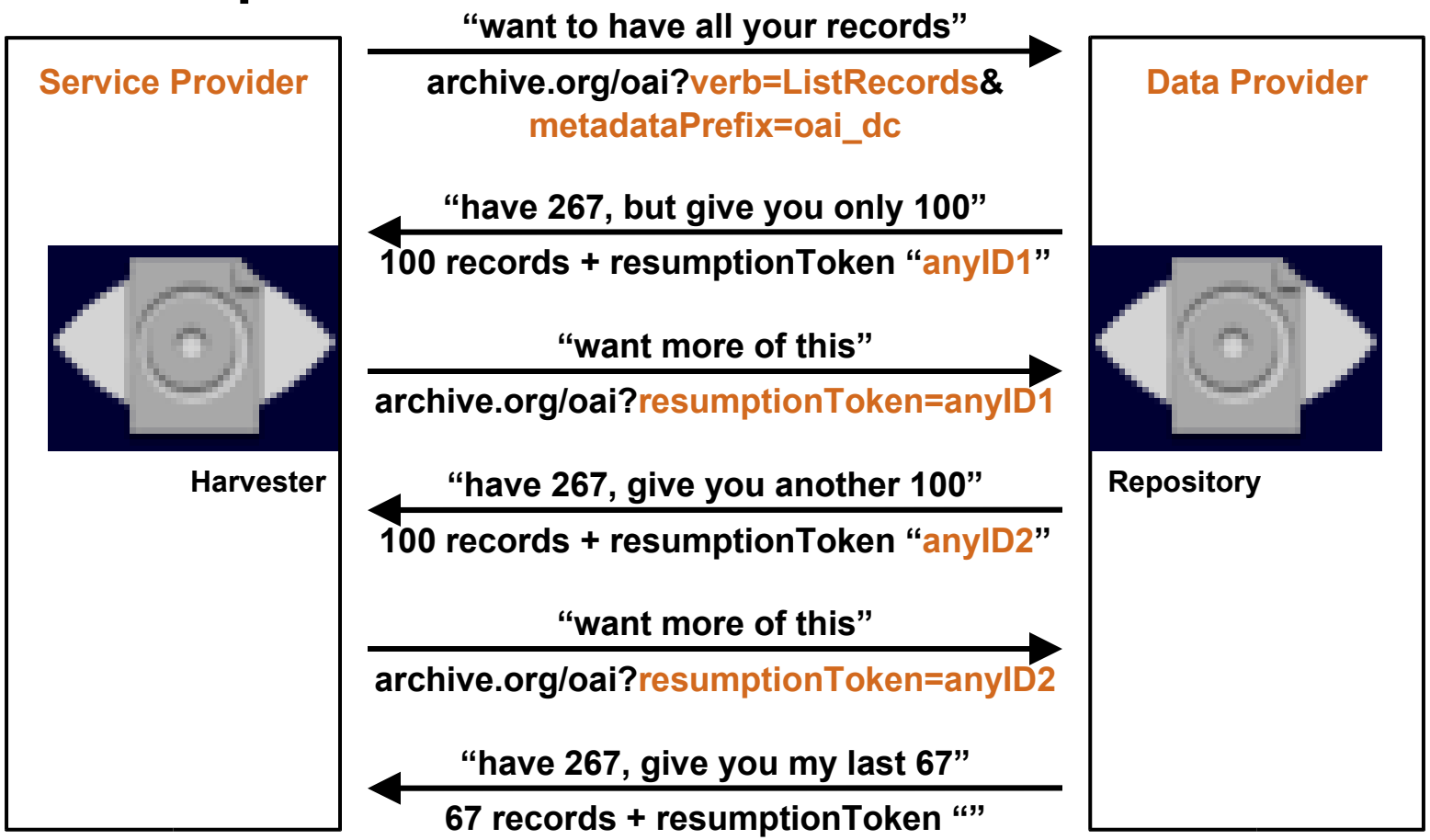


Data Provider: Resumption Token

- sollte für “große” Listen implementiert werden
- initiiert vom Data Provider
- speichere Parameter (**`set`**, **`from`**, …) und die Anzahl der bereits gelieferten Records
- Eigenschaften
  - expiration: **`expirationDate`** (optional)
  - **`completeListSize`** (optional)
  - bereits gelieferte Records: **`cursor`** (optional)
  - Wiederaufnahme nach Netzwerkfehlern (Möglichkeit, den letzten Resumption Token erneut aufzurufen)
- Probleme
  - Änderungen der Datenbank
  - zwei mögliche Lösungen
    - dupliziere Daten in einer Anfragetabelle
    - speichere Datum der ersten Anfrage mit den weiteren Parametern → nutze dies wie ein zusätzliches **`until`** Argument

Data Provider: Resumption Token (2)

Example



Resumption Token

```
- <record>
  - <header>
    - <identifier>
        oai:physdoc:http://www.logos-verlag.de/cgi-local/buch?isbn=607
      </identifier>
      <datestamp>2002-01-25T00:00:00Z</datestamp>
    </header>
  - <metadata>
    - <oai_dc:dc xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
      http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
      - <dc:title>
          Die Natur der Naturwissenschaften historisch verstehen
        </dc:title>
        <dc:date>2001-01-29</dc:date>
        <dc:identifier>http://www.logos-verlag.de/cgi-local/buch?isbn=607</dc:identifier>
      </oai_dc:dc>
    </metadata>
  </record>
  <resumptionToken expirationDate="2003-03-27T00:01:10Z" completeListSize="319"
  cursor="0">664850492</resumptionToken>
 </ListRecords>
</OAI-PMH>
```

Metadatenerzeugung

- Vorgehensweisen:
  - Bilde von der Quelle in jedes Metadatenformat ab
  - Benutze „crosswalks“ (vielleicht sogar XSLT) um weitere Formate zu generieren

source		dc		rfc1807
name	=	title	=	title
author	=	creator	=	author

Data Provider: Datenrepräsentation

- benutze empfohlene Datenrepräsentation
  - Datum
    - ✓ 2002-12-05
    - ✗ 2002-xx-xx, 2002, 05.12.2002
  - Sprachcode
    - ✓ eng, ger, ...
    - ✗ en, de, english, german
- mehrfache Werte: benutze ein eigenes XML Element für jede Entität
  - author
    - ✓ <dc:creator>Smith, Adam</dc:creator> <dc:creator>Nash, John</dc:creator>
    - ✗ <dc:creator>Smith, Adam; Nash, John </dc:creator>

Kodierung der Daten für XML

- Spezielle XML Zeichen müssen „escaped“ werden
- Konvertiere nach UTF-8 (Unicode)
- Konvertiere Entitäten (Umlaute)
- Entferne überschüssige Leerzeichen
- Konvertiere CR/LF für Paragraphen
- URLs
  - /?#=&:;+ müssen als Escape Sequenz kodiert werden

Data Provider: Komprimierung

- Methode um Netzverkehr zu verringern
- Optional for beide Seiten: Data and Service Provider
- wird über HTTP ausgehandelt
- Harvester können einen **`Accept-Encoding`** Header in ihre Anfragen einbauen
- Harvesters ohne **`Accept-Encoding`** header bekommen immer unkomprimierte Daten
- Repositories müssen HTTP **`identity`** encoding unterstützen
- Repositories sollten unterstützte Encodings durch **`compression`** Elemente in ihrer **`Identify`** Antwort angeben

mögliches Cachen von Ergebnissen

- für mehrfache Anfragen von mehreren Clients oder um Teilantworten zu beantworten
- temporäre Daten verfallen, wenn sie nicht länger gebraucht werden
- Ist dies notwendig, um Zeitraumanfragen zu beantworten, während gleichzeitig neue Einträgehinzugefügt werden, während das Harvesting durchgeführt wird?

Fehlerbehandlung

- Alle Protokollfehler werden im XML Format zurückgegeben
  - **`badVerb`**

    illegales Verb angefragt
  - **`badArgument`**

    illegale Parameterwerte oder -kombinationen
  - **`badResumptionToken`**, **`cannotDisseminateFormat`**, **`idDoesNotExist`**

    Parameters sind in korrektem Format, sind aber nicht legal unter den derzeitigen Bedingungen
  - **`noRecordsMatch`**, **`noMetadataFormats`**, **`noSetHierarchy`**

    leere Antwort

Fehler und Ausnahmen

```
- <OAI-PMH xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
  http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
    <responseDate>2003-03-26T00:06:56Z</responseDate>
    <request>http://physnet.uni-oldenburg.de/oai/oai2.php</request>
   - <error code="badVerb">
       The verb 'ListeAlles' provided in the request is illegal.
    </error>
  </OAI-PMH>
```

Vorbeugung gegen Denial-of-Service

- Gib nur Teilantworten und lass den Service Provider mit einem Resumption Token nach mehr fragen
- Benutze ggf. 503 retry-after HTTP Fehler, so dass Clients nach einer spezifierten Zeit erneut anfragen
- Benutze ggf. Zugriffslisten (access control lists), um den Zugriff auf das Archiv zu beschränken
- Warte einen gewissen Zeitraum, bevor die Ergebnisse zurückgesendet werden

Übliche Probleme

- Keine eindeutigen Identifier!
- Keine Zeitmarken (Datestamps) !
- Unvollständige Information in der Datenbank
- Neues Metadatenformat
- XML Antworten ungültig (not validating)

Keine eindeutigen Identifier

- Erzeuge eine unabhängige Abbildung auf die Identifier
- Benutze die Zeilennummern der Datenbank
- Benutze Dateinamen für die Daten in Dateien
- Erzeuge einen Hash von anderen Feldern
  - Z.B. Autor+Jahr+erstes Wort des Titels

Keine Zeitmarken (datestamps)

- Wenn es wirklich nicht anders gehen sollte...
- Ignoriere die Datestampparameter und markiere alle Records mit dem aktuellen Datum
- Erzeuge eine Datumstabelle mit dem aktuellen Datum für alle alten Einträge und aktualisiere das Datum für neue und geänderte Einträge
- Am wichtigsten: Jeder Harvesting Algorithmus, der interoperabel stabil für ein Archiv mit wirklichen Datumsangaben ist, sollte auch stabil für ein Archiv mit synthetischen Datumsangaben sein

Unvollständige Information

- Synthetisiere Metadatenfelder aus dem vorherigen Wissen über die Daten
  - Beispiel: Verlag und Sprache können für viele Archive fest kodiert sein
- Lasse Felder, die nicht korrekt gefüllt werden können, weg – es ist besser, weniger Information als falsche Information anzubieten!

Neues Metadatenformat

- Finde die Beschreibung, den Namensraum und den formalen Namen des Standards
- Finde eine XML Schema Beschreibung des Datenformats
  - Wenn keines existiert, schreib selbst eins (lass Dich ggf. von anderen OAI-Leuten beraten)
- Erzeuge eine Abbildung und teste, so dass es XML Schema Validation erfolgreich durchläuft.

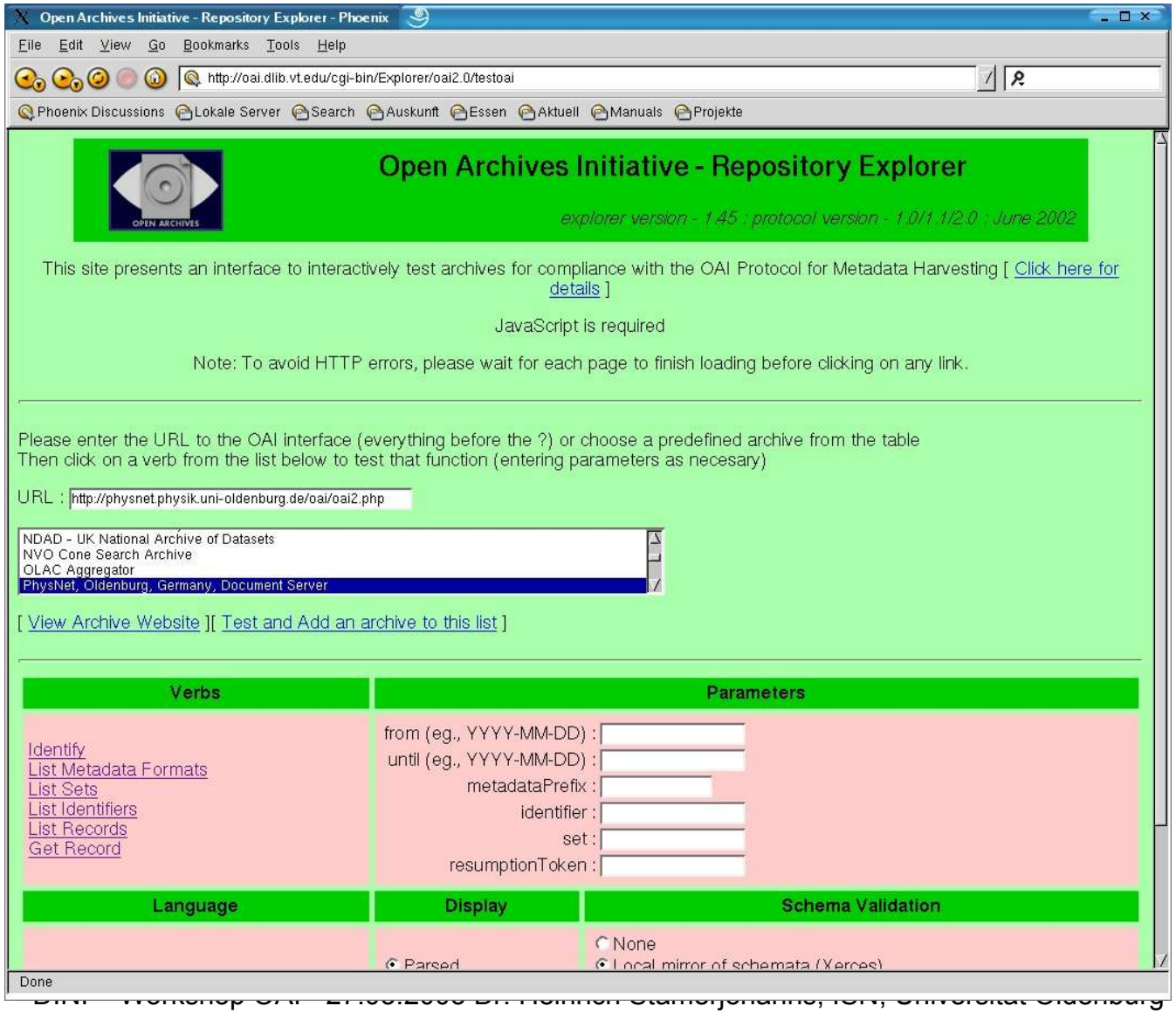
Ungültiges (not validating) XML

- Prüfe Namespaces und Schema
- Benutze den Repository Explorer im nicht-prüfenden (non-validating) Modus um die XML-Struktur, so dass nicht die Namespaces oder Schemata betrachtet werden
- Validiere das Schema selbst, wenn es kein Standardschema ist
- Sieh Dir das XML, welches von anderen Repositories erzeugt wird, an
- Beachte übliche Probleme beim Character encoding (iso8859-1 --> utf-8)

Werkzeuge zum Testen

- Repository Explorer
  - Interaktives Browsing
  - Testen von Parameters
  - Mehrfache Ansichten auf Daten
  - Multilinguale Unterstützung
  - Automatische Tests
- OAI Registry
- XML Schema Validator

RE Interaktives Browsing



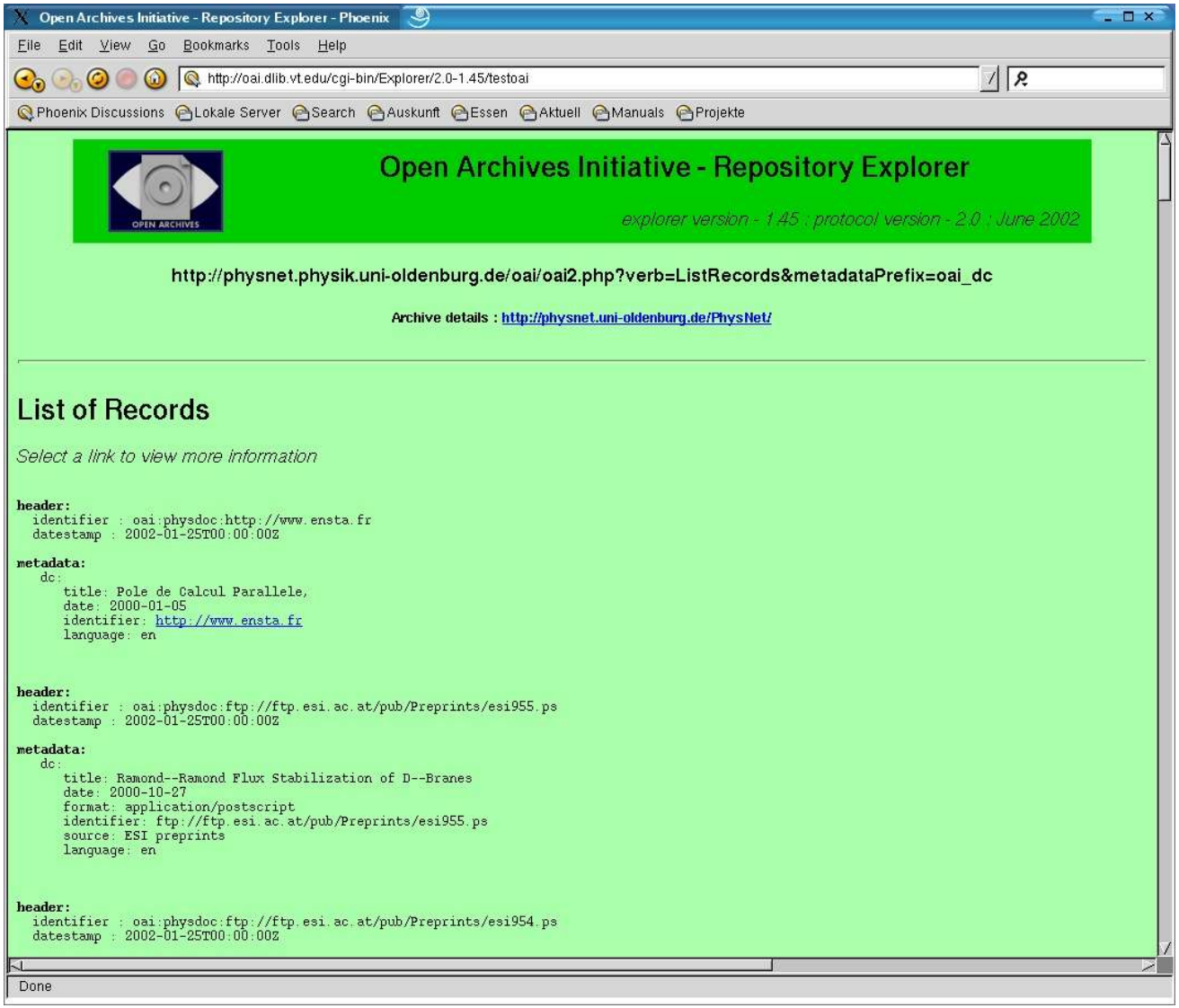
RE Parameter Test

Verbs	Parameters
Identify List Metadata Formats List Sets List Identifiers List Records Get Record	from (eg., YYYY-MM-DD) : until (eg., YYYY-MM-DD) : metadataPrefix : identifier : set : resumptionToken :

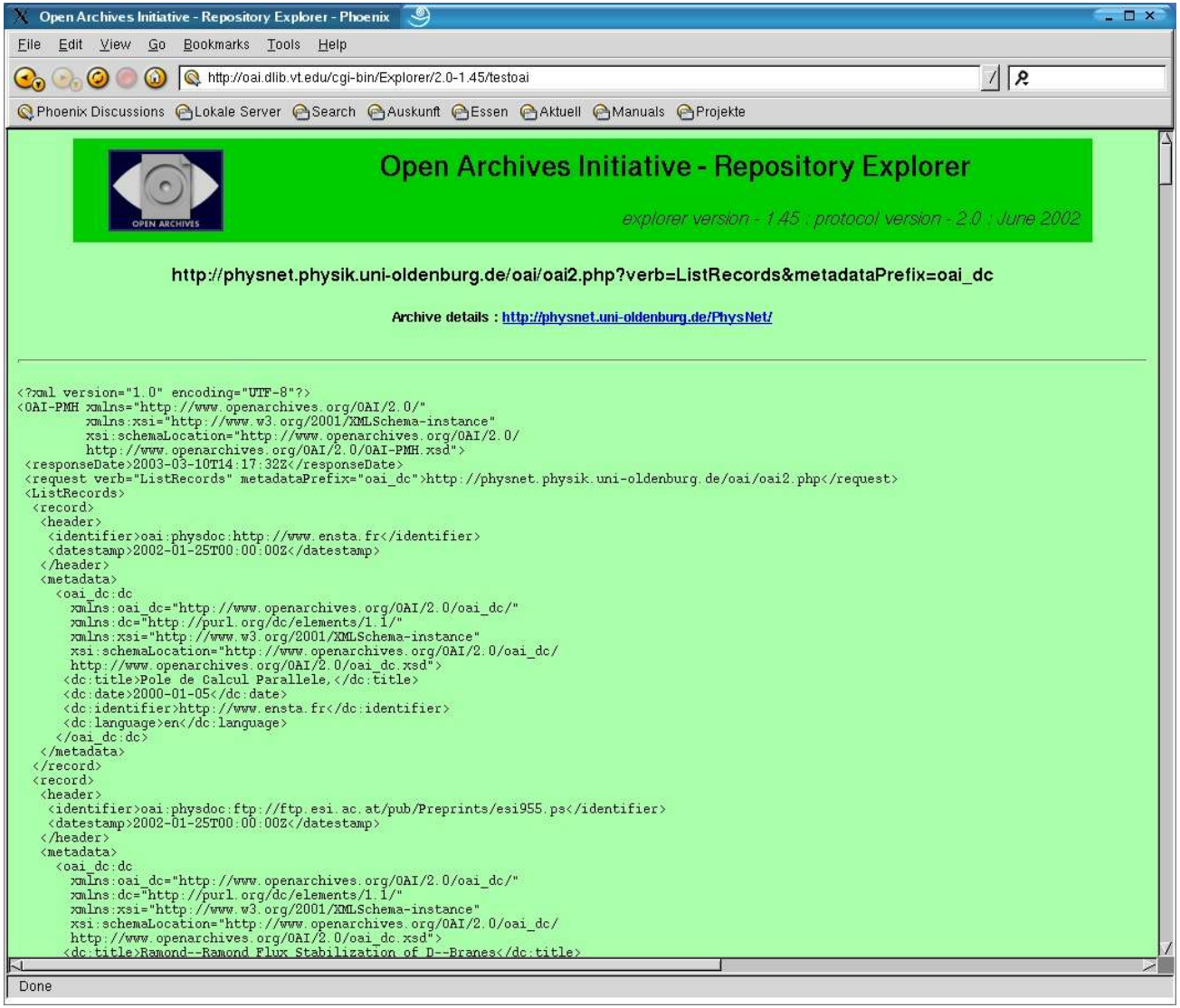
Language	Display	Schema Validation
English	◉ Parsed ○ Raw XML ○ Both	○ None ◉ Local mirror of schemata (Xerces) ○ Online schemata (Xerces) ○ Local mirror of schemata (XSV) ○ Online schemata (XSV)

home about Send all comments to hussein@vt.edu --- Digital Library Research Laboratory@Virginia Tech

RE Browsing


Open Archives Initiative - Repository Explorer - Phoenix
File Edit View Go Bookmarks Tools Help
http://oai.dlib.vt.edu/cgi-bin/Explorer/2.0-1.45/testoai
Phoenix Discussions Lokale Server Search Auskunft Essen Aktuell Manuals Projekte
Open Archives Initiative - Repository Explorer
explorer version - 1.45 ; protocol version - 2.0 ; June 2002
http://physnet.physik.uni-oldenburg.de/oai/oai2.php?verb=ListRecords&metadataPrefix=oai_dc
Archive details : http://physnet.uni-oldenburg.de/PhysNet/
List of Records
Select a link to view more information
header:
identifier : oai:physdoc:http://www.ensta.fr
datestamp : 2002-01-25T00:00:00Z
metadata:
dc:
title: Pole de Calcul Parallele,
date: 2000-01-05
identifier: http://www.ensta.fr
language: en
header:
identifier : oai:physdoc:ftp://ftp.esi.ac.at/pub/Preprints/esi955.ps
datestamp : 2002-01-25T00:00:00Z
metadata:
dc:
title: Ramond--Ramond Flux Stabilization of D--Branes
date: 2000-10-27
format: application/postscript
identifier: ftp://ftp.esi.ac.at/pub/Preprints/esi955.ps
source: ESI preprints
language: en
header:
identifier : oai:physdoc:ftp://ftp.esi.ac.at/pub/Preprints/esi954.ps
datestamp : 2002-01-25T00:00:00Z
Done

RE: Darstellung von XML



OAI Registry

The Open Archives Initiative

Registering as a Data Provider

Data providers who support the OAI protocol may choose to list their repository in the OAI registry. The goals of the registry are:

- Provide a publicly accessible list of OAI conformant repositories, making it easy for service providers to discover repositories from which metadata can be harvested.
- Provide a mechanism for data providers to ensure their conformance with the OAI protocol specification.
- Provide a means for the OAI to monitor use of the protocol and plan future activities and strategies.

This page allows you to register your repository by entering your BASE-URL in the text box at the bottom of this page. *Before* doing that, please read all of this instruction page so you understand what registration means and the choices you have.

Consequences of Registration
Protocol Testing
Conformance Testing

XSV Schema Validator

W3C XML LTG

Validator for XML Schema REC (20010502) version

XSV version: XSV 2.3-1 of 2003/02/14 09:39:35

NOTICE: This is an Beta Test of a service for a **approved recommendation**. This version is for schema documents with the namespace URI `http://www.w3.org/2001/XMLSchema` and is being actively developed: see XSV for XML Schema 20000922 version for the no longer maintained previous version, for schema documents with the namespace URI `http://www.w3.org/2000/10/XMLSchema`, and XSV for XML Schema 200004007 version for the no longer maintained even earlier version, for schema documents with the namespace URI `http://www.w3.org/1999/XMLSchema`.

Use this form for checking a schema which is accessible via the Web, and/or schema-validating an instance with a schema of your own.

Address(es):

☐ Show warnings ☐ Keep Going ☐ Contribute

Service Provider

- Voraussetzungen
- Struktur
- Architektur
- Einsammeln.. (harvest)
- Vorgehensweise
- Intermediate systems
- Werkzeug

Service Provider: Voraussetzungen

- Server, mit Internetanbindung
- Datenbank (Relational oder XML)
- Programmierumgebung
  - kann HTTP Anfragen an andere Webserver stellen
  - kann die (lokale) Datenbank abfragen
  - XML Parser
  - Anpassung von Skripten oder Programmen

Service Provider: Struktur (1)

Archiv Management

- Auswahl der Archive von den gesammelt werden soll
- manuelle Eingabe oder
- automatische Hinzufügung/Löschung von Archiven mittels der offiziellen Registry

Anfrage Komponente

- erzeugt HTTP Anfragen und sendet sie an OAI Archive (Data Provider)
- verlangt Metadaten mittels OAI-PMH
- möglicherweise selective harvesting (`set` Parameter)

Service Provider: Struktur (2)

Scheduler

- sorgt für regelmässige Abfragen von den Archiven
- einfachster Fall: manueller Start
- sonst: z.B. cron job, relationale Datenbank …

Flusskontrolle (Flow control)

- resumption token: weitere Anfragen bei Rückgabe eines Resumption Tokens
- HTTP Fehler 503 (service not available) – Analyse der Antwort um das Archiv nach “retry-after” Zeitraum erneut anzufragen

Service Provider: Struktur (3)

Update Mechanismus

- fügt alte und neue Daten zusammen (oder ersetzt diese)
- einfachster Fall: lösche alle Einträge mit “alten” Metadaten, bevor diese von einem Archive eingesammelt werden
- besser: inkrementelle Aktualisierung (`from` parameter) – füge *neue* Metadaten ein und überschreibe *geänderte / gelöschte* Metadaten (anhand der eindeutigen Identifier)

XML Parser

- analysiert die Antworten von den Archiven
- Validierung (Validation) anhand des XML Schemas
- extrahiert die Metadaten
- transformiert die Metadaten in eine interne Datenstruktur

Service Provider: Struktur (4)

Normalisierer und Mapper

- normalisiert die Darstellung (z.B. Datum, Autor, Sprachcode)
- transformiert die Daten in eine homogene Struktur (bei unterschiedlichen Metadaten Formaten)

Datenbank

- Abbildung der XML Struktur der Metadaten in eine relationale Datenbank (mehrfache Werte)
- oder: nutze eine XML Datenbank

Service Provider: Struktur (5)

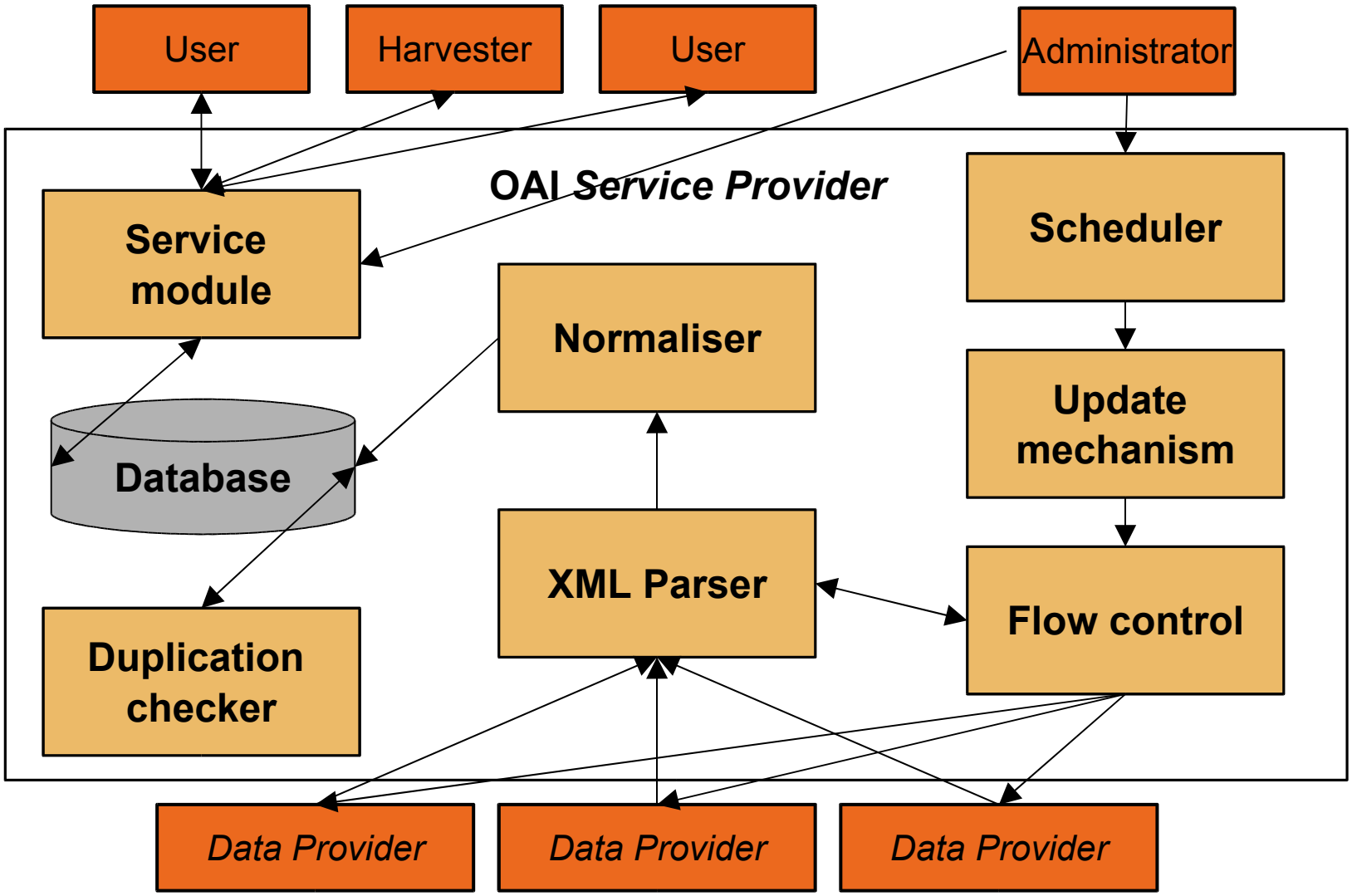
Duplication Checker

- fügt identische Records von verschiedenen Data Providern zusammen
- z.B.: eindeutiger Identifier für ein Item (z.B. URN, …)
- jedoch: oft nicht einfach zu handhaben und nicht fehlerfrei

Dienste (Service Module)

- bietet den Dienst für die “Öffentlichkeit” an
- Basis: eingesammelte und gespeicherte Records der Archive
- benutzt ausschliesslich die lokale Datenbank für Suchen

Service Provider: Architektur



Wie man einsammelt (harvest)

- Identify, um einfache Information zu bekommen
- ListIdentifiers, danach ListMetadataFormats für jeden Record, anschliessend GetRecord für jede Kombination von id/metadaten
  - Anzahl (#) der kurzen HTTP Anfragen = 1+n+n x m
    n = # der Identifier, m = # der metadatenformaten
- ListRecords für jedes benötigte Metadatenformat
  - Anzahl (#) langer HTTP Anfragen = m
    m = # of metadata formats

Vorgehensweise

- Benutze einen Zeitplan (schedule), um regelmässig einzusammeln
- Speichere das Datum des letzen Einsammeln (bevor man startet)
- Benutze eine Überschneidung von zwei Tagen (oder einen Tag, wenn das Archiv korrekte UTC datestamps benutzt)
  - Neue Einträge können für den aktuellen Tag hinzugefügt werden
  - (Zeitzonen erzeugen einen Lag von bis zu einem Tag, wenn sie ignoriert werden)
  - Wenn die Quelle korrekte UTC datestamps benutzt, dann ist nur eine Überlappung von 1 Sekunde notwendig!
- Jedes Mal, wenn ein Record behandelt wird, lösche frühere Instanzen dieses Records

Intermediate Systems

- Zugleich Data- und Service Provider
- Alle eingesammelten Daten müssen auf die Zeitmarke gestzt werden, zu welcher die Daten eingesammelt worden sind
- Identifier müssen ihren Ursprungswert behalten
- Beachte: Konsistenz in der Quelle wird weitergegeben, aber natürlich auch inkonsistente Daten!

Werkzeuge

- Prüfe OAI Website für Beispielcode
- XML parser – abhängig von der Plattform – prüfe W3C
- XML Schema validator
  - Sehr wenig verfügbar – die Referenzversion läuft, aber kann vielleicht nicht einfach zu installieren sein
  - Ignoriere Validation, wenn man der Quelle vertrauen kann
- Beispiel Data Providers – prüfe die OAI website für eine Liste von öffentlichen Archiven

OAI Communities

- gemeinsame Metadatenformate
- gemeinsame Semantik
- Geschlossene OAI Netzwerke
- OAI innerhalb von Digital Libraries (DL)

Gemeinsame Metadatenformate

- Benutze Metadatenformate, die innerhalb einer Community akzeptiert werden, um spezifischere Information zu bekommen.
- Beispiele
  - E-Print format (under development)
  - ETD-MS for theses and dissertations
  - VRA Core for multimedia
  - IMS Metadata for educational material

Gemeinsame Semantik

- Entwickle ein gemeinsames Verständnis für die Bedeutung von einzelnen Feldern
- Beispiele
  - Entwickle kontrolliertes Vokabular für Felder
  - Benutze spezifische Felder für externe Links (OAI empfiehlt die Benutzung von Identifier in DC, um dieses zu erreichen)
  - Wähle aus existierenden Standards (z.B. ISO 601-3 für Sprachnamen)

Geschlossene OAI Netzwerke

- Data Provider müssen nicht öffentlich zugänglich sein!
- Innerhalb einer Organisation kann OAI für den Transfer zwischen heterogenen Systemen benutzt werden
- Bessere Kontrolle über den Nutzen, so dass globale Optimierungen möglich sind (z.B. harvesting schedules und Wahl der Metadatenformate)

OAI innerhalb von DL

- Benutze das OAI Protokoll als Basis für Komponenten, die miteinander kommunizieren
- Beispiele
  - Suchmaschinen könnten dynamische Mengen (sets) benutzen
  - Browsing kann über Sets geführt werden
  - Reviews and Anmerkungen können unabhängige OAI Data Provider sein
- Open Digital Libraries Projekt, um diese Vorgehensweise zu untersuchen
  - http://oai.dlib.vt.edu/odl

Links

- Open Archives Initiative
  http://www.openarchives.org
- OAI Metadata Harvesting Protoco
  http://www.openarchives.org/OAI/openarchivespr
- Virginia Tech DLRL OAI Project
  http://www.dlib.vt.edu/projects/OAI/
- Repository Explorer
  http://purl.org/net/oai_explorer
- NDLTD
  http://www.ndltd.org

Noch mehr Links

- ARC Cross-Archive Search Service
  http://arc.cs.odu.edu/
- XML Schema Validator
  http://www.w3.org/2001/03/webdata/xsv
- Dublin Core Metadata Initiative
  http://www.dublincore.org
- E-Prints DL-in-a-box
  http://www.eprints.org
- XML Tools at W3C
  http://www.w3.org/XML/#software

Gliederung des Tutoriums

- Teil I - Geschichte und Überblick
- Teil II - Technische Einführung
- Teil III - Implementation von Data Provider und Service Provider
- Kaffeepause 15:45 - 16:15
- Teil IV - OAI Serviceprovider - Beispiele
- Teil V - Metadaten
- Teil VI - Sets
- Teil VII - Realisierung auf Verbundebene

Teil V Metadaten

(direkt übernommen von Andy Powell)

1. Einführung
2. XML Schema Details
3. Erweiterung von oai_dc für die eigene Anwendung
4. Nutzung von IMS Metadaten als neues Recordformat

Einführung

- OAI-PMH nutzt XML Schemas, um das Recordformat zu definieren
- man kann beliebige Daten über OAI-PMH austauschen, solange diese als XML transportiert werden können und man eine XML-Schema dafür erstellen kann
- OAI-PMH erfordert das ‘oai_dc’ XML schema
- Die OAI-PMH Dokumentation beschreibt auch die Nutzung von XML schema um

  rfc1807: a schema for rfc1807 format metadata;
  marc21: a recommended schema for MARC21 metadata, provided by the Library of Congress;
  oai_marc: a schema for MARC format metadata

  –auszutauschen

Nähere Betrachtung von oai_dc

- das einfache DC schema als notwendiges Recordformat in OAI-PMH definiert ein Container Schema
- Container Schema ist OAI-spezifisch
- Container Schema liegt auf der OAI Website
- importiert ein generisches DCMES Schema
- das generische DCMES Schema liegt auf DCMI Website
- Das gleiche Modell wird gleichsam für das ‘qualified’ DC Schema genutzt – ein Container Schema bei OAI, das generische Schema bei DCMI

Ein oai_dc Record…

- Ein Beispiel: oai_dc record (repository explorer)
- hier ist die volle GetRecord Antwort
- drei wichtige Dinge…
- Namespace für das oia_dc format

  xmlns:oai_dc=http://www.openarchives.org/OAI/2.0/oai_dc/
- Namespace für DCMES Elemente

  xmlns:dc=http://purl.org/dc/elements/1.1/
- Container schema, welches mit dem oai_dc namespace verbunden wird:

  xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
  http://www.openarchives.org/OAI/2.0/oai_dc.xsd"

XML Schemas

- Das oai_dc Container Schema
  http://www.openarchives.org/OAI/2.0/oai_dc.xsd
- importiert DCMES Schema von
  http://dublincore.org/schemas/xmls/simpledc20020312.xsd
- Definiert ein Containerelement, welches ‘dc’ genannt wird
- listet die erlaubten Elemente innerhalb des ‘dc’ Container (von dem obigen DCMES namespace/ schema)

Wem oai_dc nicht genug ist

- wenn die 15 DCMES Elemente beschränkt – z.B. das Hinzufügen von weiteren Metadatenelementen
- wenn man genauere Präzision in den Metadatenelementen benötigt – z.B. durch das Hinzufügen von ‘encoding schemes’ zu existierenden Elementen
- wenn man andere Metadatenformate austauschen möchte
  - IMS/IEEE LOM – eLearning metadata
  - ODRL – Open Digital Rights Language

Erweitern des oai_dc Schemas

- einfaches Szenario…
- man benutzt zur Zeit oai_dc schema um Records auszutauschen, möchte jedoch ein weiteres Element *accessControl* hinzufügen
- (Dies ist kein echtes Szenario)

  qualified DC ist zu kompliziert für dieses Tutorial

1 – Name des Metadatenformats

- das neue Metadatenformat braucht einen Namen
- Wir wählen z.B.

  rdn_dc
- angelehnt an die Namensgebung von OAI
- Alternative Möglichkeiten

  rdndc

  rdn

  etc.

2 – Erzeuge Namespaces

- zwei Namensräume werden benötigt...
- Namensraum für das rdn_dc Format

  http://www.rdn.ac.uk/oai/rdn_dc/
- Namensraum für die neuen Metadatenelemente (Eigenschaften), die wir in diesem neuen Format nutzen

  http://purl.org/rdn/terms/
- Beachte:
  - die Nutzung von Purl für diesen Elementnamensraume folgt der DCMI Nutzung, aber ist nicht notwendig
  - jedoch sollten beide URIs, die auf diese Namensräume hinweisen, unter eigener Kontrolle sein, um die Eindeutigkeit sicherzustellen und sich die Weiternutzung vorzubehalten
  - Hinter den URIs braucht nichts spezifisches zu finden zu sein

3 – lokale Kopie des DC Schemas

- erstelle lokale Kopie des DCMES Schema
- in diesem Falle ist die Kopie unter
  - http://www.rdn.ac.uk/oai/rdn_dc/20021204/dc.xsd

  finden
- dieser Schritt ist nicht wirklich notwendig
- wahrscheinlich ist es sogar nicht gut
- aber, derzeit gibt es kleinere Probleme mit der DCMI Kopie des Schemas
- …mit lokalen Kopien lässt sich einfacher arbeiten

4 – Schema für neue Begriffe

- erzeuge ein XML Schema für die neuen ‘rdnterms’
- in this case the schema is available at
  http://www.rdn.ac.uk/oai/rdn_dc/20021204/rdnterms.xsd
- das Schema definiert ein neues Element/ Eigenschaft
  accessControl
- und fügt es zu dc:any group
- erzeugt auch einen neuen Containertyp
  rdnterms:elementContainer
- beachte:
  Schema URI enthält ein datestamp
  dies sollte weitere Erweiterungen des Schemas vereinfachen

5 – Container Schema

- erzeuge ein neues Container Schema für das neue Recordformat
- in diesem Falle ist das Schema unter

  http://www.rdn.ac.uk/oai/rdn_dc/20021204/rdn_dc.xsd

  zu finden
- dies importiert einfach das rdnterms Schema
- dann wird ein Containerelement namens ‘rdndc’ des Typs

  rdnterms:elementContainer
- wiederum enthält der Schema URI ein datestamp

6 – validieren, validieren

- erzeuge einige test records, die das neue Schema nutzen

  http://www.rdn.ac.uk/oai/rdn_dc/20021204/test.xml

  http://www.rdn.ac.uk/oai/rdn_dc/20021204/oai-test.xml

- benutze den XML Schema validator unter

  http://www.w3.org/2001/03/webdata/xsv

7 – ListMetadataFormats

- füge Information über das neue Format zur Antwort auf die Anfrage der ‘ListMetadataFormats’ Anfrage…

```
…
<metadataFormat>
<metadataPrefix>rdn_dc</metadataPrefix>
<schema>http://www.rdn.ac.uk/oai/rdn_dc/20021113/rdn_dc.xsd</schema>
<metadataNamespace>http://www.rdn.ac.uk/oai/rdn_dc/</metadataNamespace>
</metadataFormat>
…
```

8 – andere Verben

- modifiziere die Antwort auf ‘ListSets’, ‘ListIdentifiers’, ‘ListRecords’ und ‘GetRecord’ Anfragen
- akzeptiere ‘metadataPrefix’ für den neuen Formatnamen
- gib Records, die entsprechend dem neuen Schema formatiert sind, zurück

9 – erneutes Validieren

- nutze den Repository Explorer, um zu überprüfen, dass
- alle Anfragen mit dem neuen ‘metadataPrefix’ funktionieren
- dass das oai_dc Format immer noch funktioniert!
- die richtigen Records für jedes Format zurückgegeben werden
- die Antworten korrekt validiert werden

Zusammenfassung

- entscheide Dich für ein neues Metadatenformat und die passenden Namespaces
- entwickle XML Schemas für Container und neue Elemente
- erzeuge Test Records und validiere
- modifiziere dein Repository (Quellcode und/oder Konfgurationsdateien), um das neue Format zu unterstützen
- validiere und teste das Repository

Andere Recordformate

- man kann eine ähnliche Vorgehensweise mit anderen Metadatenformaten anwenden
  - IMS/IEEE LOM
  - ODRL
- in diesen Fällen gab es bereits Übereinkunft über XML Schemas und Namespaces
- Installation dieser Formate sollte einfacher sein, weil man nicht sein eigenes Schema definieren muss…
  - Aber… XML Schemaspezifikationen werden derzeit kontinuierlich geändert, so dass es immer kleinere Anpasungen notwendig sind.

Füge Unterstützung für IMS hinzu

- modifziere die ‘ListMetadataFormats’ Antwort

```
…
<metadataFormat>
<metadataPrefix>ims</metadataPrefix>
<schema>http://www.imsglobal.org/xsd/imsmd_v1p2p2.xsd</schema>
<metadataNamespace>
  http://www.imsglobal.org/xsd/imsmd_v1p2
</metadataNamespace>
</metadataFormat>
…
```

- erweitere ‘ListSets’, ‘ListIdentifiers’, ‘ListRecords’ und ‘GetRecord’ Antworten

  akzeptiere den 'ims' ‘metadataPrefix’ und gib die entsprechenden records entsprechend formatiert zurück

Zusammenfassung

- Wir hoffen, dass Sie
- einen Überblick über die Geschichte des OAI-PMH und einen Überblick über die wichtigsten Eigenschaften bekommen haben
- einen tieferen technischen Einblick bekommen haben, wie das Protokoll funktioniert
- ein bisschen über die wichtigsten Implementationsmerkmale bekommen haben
- einige nützliche Startpunkte gefunden haben, um eigene Implementationen durchzuführen

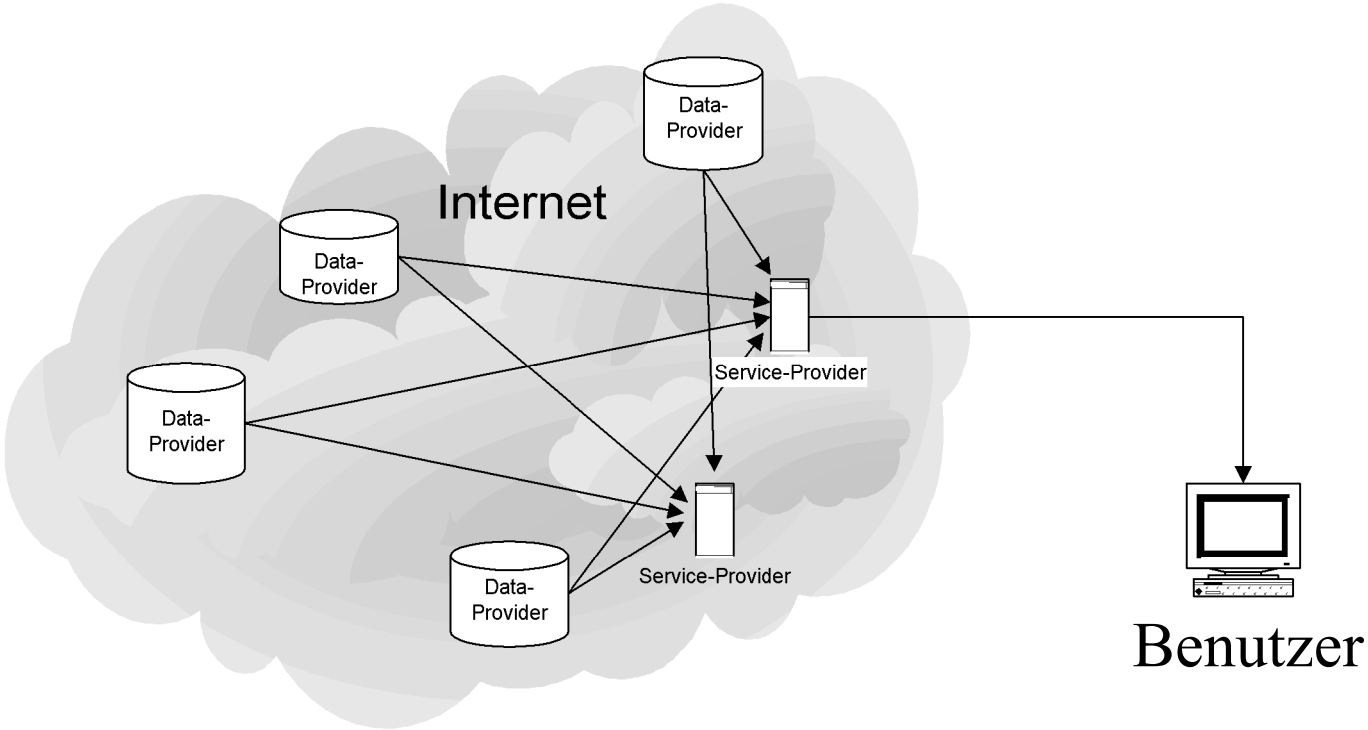
Gliederung des Tutoriums

- Teil I - Geschichte und Überblick
- Teil II - Technische Einführung
- Teil III - Implementation von Data Provider und Service Provider
- Kaffeepause 15:45 - 16:15
- Teil IV - OAI Serviceprovider - Beispiele
- Teil V - Metadaten
- Teil VI - Sets
- Teil VII - Realisierung auf Verbundebene

Übersicht zum Teil VI Sets

- Protokollspezifikation
- Möglichkeiten von Sets - Beispiele
- Empfehlungen der DINI Arbeitsgruppe
  - inhaltlicher Set
  - Formaler Set (Publikationstyp)
  - Formaler Set (Dokumenttyp)
- Zusammenfassung

Zusammenwirken von Daten - und Serviceprovidern



Aussagen der Protokollspezifikation zu Set

- **A *set* is an optional construct for grouping items for the purpose of selective harvesting.** Repositories **may** organize items into sets. Set organization **may** be flat, i.e. a simple list, or hierarchical. Multiple hierarchies with distinct, independent top-level nodes are allowed. Hierarchical organization of sets is expressed in the syntax of the `setSpec` parameter as described below. When a repository defines a set organization it **must** include set membership information in the headers of items returned in response to the `ListIdentifiers` , `ListRecords` and `GetRecord` requests.

Bedeutung der Sets

- Suche für SP nach inhaltlichen und formalen Kriterien nur über Sets möglich.
- Eine inhaltliche Strukturierung ist nur mit Sets möglich.
  - Aufbau von Hierarchien
  - Dokumente können in mehreren Sets vorkommen
- Information zu Sets muss im Header mit übergeben werden. Neu in der Version 2.0!

Bestandteile des Set

- Jedes Set besteht aus :
  - setSpec -- a colon [:] separated list indicating the path from the root of the set hierarchy to the respective node.
  - setName -- a short human-readable string naming the set.
  - setDescription -- an **optional** and repeatable container that **may** hold community-specific XML-encoded data about the set

Aufforderung zu Absprachen

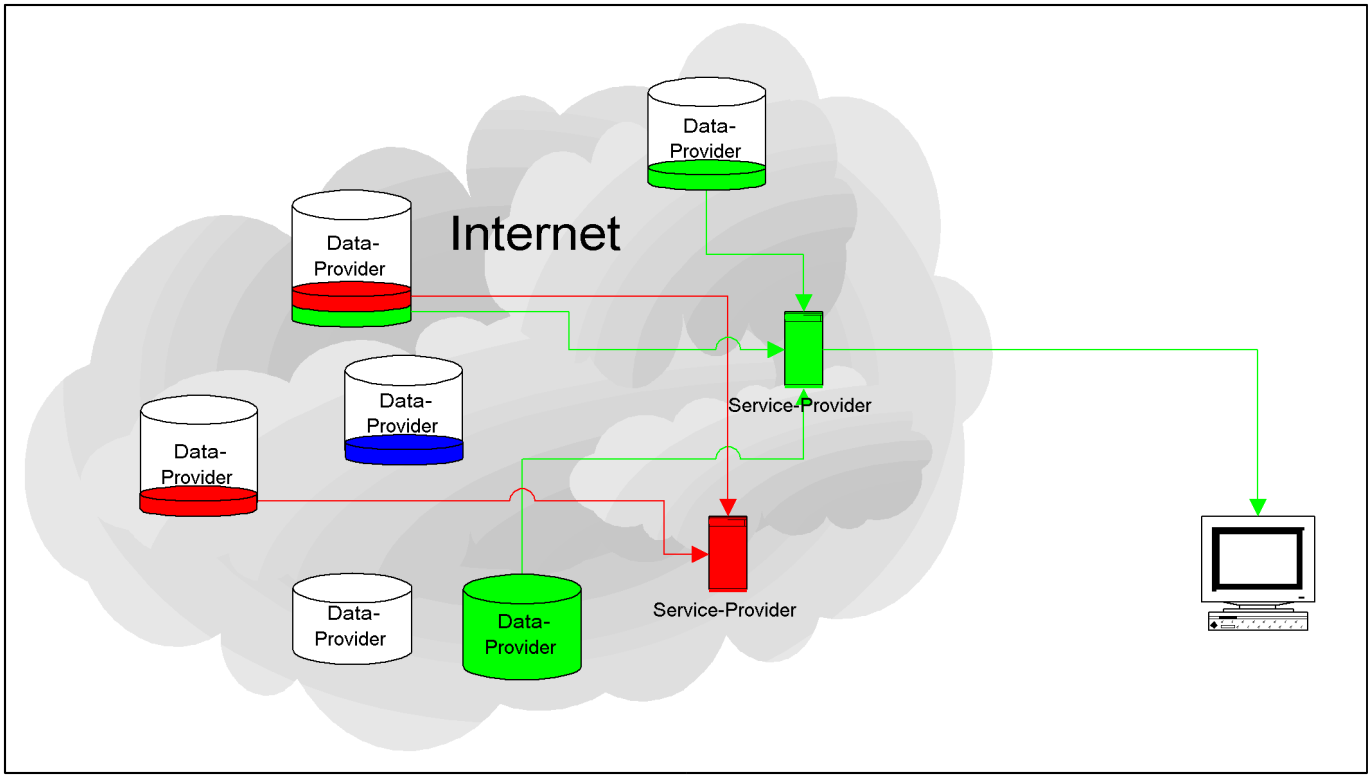
- The actual meaning of a set or of the arrangement of sets in a repository is not defined in the protocol. It is expected that individual communities may formulate well-defined set configurations with perhaps a controlled vocabulary for `setNames` and `setSpec`, and may even develop mechanisms for exposing these to harvesters.

- Dini-Empfehlungen zu Sets

Übersicht zum Teil VI Sets

- Protokollspezifikation
- Möglichkeiten von Sets - Beispiele
- Empfehlungen der DINI Arbeitsgruppe
  - inhaltlicher Set
  - Formaler Set (Publikationstyp)
  - Formaler Set (Dokumenttyp)
- Zusammenfassung

Benutzung von Sets



Beispiele für Datenprovider mit Sets

- Es gibt bei den Datenprovidern die zwei Ordnungstypen:

- Nach rein formalen Kriterien:

  Beispiel: Archiv American Memory (alt)

  Department of Electronics and Computer Science, University of Southampton (Status)

- Und nach rein inhaltlichen Kriterien:

  Beispiel: CogPrint (alt)

Open Archives Initiative - Repository Explorer

explorer version - 1.1 : protocol version - 1.0 : April 2001

http://memory.loc.gov/cgi-bin/oai1_0?verb=ListSets

Archive details : http://memory.loc.gov

List of Sets

Click on the link to list the contents

LC Maps
LC Dance Instruction Manuals
LC Sheet Music
LC Early Motion Pictures

Request URL : http://memory.loc.gov/cgi-bin/oai1_0?verb=ListSets
Response Date : 2001-06-16T15:30:52-04:00

Beispiel American Memory

```
<?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>2003-03-26T14:58:25Z</responseDate>
<request verb="ListSets">http://memory.loc.gov/cgi-bin/oai2_0</request>
 <ListSets>
   <set>
     <setSpec>manz</setSpec>
     <setName>Ansel Adams&amp;apos;s Photographs of Japanese-American Internment at Manzanar</setName>
     <setDescription>
              <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/" xm
                       <dc:title xml:lang="en">Records for &amp;quot;Suffering Under a Great </dc:title>
                       <dc:creator>Library of Congress</dc:creator>
                       <dc:description>In 1943, Ansel Adams (1902-1984), America's best-
              </oai_dc:dc>
</setDescription>

   </set>
   <set>
     <setSpec>ecur</setSpec>
     <setName>Curtis (Edward S.) Collection (Photographs)</setName>
     <setDescription>
              <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/" xmlns:dc="http://purl.org/dc/elements
```

Ein Beispiel für eine Hierarchie innerhalb eines Archiv

```
<set>
  <setSpec>bio</setSpec>
  <setName>Biology</setName>
 </set>
 <set>
  <setSpec>bio:bio-ani-behav</setSpec>
  <setName>Animal Behavior</setName>
 </set>
 <set>
  <setSpec>bio:bio-ani-cog</setSpec>
  <setName>Animal Cognition</setName>
 </set>
```


Open Archives Initiative - Repository Explorer - Netscape
Datei Bearbeiten Ansicht Gehe Communicator Hilfe
Lesezeichen Adresse: http://rocky.dlib.vt.edu/~oai/cgi-bin/Explorer/oai1.0/testoai
Verwandte Objek
OPEN ARCHIVES
Open Archives Initiative - Repository Explorer
explorer version - 1.1 : protocol version - 1.0 : April 2001
http://cogprints.soton.ac.uk/perl/oai?verb=ListSets
Archive details : http://cogprints.soton.ac.uk/
List of Sets
Click on the link to list the contents
Biology
Animal Behavior
Animal Cognition
Behavioral Biology
Ecology
Ethology
Evolution
Population Biology
Primatology
Sociobiology
Dokument: Übermittelt
Start
Open Archives Initiati...
20:54

Open Archives Initiative - Repository Explorer

explorer version - 1.45a : protocol version - 1.0/1.1/2.0 : March 2003

http://wo.uio.no/as/WebObjects/theses.woa/wa/oai?verb=ListSets

Archive details : http://www.digbib.uio.no/publ/english/

List of Sets

Click on the link to list the contents

Document types

Thesis

Dissertation

Spesiale

Series titles

Master thesis

Series

Magistergradsavhandling

Frequent occurrences of languages in the database

set description:

Empfehlungen der DINI Arbeitsgruppe

- Die DINI-Arbeitsgruppe Elektronisches Publizieren empfiehlt eine,
  - Inhalt - fachliche Zuordnung
  - Publikationstyp - formale Zuordnung
  - Dokumentationstyp - formale Zuordnung
- um den Aufbau von spezifischen Datenprovider- und Serviceproviderdiensten zu erleichtern.

Set inhaltliche Beschreibung

- Eine grobe inhaltliche Einordnung ermöglicht eine fachliche Strukturierung und Selektionsmöglichkeit.

  DNB Sachgruppen

- Die Bezeichnung des jeweiligen Elements sollte über die Sachgruppennummer erfolgen.

  Spezifikation (SetSpec): dnb:Nummer

  Bezeichnung (SetName): Englisch

Sachgruppen der Deutschen Nationalbibliographie als inhaltliches Set

SetSpec	SetName
dnb:01	Knowledge and Culture in General
dnb:02	Books and Libraries, Information and Documentation
dnb:03	Reference Books, Bibliographies
dnb:04	Directories and Phone Books
dnb:05	Calendars
dnb:06	Journalism
.	.
.	.
.	.

Quelle

Ordnungskriterium

Bezeichnung

Beispielelement Plain-XML Darstellung:

- <set>
- <setSpec>**dnb**</setSpec>
<setName>**DNB classified objects**</setName>
- </set>
- <set>
  - <setSpec>**dnb:01** </setSpec>
  - <setName>**Knowledge and Culture in General** </setName>
  - </set>
  - <set>
  - <setSpec>**dnb:30** </setSpec>
  - <setName>**Chemistry** </setName>
  </set>

Set – Formaler Publikationstyp

SetSpec	SetName
pub-type	Documents having a formal publicationtype
pub-type:monograph	Books, Monographs
pub-type:article	Journal Articles
pub-type:dissertation	Dissertations and Professional Dissertations
pub-type:masterthesis	Diploma Theses
pub-type:report	Report
pub-type:paper	Paper
pub-type:conf-proceeding	Conference Proceedings
pub-type:lecture	Lectures
pub-type:music	Music
pub-type:program	Programs
Pub-type:play	Play
Pub-type:news	News
Pub-type:standards	Standards

Beispielelement Plain-XML Darstellung:

```
<set>
    <setSpec>pub-type </setSpec>
    <setName>Documents having a formal
        publication-type</setName>
</set>
<set>
        <setSpec>pub-type: monograph</setSpec>
        <setName>Books, Monographs</setName>
</set>
```

Formaler Set - Dokumentationstyp

- SetSpec
- doc-type
- doc-type:text
- doc-type:notes
- doc-type:image
- doc-type:audio
- doc-type:video
- doc-type:multimedia
- doc-type:data
- doc-type-binary

- SetName
- formal document-type
- Text
- Notes
- Image
- Audio
- video
- multimedia
- data
- Binary data, (executable) program

Beispielelement Plain-XML Darstellung

```
<set>
    <setSpec>doc-type </setSpec>
    <setName>formal document-type </setName>
</set>
<set>
    <setSpec>doc-type: video</setSpec>
    <setName>Video </setName>
</set>
```

Qualitativer Set

- Soll in Anlehnung an Quality - Schemes erfolgen, sobald die Elemente festgelegt sind.

Teil VII Verbundlösung in NRW

Empfehlungen für die Dokumentenserverbetreiber und dem Hochschulbibliothekszentrum in Köln

Dr. Bruno Klotz-Berendes

Universitätsbibliothek Dortmund

bruno.klotz-berendes@ub.uni-dortmund.de

Ausgangslage

- Jede Bibliothek / Hochschulrechenzentrum betreibt einen eigenen Dokumentenserver
- Die Digitale Bibliothek NRW wird vom Hochschulbibliothekszentrum (HBZ) für alle Bibliotheken mit einer lokalen Sicht betrieben.
- Der erste Versuch einer gemeinsamen Dokumentenserversuche ist gescheitert.

OAI ist die Lösung

- Die Expertengruppe hat sich auf die folgenden Punkte verständigt:
- OAI erlaubt jeder Institution den Betrieb einer eigenen Lösung für den Dokumentenserver
- Einige Dokumentenserver verfügen bereits über eine solche Schnittstelle
- Hilfestellung bei der Implementierung der Schnittstelle durch die OAI-Community
- Support des HBZs für OPUS

OAI ist die Lösung

- Serviceprovider des HBZ – Verbundes
  Das HBZ sammelt alle Metadaten der Datenprovider ein
- Integration in die Suchfunktion der Digitalen Bibliothek
- Dokumentenserver – Hosting für kleinere Institutionen (mit OAI-Schnittstelle)
- Umsetzungen der Set – Empfehlungen
- Erweiterung der Sets um eine regionale Komponente

OAI ist die Lösung

- Aufbau eines aggregierenden Datenproviders beim HBZ
- Ausweitung der SP – Funktionen und Archive

- Nutzen wir unsere Verbundstrukturen für den Aufbau neuer Strukturen auf Dokumentenserver-ebene um die wissenschaftlichen Dokumente unserer Hochschulen in der internationalen Wissenschaft bekannt zu machen.

Danksagung

Unseren Mitstreitern in der DINI Arbeitsgruppe
Susanne Dobratz, HU Berlin,
Frank Scholz, UB Stuttgart und
Bernd Diekmann, BIS Oldenburg

Uwe Müller und Andy Powell, deren Tutorial
in Lissabon als Grundlage diente, sowie allen,
anderen, bei denen wir uns Folien „entliehen“ haben,

und Ihnen Danke wir für Ihre Aufmerksamkeit !