



Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Projekt-Ausarbeitung

Sven Schliesing

Metadaten

Sven Schliesing

Thema der Projekt-Ausarbeitung

Metadaten

Stichworte

Metadaten

Kurzzusammenfassung

In dieser Ausarbeitung wird der Verlauf und das Ergebnis des Teilprojekts "Metadaten" dargestellt. Das Teilprojekt war Bestandteil des Projekts "Ferienclub" im Studiengang "Informatik Master". Das Ziel des Teilprojekts war die Erstellung einer Architektur zur Sammlung, Verwaltung und Anreicherung von Metadaten aus unterschiedlichen Programmen.

Basierend auf dem Projektverlauf werden Architektur, verwendete Technologien sowie erwonnene Erkenntnisse dargestellt.

Inhaltsverzeichnis

1	Einleitung	4
2	Struktur	5
3	Das System „Metamind“	6
3.1	Metabase	6
3.1.1	Metapic	6
3.1.2	Die Servlets	9
4	Kommunikation	10
5	verwendete Technik	11
6	Problematik	11
7	Fazit	12

1 Einleitung

Im Rahmen des Projekts „Ferienclub“ wurde von den Studenten des Studiengangs „Informatik Master“ eine Vielzahl von Themen behandelt. Eine Teilgruppe (Lutz Behnke und Sven Schliesing) behandelten hierbei das Thema der Speicherung und Verwaltung von Daten und ihren zugehörigen Metadaten. Hierbei beschäftigte sich Lutz Behnke mit der Frage, wie Daten am intelligentesten in einer verteilten Umgebung gespeichert werden können (Behnke, 2005). Die notwendige Interaktion mit dem Benutzer sollte hierbei so klein wie möglich gehalten werden. Diese Speicherung von Daten bezog sich allerdings nur auf die reinen Daten. Eine Speicherung und Verwaltung von, zu diesen Daten gehörenden, Beschreibungen (Metadaten) wurde ausgelagert und im weiteren von Sven Schliesing in einem eigenen Teilprojekt behandelt.

Basis dieser Arbeiten war die Ausarbeitung im Rahmen von „Anwendungen 1“ im vorhergehenden Semester (SS 05), sowie der Vortrag und die Ausarbeitung (Schliesing, 2005) im Rahmen des „Seminar Ringvorlesung“ im gleichen Semester (WS 05/06).

In dieser Ausarbeitung soll nun genauer beschrieben werden, wie die Arbeit im Projekt verlaufen ist. Im Abschnitt „Hintergrund“ wird kurz auf die Problematik sowie die verwendete Technik eingegangen. Die bereits getätigten Vorarbeiten, z.B. im Seminar, werden im Abschnitt „Vorarbeiten“ dargestellt. „Realisierung“ beinhaltet eine genauere Beschreibung der Architektur. Den Abschluss bildet das „Fazit“ mit einer Betrachtung der erreichten Ziele sowie möglichen weiterführenden Arbeiten.

2 Struktur

Die ursprüngliche Idee zu diesem Teilprojekt entstand als Lutz Behnke das Konzept seiner Projektarbeit erstellte. Ziel seines Systems „Wotan“ war es, beliebige (binäre) Daten (z.B. Bilder oder Videos) von einem Client anzunehmen, abzuspeichern und mit einem eindeutigen Schlüssel zu versehen.

Ein Punkt der bei diesem Ansatz jedoch außen vor blieb, war die Speicherung und Verwaltung der anfallenden Metadaten. Angefangen bei simplen Eigenschaften wie dem Dateinamen bis hin zu Schlüsselwörtern mit denen die entsprechenden Daten wiedergefunden werden können. Auch sollte der Benutzer nie mit dem, von „Wotan“ generierten, Schlüssel in Verbindung kommen, da dieser voraussichtlich ein schwer zu merkender und für den normalen Benutzer recht unansehnlicher Hash-Wert werden würde.

Aus diesem Problem wurde das Teilprojekt „Metamind“ geboren, dass vom Author bearbeitet wurde.

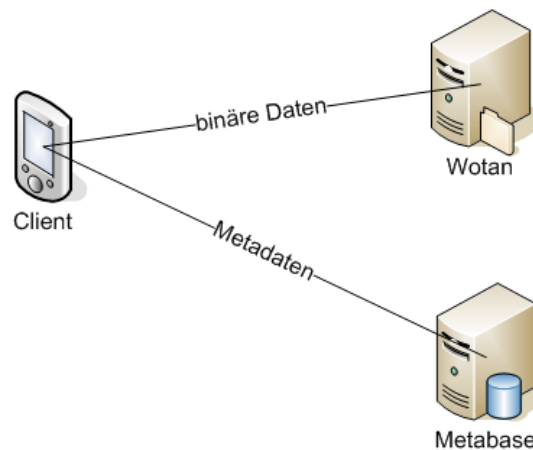


Abbildung 1: angedachte Systemstruktur

Wie im Schaubild 1 zu erkennen ist, folgt eine recht simple und auch strikte Trennung: Die Binärdaten werden für den Client von „Wotan“ übernommen und gespeichert, während alle beschreibenden Daten, also die Metadaten, an „Metabase“ (Server des Systems „Metamind“) gehen.

In dieser Darstellung wird ein Detail nicht dargestellt. Für jedes System, „Wotan“ sowie „Metamind“, existiert auf dem Client ein entsprechendes Client-Programm, was alle entsprechenden Aufgaben weitestgehend für den User transparent übernimmt.

Der Aufbau der Datenstruktur zeigt Abbildung 2.

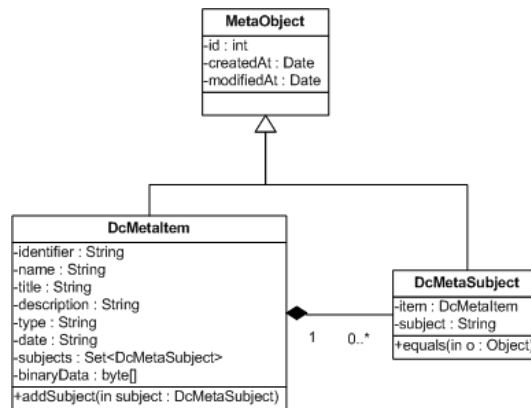


Abbildung 2: Datenstruktur

3 Das System „Metamind“

Das System „Metamind“ stellt eine klassische Client-Server-Architektur dar. Der Name ist abgeleitet von dem Spiel „Mastermind“ und dem Begriff „Metadaten“. Die zum Einsatz kommende Technik wird näher im Abschnitt „verwendete Technik“ erläutert.

3.1 Metabase

Der Server wird „Metabase“ genannt. Hier erfolgt die Speicherung und Anreicherung der vom Client angelieferten Daten. In der Realisierung wurde eine J2SE-Anwendung geschrieben, welche in einem Tomcat-Server läuft. Diese stellt dem Anwender, bzw. dem Client-Programm drei Servlets zur Verfügung. In Abbildung 3 werden 2 (Importer- und SearchServlet) dieser Servlets gezeigt. Es handelt sich hierbei um diejenigen, die für die Kommunikation zwischen Client-Programm (z.B. „Metapic“) und dem Server („Metabase“) zuständig sind. Das dritte (EnrichServlet) wird in Abbildung 7 gezeigt.

3.1.1 Metapic

Eine Beispiel-Client-Anwendung wurde mit „Metapic“ (Abbildung 4) realisiert. Die Kommunikation zwischen dem User und dieser Anwendung wird in Abbildung 6 gezeigt.

Ziel dieser Anwendung war es nicht, eine voll ausgereifte Verwaltung für Bilder zu schaffen. Sie sollte lediglich als „Proof of concept“ für das gesamte System dienen. Genau wie diese Beispielanwendung auf der zentralen Server „Metabase“ zugreift, sollte es auch Dritt-Programmen möglich sein, dessen Dienste zu nutzen.

In dieser Version des Programms ist es möglich, Bilder aus dem Speicher des PDAs zu laden, zu annotieren (Abbildung 5) und diese Informationen an den Server zu übertragen. Weiterhin ist es möglich, eine Suche (nach Keywords) durchzuführen.

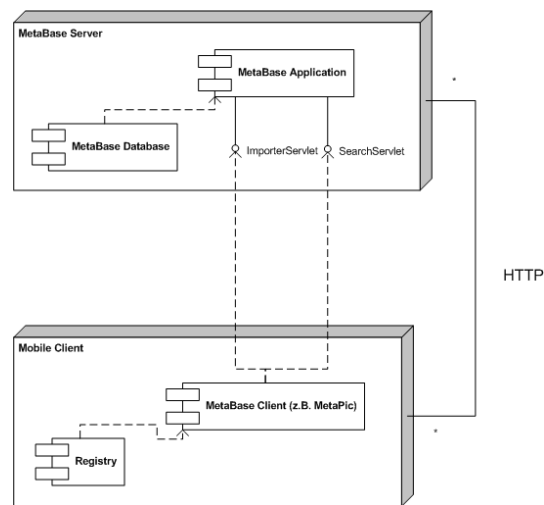


Abbildung 3: Struktur der Client-Server-Interaktion

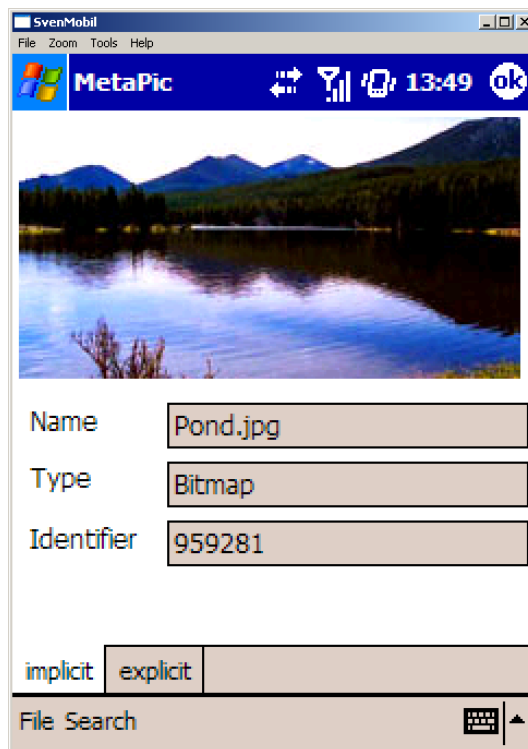


Abbildung 4: Beispielanwendung: Metapic

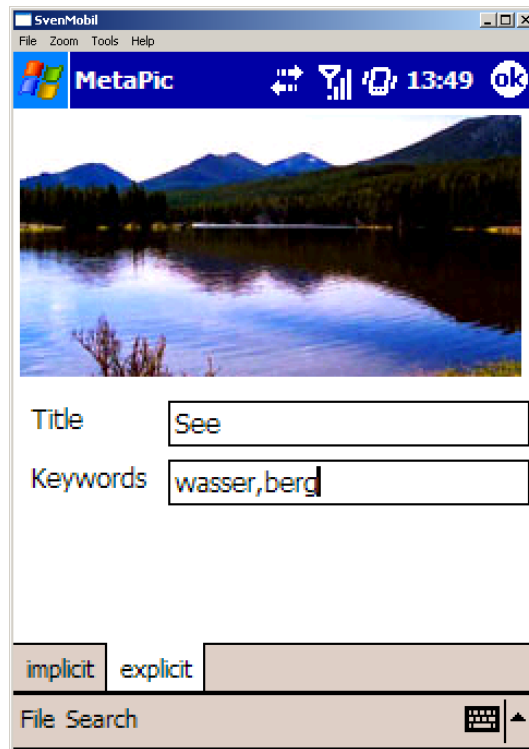


Abbildung 5: Beispielanwendung: Metapic, Annotierung

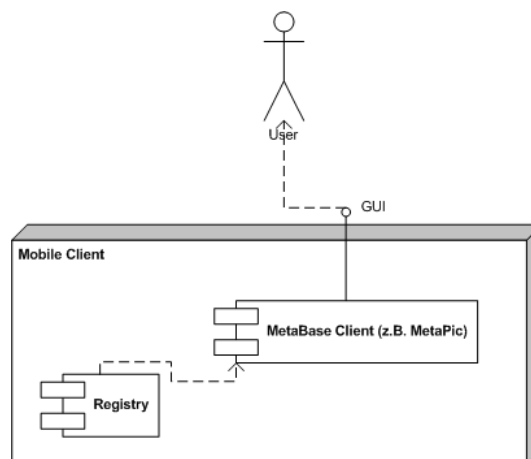


Abbildung 6: Zugriff des Users auf die Client-Applikation

3.1.2 Die Servlets

Wie schon beschrieben erfolgt die Kommunikation zwischen Client und Server, aber auch zwischen User und Server(-Anwendung) über Servlets. Im folgenden werden diese beschrieben:

- **ImporterServlet:** Hierdurch erfolgt die Datenübernahme. Das Servlet empfängt die Nachricht von der Client-Anwendung (z.B. „Metapic“), analysiert diese und übernimmt sie in die interne Datenstruktur (Abbildung 2).
- **SearchServlet:** Wird vom Client eine Suchanfrage geschickt, verarbeitet dieses Servlet diese. Es sucht dann in der Persistenzschicht nach den in Frage kommenden Objekten und gibt diese an den Client zurück.
- **EnrichServlet:** Dieses Servlet bietet im Gegensatz zu dem Importer- und SearchServlet eine Ausnahme. Es wird nicht vom Client-Programm angesprochen, sondern vom User direkt via Web-Interface angesprochen. In einer späteren Phase des Projekts hätte dieses Servlet desweiteren keine Daseinsberechtigung mehr sondern wäre durch einen zyklisch ablaufenden Job abgelöst worden. Aus Gründen der Zeit und des kontrollierten Ablaufs wurde jedoch auf diesen Schritt verzichtet. Aufgabe des EnrichServlet ist die Anreicherung der für ein Objekt eingetragenen Keywords. Genauer gesagt kommuniziert es mit dem „EnrichManager“, der aufgrund der bereits vorhandenen Keywords eine Menge an zusätzlichen Keywords zurückgibt. Das EnrichServlet fügt diese nun der schon vorhandenen Menge von Keywords hinzu.

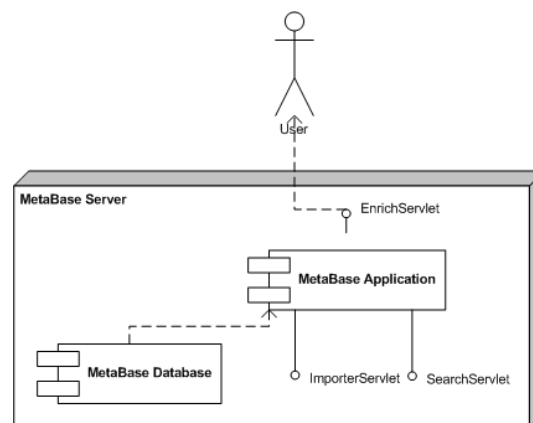


Abbildung 7: Benutzung des Enrich-Servlets

4 Kommunikation

Die Kommunikation zwischen dem Client und dem Server erfolgt, wie Abbildung 3 zeigt via http. Ursprünglich war die Anbindung via Web Services geplant. Der unvollständigen Dokumentation sowie teilweise fehlerhaften Implementation von Axis2 ist es jedoch zuzuschreiben, dass eine einfachere Variante, sozusagen als Fallback-Lösung, verwendet wurde. In dieser Lösung wird jegliche Kommunikation in ein XML-Dokument verpackt und via http an den jeweiligen Kommunikationspartner verschickt.

Beispiel einer Import-Anfrage:

```
<?xml version="1.0" encoding="utf-8"?>
<metabaseentry>
  <title>See</title>
  <description>Pond.jpg</description>
  <type>Bitmap</type>
  <date>02.02.06</date>
  <subject>wasser</subject>
  <subject>berg</subject>
  <identifizier>1378029</identifizier>
</metabaseentry>
```

Beispiel einer Such-Anfrage:

```
<?xml version="1.0" encoding="UTF-8"?>
<dc>
  <title>Bergsee</title>
  <description>Pond.jpg</description>
  <type>Bitmap</type>
  <date>22.12.05</date>
  <identifizier>949625</identifizier>
  <subject>berge</subject>
  <subject>blau</subject>
  <subject>frisch</subject>
  <subject>wasser</subject>
</dc>
```

Dieser Ansatz hat gegenüber Web Services den großen Nachteil, dass die Konstruktion der Nachrichten sowie das Lesen selbiger von der Anwendung implementiert werden muss. Hierbei kann es leicht zu Fehlern kommen und das Debugging ist aufwendig. Allerdings gibt es auch den Vorteil, dass diese Art schneller und weniger Rechenintensiv ist als der Ansatz mit Web Services. So spart man sich z.B. den Overhead der SOAP-Implementation. Auf mobilen Geräten ist dies nicht ganz unerheblich.

5 verwendete Technik

Hier wird nun auf die verwendeten Techniken eingegangen.

- **Server „Metabase“:** Wie schon erwähnt ist die Anwendung „Metabase“ eine J2SE-Anwendung die in einem Tomcat-Server läuft. Als Datenbasis wurde MS SQL-Server verwendet, die Persistenzschicht ist „subpersistence“, welche wiederum Hibernate verwendet. Als Entwicklungsumgebung kam Eclipse (3.2.0) zu Einsatz.
- **Client „Metapic“:** Die Anwendung „Metapic“ wurde für die Plattform „Windows Mobile 2003“ entwickelt. Es wurden jedoch keine bekannten Besonderheiten benutzt, so dass eine Portierung auf z.B. „Windows Mobile 5“ kein Problem darstellen sollte. Die Entwicklungsumgebung wurde, während des Projektes, von „Visual Studio 2003“ auf „Visual Studio 2005“ umgestellt. Ein Schritt der Unter „Problematik“ weiter erläutert wird.

6 Problematik

Wie schon erwähnt wurde mitten im Projekt die Entwicklungsumgebung von „Visual Studio 2003“ auf „Visual Studio 2005“ umgestellt. Dieser Schritt ist rationell nicht hinreichend zu begründen. Lediglich die verbesserte Anbindung an das mobile Gerät (für Zwecke des Debugging) sollte ein Grund gewesen sein. Die natürliche Neugier nach Neuerscheinungen überwog jedoch bei der Entscheidung, die einige Wochen später bereut wurde. Abbildung 8 zeigt einen typischen Fehler für den weder die Online-Hilfe noch langwierige Suchen im Internet eine entsprechende Lösung brachten.

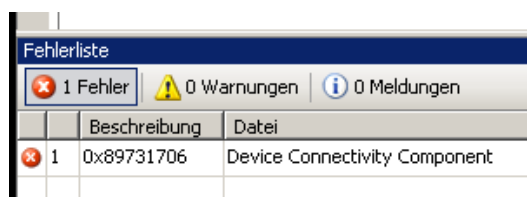


Abbildung 8: Nicht behebbarer Fehler

Ebenso wurde lange Zeit mit der Beta-Version von Hibernate (3.1) gearbeitet, bis zu viele unerklärliche Phänomene zum logischen Rückschritt auf eine stabile Version (3.0.5) führten. Wie bereits erwähnt sorgte ebenfalls die fehlerbehaftete Implementierung von Axis2 sowie dessen unvollständige Dokumentation für Ausfälle in der Produktivität.

7 Fazit

Im Projekt wurde ein Ziel sicher erreicht: Die Akzeptanz-Stufe für fehlerhafte Software wurde weit nach unten gesenkt. Die aufgetretenen Probleme mit „Visual Studio 2005“ sowie „Axis2“ lehrt, dass neuere Versionen vielleicht mehr Funktionalität bieten, diese aber ohne bestehende Stabilität nicht für den Produktivbetrieb geeignet ist.

Desweiteren ist klar geworden, dass die Ähnlichkeit zwischen C# und Java auf den ersten Blick hilfreich scheint. Allerdings gibt es kleine aber feine Unterschiede die im „Programmieralltag“ all zu oft Probleme bereiten, wo keine Probleme vermutet werden.

Abgesehen von Problemen mit Drittsoftware ist das Projektziel größtenteils erreicht worden. Die geplante Verschmelzung mit dem Projekt „Wotan“ von Lutz Behnke ist leider aus Gründen der Komplexität von Lutz' Thema und seiner Umstellung von C# zurück auf Java nicht möglich gewesen. Dies wäre zum Projektstatus von „Metamind“ allerdings nur noch ein kleiner Schritt. Es müsste lediglich der Part implementiert werden, wo das Objekt an „Wotan“ zur Verarbeitung geschickt wird. Mit mehr Zeit und besserer Koordination wäre dies unter günstigen Umständen im Rahmen des Projekts möglich gewesen.

So lässt sich dieser Punkt als „weiterführende Arbeit“ verbuchen.

Die ursprüngliche Idee war, eine Implementation des „METS“ (Metadata Encoding and Transmission Standard) (metstutorial, 2005). Das METS ist sehr Eng mit dem Konzept von „OAIS“ (Brübach, 2003) verbunden. Diese Idee wurde jedoch sehr schnell verworfen, da der Implementierungsaufwand den Rahmen des Semesterprojekts weit gesprengt hätte. Außerdem wäre eine Integration von „Wotan“ in diesen Ansatz ebenfalls nicht ganz trivial gewesen.

Literatur

- [metstutorial 2005] : *METS: An Overview & Tutorial*. Mai 2005. – URL <http://www.loc.gov/standards/mets/METSOverview.v2.html>
- [Behnke 2005] BEHNKE, Lutz: *Thesis Outline*. 2005. – URL <http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master05-06/behnke/abstract.pdf>
- [Brübach 2003] BRÜBACH, Nils: *OAIS–Das “Open Archival Information System“: Ein Referenzmodell zur Organisation und Abwicklung der Archivierung digitaler Unterlagen*. August 2003. – URL http://www.sachsen.de/de/bf/verwaltung/archivverwaltung/pdf/pdf_onlinepublikationen/pp_bruebach.pdf
- [Schliesing 2005] SCHLIESING, Sven: *Ausarbeitung Seminar: Metadaten*. 2005. – URL <http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master05-06/schliesing/abstract.pdf>
- [Ullmann 2004] ULLMANN, Richard: *ISO 14721:2003 - OAIS; A Reference Model for an Open Archival Information System*. Juli 2004. – URL <http://apan.net/meetings/cairns2004/presentation/escience-ullman.ppt>