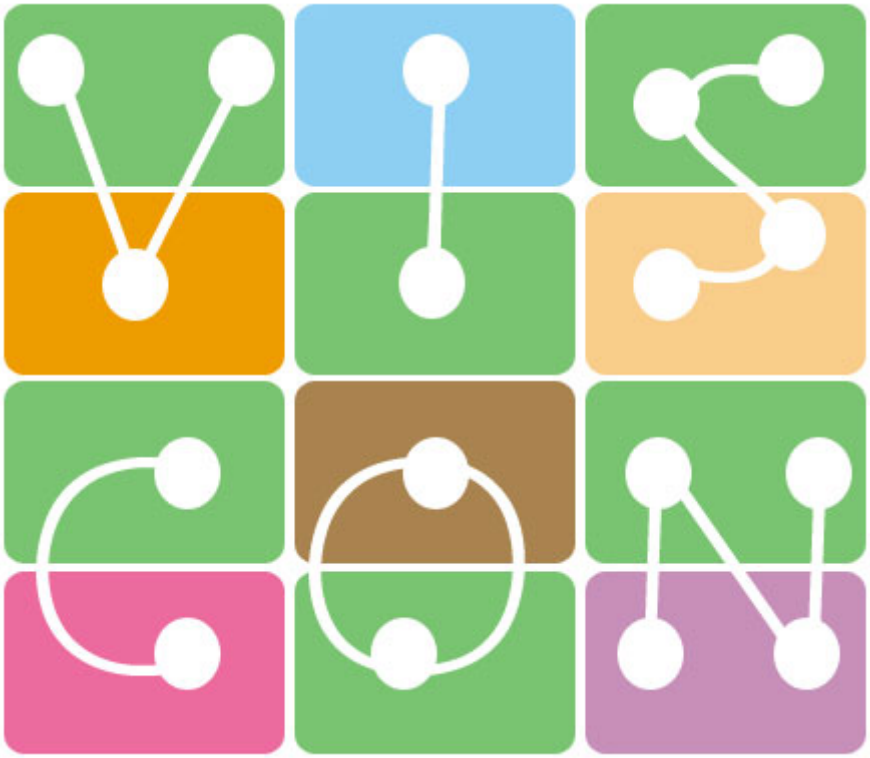
Entwicklerdokumentation

für das Fach
Multimediakonzeption

im Wintersemester 2006/2007



für die Anwendung
NepotistAnalyzer

Bezugsquelle:
http://sourceforge.net/projects/viscon/

Stand: 20.01.2007

Inhaltsverzeichnis

1. Einführung

Diese Einführung beschreibt den Umgang und die Verwendung der Anwendung NepotistAnalyzer. Die Dokumentation enthält Erläuterungen für die reine Benutzung aber auch wichtige Informationen für Entwickler, welche die Anwendung verstehen und verbessern möchten.

1.1 VISCON

Das Projekt Viscon beschäftigt sich mit der Visualisierung von Unternehmensstrukturen deutscher Unternehmen und den Beziehungen untereinander. Hierbei entstand die Anwendung NepotistAnalyzer. Dies ist ein Versuch die Beziehungen der Vorstände und Aufsichtsräte der führenden deutschen Unternehmen aufzuzeigen. Die Anwendung behält sich vor weder vollkommen korrekt noch auf dem aktuellsten Stand zu sein. Bei der Recherche wurde die Testdatenbank des Unternehmens Hoppenstedt verwendet. Diese stellt nach Anlegen eines Testzuganges dem Nutzer die Möglichkeit zur Verfügung einige Seiten im HTML Format auf dem eigenen Rechner zu speichern. Diese werden dann mit Hilfe eines in Java geschriebenen, speziell auf diese HTML Struktur angepassten Dokumentenparser bearbeitet. Der Parser analysiert das Dokument und extrahiert die wichtigen Informationen anhand der Formatierung und wandelt diese im Anschluss in eine gültige XML Datei um. Diese Datei wird benötigt um im weiterem die Anwendung einzusetzen. Mit Hilfe einer XSL/XSLT Datei wird aus der XML Datei nun eine SVG Dokument erzeugt. Die Erzeugen der SVG Datei übernimmt die OpenSource Anwendung SAXON. Wir verwendeten während unserer Arbeit die Version 6.5.5. Die so erhaltene SVG Datei kann dann in einem Browser angezeigt werden und bietet dem Nutzer interaktive Möglichkeiten der Platzierung einzelner Unternehmen. Die Interaktionen im Browser wurden mit Hilfe von JavaScript realisiert. Es ist zu beachten das einige Browser den Standard SVG noch nicht oder nur teilweise unterstützen.
Die Anwendung setzt sich also auf den folgenden drei Komponenten zusammen.

- Dokumentenparser
- XSL/XSLT Stylesheet
- SVG Dokument

1.2 Lizenz und Haftungsausschluss

Die Anwendung NeoptistAnalyzer wurde unter der General Public License (GPL) entwickelt. Nähere Informationen dazu unter www.opensource.org. Die Anwendung behält sich vor weder vollkommen korrekt noch auf dem aktuellsten Stand zu sein. Bei der Entwicklung der Daten wurden die Daten der Hoppenstedt Datenbank aus dem Jahre 2000 verwendet.

1.3 Weiterentwicklung

Die Anwendung NepotistAnalyzer bietet jedem Interessenten die Möglichkeit den Quellcode nach seinen eigenen Vorstellungen anzupassen und weiterzuentwickeln. Ebenso besteht die Möglichkeit einen Einblick in den Umgang mit den Auszeichnungssprachen XML, XSL, XSLT, SVG zu bekommen. Es besteht durch den allgemein geschrieben Dokumentenparser die Möglichkeit seine eigene XML Datei mit den Firmen seiner Wahl zu erstellen und anzeigen zu lassen. Hierbei kommt es nur darauf an für welche Firmenprofile man sich aus der Hoppenstedt Datenbank interessiert. Um einen benutzerfreundlicheren Umgang mit der Anwendung zu ermöglichen wären noch einige Weiterentwicklungen denkbar und sinnvoll. Automatisches einlesen und auswählen von Firmenprofilen, Auslagerung der Fehlerbehandlung, Extrahieren weiterer Informationen. Weiterentwicklung des XSL/XSLT Stylesheets hierbei insbesondere die Darstellung der Personen welche in mehreren Unternehmen tätig sind.

2. Voraussetzungen, Installation, Verwendung

2.1 Systemvoraussetzungen für die Entwicklung

Um erfolgreich die Anwendung weiterentwickeln zu können benötigt man mindestens folgende Werkzeuge.

- Java Development Kit 5.0 oder Java Standard Edition J2SE
- Adobe SVG Viewer Version 3.03
- XML Editor nach Wahl (z.B. Cooktop)
- SAXON XSLT Processor Version 6.5.5
- Entwicklungsumgebung nach Wahl (z.B. Eclipse)
- Internet Browser (Internet Explorer 7)

Bezugsquelle SAXON http://sourceforge.net/projects/saxon

2.2 Installation der Anwendung für Entwickler

1. Herunterladen des Package viscon Release developerpack-alpha
2. Entpacken der Datei Viscon.zip
3. Anpassen der Pfadangaben in der Javaklassen

Bemerkung:
Die stellen im Quellcode an denen Pfadangaben angepasst werden müssen sind durch Kommentare gekennzeichnet.

developerpack-alpha enthält alle Dateien die auf dem CVS von http://sourceforge.net/projects/viscon/ liegen. Falls Sie neue Firmen einbinden möchten können Sie sich die Daten bei der Hoppenstedt Datenbank einholen (Testzugang reicht aus!).
Ihre neuen Firmenprofile legen sie in der Dateistruktur des Projektes ab. Ebenfalls müssen Sie die neuen Firmenprofile der Javaklasse im Code bekannt machen. Nach dem Hinzufügen der Profile oder Änderungen im Code muss die Datei neu ausgeführt werden. Dabei wird die alte Datei Viscon.xml von der neuen überschreiben.
Eine genaue Beschreibung des Einfügens befindet sich als Kommentar im Quellcode der Javadateien.

Bei Veränderungen an der XSL Datei ist es unbedingt notwendig die SVG Datei mit Hilfe von SAXON neu erstellen zu lassen um damit die Änderungen wirksam werden zu lassen. Über den Umgang mit SAXON informieren Sie sich bitte auf der Projektseite.

2.3 Systemvoraussetzungen für die Nutzung

Um die Anwendung benutzen zu können müssen auf dem System mindestens folgende Anwendungen installiert haben.

- Internet Browser (Internet Explorer 7)
- Adobe SVG Viewer Version 3.03

Bemerkung:
Vergewissern Sie sich das sie JavaScript bei ihrem Browser aktiviert haben.

2.4 Installation der Anwendung für die Nutzung

1. Herunterladen der SVG Datei mit dem Namen Viscon.svg
2. Speichern der Datei an einen Ort Ihrer Wahl
3. Öffnen der Datei mit dem Internet Explorer 7

2.5 Bedienung

Die Bedienung der Anwendung ist denkbar einfach. Durch einmaliges Anklickens eines Unternehmensnamen auf der linken Seite erscheint das Unternehmen auf dem Bildschirm. Durch onMouseOver auf das Unternehmen erscheint ein Menü mit den Punkten „Aufsichtsrat“ und „Vorstand“, nach anklicken eines der Punkte erscheinen die Personen der ausgewählten Gesellschaft. Durch klicken und gedrückt halten der linken Maustaste lässt sich ein Unternehmen verschieben. In roter Schrift werden die Geschäftsführer angezeigt, die in mehreren Unternehmen tätig sind. Klickt man den Personennamen einfach an werden alle Unternehmen rot angezeigt in welche diese Person noch tätig ist.

2.6 Programmiersprachen

Der Dokumentenparser wurde in JAVA 5.0 geschrieben. Die Weiterverarbeitung der XML Datei erfolgt mit Hilfe der Auszeichnungssprachen XSL und XSLT. Weiterhin kommt JavaScript zum Einsatz um Interaktivität herzustellen.

2.7 Weitere Hinweise (SAXON, Browser)

Das Programm SAXON kann man unter dem Link http://sourceforge.net/projects/saxon beziehen. Auf dieser Seite befindet sich dafür eine ausführliche Anleitung zur Installation und Verwendung. Dabei muss zum Erstellen der SVG Datei mit Hilfe des SAXON Processor, die XSL und XML Datei im entsprechenden SAXON Ordner abgelegt sein. Durch ein DOS – Fenster kann mit dem Befehl java –jar saxon.jar –o AngabeSVGDatei.svg AngabeXMLDatei.xml AngabeXSLDatei.xsl entsprechend die SVG Datei erstellt werden; nähere Informationen befinden sich in der SAXON Dokumentation.

SVG ist ein Standard der noch nicht von allen Browsern vollständig unterstützt wird. Bei der Entwicklung wurde der Internet Explorer 7 verwendet. Falls man einen anderen Browser zur Anzeige verwenden möchte lesen Sie bitte in der Dokumentation des jeweiligen Browser nach ob und in wie weit SVG bzw. XSL, XSLT unterstützt wird.

3. Softwarekomponenten

3.1 Dokumentenparser

Der folgende Text ist keine genaue Beschreibung der erstellten Klassen und Methoden. Es ist nur eine kleine Hilfe um ein besseres Verständnis für den Erstellungsvorgang der benötigten XML Datei zu bekommen. Hier wird die Grundlage für die Anwendung erstellt.

Eine genaue Beschreibung der Methoden befindet sich in der JavaDoc!

In der Klasse ExtractFileToXml werden die einzelnen HTML Firmendateien angelegt. Diese Firmendateien werden im Anschluss nach bestimmten Tags durchsucht. Es wird nach den Aufsichtsräten, Vorständen, Firmennamen und Adressen gesucht. Sind diese Tags in einer Datei vorhanden wird der Inhalt dieses Tags ausgelesen und als String oder String Array abgelegt. Es erfolgt eine weitere Zerlegung der Daten in einzelne Attribute wie z.B. Nachname, Vorname, PLZ usw. Die Inhalte der einzelnen HTML Firmendateien werden dann in zwei verschiedenen Listen abgelegt. Es werden die Personen in eine Personenliste abgelegt. Die Firmeninformationen werden in einer Firmenliste gespeichert.

Die Methode Check überprüft zusätzlich ob eine Person schon einmal in einer anderen Firmendatei gefunden worden ist. Falls dies der Fall ist wird die Firma der Person zugeordnet. Falls die Person noch nicht in der Personenliste enthalten ist wird ein neuer Eintrag erzeugt.

Sind alle Firmendateien extrahiert worden, werden die Listen weiterverwendet. Aus den Listen wird nun eine XML Datei erstellt. Diese Datei wird dann auf der Festplatte gespeichert.

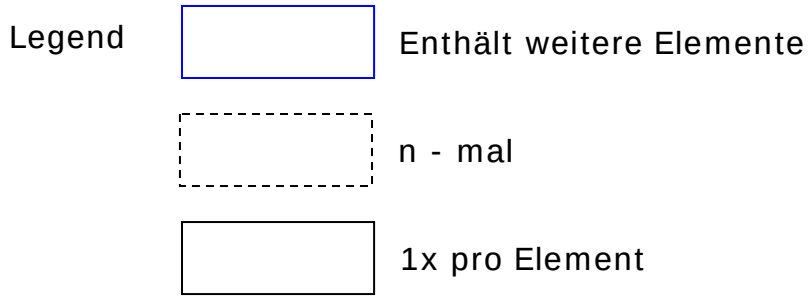
3.2 XML Familie

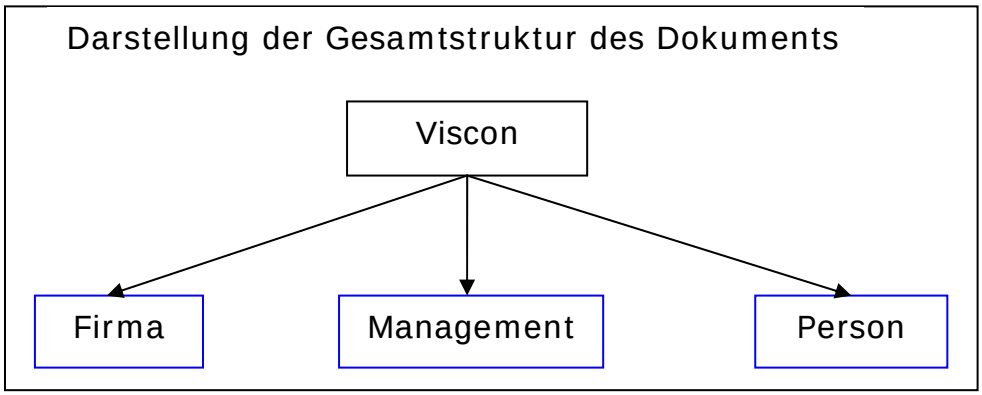
3.2.1 XML

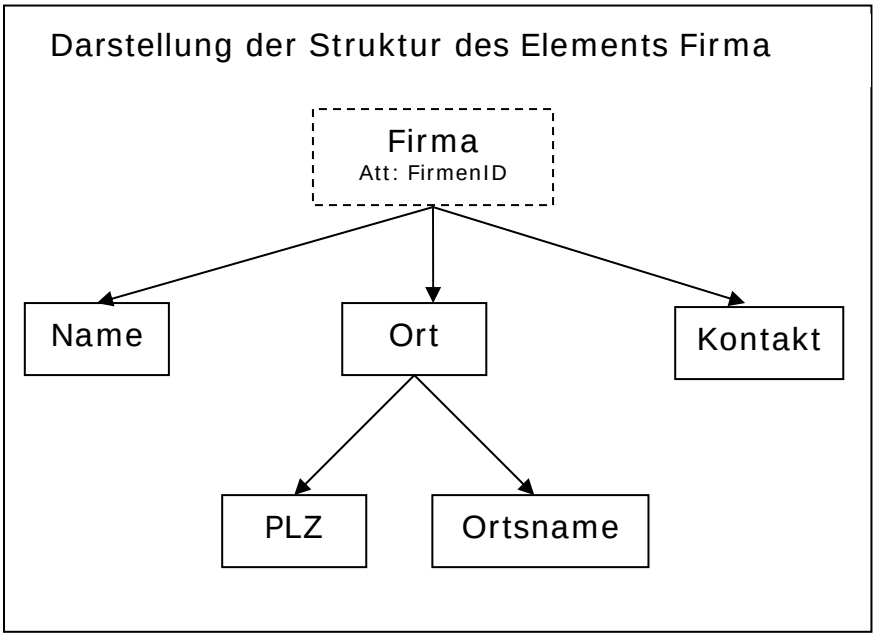
XML ist eine seit 1998 vom W3C empfohlener Standard. XML spielt eine Zentrale Rolle in der Web Entwicklung, Content & Dokumentenmanagement, Programmierung und Datenbanken. Es ist eine formale Beschreibung von jeglicher Art von sog. Semi-Strukturierten Daten. Durch die Verwendung von XML wird versucht die Daten und die Darstellung von einander zu trennen. XML eignet sich besonders gut dafür Daten und Dokumente zwischen verschiedenen Anwendungen auszutauschen. Aber auch hilft XML dabei Dokumente auf verschiedenen Medien wie Bildschirm, Drucker, Handy angemessen angepasst darzustellen. XML ist eine Auszeichnungssprache.

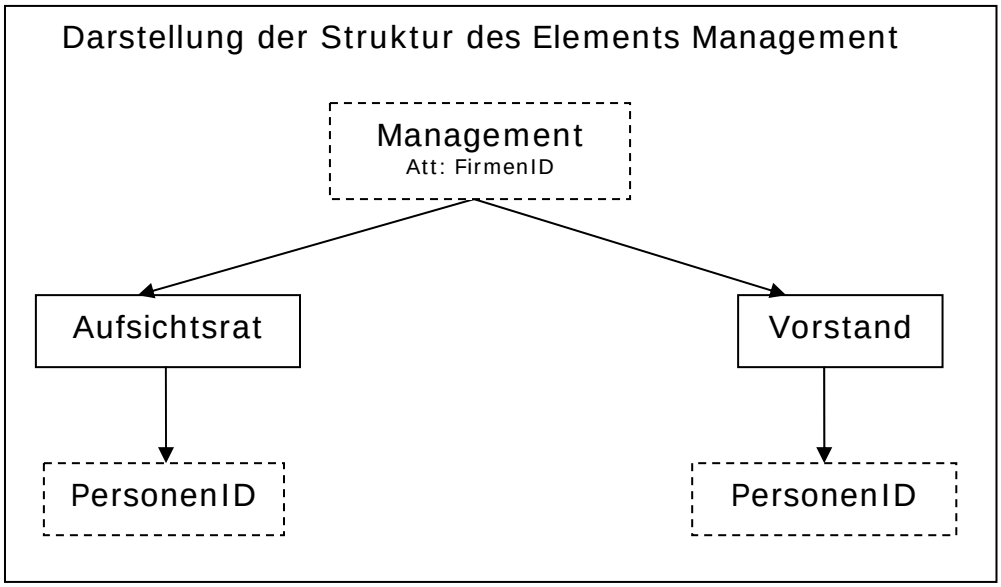
3.2.1.1 Die Datei Viscon.xml

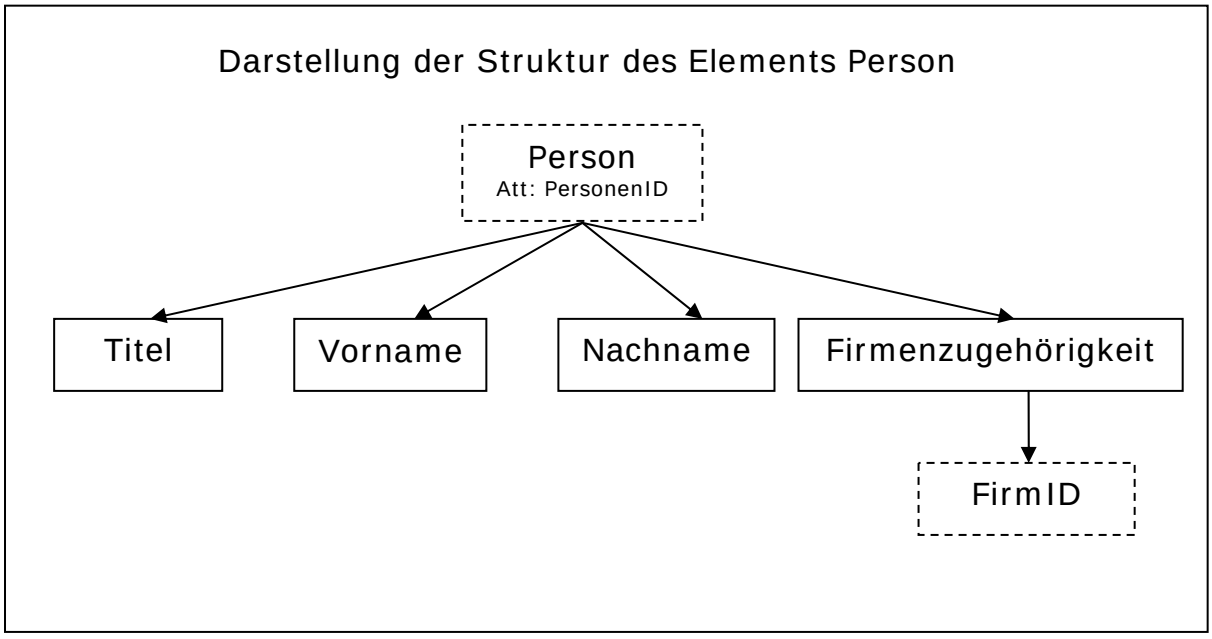
Die Datei Viscon.xml ist für die Ablage und Strukturierung der Daten über die Unternehmen zuständig. Hier werden die Daten und die Darstellung voneinander getrennt. Es wird versucht durch die verwendeten Tags ein für unsere Zwecke optimale Struktur der Informationen zu bekommen. Zum besseren Verständnis werden hier in der Dokumentation alle XML betreffenden Angaben in „rot“ geschrieben.
Die Datei beginnt wie für XML üblich mit der <?xml version="1.0" encoding="utf-8"?> Angabe. Danach folgt auch schon das Wurzelelement <Viscon>. Danach folgen die Kindknoten <Firma> diese haben ebenfalls weitere Kindknoten <Name>, <Ort>, <Kontakt>. Der <Ort> hat ebenfalls noch zwei weitere Kindknoten <PLZ>, <Ortname>.
In dem Knoten <Firma> werden also alle Informationen zu einem Unternehmen abgelegt. Zur besseren Verarbeitung wird dem Tag Firma noch das Attribut FirmenID definiert.
Im Anschluss an die Firmen folgen die Kindknoten <Management> mit ihren Kindknoten <Aufsichtrat> und <Vorstand>. Diese beiden Knoten besitzen ebenfalls Kindknoten <PersonenID>. Das Management Tag besitzt das Attribut FirmenID dadurch kann das Management einer Firma genau zugeordnet werden. Nach den Management Informationen folgen im weiteren die Kindknoten der einzelnen Personen. Kindkonten <Person> besitzt das Attribut PersonenID um es einzigartig zu machen, weiterhin enthält es weitere Knoten <Titel>, <Vorname>, <Nachname>, <Firmenzugehörigkeit>. Der Firmenzugehörigkeitsknoten besitzt das Element <FirmenID> in diesen werden die Informationen gespeichert in welchem Unternehmen die Person alles beschäftigt ist.
Im folgendem wird versucht die Struktur graphisch zu veranschaulichen.


Legend
Enthält weitere Elemente
n - mal
1x pro Element


Darstellung der Gesamtstruktur des Dokuments
Viscon
Firma
Management
Person


Darstellung der Struktur des Elements Firma
Firma
Att: FirmenID
Name
Ort
Kontakt
PLZ
Ortsname


Darstellung der Struktur des Elements Management
Management
Att: FirmenID
Aufsichtsrat
Vorstand
PersonenID
PersonenID


Darstellung der Struktur des Elements Person
Person
Att: PersonenID
Titel
Vorname
Nachname
Firmenzugehörigkeit
FirmID

3.2.2 XSL Style Sheet

Für die bessere Verwaltung von großen Datenmengen ist die Transformationssprache XSL spezifiziert worden. Dabei erfüllt diese Sprache komplexere und strukturellere Formatierungsanforderungen an XML Dokumenten. XSLT (Extensible Stylesheet Language Transformation) ermöglicht, XML Dateien durch ein XSL Style Sheet in ein anderes Dokument zu konvertieren. Dabei benutzt XSLT XPath um auf bestimmte Teile eines XML Dokuments zuzugreifen. XPath ist eine XML - Applikation, in der ein XML Dokument als Baumstruktur abgebildet wird. Durch eine Notation, die an die Baumstruktur von Verzeichnisbäumen angelehnt ist, können einzelne Elemente oder ganze Unterbäume ausgewählt werden. XSLT und XPath ermöglichen zusammen ein XML Dokument nach bestimmten Parametern des XSL Style Sheets zu durchsuchen und dann diese Datenmenge in die gewünschte Form zu konvertieren und abzuspeichern; im Falle unseres Projektes in eine SVG Datei. XSLT ist seit November 1999 ein empfohlener Standard der W3C.

3.2.3 SVG – Scalable Vector Graphics

SVG ist eine Sprache, die benutzt wird um zweidimensionale Grafiken in XML zu beschreiben. Dabei können drei verschiedene Grafik Objekte, entweder eine Vektor Grafik Form, ein Bild oder ein Text beschrieben werden. Diese Objekte können dann gruppiert werden, bestimmten Stilen zugeschrieben werden, transformiert werden und in bestimmte Objekte zusammengefasst werden. Dabei können z.B. auch bestimmte Filter Effekte angewendet werden.
Mit SVG können auch Interaktionen und dynamische Bewegungen mit den Zeichnungen gemacht werden. Dabei sind Animationen möglich, die definiert und zu bestimmten Zeitpunkten ausgelöst werden können. Auch wird Skripting ermöglicht, indem verschiedene Skriptsprachen implementiert werden können, die dann mit dem SVG DOM (Document Object Model) bearbeitet werden können. DOM ermöglicht auf alle Elemente einer SVG Datei zuzugreifen und dessen Attribute oder Eigenschaften zu verändern oder auszulesen. SVG Objekte können auch mit Event Handling wie z.B. onMouseOver und anderen Events bearbeitet werden.

3.2.4 Datei Viscon.xsl

3.2.4.1 XSL und SVG

Die Datei Viscon.xsl ist für die Darstellung der Daten verantwortlich. Dabei wird die Transformationssprache XSLT, auf die in 3.2.1 eingegangen ist,

verwendet. Für die Formatierung der XML Daten, wurden SVG Elemente angelegt und mit JavaScript Funktionen geschrieben. Dabei werden für besseres Verständnis alle XML Element in der Farbe „rot“, alle XSL Elmente in der Farbe „blaugrün“ und die SVG Elemente in der Farbe „pflaume“ angezeigt.

Jede Datei die der XML Familie angehört beginnt mit einer XML Deklaration, in unserem Fall wie folgt:

```
<?xml version="1.0" encoding="utf-8"?>
```

Mit dem Root Element <xsl:stylesheet> wird die XSL Datei eingeleitet. Um auf alle Attribute, Elemente und Eigenschaften der XSLT Sprache zugreifen zu können, muss der korrekte Namensraum angegeben werden. Dabei wird nach dem empfohlenen Standard der W3C ein XSLT immer wie folgt eingeleitet:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

Dabei gibt xmlns:xsl="http://www.w3.org/1999/XSL/Transform" den Namensraum an.
Mit dem Element <xsl:template> wird ein Template deklariert und mit dem **match** Attribut wird eine Verbindung zwischen dem Template und einem bestimmten Element aus dem XML Dokument hergestellt. In unserem Fall soll auf das ganze XML Dokument verwiesen werden und deswegen wird der Wert „/“ an das **match** Attribut übergeben.
Mit dem <svg> Element wird ein Scalable Vector Graphic eingeleitet. Dem Element wurden in unserem Style Sheet bestimmte Werte, wie Größe der Ausgabe Datei übergeben. Hierbei müssen auch wieder Namensräume angegeben werden. Mit **onload="addEvent()"** wird beim Laden der Seite die Funktion „addEvent()“ aufgerufen, die verschiedene Events für den jeweiligen benutzten Browser hinzufügt. Auf die jeweiligen JavaScript Funktionen wird unter ***3.2.4 JavaScript Funktionen*** eingegangen.
Das XSL <xsl:for-each> Element wird benutzt, um alle XML Elemente eines bestimmten Knotens auszuwählen. Dabei kann mit dem **select** Attribut der gewünschte Knoten-Pfad angegeben werden, wie in unserem Fall:

```
<xsl:for-each select="Viscon/Firma">
    <xsl:sort select="Name" />
```

Dabei gibt das XSL <xsl:sort> Element einfach nur an nach was alphabetisch angeordnet werden soll. Der Inhalt des **select** Elements ist ein XPath Ausdruck. Ein XPath Ausdruck funktioniert wie das Navigieren in einem Dateisystem. Ein Schrägstrich „/“ leitet ein Unterverzeichnis ein.
Mit dem XSL <xsl:variable> Element kann eine Variable angelegt werden, in unserem Fall werden zwei Variablen deklariert, in dem der Name der Firma und in der anderen die ID der Firma gespeichert wird. Diese Variablen werden später in der SVG Datei noch gebraucht.

Die folgende Variablen sind nur x und y Positionen die berechnet werden, um die Position der einzelnen Firmennamen anzugeben.

```
<xsl:variable name ="yr" select="(position() * 10)"/>
<xsl:variable name ="ya" select="(position() *50)"/>
<xsl:variable name ="xa" select="(position() *50)"/>
```

Mit dem SVG <a xlink:href="" target="_top"> Element wird ein Link, wobei in SVG von XLinks gesprochen wird, eingeleitet und mit geschlossen. Dabei muss im einleitenden SVG Element am Anfang der Datei der Namensraum **„xmlns:xlink=http://www.w3.org/1999/xlink“** angegeben werden. In unserem Fall wird nur ein leerer XLink deklariert, nur um das SVG Text Objekt als Link für den Nutzer erkenntlich zu machen. Mit dem SVG <text> Element wird ein Text Objekt eingeleitet. Hierbei werden verschiedene Attribute und Eigenschaften für das Element deklariert. Zuerst kann mit den x und y Attributen die x – Koordinate und y – Koordinate angegeben werden. Mit {$yr} wird der Wert aus der vorher deklarierten Variable ausgelesen. Das Attribut fill wird für ein späteres Verändern der Farbe gebraucht. Mit den font-family und font-size Attributen wird der Schriftstil angegeben. Der onclick ist ein Mouse Event, das mit der JavaScript Funktionsaufruf verbunden wird. Mit visibility kann ein Objekt auf sichtbar (visible) oder unsichtbar (hidden) gesetzt werden.

```
<text x="0" y="{$yr}" fill="" font-family="Verdana" id="{$fName}" font-size="10"
onclick="handleClick(evt)" visibility="visible">
```

Natürlich besitzt das <text> Element noch andere Attribute, diese können alle auf der W3C Seite nachgelesen werden. Das XSL <xsl:value-of> Element gibt den Wert eines bestimmten Knotenpunktes aus. Somit kann ein XML Element ausgewählt werden und der Knoteninhalt für die Ausgabe in der SVG Datei benutzt werden.
Mit dem SVG <g> Element können verschiedene Objekt gruppiert werden. Der Vorteil dieses Container Elements ist, das alle Elemente innerhalb einer Gruppe, nur durch ein Gruppenmitglied angesprochen werden können. Wird ein Attribut in dem <g> Element deklariert, besitzt jeder in der Gruppe das gleiche Attribut.

```
<g id="g_{$fName}" transform="translate({$xa},{$ya})" visibility="hidden"
onmousedown="ClickObj(evt)" onmousemove="MoveObj(evt)"
onmouseup="OutOfObj(evt)" onmouseout="OutOfObj(evt)">
```

In unserem Fall konnten für verschiedene Elemente gleiche onMouse Events ausgelöst werden. Durch das **id** Attribut wird das eindeutige Adressieren eines Elementes ermöglicht.
Mit dem SVG <image> Element kann eine Datei in das Koordinatensystem erfasst werden oder ein Bild eingeleitet werden. In unserem Fall wurde ein GIF als Standbild deklariert. Möglich wäre es nur durch Angabe eines Pfades innerhalb des Koordinatensystems ein Bild malen zu lassen.

```
<image id="{$fName}" height="82" width="135"  xlink:href="fabrik_1.gif"  />
```

Auch ist es mögliche innerhalb einer SVG Datei ein weiteres SVG Element einzuleiten. Somit wurde für das Menü, das beim onMouseOver über das Bild eines Unternehmens erscheint, ermöglicht. Innerhalb dieser SVG wird auch eine Animation eingeleitet, die das Erscheinen und Verschwinden des Menüs ermöglicht.

```
<svg id="menu" width="200" height="200" visibility="hidden">
<animate begin="g_{$fName}.mouseover" dur="0.1s" attributeName="visibility"
from="hidden" to="visible" fill="freeze"/>
<animate begin="g_{$fName}.mouseout" dur="0.1s" attributeName="visibility"
from="visible" to="hidden" fill="freeze"/>
<a xlink:href="" target="_top">
<rect x="90" y="30" width="70" height="30" style="fill:#88c99d;stroke:white;stroke-
width:1"/>
<text font-family="Verdana" font-size="10" style="fill:white">
<tspan x="95" y="43" id="{$fName}" onclick="ShowRat(evt)">
Aufsichtsrat
</tspan>
<tspan x="95" y="53" id="{$fName}" onclick="ShowVor(evt)">
Vorstand
</tspan>
</text>
</a>
</svg>
```

Für die verschiedenen Aufsichtsräte und Vorstände wurden je nach Firmenzugehörigkeit, alle Personen in Gruppen gegliedert. Zuerst wird die Gruppe der Aufsichtsräte eingeleitet dann mit dem XSL <xsl:for-each> Element die entsprechende Firmenzugehörigkeit ermittelt. Dabei musste noch abgefangen werden ob eine Person in verschiedenen Firmen zuständig ist. Dies wurde in unserem Fall durch die Zähl – Funktion **count()** ermöglicht.

```
<xsl:variable name="countFirmID"
select="count(Firmenzugehoerigkeit/FirmID)"></xsl:variable>
```

In der **countFirmID** Variable wird je nach dem wie viele Firmen bei der Person stehen die Anzahl gespeichert und dann durch das XSL <xsl:choose> Element abgefangen ob mehr als eine Firma vorhanden ist. Falls dies der Fall ist, wird die Person in dem Browser dann rot angezeigt. Die Personen mit allen Firmenzugehörigkeiten werden dann in einer weiteren Gruppe abgelegt, um dann später eventuelle Verbindungen zu verwirklichen. Das gleiche wird mit den Vorständen gemacht.

3.2.4.2 JavaScript Funktionen

Um Manipulationen innerhalb der SVG Datei zu ermöglichen wurden verschiede JavaScript Funktionen geschrieben. Hier wird nur kurz auf die verschiedenen Funktionen eingegangen, dabei kann man am Besten durch

eigene Recherche, die einzelnen Funktionen verstehen. Alle Funktionen wurden innerhalb der Viscon.xsl gut dokumentiert.

Abbildung 1

Die Funktion addEvent() fügt alle gebrauchten Events beim Laden der SVG Datei zum jeweiligen genutzten Browser hinzu.
Die Funktion MoveObject() ermöglicht die Bewegung der Firmenbilder (siehe dazu *Abbildung 1*) innerhalb des Browsers. Dabei muss zuerst die Funktion ClickObject(evt) aufgerufen worden sein, die beim Mausklick auf das entsprechende Gruppen Element den Event auslöst. Innerhalb der ClickObject(evt) Funktion wird aktiv auf true gesetzt und dann innerhalb der MoveObject() Funktion kann durch das Abfangen der x – und y-Koordinaten mit .clientX() und .clientY() ein Bewegen mit dem Mauszeiger ermöglicht werden. Nach einem weiteren Mausklick setzt die Funktion OutOfObject() aktiv wieder auf false und die Bewegungen mit den Firmenbildern enden.
Die Funktion ShowOtherFirms(evt) wird aufgerufen wenn eine Person anklickt wird, die in verschiedenen Firmen tätig ist. Dabei werden alle Firmen Namen in der die Person noch zusätzlich ist rot angezeigt.
Die Funktionen ShowRat(evt) und ShowVor(evt) arbeiten fasst gleich, nur das die erste den Aufsichtsrat der Firma anzeigt und die zweite den Vorstand anzeigt. Dabei werden diese Funktionen innerhalb eines Menüs, das beim onMouseOver über einer der bestimmten Firmenbilder erscheint,

durch anklicken einer der Menüpunkte Aufsichtsrat oder Vorstand aufgerufen. Die Personen werden durch eine Kreisberechnung, kreisförmig um die entsprechenden Firmenbilder angezeigt. Durch einen weiteren Klick auf einer der Menüpunkte werden die Personen wieder unsichtbar gesetzt.
Die Funktion handleClick(click_evt) wird beim anklicken einer der Firmen Namen aufgerufen und setzt bis zu fünf Firmenbilder auf die Anzeigefläche.

4. Quellen

W3C The World Wide Web Consortium
http://www.w3.org/

W3 Schools
http://www.w3schools.com

Selfhtml
http://de.selfhtml.org/

SVG Tutorial
http://svg.tutorial.aptico.de/index.php

SVG – Learning By Coding
http://svglbc.datenverdrahten.de/