# Ariadne

-

Ein System zur Integration von Web-Daten

Markus Kirschmann
(mk10@informatik.uni-ulm.de)

Mai 2001

Um Informationen von verschiedenen Web-Seiten in eine einzige Seite zu integrieren, mußten bis jetzt immer zuerst spezielle Anwendungen erstellt werden. Dieses Erstellen ist meist wegen der Menge der zu integrierenden Daten eine zeitaufwendige und damit auch kostenintensive Arbeit. Mit Ariadne wird nun ein Werkzeug bereit gestellt, das die schnelle Herstellung solcher Anwendungen ermöglicht. Ariadne stellt verschiedene Module zur Extraktion, Verifikation und dem Einbinden von Daten verschiedener Web-Ressourcen in eine einzige Anwendung zur Verfügung. Außerdem verfügt Ariadne über ein Modul das die Antwort - Zeit durch Query - Planung und optimale Nutzung eines Caches verbessert. Zusätzlich kann Ariadne selbständig Änderungen in den eingebundenen Web-Ressourcen entdecken und verarbeiten.

## 1. Einleitung und Grundlagen

Heute, im Zeitalter der neuen Medien gibt es zu eine Unmenge von Daten und Informationen zu jedem Thema, die über das Internet (meistens) frei erhältlich sind. Diese Datenflut nimmt von Tag zu Tag zu. Allerdings sind die Möglichkeiten, diese Informationen zu suchen und einfach zu integrieren meist sehr beschränkt. Oft sind nur einfache Suchmaschinen vorhanden, die kontext - frei nach Stichworten suchen. Um aber eine effektive Nutzung der Daten zu ermöglichen, wäre es wünschenswert, nach bestimmten Themengebieten suchen zu können, bzw. verschiedene Daten automatisch kombinieren zu lassen. Sucht man z.B. die Hauptstätte zu den NATO - Ländern, so muß man zuerst nach den Ländern suchen, dann diese Auswahl selektieren und dann noch auf den Seiten der jeweiligen Länder die Hauptstädte herauslesen. Wesentlich effektiver wäre es, dies alles von einem Programm erledigen zu lassen.
Durch Ariadne wird ein erster Schritt in diese Richtung gemacht. Ariadne ermöglicht es, die Daten von verschiedene Seiten gezielt über eine Schnittstelle (sprich eine Webseite) anzusprechen und zu kombinieren. Ariadne kann z.B. so in eine Webseite integriert werden, daß eine Anfrage (z.B. über ein Eingabe-Feld) nach den Hauptstädten der NATO-Länder sofort das gewünschte Ergebnis liefert. Zudem entdeckt Ariadne automatisch, wenn die verwendeten Webseiten aktualisiert wurden und übernimmt diese Änderungen dann selbständig.
Ariadne besteht aus verschiedenen Modulen, und kann dadurch leicht angepaßt werden

## Grundlagen

- SIMS, Wrapper -

Ariadne ist eine Weiterentwicklung des SIMS - Modells für Informationen aus dem Internet. Dieses Modell basiert auf der Idee, verschiedene "Datenstrukturen" (z.B. Web-Seiten, Datenbanken, Sensoren, oder andere Daten-Quellen) über eine einzige Schnittstelle, dem Information-Mediator, anzusprechen. Dieser Information-Mediator (in unserem Fall Ariadne) zerlegt und optimiert die Anfragen des Benutzers und gibt diese dann an sogenannte Wrapper weiter. Diese Wrapper übersetzen die an sie gestellte Teilanfrage in eine Sprache, die die jeweilige, dem Wrapper zugeordnete, Datenstruktur akzeptiert. Dann übergibt der Wrapper die Anfrage

an die Datenstruktur oder liest die Daten direkt aus der Struktur und übersetzt das Ergebnis wieder zurück in die vom Information-Mediator benutzte Sprache. Ist die Datenstruktur z.B. eine einfache HTML-Seite, so sucht der Wrapper die jeweiligen Stellen auf der Seite, die den angefragten Daten-Feldern entsprechen , extrahiert dann die Daten und liefert diese in der Sprache des Mediators zurück.
Über die selbe Sprache können die Wrapper auch untereinander kommunizieren, wodurch die Anzahl der verschiedenen Übersetzungsmechanismen zwischen den Datenstrukturen gering gehalten werden kann.
Ariadne ermöglicht es nun, diese Idee für Datenstrukturen aus dem Web zu implementieren. Zusätzlich stellt Ariadne intelligente Module zur Wrapper-Erstellung, Datenextraktion und -verifikation aus Websites und zur Anfragen-Optimierung zur Verfügung.
Die Module sind im einzelnen :

- Module zur Wrapper-Erstellung :
  - Erstellen der Extraktionsregeln: **Stalker**
  - Optimale Auswahl der Beispiel-Seiten : **Co-Testing**
- Modul zur Datenverifikation: **Data Pro**
- Module zur Query-Planung
  - Modul zur Aufwands-Planung
  - Modul mit Regeln um die Anfrage zu optimieren (Relationen Algebra)
  - Modul, mit den Strategien zur Optimierung
- Modul zur Cache-Verwaltung : **Cluster and Merge**

Da davon ausgegangen werden kann, daß viele Seiten im Web semi-strukturiert sind (z.b. Listen, Kataloge) extrahiert Ariadne die Daten "kontext-frei", also nicht aus dem sprachlichen Zusammenhang.
Eine Internet - Domain, auf der z.B. verschiedene Bauteile angeboten werden, wird eine Index-Seite in Form einer Liste mit allen Bauteilen enthalten. Aus dieser Liste erhält man dann einen Link auf die Unterseiten mit den Details zum jeweiligen Bauteil. Diese Seiten werden sehr wahrscheinlich alle das gleiche Layout haben, also ist diese Domain semi-strukturiert.

## 2. Struktur und Idee

Hier soll nun anhand eines Beispieles die Funktionsweise und Idee von Ariadne skizziert werden. In unserem Beispiel wollen wir eine Applikation erstellen, die dem Benutzer Anfragen über geographische, demografische und politische Daten aller Länder ermöglicht. Alle diese Informationen sind im Internet vorhanden, aber auf verschiedenen Seiten und Domains verstreut.
In Abbildung 1 wird die Struktur von Ariadne, die in Abschnitt 2 beschrieben wurde, für dieses Beispiel dargestellt.

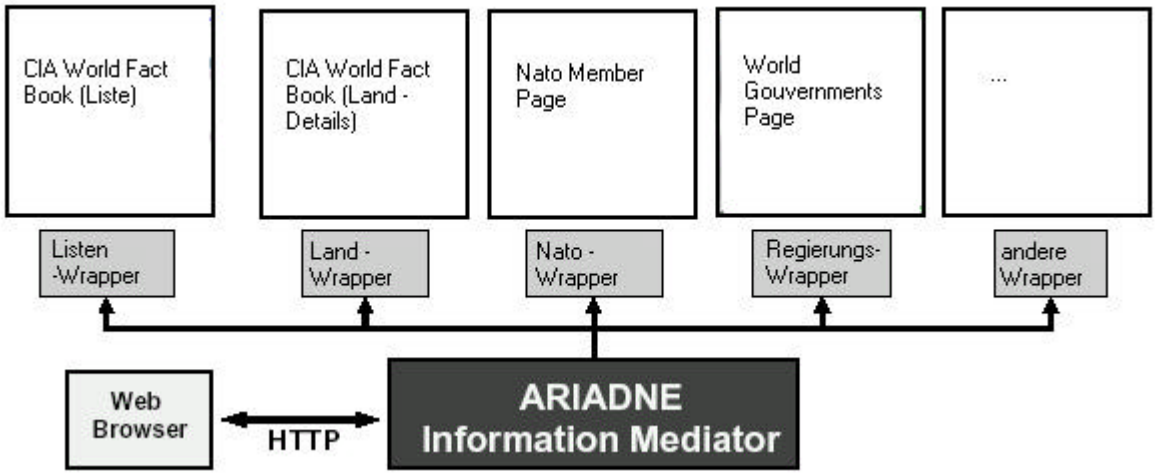



Abb. 1: Ariadne - Struktur

# 3. Datenbank-Modell

Um eine solche Applikation erstellen zu können, überlegen wir uns zuerst, woher wir unsere Daten bekommen, bzw. auf welchen verschiedenen Seiten sie liegen. Dabei sollten die ausgewählten Seiten die Daten möglichst strukturiert darstellen, um eine sicherere Datenextraktion zu gewährleisten. Dies bedeutet aber keine große Einschränkung, da die meisten Seiten im Web irgendeine Art von Struktur enthalten.
Dann setzen wir die Daten über ein Modell zueinander in Beziehung (z.B. über ein Entity-Relationship-Modell). In unserem Fall finden wir z.B. im "CIA World Fact-Book" (Index-seite) die Namen aller Länder, eine Ebene tiefer dann die geographischen Daten (z.B. Längen- und Breitengrad) zu den jeweiligen Ländern. Auf der einer anderen Seite, im folgenden "NATO-Seite" genannt, finden wir politische Informationen über die Mitglieder der NATO, auf der "World Gouvernment Page" finden wir politischen Informationen über die meisten Länder. Nun erstellen wir aus diesen Daten ein Modell, in dem festgelegt wird, welche Seiten (oder Domains) welche Informationen und Attribute liefern. Anstatt nun im Model die einzelnen Objekte mit ihren Attributen einzutragen, werden nun Wrapper eingesetzt, die als Funktionen auf den Objekten angesehen werden können.
Der Land - Wrapper z.B. liefert auf die Eingabe der Land-URL die jeweiligen Attribute des Landes. Wenn wir nun die Wrapper in das Modell integrieren, so erhalten wir für unsere Beispiel-Anwendung folgendes Modell : (Abb. 2)

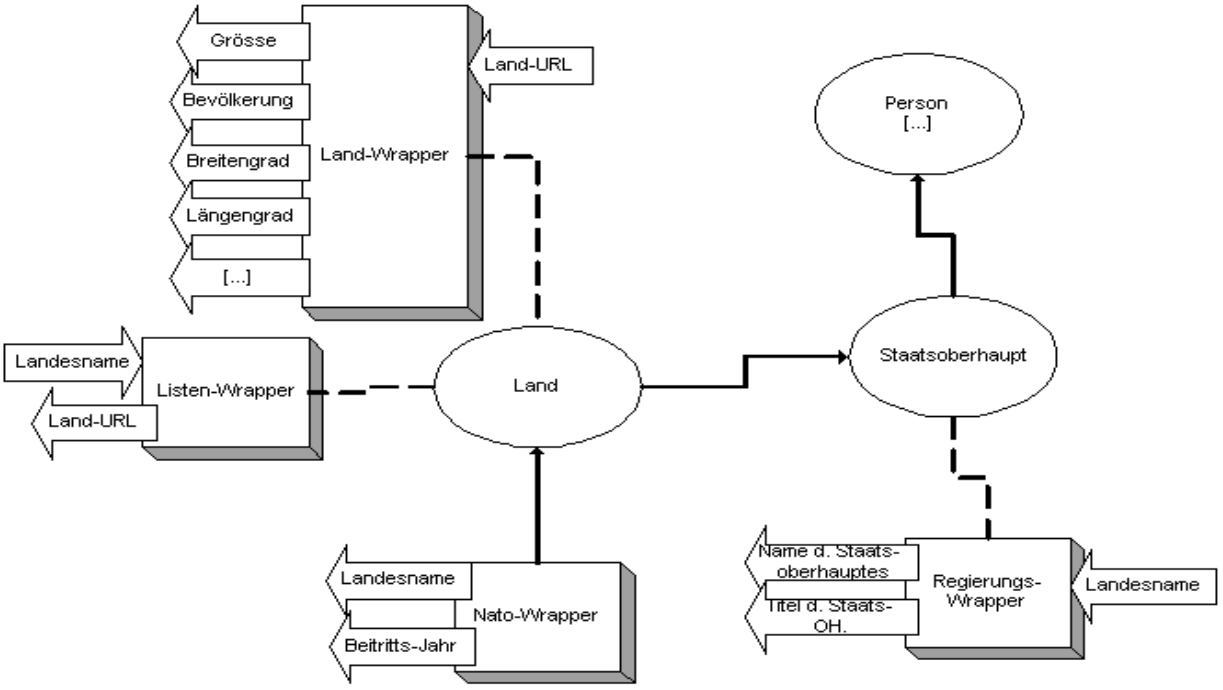


Abb. 2 : Datenbank-Modell mit Wrappern.

# 4. Wrapper

## 4.1 Was sind Wrapper ?

Wrapper sind Programm-Einheiten, die eine globale Syntax bzw. Sprache in die lokale Sprache der jeweiligen Datenstruktur übersetzen. Sie übergeben die Anfrage an die Datenstruktur oder lesen die Daten direkt aus derselben und liefern das Ergebnis wieder zurück in der globalen vom Mediator (= Ariadne) verwendete Sprache. Bei solchen Datenstrukturen kann es sich z.B. um einfache HTML-Seiten handeln,

oder aber auch um Seiten mit Eingabe-Feldern oder Auswahl-Listen. Wrapper werden unter anderem deshalb verwendet, da es einfacher ist, die Daten der Seiten über das normale HTML-Protokoll auszulesen, als die jeweilige Datenbankstruktur der Server direkt anzusprechen.

## 4.2 Wrapper erstellen

Um nun solche Wrapper zu erstellen, gibt es in Ariadne die Möglichkeit, dies anhand von ausgewählten Beispiel-Seiten für jeden Wrapper zu tun. Dabei werden die Daten in den jeweiligen Seiten z.B. über eine Benutzeroberfläche markiert, und STALKER, ein Modul von Ariadne, erstellt dann aus den Beispiel-Seiten Regeln, die zum Extrahieren der Daten benutzt werden. Diese Regeln werden mit jeder weiteren Seite, die der Wrapper verarbeitet, verifiziert oder ggf. überarbeitet. Für das Markieren der Daten stellt Ariadne eine Benutzeroberfläche zur Verfügung : DoUI (demonstration-oriented user interface). In dieser Oberfläche können die Daten durch Cut&Paste markiert werden (Abb. 3)

## 4.3 Daten-Extraktion

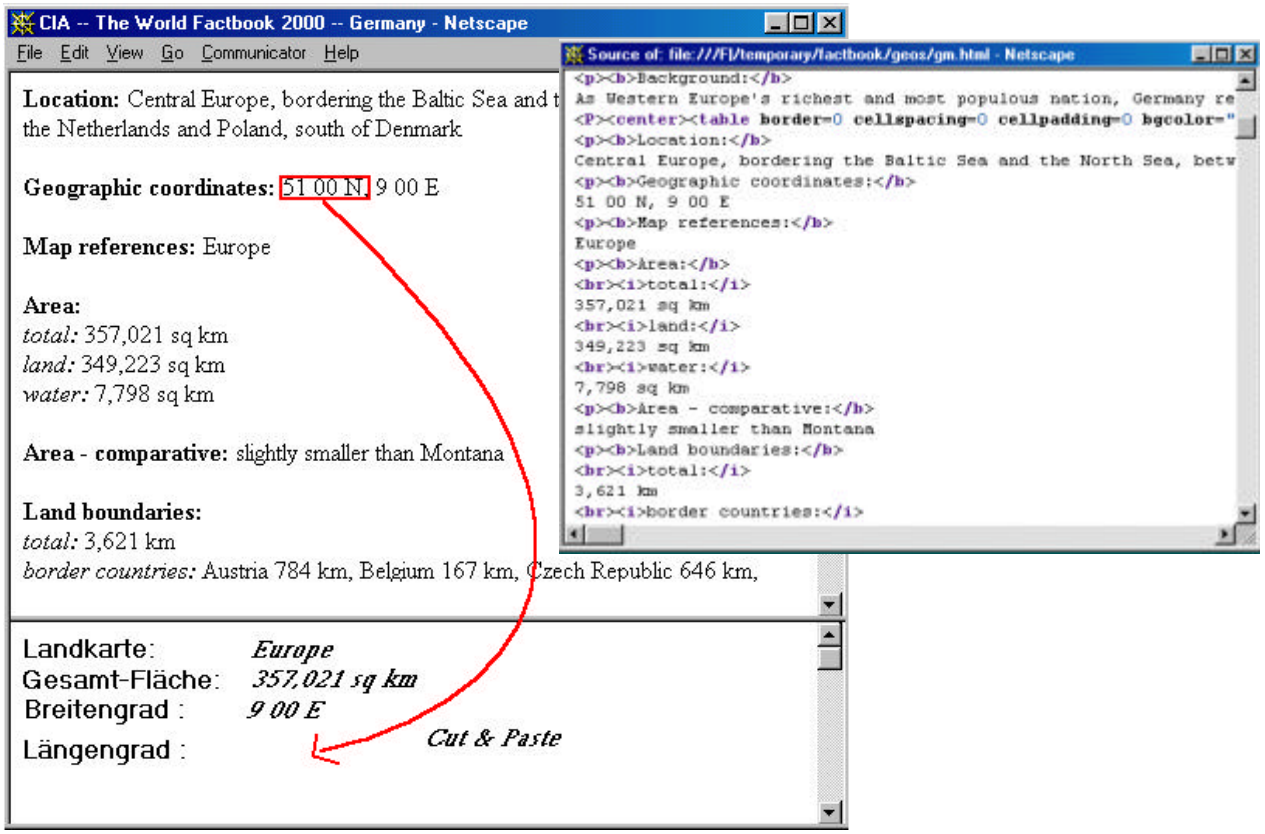


Abb. 3: CIA Fact Book (Oberfläche + Code)

## 4.3.1 Regeln erstellen

In unserem Beispiel würden wir nun aus der obigen Seite die Felder "Europe" als Landkarte, "357,021 sq km" als Gesamt-Fläche, etc markieren. STALKER generiert nun aus diesen Markierungen Extraktionsregeln. Für das Feld "Landkarte" könnte eine Regel z.B. so aussehen :

R1 : SkipTo "Map Reference: </h>"

Diese Regel würde für den Wrapper folgendes bedeuten: Gehe vom Anfang der Seite solange nach unten, bis die Zeichenkette „Map Reference: </h>“ auftritt, dann lies das Datenfeld *„Landkarte“*
Es gibt natürlich mehrere Möglichkeiten, ein Feld zu identifizieren, und auch sog. Wildcards (_capitalized_, _number_ , _HTML-Tag_ ..) sind erlaubt:

R2 : SkipTo "Map Reference :" SkipTo _capitalized_

Hier würde der Wrapper dann wie oben nach „Map Reference“ suchen und dann jedoch noch bis zum ersten Großbuchstaben weitergehen.
Damit auch einfache Formatierungs-Unterschiede auf den Seiten, wie z.b. kursiv oder normal, die Extraktion nicht behindern, werden mehrere alternative, disjunkte Regeln aufgestellt,
z.B. für Regel 2, falls “Landkarte” auf einer Beispiel-Seite fett (entspricht HTML-TAG </b>) geschrieben ist.

R2: : SkipTo "Map Reference : " ; SkipTo </h></b>“

Für STALKER genügen meist schon 2 bis 10 Beispiel-Seiten, aus denen er dann mehrere Regeln erstellt, und diese dann auf die anderen Seiten anwendet. Um minimal viele perfekte Regeln zu erhalten, erstellt STALKER eine hierarchische Struktur des Dokumentes, aus dem er die Daten extrahieren soll. Eine perfekte Regel wurde von mindestens einem gegebener Beispiel-Seite abgeleitet und liefert auf jeder Seite auf der sie gilt, das richtige Ergebnis.
Dann wandelt er komplexere Regeln (wie z.B. eine Regel, die *Landkarte* und *Gesamte_Fläche* auf einmal markiert), in kleinere, einfachere Regeln um. STALKER erstellt aus den Beispielen eine Menge C an Regeln, auf die er dann durch die folgenden 3 Schritte optimiert:

- suche die am besten passende Regel
- redefiniere diese Regel
- füge das Ergebnis wieder in C ein.

Regeln, die auf die vielen Seiten nicht passen, werden aus C aussortiert, so daß am Ende nur noch perfekte Regeln vorhanden sind.

## 4.3.2 Auswählen von Beispiel-Seiten

- Co-Testing -

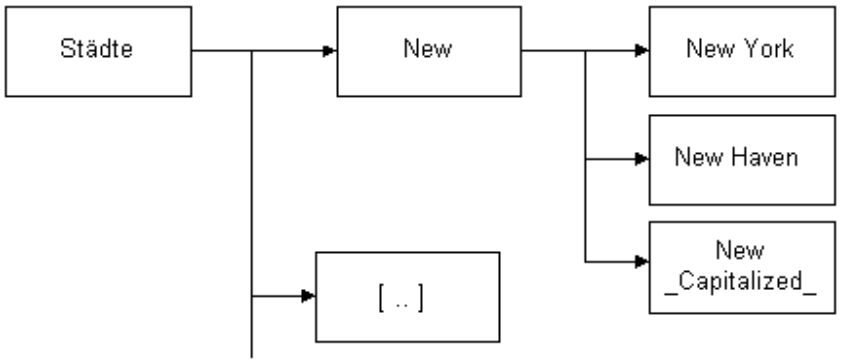
STALKER liefert bessere Ergebnisse, wenn die Beispiel-Seiten nicht zufällig ausgewählt wurden, sondern danach, ob sie sich zum Erlernen von Regeln besonders eignen. Da der Programmierer der Anwendung aber nicht alle Seiten nach diesen Kriterien durchsuchen kann, wurde der Co-Testing-Algorithmus entwickelt. Dieser durchsucht die noch nicht markierten Seiten nach geeigneten Beispiel-Seiten. Da eine Information auf verschiedenen Wegen lokalisiert werden kann, erstellt Co-Testing zu jeder Vorwärts- eine Rückwärts-Regel. Eine Rückwärts-Regel gibt die Position des Datenfeldes relativ zum Ende des

Dokumentes an, eine Vorwärts-Regel liefert Gegensatz dazu die relative Position vom Anfang des Dokumentes. Da Vorwärts- und Rückwärts-Regel absolut gleichwertig sind, müssen die Regeln auf allen Seiten das selbe Ergebnis liefern. Co-Testing wendet diese Regeln nun an und vergleicht die daraus resultierenden Ergebnisse. Sind nun die Ergebnisse unterschiedlich, so wird diese Seite dem Programmierer als weitere Beispiel-Seite vorgelegt.
Da die Regeln aus unterschiedlichen Beispiel-Seiten resultieren, ist es sehr unwahrscheinlich, daß verschiedene Regeln das selbe falsche Ergebnis liefern. Wenn nun eine Regel der anderen widerspricht, so muß eine der beiden Regeln falsch sein. Welche dies ist, so wird vom Programmierer durch das Markieren der Daten in der neuen Beispiel-Seite, in der sich die Regeln widersprechen, bestimmt.

## 4.4 Datenverifikation

- Data Pro -

Schließlich gibt es noch ein anderes Problem beim Extrahieren der Daten: die meisten Seiten werden sehr oft aktualisiert und bearbeitet. Dadurch können die vom Wrapper gelieferten Daten falsch sein, da alten die Markierungspunkte nicht mehr gelten. Deshalb wurde Data-Pro entwickelt.
DataPro versucht für jedes Daten-Feld aus den Beispielen ein Muster zu entwickeln.
In unserer Beispiel-Anwendung soll auch die Adresse des Regierungssitzes des jeweiligen Landes abgefragt werden. Nach dem wir die Datenfelder der Adresse markiert haben, erstellt Data Pro Muster für diese Felder.
Für die Beschreibung der Straße in einer Adresse z.B. gibt es mehrere Möglichkeiten : Birkenweg 10, Münsterplatz 10, Hauptstrasse 11. Alle diese Daten beginnen mit einem großen Buchstaben, enden mit einer Zahl und enthalten „weg“, „strasse“ oder „platz“.
Außerdem können auch bestimmte „negative“ Merkmale mit in das Muster einbezogen werden, z.B. kommt normalerweise kein „m²“ oder eine 5-Stellige Nummer im Straßen-Teil vor.
DataPro findet nun alle wichtigen[1] Merkmale zu den Datenfeldern aus den Beispielen. Um ein möglichst gutes Muster zu erzeugen, erstellt DataPro aus den erkannten Daten eine Baumstruktur. In diesem Baum steht an erster Stelle das allgemeinste Merkmal, danach kommen die spezielleren Merkmale.




Das Muster für das Datenfeld, das den jeweiligen Ort enthält könnte bei den Städte-Namen *New York* und *New Haven* z.B. wie nebenstehend als Baum dargestellt werden.
Nachdem der Baum erstellt wurde, wird ausgewertet, welcher der Einträge statistisch am häufigsten vorkommt. Wenn nun 10 mal *New York* und 10 mal *New Haven* in den Beispiel-Seiten vorkommt, so wird *New _capitalized_* nur dann als wichtigstes Merkmal gewertet, wenn noch mindestens 20 andere Städte mit *New + Großbuchstabe* verarbeitet worden sind. Auf diese Weise werden die erhaltenen Daten immer mit den Merkmalen verglichen und dann bewertet. Nun werden die von den Wrappern zurück gelieferten Daten durch diese Muster überprüft. Läßt sich das Ergebnis nicht mit den Merkmalen in Übereinstimmung bringen oder fällt die Bewertung unter einen bestimmten Wert, so wird die jeweilige Seite

1 Ein wichtiges Merkmal ist ein Merkmal, das öfters als erwartet in den (zufällig) ausgewählten Beispielen auftritt. Wie wichtig ein Merkmal ist, könnte z.B. durch stohastische Methoden (Erwartungswert, Varianz) festgelegt werden.

erneut dem Programmierer der Anwendung zum Markieren der Daten vorgelegt und es werden wieder neue Regeln erstellt.

### 4.5 Automatische Wrapper-Reparatur

Viele Aktualisierungen von Webpages bedeuten nur eine Veränderung des Inhaltes oder kleinere formale Korrekturen, meistens bleibt aber Das Layout der Seite erhalten. Um solche Aktualisierungen automatisch verarbeiten zu können, gibt es einen Algorithmus, der den Wrapper dann automatisch repariert bzw. aktualisiert. Nachdem das gesuchte Datenfeld lokalisiert wurde, werden auf den neuen (geänderten) Seiten wiederum Regeln definiert, die den Anfang- und das Ende der Datenfelder beschreiben. Diejenigen neuen Regeln, die dann längere oder kürzere Zeichenketten als die ursprünglichen Regeln liefern, werden aussortiert. Dann werden die übrigen Regeln nach gemeinsamen Merkmalen in Gruppen sortiert. Diese Gruppen werden dann wiederum nach Kriterien wie z.B. "relative Seitenposition", "für den Benutzer sichtbar" , etc sortiert. Mit dieser Ordnung und dem Übereinstimmungsgrad mit der Original-Regel, ergibt sich eine Bewertung für die jeweilige Gruppe. Die Gruppe mit der besten Bewertung sollte dann die jeweiliges richtigen Daten zu den Datenfeldern liefern.

## 5 Optimierungen

### 5.1 Query-Planung

- Planning by Rewriting -

Würden alle Anfragen direkt ohne Optimierung an die jeweiligen Seiten gehen, so würde eine Anfrage, teils wegen der jeweiligen Antwort-Zeiten und teils wegen der zurückgelieferten Daten-Mengen inakzeptabel lange dauern.
Würde wir z.b. diese Anfrage stellen : *„Welche Länder mit einer Bevölkerungsanzahl über 10 Millionen sind Mitglied in der NATO und wann sind sie eingetreten ?"*, so würden ohne Optimierung erst **alle** Bevölkerungszahlen **aller** Länder der Welt abgefragt, und dann erst die Mitglieder der NATO herausgesucht. Wenn nun aber zuerst die Mitglieder der NATO gesucht werden, und diese dann nach ihrer Bevölkerungsanzahl aussortiert werden, so müssen weniger Daten (= weniger Zeitaufwand) durchsucht werden. Deshalb wird jede Anfrage des Benutzers vorher optimiert.

Vor der Optimierung wandelt Ariadne das Datenbank-Modell, bzw. das Wrapper-Model (Abb. 2) in ein Modell aus Axiomen um. Dies geschieht nach diesen 5 Regeln :

| | |
|---|---|
| 1. Direktes übernehmen : | Invertiert die Datenbank-Beschreibung, d.h. Umwandeln der Objekte und ihrer Attribute in Wrapper |
| 2. Covering-Regel: | Extrahiert Klassen, Unterklassen und deren Beziehungen aus dem Datenbank - Modell |
| 3. Definitionen Übernehmen: | Übernimmt die Definitionen der einzelnen Klassen |
| 4. Vererbungs-Regel | Durchsucht die Klassen nach vererbten Attributen der Oberklassen (über Schlüssel-Wörter) |
| 5. Vereinigungs-Regel | Vereinigt einzelne Klassen um zusätzliche Attribute zu erhalten |

In unserer Beispiel-Anwendung könnten die Axiome so aussehen :

1. Schritt : Umschreiben der Objekt-Attribute auf Wrapper [Regel 1]:

| | | |
|---|---|---|
| 1.1 Land (Name,Größe, Bevölkerung, L-Grad, Br-Grad ..) | ⇔ | Land-Wrapper (Name, Größe ,Bev...) |
| 1.2 Staatsoberhaupt (Titel, Personen-Name) | ⇔ | Regierungs -Wrapper (Titel ...) |
| 1.3 NATO-Land ( Name, Beitritts-Jahr) | ⇔ | NATO-Wrapper ( Name, ..) |

2. Schritt : Vererbungs-Attribute suchen (Land = Oberklasse von NATO-Land) [Regel 4]

| | | |
|---|---|---|
| 2.1 NATO-Land( Beitritts-Jahr, Name, Größe,...) | ⇔ | NATO-Wrapper & Land - Wrapper |

3. Schritt: Vereinigung einzelner Klassen um möglichst viele Attribute zu erhalten [Regel 5]

| | | |
|---|---|---|
| 3.1 Land (Größe, Bevölkerung, Längengrad, .. , Oberhaupt) | ⇔ | Land-Wrapper & Regierungs-Wrapper |
| 3.2 NATO-Land (Größe, .. ,Oberhaupt,Beitritts-Jahr) | ⇔ | NATO - Wrapper & Land - Wrapper & Regierungs - Wrapper |

Nun wird jede Anfrage des Benutzers durch 2 Schritte optimiert: Zuerst wird die Eingabe übersetzt, vereinfacht und so umgeschrieben, daß nur noch Operatoren der Relationalen Algebra und Klassen asu dem Axiomen-Modell darin vorkommen. Dabei wird jeweils die speziellste Klasse, die alle nötigen Attribute enthält, aus dem Axiomen-Modell verwendet. Würde z.B. nach einem Land, das in der NATO Mitglied ist gefragt, so würde in der Anfrage nach dem Umschreiben „NATO-Land“ (2.1) verwendet werden.
In einem zweiten Schritt wird die Methode „Planning by Rewriting“ angewendet. In diesem Schritt wird nun zuerst ein vorläufiger Query-Plan erstellt. Dieser Plan entsteht direkt aus der Anfrage und ist noch nicht optimiert.
Der Plan wird dann unter Benutzung der Regeln der Relationen - Algebra so lange umgeformt, bis er optimal ist. Diese Regeln enthalten unter anderem mathematische Gesetze zu den Anfrage - Operatoren (Vereinigung , Durchschnitt, Differenz, Selektion ...) und die zugehörigen algebraischen Gesetze (z.B. Kommutativ- und Assoziativgesetz bzgl. der Vereinigung)
Um eine optimale Umformung zu erreichen, enthält der Query-Planer von Ariadne verschiedene Module :

- ein Modul für die Berechnung des Aufwandes des jeweiligen Query-Planes,
- ein Modul für die Umformung nach den Regeln der Relationen Algebra und
- ein Modul, das die Strategie zum Umschreiben enthält.

Durch diese Modul-Bauweise ist der Query-Planer sehr flexibel und kann durch das Austauschen einzelner Module sehr schnell angepaßt werden.
Wenn nun z.B. wiederum die Anfrage *„Welche Länder mit einer Bevölkerungsanzahl über 10 Millionen sind Mitglied in der NATO und wann sind sie eingetreten ?“* gestellt würde, so würde im ersten Schritt das Axiom 2.1 gewählt werden, da es alle benötigten Attribute (Name, Eintrittsjahr, Bevölkerung) enthält. Daraus würde dann der folgende vorläufiger Startplan entstehen:

1. Suche alle Namen und zugehörigen Bevölkerungszahlen aller Länder.
   Suche alle Namen und das Beitrittsjahr aller NATO-Länder
2. Vereinige diese über dem Attribut „Name“
3. Selektiere dann nach der Bevölkerungsgröße

Dieser Plan ist jedoch noch nicht optimiert, da es sehr viel mehr Länder als NATO-Länder gibt. Somit wäre es sinnvoller, zuerst alle Nato-Länder zu suchen, und dann nur diesen dann nach den restlichen Kriterien zu filtern. Zur Optimierung werden durch Regeln der Relationen - Algebra aus dem Startplan verschiedene Alternativ - Pläne generiert. Diese werden nach ihrem Aufwand, also Faktoren wie Datenmengen, Antwortzeiten etc. bewertet. Aus dem obigen Plan und den erstellten Alternativ - Plänen würde dann folgernder optimaler Plan ausgewählt:

1. Suche alle Namen und das Beitrittsjahr aller NATO-Länder
2. Suche dann zu diesen Ländern (und nur zu diesen) die Bevölkerungszahlen
3. Selektiere dann nach der Bevölkerungsgröße

## 5.2 Optimierung durch Cachen

- Cluster and Merge (CM) -

Um die Antwort-Zeit auf die Anfrage weiter zu optimieren, werden Teile der Daten lokal in einem Cache zwischengespeichert und als zusätzliche Datenquelle in Ariadne eingebunden. Die einfachste Methode wäre es, einfach alle Daten lokal zu halten. Dies ist aber allein schon deshalb unmöglich, weil dies sehr viel Platz beanspruchen würde. Außerdem wäre eine Konsistenz zwischen den lokalen und den Daten im Web nur mit großem Aufwand zu erhalten.
Um zu ermitteln, welche Daten am besten zwischengespeichert werden , wurde CM (Cluster and Merge) entwickelt.
Um die Laufzeit des Query-Planers möglichst gering zu halten, ist es wichtig, möglichst wenig zusätzliche Datenquellen einzubinden. Deshalb analysiert CM die Benutzeranfragen nach möglichst zusammenhängenden Themen-Gebieten, die dann als **eine** neue (lokale) Datenquelle eingebunden werden können.
Würden nun z.B. viele Anfragen nach europäischen Ländern und deren geographischen Daten gestellt werden, so würde CM aus den Anfragen eine neue Datenquelle “Europäische Länder” erstellen, in der dann alle geographischen Daten aller europäischen Ländern lokal abgelegt würden. Der Query-Planer würde dann bei erneuten Anfragen die lokale Datenquelle benutzen.
Indem alle (in der jeweiligen Anwendung) möglichen Anfragen nach ihrem Aufwand und unter Beachtung der jeweiligen Abfrage-Eigenschaften der Datenstrukturen analysiert werden, können zusätzliche, besonders zur Zwischenspeicherung geeignete, Daten ermittelt werden.
Die Anfrage nach allen europäischen Ländern würde den Land-Wrapper dazu veranlassen, alle Länder nach “Landkarte” = “Europe” zu durchsuchen, da die Index-Seite des CIA-Fact-Book keine direkte Abfrage nach dieser Eigenschaft enthält. Da dies ein großer Aufwand ist, wird zur Optimierung weiterer Anfragen, die Beziehung „Land - Kontinent“ in einer lokalen Datenquelle mit geeigneten Strukturen abgelegt.
Natürlich muß auch darauf geachtete werden, daß die lokalen Daten mit den Original-Daten aus dem Web konsistent sind. Deshalb muß festgelegt werden, wann und wie oft eine lokale Datenquelle erneut aktualisiert werden muß. Diese Überlegung wird dann ebenfalls mit einbezogen, wenn es um das Zwischenspeichern von Daten geht.
Das CIA-Fact-Book, z.B. wird nur jedes Jahr neu bearbeitet, deshalb können diese Daten ohne Probleme lokal abgelegt werden.

## 6. Die Zukunft von Anwendungen zur Integration von Web-Daten

Da es immer mehr Daten im Internet geben wird, wird es für den Benutzer immer unbefriedigender, einfache Suchmaschinen zu benutzen. Diese liefern zwar eine Unmenge Daten zurück, die jedoch nur das Suchwort enthalten und oft nicht zum gesuchten Thema passen. Deshalb werden Applikationen, die verschiedene Webseiten auf einmal in Zusammenhang bringen und abfragbar machen, immer wichtiger. Der große Vorteil dieser Applikationen liegt darin, daß sie nicht nur einfach irgendwelche Daten zurückliefen, sondern nur Daten, die auch zu den jeweiligen Themen der Applikation passen. Zwar gibt es mittlerweile auch XML, wodurch eine Abfrage der Web-Sites nach einzelnen Daten-Feldern möglich ist, jedoch liefern auch diese nur unbefriedigende bzw. zu wenig Ergebnisse.
Ein Grund hierfür ist, daß viele Daten-Felder unterschiedlich bezeichnet werden, z.b. das Feld „Name“ kann auch in „Vorname“ und „Nachname“ aufgespalten werden. Zusätzlich können die XML - Bezeichnungen mehrdeutig sein und so falsche Daten liefern. Dadurch kommt wiederum nur ein unscharfes Suchergebnis zustande. Deshalb wird es in Zukunft immer mehr Applikationen geben, die verschiedene andere Seiten einbinden um so zu einem Themengebiet eine bessere Informationsquelle bereit zu stellen.

## 7. Verwendete Quellen


- G. Barish, Craig A. Knoblock, Yi-Shin Chen, Steven Minton, Andrew Philot, Cysus Shahabi: **The TheatherLoc Virtual Application**

- Craig A. Knoblock, Kristian Lerman, Steven Minton, Ion Muslea : **Accurately and reliably extracting data from the web: A machine learning approach**

- Josè Luis Ambite, Craig A. Knoblock: **Agents for information gathering** (IEEE-Expert September/Oktober 1997)

- Craig A. Knoblock, Steven Minton: **Ariadne approach to web-based information integration** (IEEE-Expert September/Oktober 1998)

- Craig A. Knoblock, Steven Minton, Josè Luis Ambite, Neveen Ashish, Ion Musela, Andrew G. Philot, Sheila Tejada : **The ariadne approach to web-based information integration** (extended version of IEEE-Expert September/Oktober 1998)

- CIA-World Factbook : **http://www.cia.gov/cia/publications/factbook/index.html**

- Ariadne - Homepages und Beispiel-Anwendungen **http://www.isi.edu/info-agents/ariadne/index.html**