Tutorium

Die Open Archive Initiative

Uwe Müller
Humboldt University Berlin
u.mueller@cms.hu-berlin.de
Dr. Bruno Klotz-Berendes
Universitätsbibliothek Dortmund
bruno.klotz-berendes@ub.uni-dortmund.de

Gliederung des Workshops

- Teil I - Geschichte und Überblick
- Teil II - OAI Serviceprovider - Beispiele
- Teil III Technische Einführung
- Mittagspause
- Teil IV Implementation data provider and service provider
- Part V Sets
- Part VI Realisierung auf Verbundebene
- Part VII Metadaten

Acknowledgements

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

  Heinrich Stammerjohanns
  Andy Powell
  Herbert Van de Sompel
  Carl Lagoze
  Hussein Suleman
  Michael Nelson
  Simeon Warner
  (and others probably!)

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; bad

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)

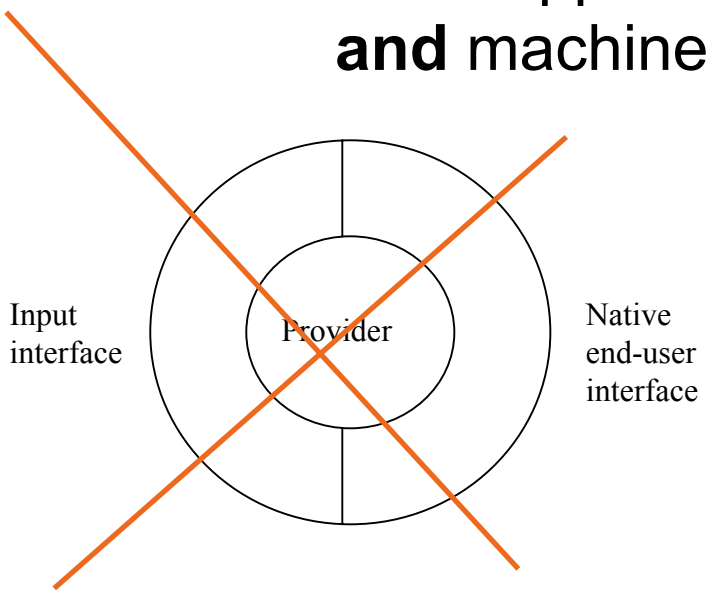
  http://www.dlib.org/dlib/february00/02contents.html
- UPS wurde kurz darauf in Open Archives Initiative umbenannt. (OAI) http://www.openarchives.org/

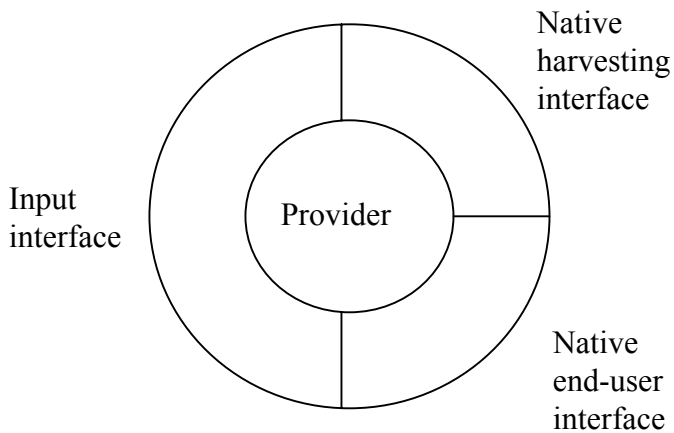
Daten- und Serviceprovider

- 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.

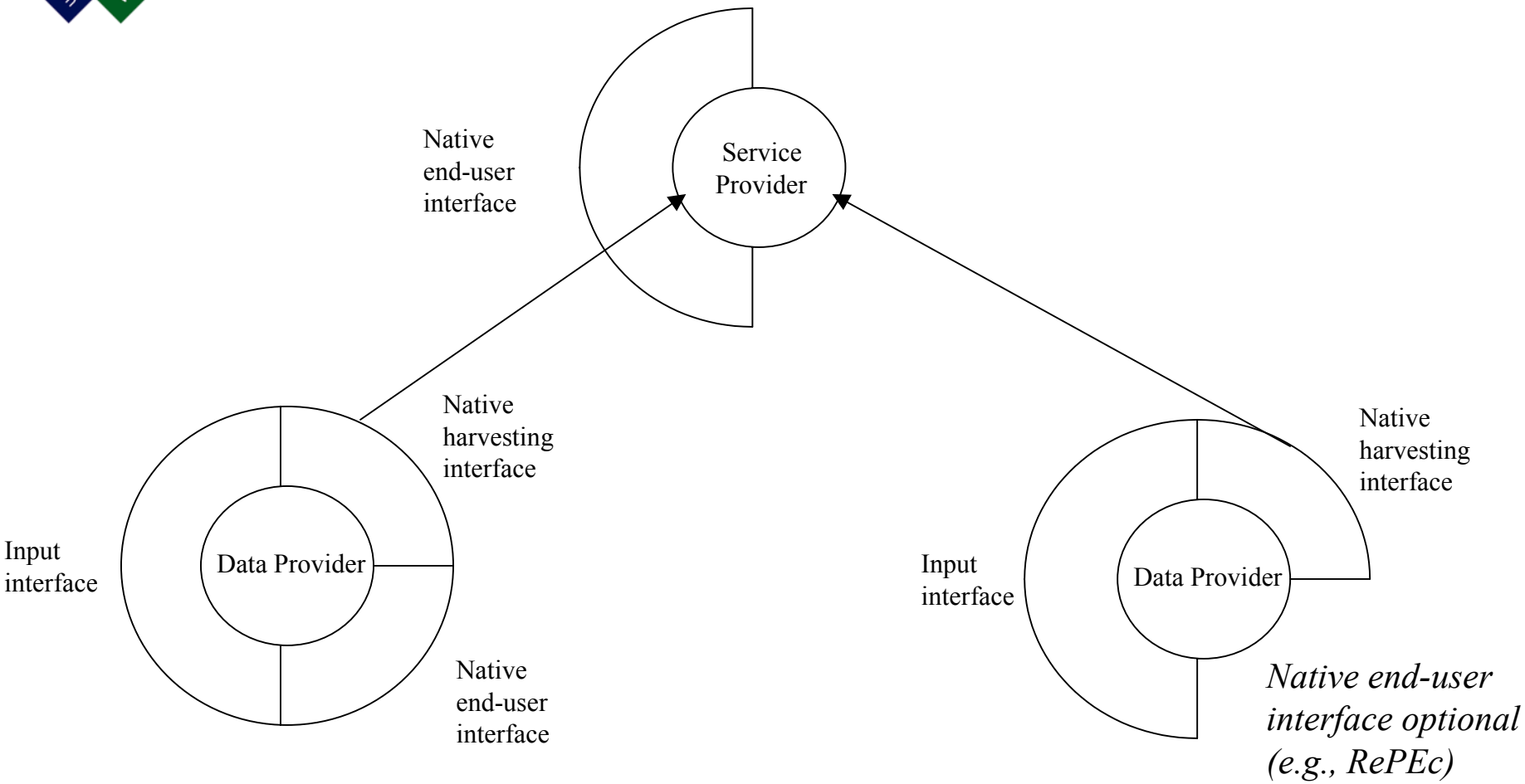
Human vs. machine interfaces

- 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”

OAI-PMH v 1.0 [01/2001]

- goal: optimise discovery of document-like objects
- inputs…
  - Santa Fe Convention
  - various DLF meetings on metadata harvesting
  - deliberations at Cornell
  - alpha-testers of OAI-PMH v 1.0
  - 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?

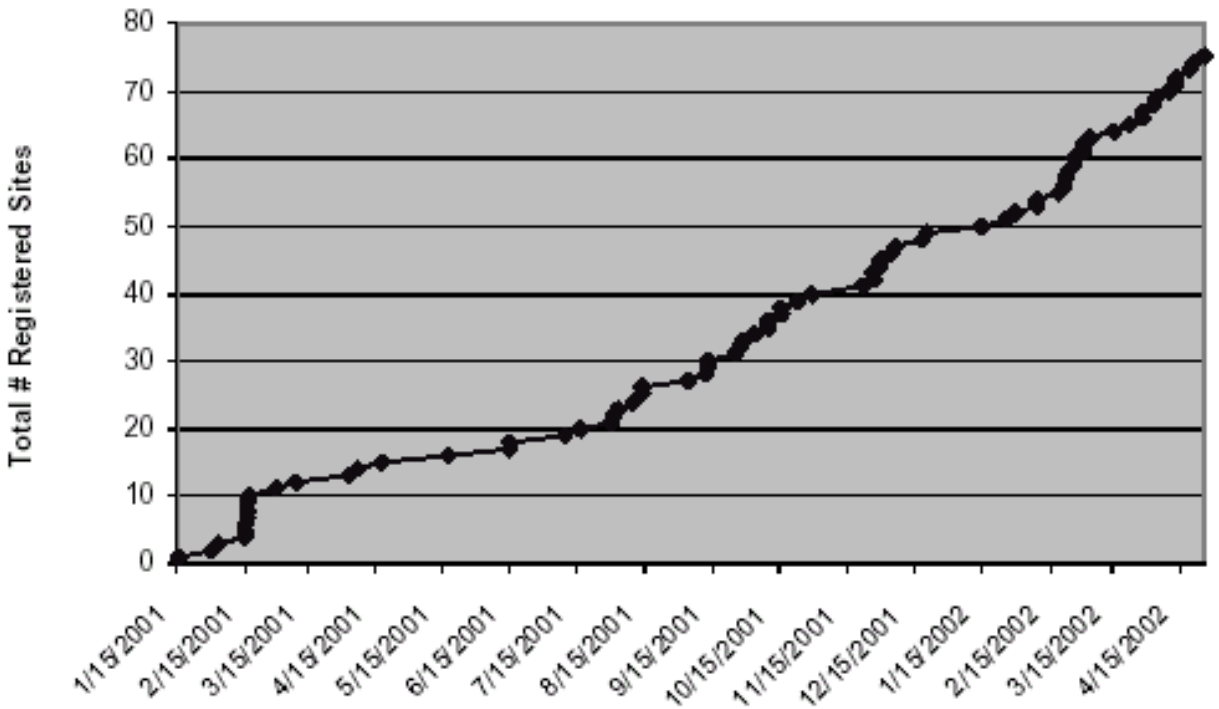
Open Archives Initiative

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.

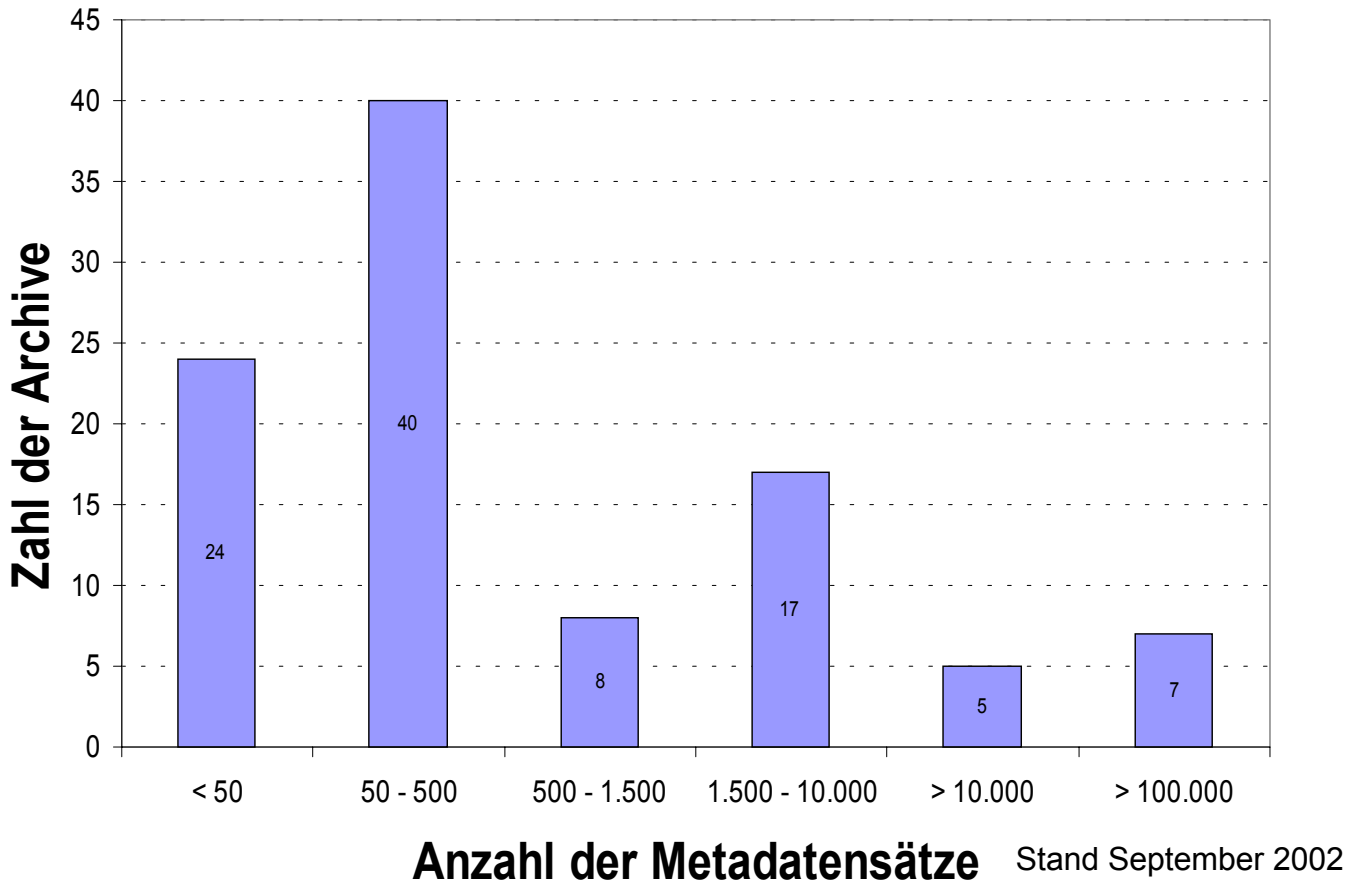
OAI is happening at break-neck speed...

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


45
40
35
30
25
20
15
10
5
0
Zahl der Archive
24
40
8
17
5
7
< 50
50 - 500
500 - 1.500
1.500 - 10.000
> 10.000
> 100.000
Anzahl der Metadatensätze
Stand September 2002

OAI-PMH Version 2.0 veröffentlicht Juni 2002

- Ziel: Dauerhafter Austausch der Metadaten zwischen Datenprovidern und Serviceprovidern
- inputs:
  - OAI-PMH v.1.0
  - Feedback der OAI-implementers
  - 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]

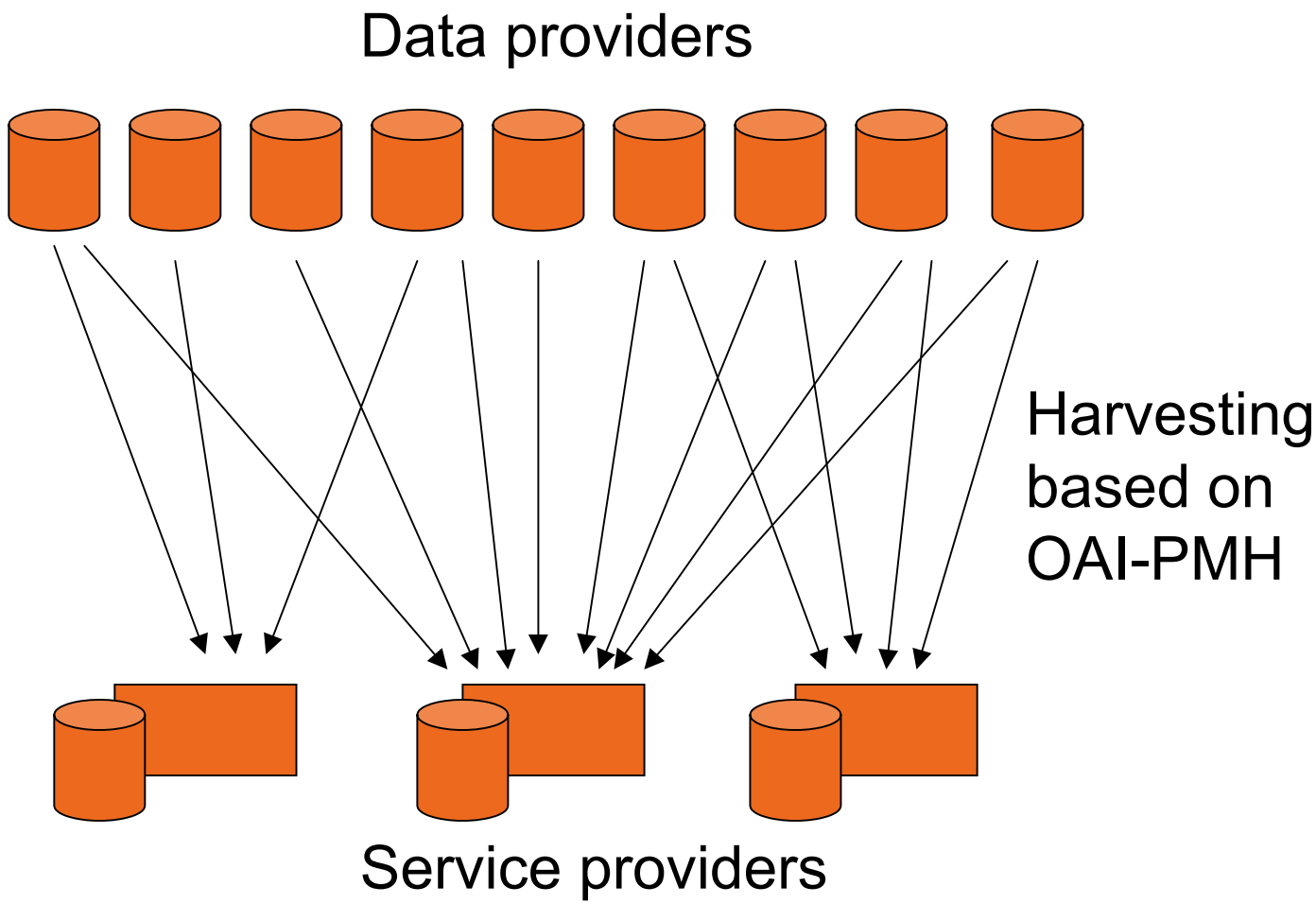
- low-barrier interoperability specification
- Metadaten Harvesting model: Datenprovider / Serviceprovider
- Metadaten der digitalen Objekte (resources)
- Unabhängiges Protokoll
- HTTP basiert
- XML basiert
- mindestens Dublin Core, ohne Qualifier
- Stabil - Produktionsbasis

	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

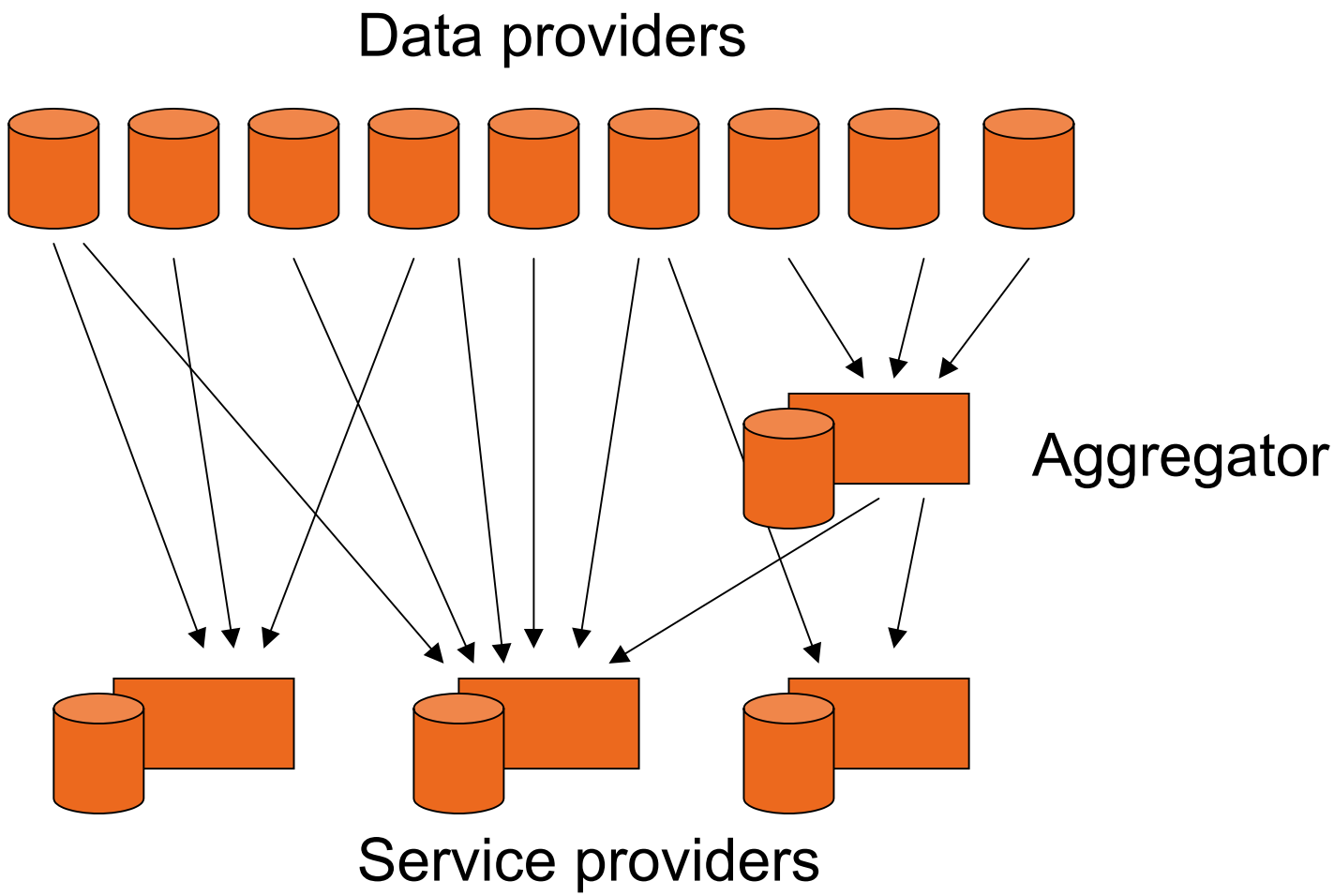
Flexible Weiterentwicklung

- 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.

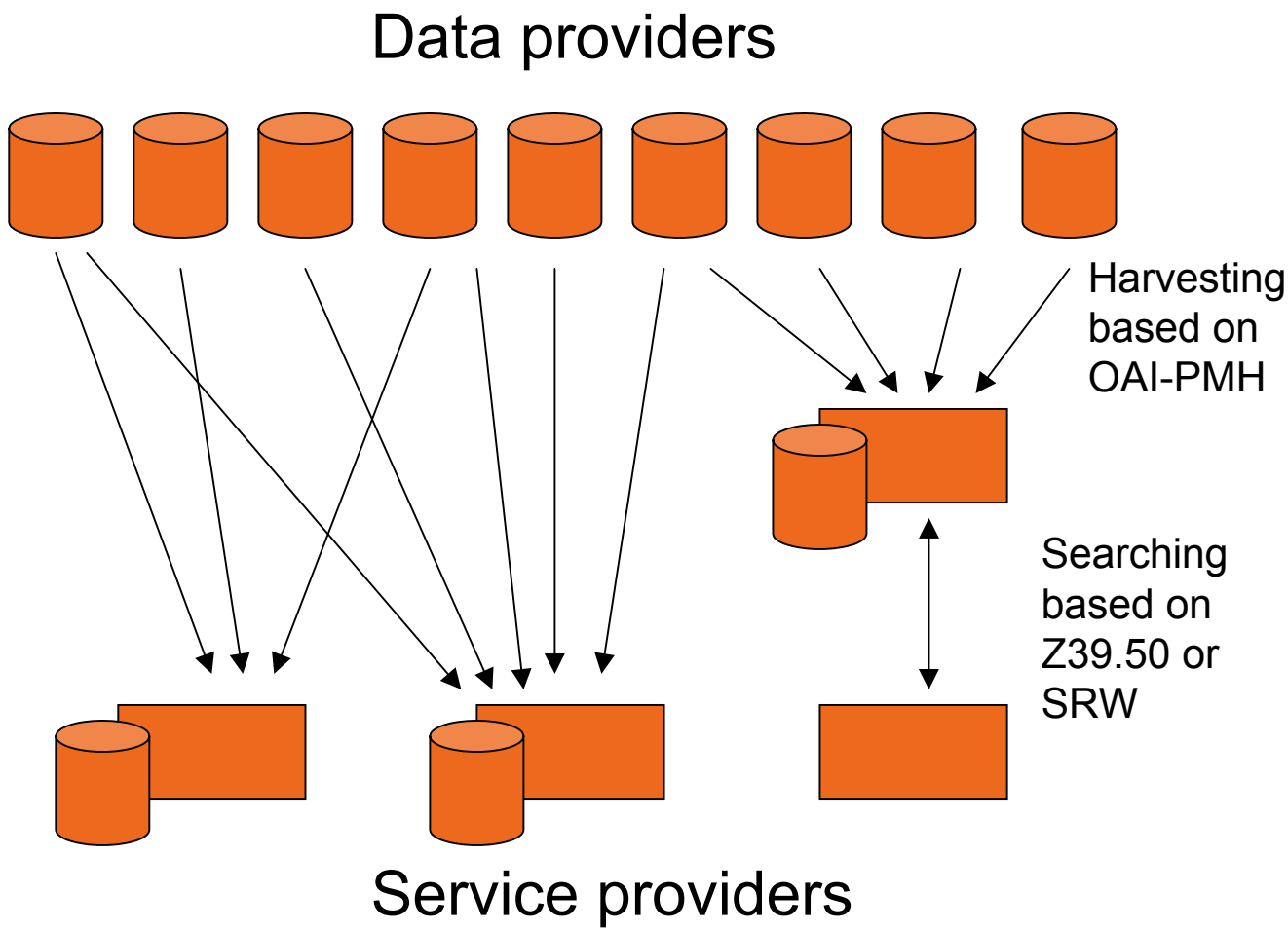
Multiple data and service p's



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

Tutorium
zur Open Archive Initiative

Teil II: Beispiele für Serviceprovider

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

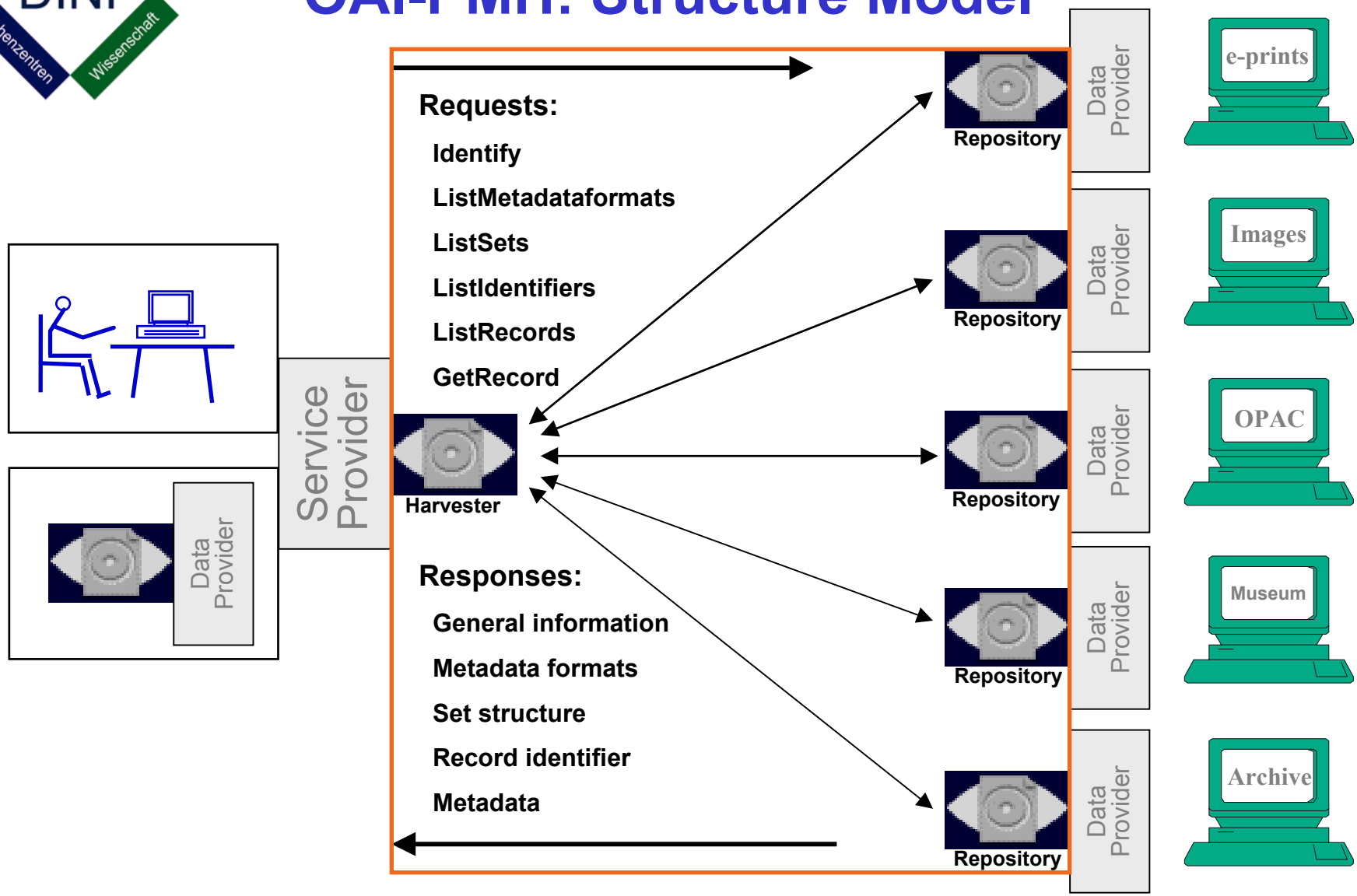
Gliederung des Workshops

- Teil I - Geschichte und Überblick
- Teil II - OAI Serviceprovider - Beispiele
- Teil III Technische Einführung
- Mittagspause
- Teil IV Implementation data provider and service provider
- Part V Sets
- Part VI Realisierung auf Verbundebene
- Part VII Metadaten

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: Structure Model



Serviceprovidertypen

- Man kann inzwischen zwei Typen von Serviceprovidern unterscheiden:

- Reine Suchfunktionalität
  Beipiele ARC, Scirus und HU

- Erweiterte Funktionen
  Beipiele: Citebase Search, Torii und MyOAI

Die online Präsentation der zuvor aufgelisteten Serviceprovider musst aufgrund einer technischen Störung der Internetanbindung entfallen.

Tutorium

Die Open Archive Initiative

Teil III

Technische Einführung

Uwe Müller
Humboldt-Universität zu Berlin, Computer- und Medienservice
u.mueller@cms.hu-berlin.de

Agenda

- Grundlagen des Protokolls
- Protokolldetails
- Anfragetypen
- Beispiele

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. Cross Search

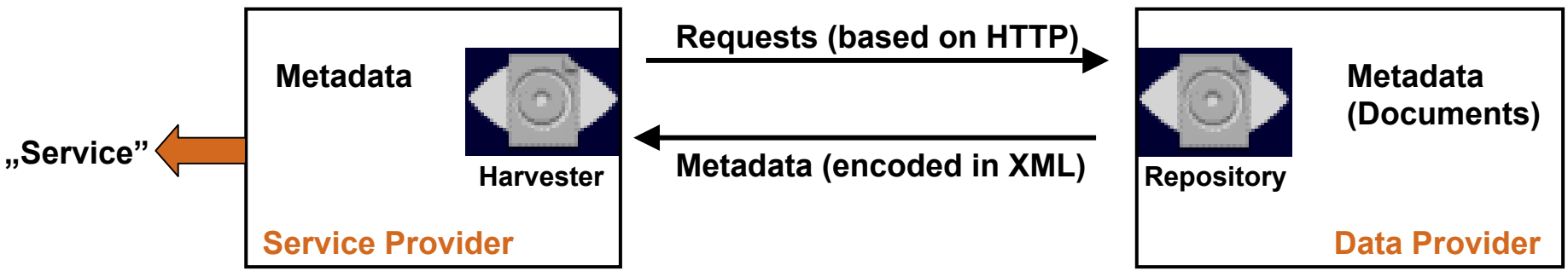
- Unterschiedliche Strategien, um Interoperabilität zu erreichen
  - Cross Search: Dienste laufen verteilt auf verteilten Daten (z.B. Metasuchmaschine, Z39.50)
  - Harvesting: Daten/Metadaten werden von verteilten Quellen zum Ziel, wo Dienste angeboten werden, transferiert (z.B. Union catalogues)
- Cross Search 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)

- Zentrale Ideen
  - weltweite Konsolidierung wissenschaftlicher Archive
  - freier Zugang zu den Archiven (mindestens: Metadaten)
  - konsistente Schnittstellen für Archive und Service Provider
  - “low barrier protocol” / einfache Implementation
  - auf existenten Standards basierend (e.g. HTTP, XML, DC)
- Grundsätzliche Funktionsweise



Anforderungen an das Protokoll

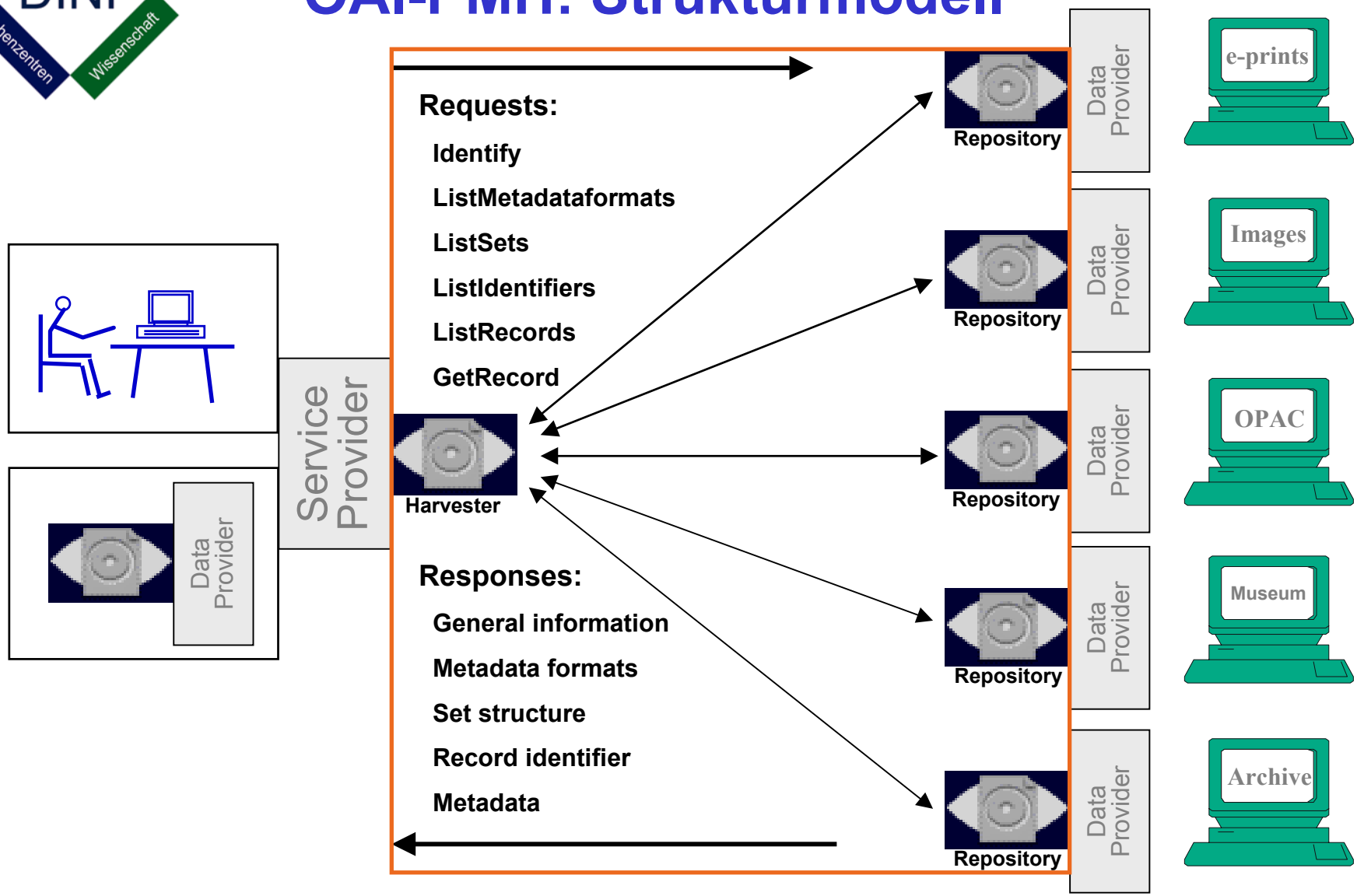
Das Protokoll sollte ...

- auf einem maschinenlesbaren Format basieren
- in einem strikten Format kodiert werden, welches validiert werden kann
  - character-encoding
  - metadata-encoding
- verschiedene Content-Modelle unterstützen
  - metadata formats
- existierende Standards nutzen (HTTP, XML, DC)
  - einfach zu implementieren
  - einfach anzupassen

OAI: Grundsätzliche Annahmen

- zwei unterschiedliche Gruppen von ‘Teilnehmern’
- *Data Provider* (Open Archives, Repositories)
  - freier Zugriff auf Metadaten
  - nicht notwendigerweise: freier Zugriff auf Volltexte / vollständige Objekte
  - einfach zu implementieren, niedrige Schwellen
- *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 Mehrwertdienste, die auf den Metadaten basieren

OAI-PMH: Strukturmodell



OAI-PMH: Protokoll-Überblick

- 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 (mindestens: 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

Agenda

- Grundlagen des Protokolls
- Protokolldetails
- Anfragetypen
- Beispiele

Protokolldetails: Definitionen

Harvester (Ernter, Einsammler)
- Client-Anwendung, die OAI-PMH-Anfragen stellen kann

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 für ein Item in einem Repository

Set (Menge)
- optionales Konstrukt um Items in einem Repository anzuordnen

Protokolldetails: Definitionen (2)

resource

item = identifier

all available metadata about *David*

item

Dublin Core metadata

MARC metadata

SPECTRUM metadata

records

Protokolldetails: Records

- Metadaten einer Resource in einem spezifischen Format
  - header (Kopf) (notwendig)
    - Identifier (1)
    - Datestamp (1)
    - setSpec-Elemente (*)
    - Statusattribut für gelöschte Items (?)
  - metadata (notwendig)
    - XML kodierte Metadaten mit “root tag”, Namespace
    - Repositories müssen zumindest Dublin Core unterstützen
  - about (über) (optional)
    - Angaben über (Urheber)Rechte
    - Angaben zur Herkunft der Metadaten / der Objekte

Protokolldetails: OAI-Record (Beispiel)

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

```
<record>
  <header>
    <identifier>oai:HUBerlin.de:3001634</identifier>
    <datestamp>2003-05-08</datestamp>
    <setSpec>tutorials</setSpec>
  </header>
  <metadata>
    <oai_dc>
      <title>OAI Tutorial</title>
      <creator>Uwe Müller</creator>
      <language>ger</language>
    </oai_dc>
  </metadata>
  <about>
    <metadataID>oai:HUBerlin.de:md3001634</metadataID>
  </about>
</record>
```

Protokolldetails: Zeitstempel

- Datum der Erstellung / letzten Modifikation des Metadatensatzes
- Pflichtinformation zu jedem Item
- zwei erlaubte Granularitäten: YYYY-MM-DD, YYYY-MM-DDThh:mm:ssZ
- Funktion: Information über Metadaten, selektives Harvesting (**`from`**- und **`until`**-Argumente)
- Anwendung: inkrementelles Update
- Bei Neuerstellung, Veränderung und Löschen von Datensätzen muss nicht das gesamte Archiv neu abgesucht werden
- Löschen: drei erlaubte ‘support levels’
  - no, persistent, transient

Protokolldetails: Metadatenformate

- OAI-PMH unterstützt das Anbieten / Übertragen mehrerer unterschiedlicher Metadatenformate durch ein Repository (auch: für ein Item)
- Eigenschaften von Metadatenformaten (im Sinne der OAI)
  - id, um das Format anzugeben (**`metadataPrefix`**)
  - Metadaten Schema URL (XML Schema, um XML zu validieren)
  - XML Namespace URI (Globaler Identifier für Metadatenformat) string to specify the format (**`metadataPrefix`**)
- Repositories müssen mindestens Dublin Core unterstützen
- aber: Es können beliebige Metadatenformate definiert und mit dem OAI-PMH übertragen werden
- Metadaten müssen mit XML Namespace Spezifikation übereinstimmen
- Z.B. MARC (Bibliotheken), IMS (Lehre), ETDMS (Diplomarbiten/Dissertationen), RFC1807 (Bibliographien)

Protokolldetails: Metadatenformate (2)

- Mindeststandard: unqualified Dublin Core
  - http://dublincore.org/
  - Metadatenformat von Dublin Core enthält 15 Elemente
  - Elemente sind optional
  - Elemente sind wiederholbar

The Dublin Core Metadata Element Set:

Title	Contributor	Source
Creator	Date	Language
Subject	Type	Relation
Description	Format	Coverage
Publisher	Identifier	Rights

Protokolldetails: Metadatenformate (3)

- XML-Namensräume und Schema
  - Konsistenz und Datenqualität werden durch den Gebrauch von XML-Schemabeschreibungen 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 OAI-PMH unterstützen

Protokolldetails: 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, Suchmaschine für Dissertationen, …
- 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)
  z.B. 302 (redirect), 503 (service not available)
- Kompression: optional bei OAI-PMH,
  nur identity encoding is notwendig
- Antwortformat: wohlgeformetes XML mit Markup:
  - XML declaration
    (**`<?xml version="1.0" encoding="UTF-8" ?>`**)
  - Root Element namens **`OAI-PMH`** mit drei Attributen
    (**`xmlns`**, **`xmlns:xsi`**, **`xsi:schemaLocation`**)
  - 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

Protokolldetails: Flusskontrolle

- 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 (2)

- 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
    + Verfallsdatum, Größe der Gesamtliste, 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 (3)

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`**

Agenda

- Grundlagen des Protokolls
- Protokolldetails
- Anfragetypen
- Beispiele

Request Types

- sechs verschiedene Anfragetypen
  - Identify
  - ListMetadataFormats
  - ListSets
  - ListIdentifiers
  - ListRecords
  - GetRecord
- Harvester muss nicht alle Typen benutzen
- Repository muss alle Typen implementieren
- Notwendige und optionale Argumente, die von den Anfragetypen abhängen

Anfragetyp: Identify

Funktion

Beschreibung eines Archives

Beispiel

archive.org/oai-script?**verb=Identify**

Parameter

keine

Fehler / Ausnahmen

`badArgument`

z.B. archive.org/oai-script?verb=Identify**&**
set=biology

Anfragetyp: Identify (2)

Antwort-Format

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, …	*

Anfragetyp: ListMetadataFormats

Funktion

Abfragen verfügbarer Metadatenformate (u.a. für ein Item)

Beispiel

archive.org/oai-script?**verb=ListMetadataFormats**&
identifier=oai:HUBerlin.de:3000218

Parameter

`identifier` (optional)

Fehler / Ausnahmen

`badArgument`

`idDoesNotExist`

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

`noMetadataFormats`

Anfragetyp: ListSets

Funktion

Abfragen der logischen Set-Struktur eines Archives

Beispiel

archive.org/oai-script?**verb=ListSets**

Parameter

`resumptionToken` (exklusiv)

Fehler / Ausnahmen

`badArgument`

`badResumptionToken`

z.B. archive.org/oai-script?verb=ListSets**&**
resumptionToken=any-wrong-token

`noSetHierarchy`

Anfragetyp: ListIdentifiers

Funktion

Abfragen der Header-Informationen einer Menge von Items

Beispiel

archive.org/oai-script?**verb=ListIdentifiers**&
metadataPrefix=oai_dc&**from=2002-12-01**

Parameter

`from` (optional)
`until` (optional)
`metadataPrefix` (erforderlich)
`set` (optional)
`resumptionToken` (exklusiv)

Fehler / Ausnahmen

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

Anfragetyp: ListRecords

Funktion

Abfragen von Records (Metadaten) einer Menge von Items

Beispiel

archive.org/oai-script?**verb=ListRecords**&
metadataPrefix=oai_dc&**set=biology**

Parameter

- **`from`** (optional)
- **`until`** (optional)
- **`metadataPrefix`** (erforderlich)
- **`set`** (optional)
- **`resumptionToken`** (exklusiv)

Fehler / Ausnahmen

- **`badArgument`**
- **`badResumptionToken`**
- **`cannotDisseminateFormat`**
- **`noRecordsMatch`**
- **`noSetHierarchy`**

Anfragetyp: GetRecord

Funktion

Abfrage eines Metadatensatzes für ein bestimmtes Item

Beispiel

archive.org/oai-script?**verb=GetRecord**&
identifier=oai:HUBerlin.de:3000218&
metadataPrefix=oai_dc

Parameter

`identifier` (erforderlich)
`metadataPrefix` (erforderlich)

Fehler / Ausnahmen

`badArgument`
`cannotDisseminateFormat`
`idDoesNotExist`

Agenda

- Grundlagen des Protokolls
- Protokolldetails
- Anfragetypen
- Beispiele

Beispiel: http://edoc.hu-berlin.de/OAI-2.0?

verb=ListIdentifiers&from=2002-01-06&until=2002-01-08&
metadataPrefix=oai_dc&set=doctypes:dissertations

```
<?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>2002-10-22T17:49:49+01:00</responseDate>
  <request verb="ListIdentifiers" from="2002-01-03" until="2002-01-08" metadataPrefix="oai_dc"
           set="doctypes:dissertations">http://edoc.hu-berlin.de/OAI-2.0</request>
  <ListIdentifiers>
    <header>
      <identifier>oai:HUBerlin.de:3000819</identifier>
      <datestamp>2002-01-08</datestamp>
      <setSpec>doctypes</setSpec>
      <setSpec>doctypes:dissertations</setSpec>
      <setSpec>dnb</setSpec>
      <setSpec>dnb:dnb33</setSpec>
    </header>
    <header>
      <identifier>oai:HUBerlin.de:3000831</identifier>
      <datestamp>2002-01-07</datestamp>
      <setSpec>doctypes</setSpec>
      <setSpec>doctypes:dissertations</setSpec>
      <setSpec>dnb</setSpec>
      <setSpec>dnb:dnb27</setSpec>
    </header>
  </ListIdentifiers>
</OAI-PMH>
```

Beispiel: http://edoc.hu-berlin.de/OAI-2.0?verb=GetRecord&identifier=oai:HUBerlin:3000819&metadataPrefix=oai_dc

```
<?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>2002-11-27T14:57:01+01:00</responseDate>
   <request verb="GetRecord" metadataPrefix="oai_dc"
            identifier="oai:HUBerlin.de:3000819">http://edoc.hu-berlin.de/OAI-2.0</request>
   <GetRecord>
     <record>
       <header>
         <identifier>oai:HUBerlin.de:3000819</identifier>
         […]
       </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>EinfluÃŸ genetischer Variationen im Tumor Nekrose […]</dc:title>
           <dc:creator>SchÃ¼ttlÃ¶ffel, Antje</dc:creator>
           […]
       </metadata>
     </record>
   </GetRecord>
</OAI-PMH>
```

Technische Einführung: Fragen?

OAI – official site

http://www.openarchives.org/

protocol specification

http://www.openarchives.org/OAI/openarchivesprotocol.html

general mailing list

http://www.openarchives.org/mailman/listinfo/OAI-general/

implementers mailing list

http://www.openarchives.org/mailman/listinfo/OAI-implementers/

Tutorium

Die Open Archive Initiative

Teil IV
Implementation von
Data Providern und Service Providern

Uwe Müller
Humboldt-Universität zu Berlin, Computer- und Medienservice
u.mueller@cms.hu-berlin.de

Agenda

- Grundsätzliche Überlegungen
- Data Provider
- Service Provider

Grundsätzliches: 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?
- Wie müssen Metadaten weiterverarbeitet werden?

Data Provider & Service Provider

- Auf welche Aspekte muss man sich einigen?
- Metadatenformate ...

Abbilden von Metadaten (Mapping)

- 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 öffentlich zugänglich sein sollte)
- definiere eine Set-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 einfach möglich

Agenda

- Grundsätzliche Überlegungen
- Data Provider
- Service Provider

Data Provider: Voraussetzungen

- Metadaten über “Objekte” (“items”)
  - sollten in (SQL)-Datenbank vorliegen
  - auch möglich: Dateisystem
  - eindeutiger Bezeichner für jedes Item
- über das Internet erreichbarer Webserver
  - z.B. Apache, IIS
- Programmiererweiterung / API
  - z.B. Perl, PHP, Java-Servlet
  - Erweiterung des Webservers
  - Zugriff auf die Datenbank (Dateisystem)
  - nicht erforderlich: Session Management

Data Provider: Voraussetzungen (2)

- Archiv-Bezeichner / Basis-URL
- eindeutige Bezeichner für Items
- Metadatenformat(e) (mindestens: unqualified Dublin Core) inkl. XML-Schema
- Zeitstempel für jedes Item
- logische Set-Struktur (optional)
  - Übereinkunft innerhalb der (Fach)Communities
- Flusskontrolle / Implementation des Resumption-Tokens (optional, ‘größere’ Archive sollten dies tun ...)

Data Provider: 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)

Data Provider: Eindeutige Identifier

- Jedes Item muss einen eindeutigen Identifier besitzen
- Identifier müssen gültige URIs sein
- Beispiel

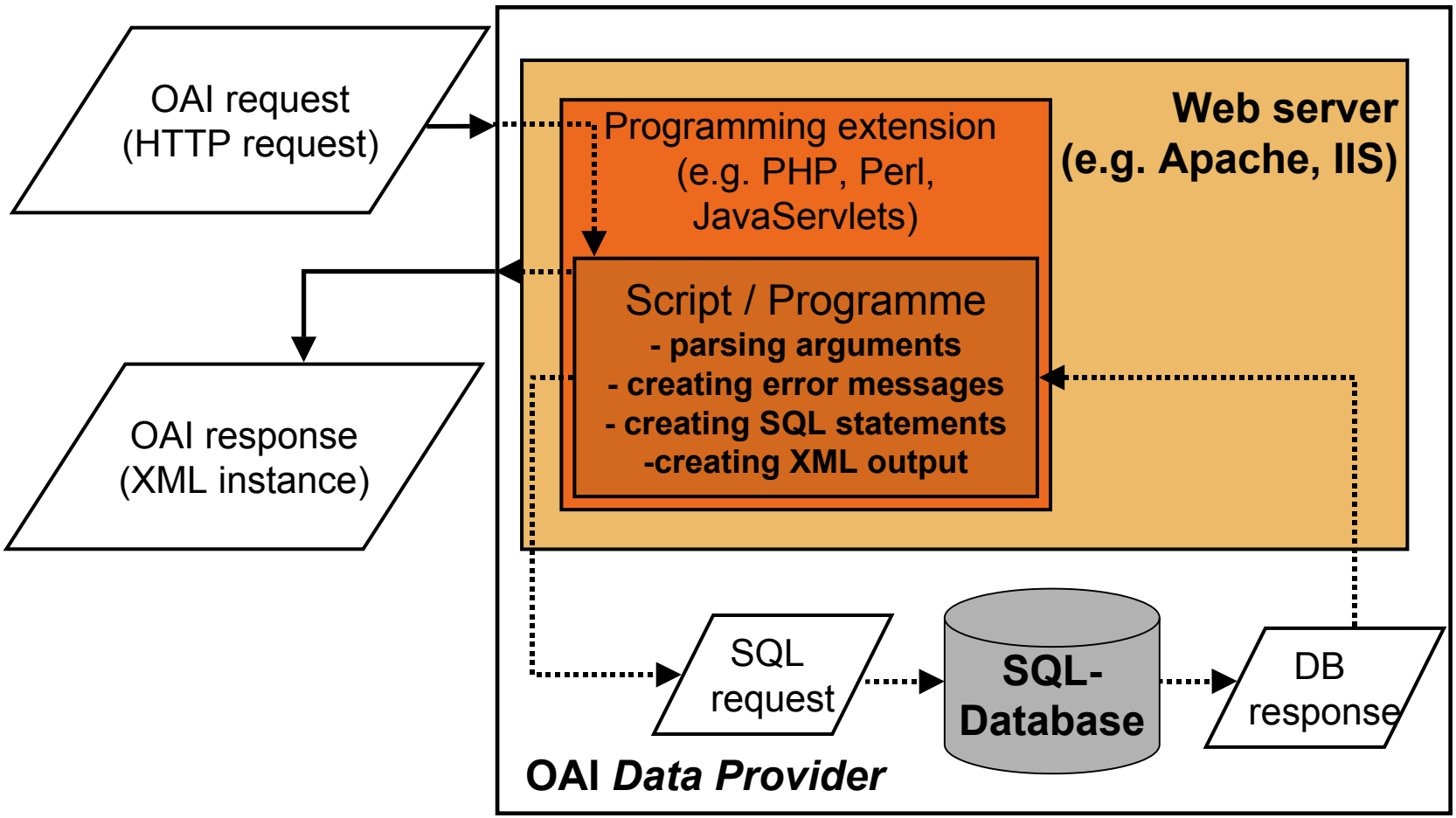
  oai:<archiveId>:<recordId>

  oai:HUBerlin.de:30001634
- Jeder Identifier muss zu einem einzigen und immer dem gleichen Record aufgelöst werden (für ein vorgegebenes Metadatenformat)

Data Provider: 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 Zeitplänen korrekt weitergegeben werden kann.

Data Provider: Architecture



Data Provider: Grundstruktur

Argument-Parser

- validiert OAI-Anfragen

Erzeugen von Fehlermeldungen

- erzeugt XML-kodierte Fehlermeldungen

Datenbankanfrage / lokales Auslesen der Metadaten

- Mapping auf das angeforderte Metadatenformat

XML-Generator

- erzeugt XML-kodierte Antworten, die die Metadaten enthalten

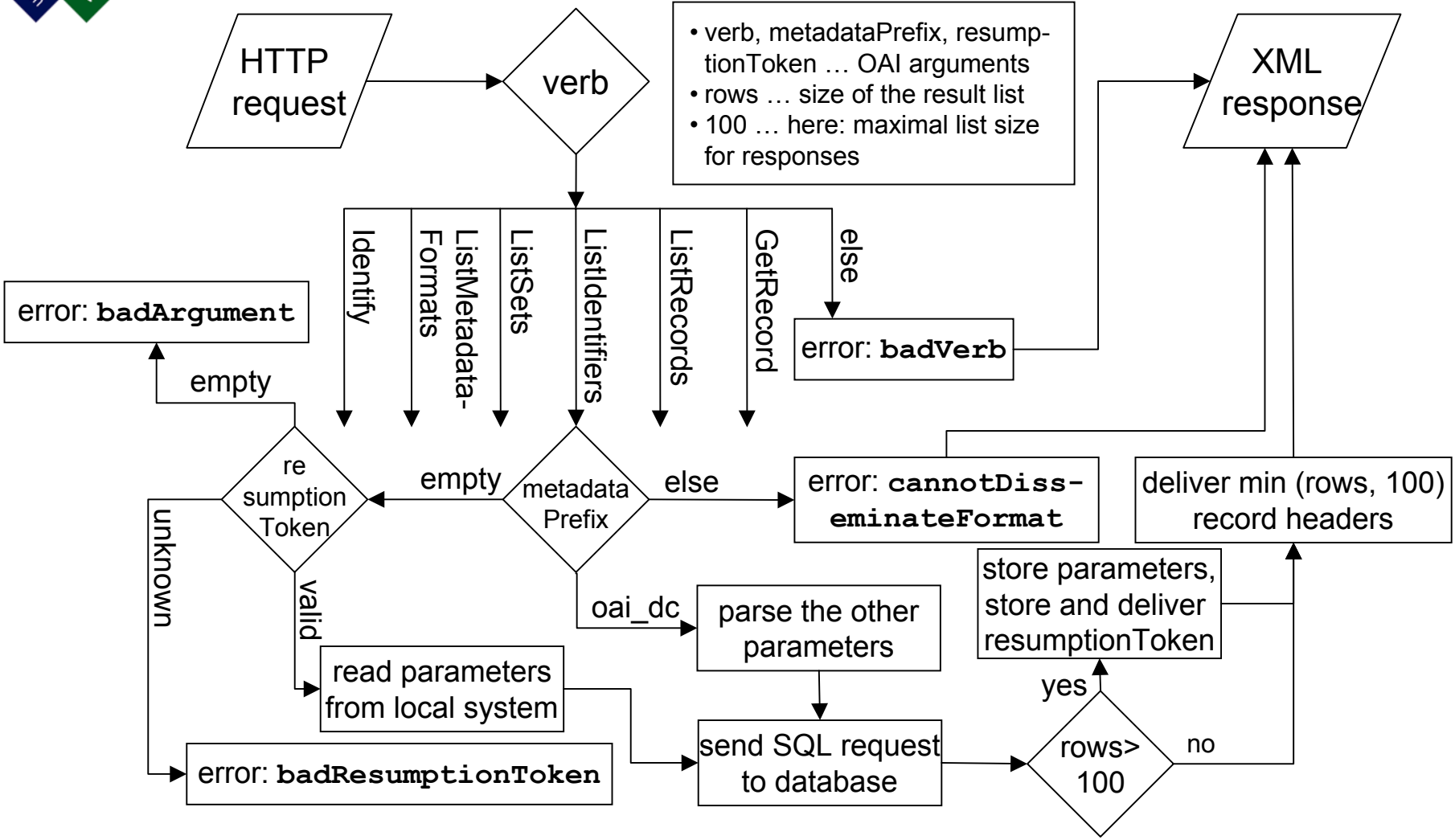
Flusskontrolle

- ermöglicht die Ausgabe einer FOlge von unvollständigen Listen, deren Summe die Gesamtliste der angeforderten Daten darstellt
- benutzt den Resumption-Token

Data Provider: 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: Flussdiagramm als Beispiel



Data Provider: Resumption Token

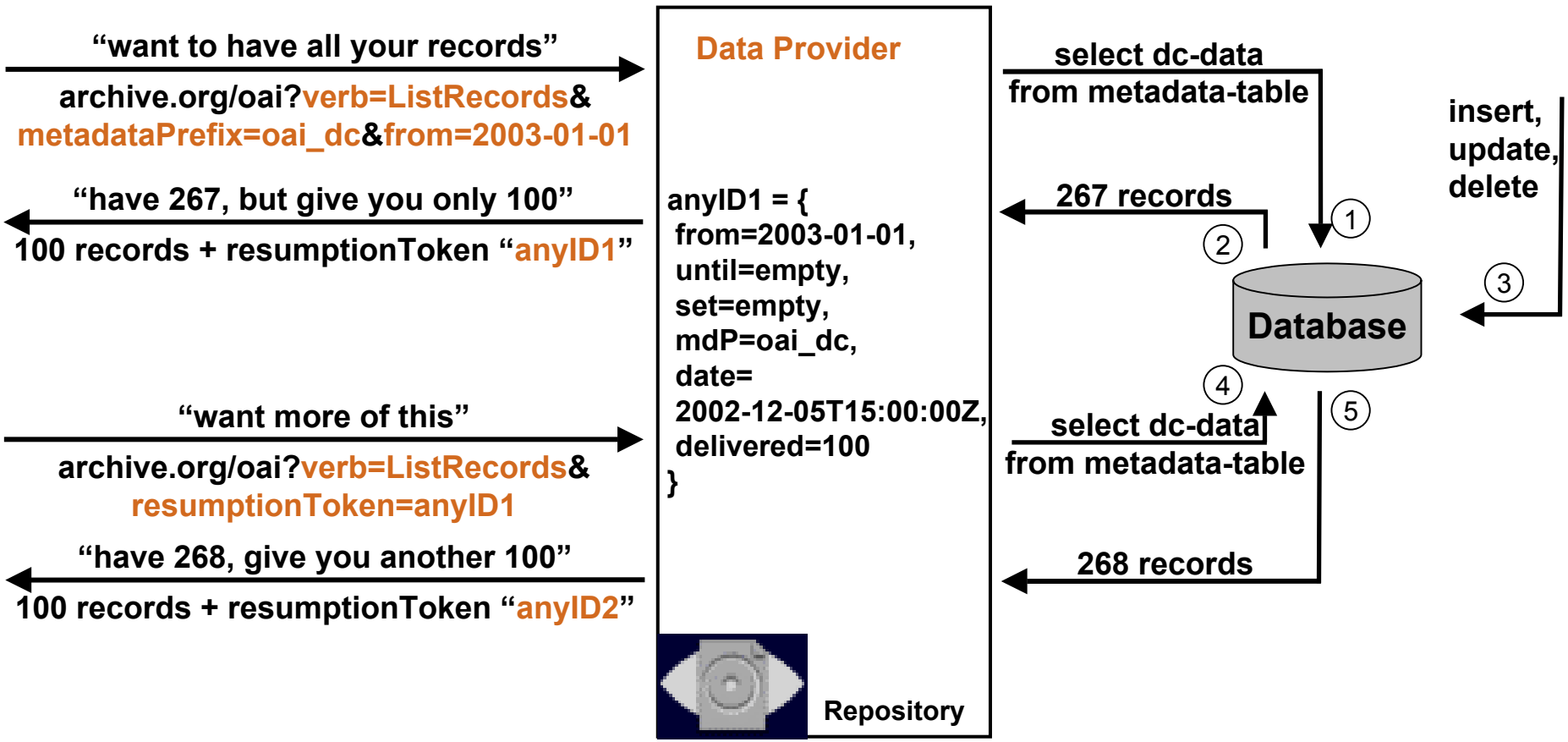
- sollte für “große” Listen implementiert werden
- initiiert vom Data Provider
- speichere Parameter (**set**, **from**, …) und die Anzahl der bereits gelieferten Records
- Eigenschaften
  - expiration: **expirationDate** (optional)
  - **completeListSize** (optional)
  - bereits gelieferte Records: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

Data Provider: Resumption Token (2)

Beispiel

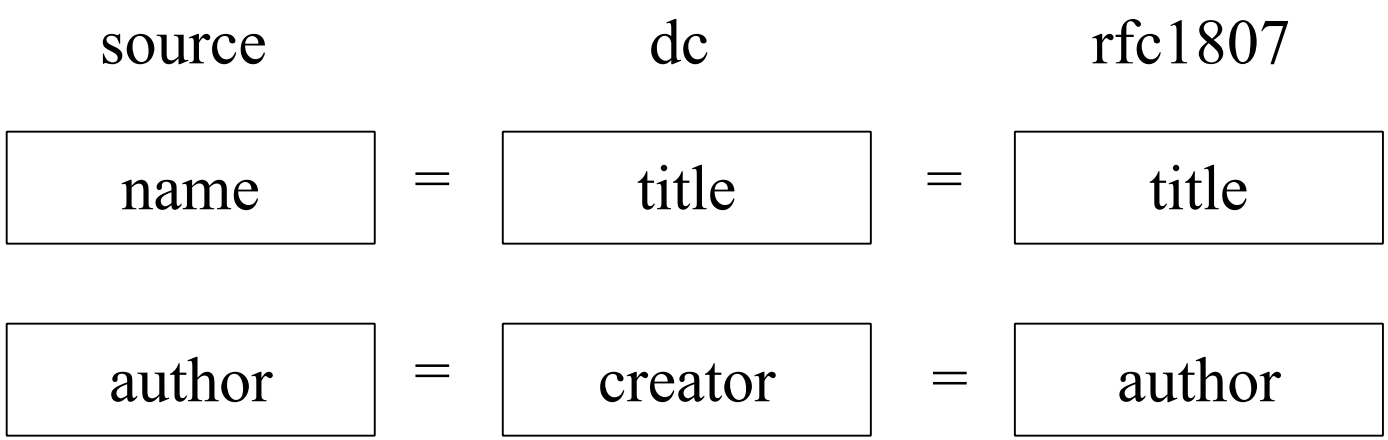
Data Provider: Resumption Token (3)

Beispiel (2)



Data Provider: Metadatenerzeugung

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



Data Provider: Datenrepräsentation

- Benutzung empfohlener Datenrepräsentationen
  - Datum
    - 2002-12-05
    - 2002-xx-xx, 2002, 05.12.2002
  - Sprach-Codes
    - eng, ger, ...
    - en, de, english, german
- Mehrfach-Werte: Verwendung eines eigenen XML-Elements für jede Einheit
  - Autor
    - <dc:creator>Smith, Adam</dc:creator> <dc:creator>Nash, John</dc:creator>
    - <dc:creator>Smith, Adam; Nash, John </dc:creator>

Data Provider: Kodierung der Daten für XML

- Spezielle XML Zeichen müssen „escaped“ werden
- Konvertieren nach UTF-8 (Unicode)
- Konvertieren von Entitäten (Umlaute)
- Entfernen überschüssiger Leerzeichen
- Konvertieren von CR/LF für Absätze
- URLs

  /?#=&:;+ müssen als Escape-Sequenz kodiert werden

Data Provider: Komprimierung

- Methode um Netzverkehr zu verringern
- Optional for beide Seiten: Data and Service Provider
- wird auf HTTP behandelt
- 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

Data Provider: 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

Data Provider: Vorbeugung gegen DoS

- Problem: so genannte Denial-of-Service-Attacken
  - Lahmlegen des Servers durch eine Flut von Anfragen
- Lösungsmöglichkeiten:
  - Rückgabe von Teilantworten und den Service Provider mit einem Resumption-Token nach weiteren Teilen fragen lassen
  - Benutzung von HTTP-Fehlermeldung ‘503 retry-after’, so dass Clients erst nach einer spezifierten Zeit erneut anfragen
  - Benutzung von Zugriffslisten (access control lists), um den Zugriff auf das Archiv zu beschränken
  - Abwarten eines gewissen Zeitraums, bevor die Ergebnisse einer Anfrage zurückgesendet werden

Data Provider: Test und Registrierung

- Erzeugen und Verwenden eigener OAI-PMH-Anfragen – Überprüfen der Ergebnisse
- Benutzen des Repository Explorer (VT University)
  - http://oai.dlib.vt.edu/cgi-bin/Explorer/oai2.0/testoai/
  - Eingabe der Argumente über ein HTML-Formular
  - Antworten werden automatisch validiert
  - Möglichkeit des ‘Browsing’ zu anderen Anfragen
  - automatisches Programm zum Testen der Korrektheit
- offizielle Registrierungsseite
  - http://www.openarchives.org/data/registerasprovider.html
  - Angeben der Basis-URL
  - genauer Korrektheitstest (inkl. Fehlermeldungen usw.)
  - Information über inkorrektes Verhalten des Data Providers
  - im Falle des korrekten Verhaltens: Hinzufügen zur offiziellen Liste registrierter Data Provider
  - regelmäßige Überprüfungen

Data Provider: Übliche Probleme

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

Agenda

- Grundsätzliche Überlegungen
- Data Provider
- Service Provider

Service Provider: Beispiele

- Repository Explorer:
  - http://oai.dlib.vt.edu/cgi-bin/Explorer/oai2.0/testoai/
- Suchmaschinen / Subject Gateways
  - Cross Archive Searching Service: http://arc.cs.odu.edu/
  - DINI: http://edoc.hu-berlin.de/oaisearch/
  - Physnet: http://physnet.uni-oldenburg.de/oai/query.php
  - NCSTRL: http://www.ncstrl.org
- Mehrwertdienste
  - ProPrint: http://www.proprint-service.de
  - Citation Indexing: http://icite.sissa.it:8888
  - MyOAI: http://www.myoai.org/

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

- 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 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 (Mehrfach-Werte)
- oder: Nutzung einer XML-Datenbank

Service Provider: Struktur (5)

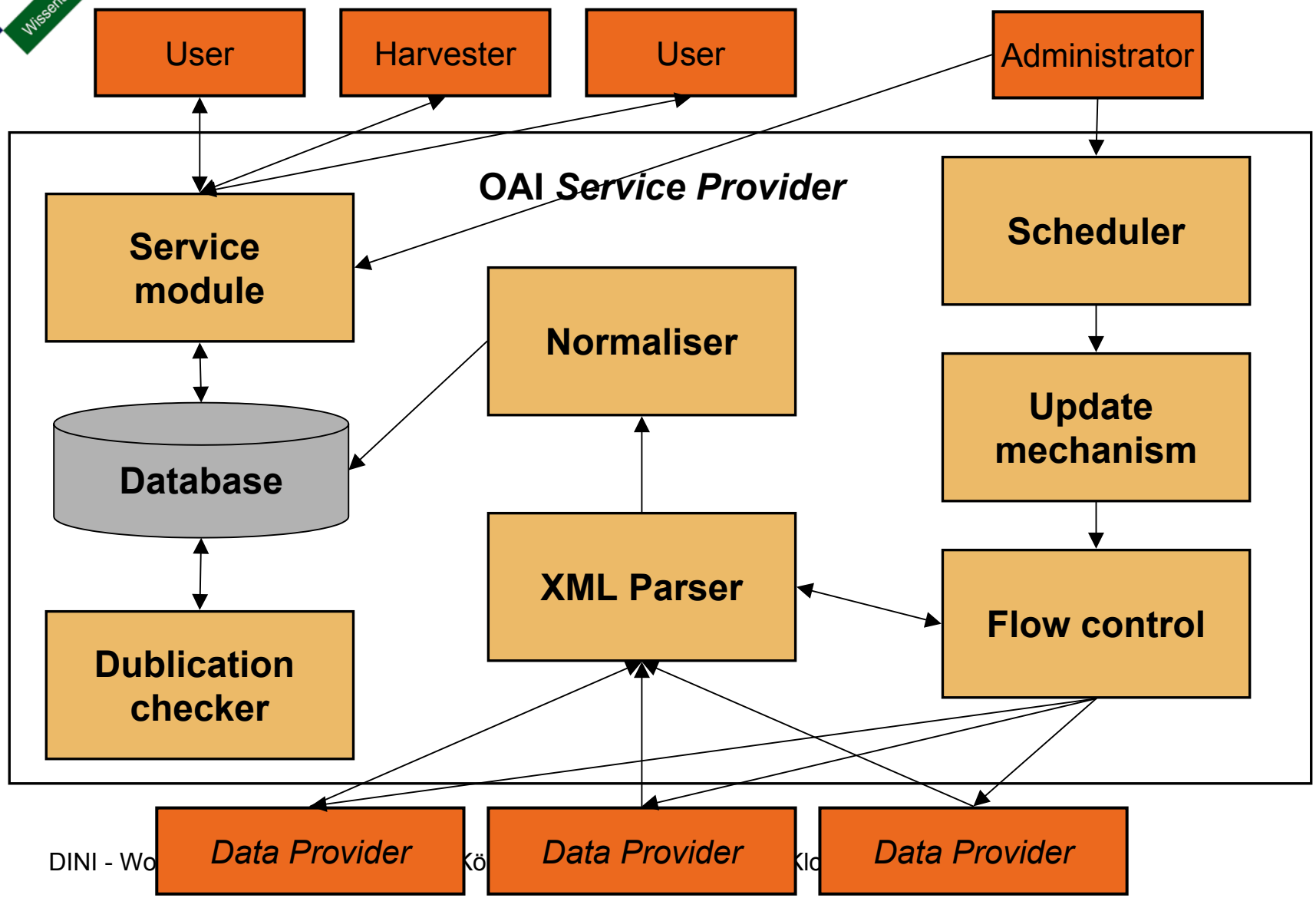
Dubletten-Checker

- führt 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

Service-Modul / Dienst

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

Service Provider: Architektur



Service Provider: Vorgehen beim 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*m
  n = # der Identifier, m = # der metadatenformaten
- ListRecords für jedes benötigte Metadatenformat

  Anzahl (#) langer HTTP Anfragen = m
  m = # of metadata formats

Service Provider: Vorgehensweise

- Benutze einen Zeitplan (schedule), um regelmässig einzusammeln
- Speichere das Datum des letzen Einsammelns (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

Service Provider: Intermediate Systems

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

Service Provider: Werkzeuge

- Beispielcode findet sich auf der OAI-Website
- XML-parser – abhängig von der Plattform – siehe W3C-Website
- XML Schema validator
- Beispiele von Data Providern – siehe OAI-Website für eine Liste von öffentlichen Archiven

Service Provider: OAI Communities

- gemeinsame Metadatenformate
  - z.B. E-Print-Format, ETD-MS, VRA Core, IMS
- gemeinsame Semantik
  - kontrolliertes Vokabular für Felder
  - spezifische Felder für externe Links (DC: Identifier)
- Geschlossene OAI-Netzwerke
  - Transfer von Daten zwischen heterogenen Systemen innerhalb einer Organisation
  - globale Optimierungen möglich (Sets, Metadatenformate)
- OAI innerhalb von Digital Libraries (DL)
  - z.B. Browsing über Sets
  - Reviews, Annotations können unabhängige OAI-Data Provider sein
  - Open Digital Library Project: http://oai.dlib.vt.edu/odl

Service Provider: Test und Registrierung

- Abfragen registrierter (→ OAI-kompatibler) Data Provider
- Testen des Verhaltens des Service Providers
- Offizielle Registrierungsseite
  - http://www.openarchives.org/service/registerasprovider.html
  - Angeben von Information über das Archiv
  - Webseite, Email-Adresse, ...

Links

- Open Archives Initiative

  http://www.openarchives.org

- OAI Metadata Harvesting Protocol

  http://www.openarchives.org/OAI/openarchivesprotocol.htm

- 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

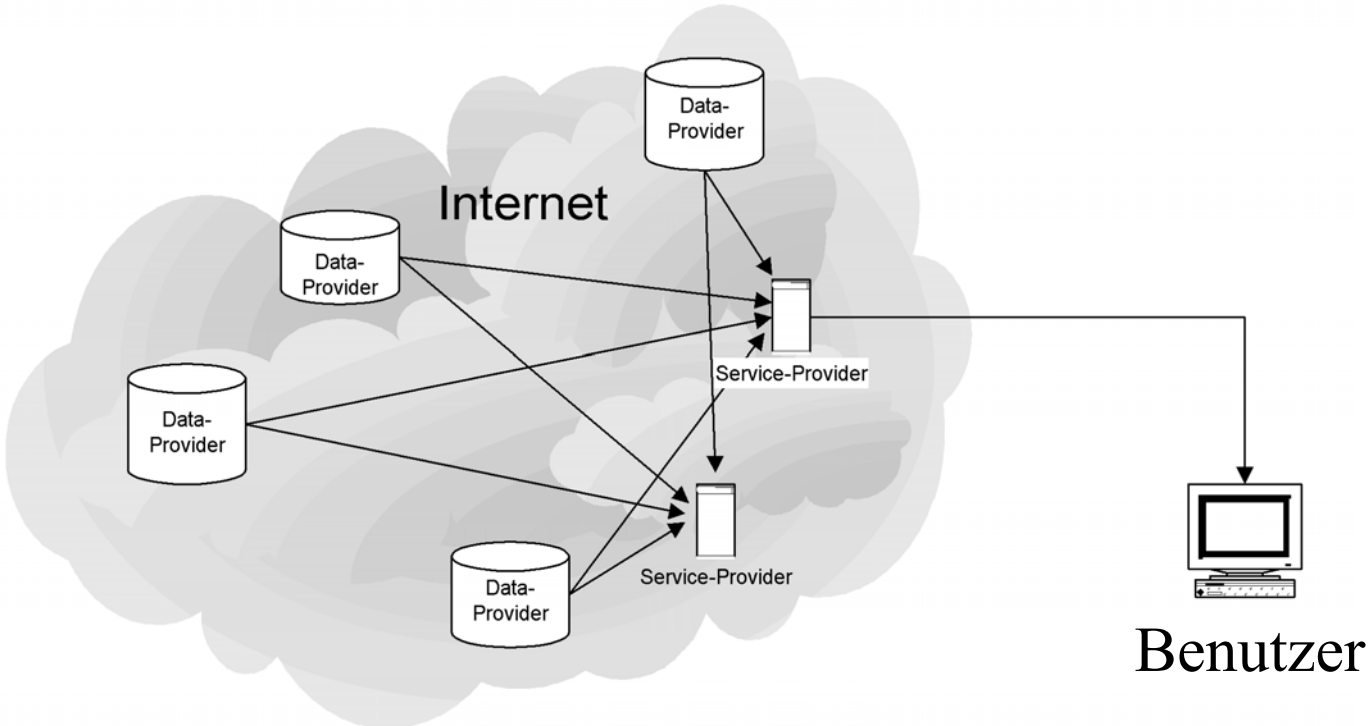
Gliederung des Workshops

- Teil I - Geschichte und Überblick
- Teil II - OAI Serviceprovider - Beispiele
- Teil III Technische Einführung
- Mittagspause
- Teil IV Implementation data provider and service provider
- Part V Sets
- Part VI Realisierung auf Verbundebene
- Part VII Metadaten

Übersicht zum Teil 5 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.

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

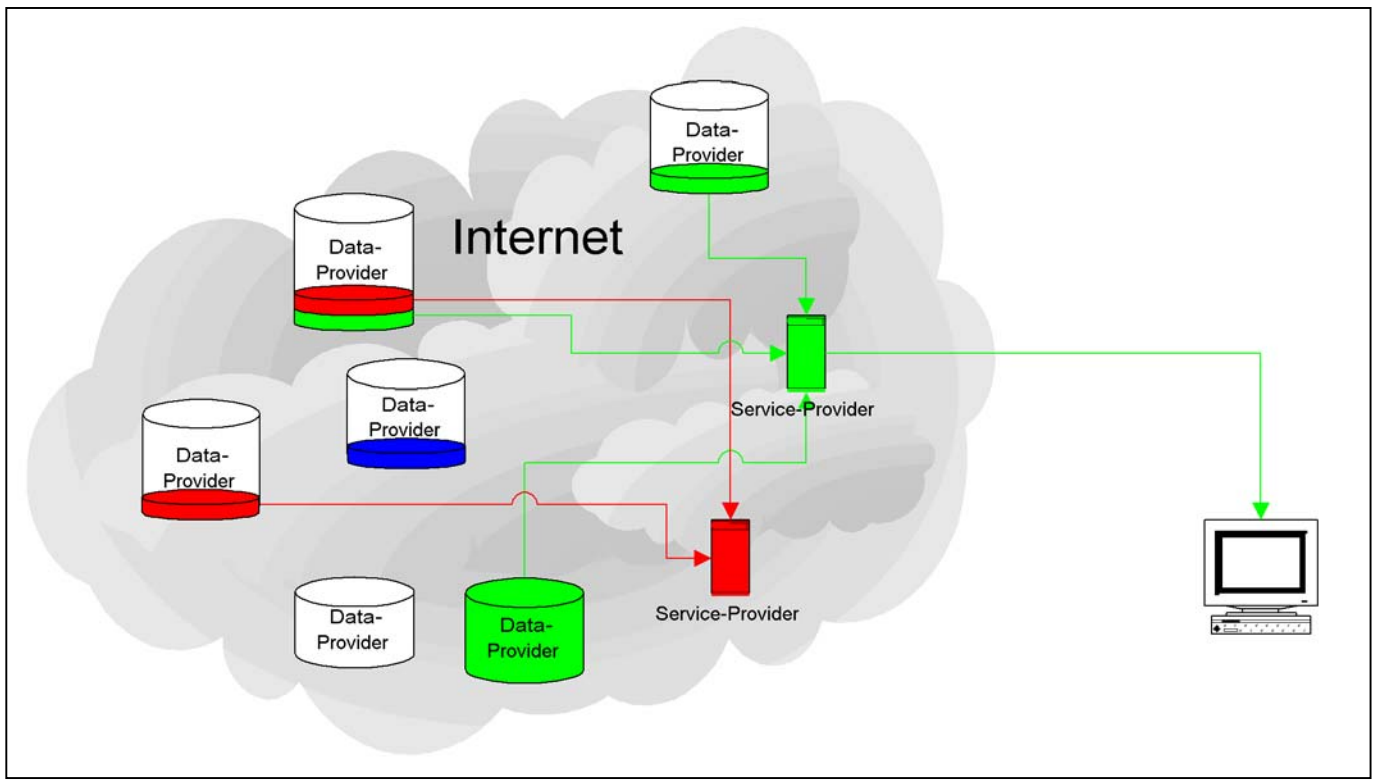
- The actual meaning of a set or of the arrangement of sets in a repository is not defined in the protocol. It is expected that individual communities may formulate well-defined set configurations with perhaps a controlled vocabulary for setNames and setSpec , and may even develop mechanisms for exposing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

- Dini-Empfehlungen zu Sets

Übersicht zum Teil 5 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	SetName
doc-type	formal document-type
doc-type:text	Text
doc-type:notes	Notes
doc-type:image	Image
doc-type:audio	Audio
doc-type:video	video
doc-type:multimedia	multimedia
doc-type:data	data
doc-type-binary	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.

Part VI 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

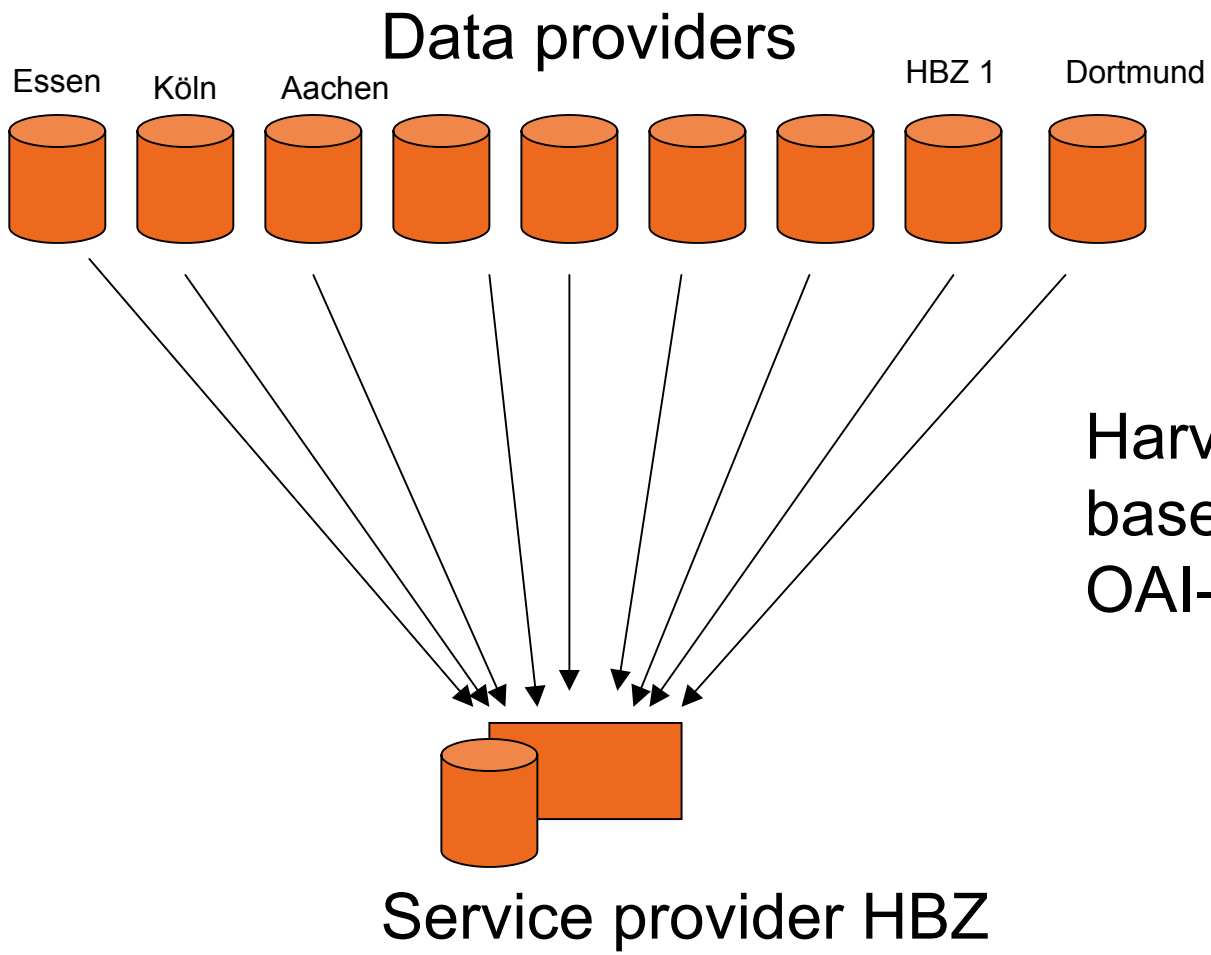
Ausgangslage

- 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.

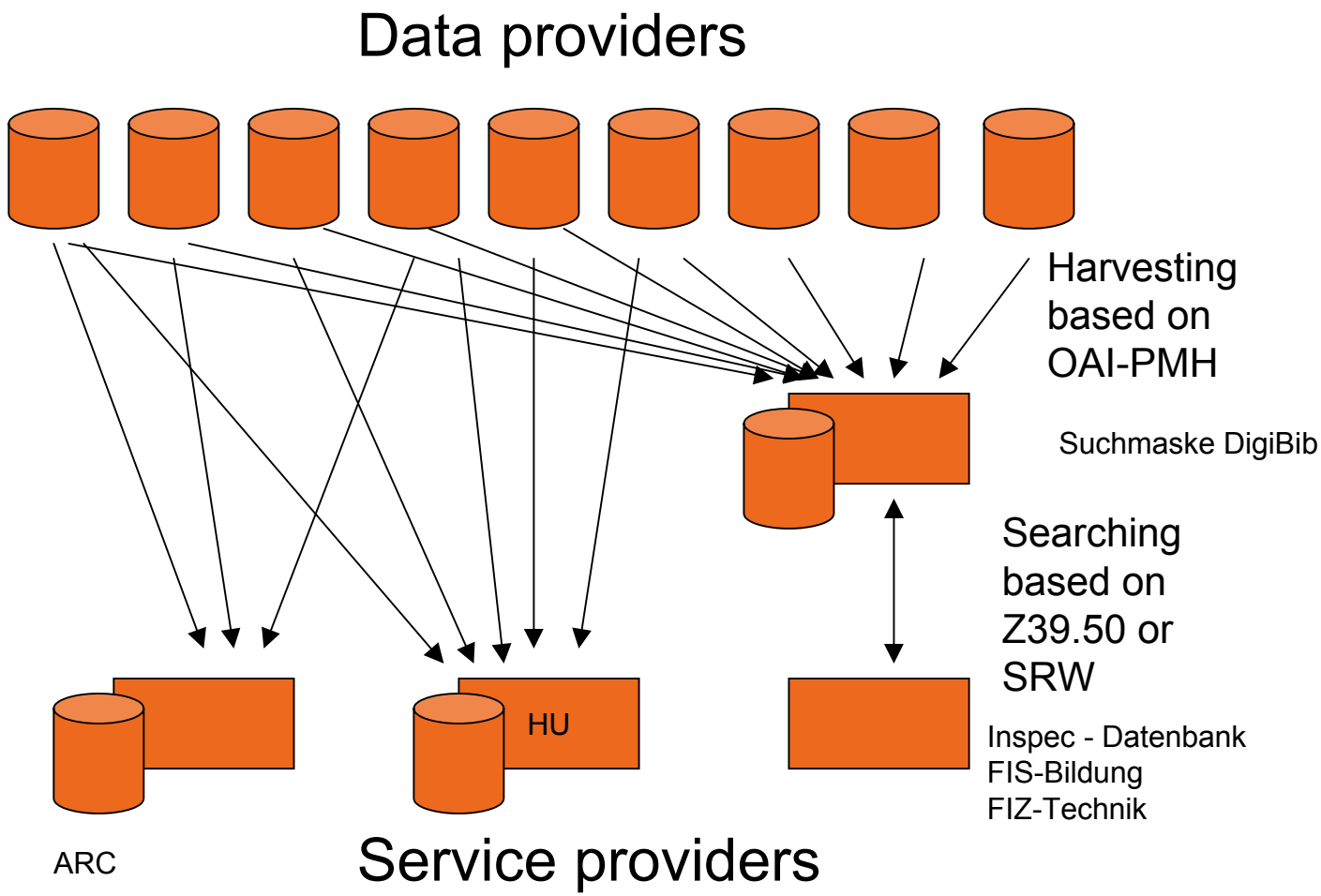
OAI ist die Lösung

- 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

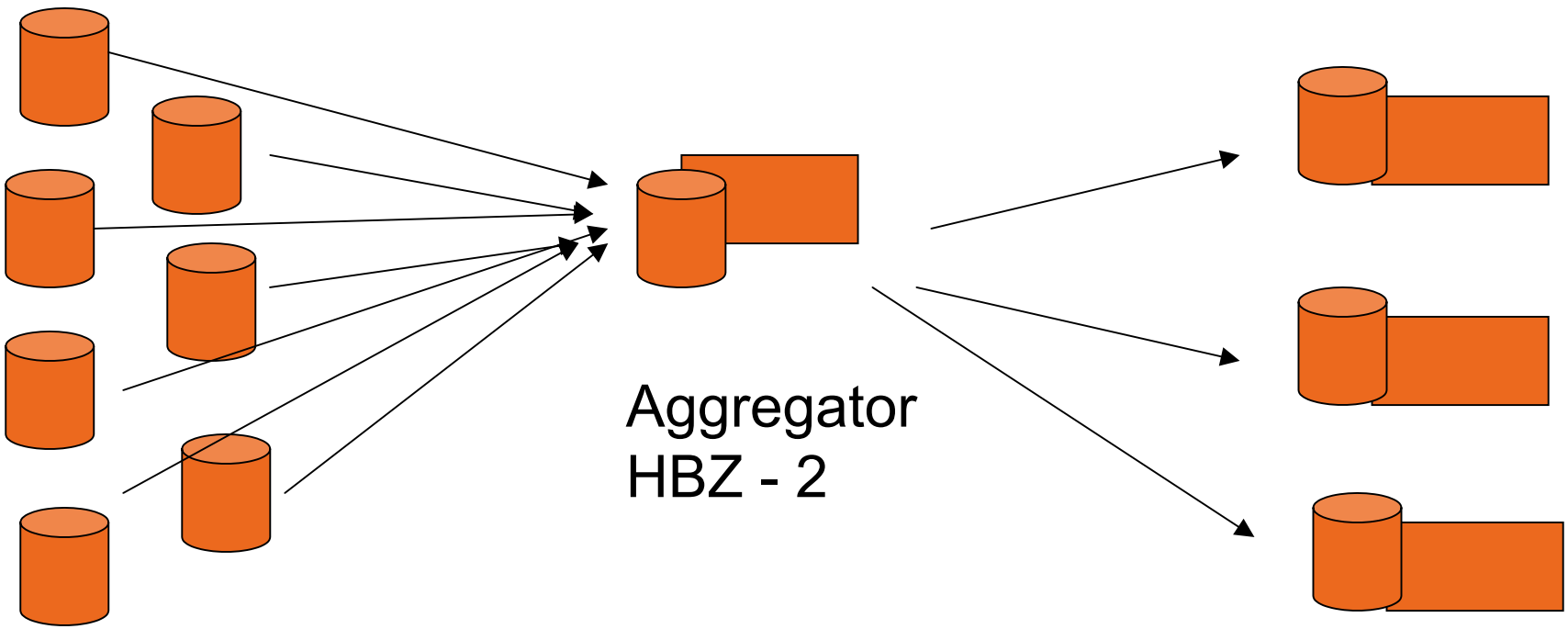
Datenprovider und SP HBZ



Integration in die DigiBib



Aggregierender DP im HBZ



Data providers - NRW

Service providers

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.

Teil VII Metadaten

(direkt übernommen von Andy Powell)

- Einführung
- XML Schema Details
- Erweiterung von oai_dc für die eigene Anwendung
- 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

  OLAC (Open Language Archive Community)

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

Danksagung

Meinen Mitstreitern in der DINI Arbeitsgruppe
Susanne Dobratz HU Berlin,
Uwe Müller HU Berlin,
Frank Scholz UB Stuttgart und
Bernd Diekmann BIS Oldenburg
Heinrich Stamerjohanns, Universität Oldenburg

Allen, bei denen wir uns Folien „entliehen“ haben,

und Ihnen Danken wir für Ihre Aufmerksamkeit !