Das Link-System: Von Weblinks zu universellen, sensitiven Hyperlinks

Dirk Fischer

Rohde & Schwarz GmbH & Co. KG,
Bereich Standardisierung und Entwicklungssupport
Mühldorfstraße 15, 81671 München
Dirk.Fischer@rohde-schwarz.com

Abstract: Der Artikel beschreibt die Lösungsidee eines universellen Hyperlinkings zwischen beliebigen Applikationen. Dies ermöglicht eine internet-artige Navigation zwischen Applikations-Inhalten. Damit wird eine für den Benutzer intuitive, transparente und vor allem effektive Integration dieser Applikationen geschaffen, die nur wenig invasiv ist und ein hohes Maß an Wertschöpfungspotenzialen schafft, wie in einer ersten Evaluierung in der Firma Rohde & Schwarz GmbH & Co. KG aufgezeigt wurde.

Zur Universalität dieser Hyperlinks trägt das im zweiten Teil des Artikels beschriebene Konzept der sensitiven Hyperlinks bei. Diese erlauben die punktgenaue Lokalisierung von Informationen innerhalb von Texten auch in den Fällen, in denen die zugehörigen Textpassagen nicht explizit durch ihren Autor als Link-Ziele markiert wurden. Bei der Navigation wird berücksichtigt, dass die Textpassagen ggf. verändert wurden, seitdem sie sensitiv verlinkt worden waren.

1 Die Ausgangssituation

Typische Fragestellungen bei der Erarbeitung einer Information sind: „Wie kommt dieses Ergebnis zustande, wie ist es zu erklären?“, „Wurden alle Hintergrund-Informationen berücksichtigt?“ oder „Könnte eine andere Information auf gleiche Weise begründet und hergeleitet werden?“. Ebenso stellt sich oft die Frage, was mit einer Referenz gemeint sein könnte. Beispiele dazu zeigt das Literaturverzeichnis dieses Artikels.

Heutzutage werden die meisten Informationen in IT Systemen erarbeitet und veröffentlicht. Trotzdem ist es Autoren oft nicht möglich, ihren Informationen Hintergründe in einer Form beizufügen, die es Lesern mit den für sie geringst möglichen Kosten erlaubt, auf diese Hintergründe zuzugreifen. Selbst wenn ein IT System für die Bereitstellung von Hintergrund-Informationen das Konstrukt von Hyperlinks erlaubt, werden diese oft wegen ihrer für Menschen unbrauchbaren Form so erheblich zu Störinformationen in der eigentlichen Information, dass ein Autor auf deren Einfügen verzichtet. Dies widerspricht seiner oft jahrelang geübten Praxis des Einfügens von Literaturverweisen, wie in jedem wissenschaftlichen Artikel, sogar in diesem, verlangt.

1.1 Informationsbeschaffungskosten

Die Kosten für die Beschaffung einer Information können wie folgt unterteilt werden:

- **Identifikationskosten**: Kosten für die Feststellung, dass Informationen benötigt werden und welche dies sind.

  Beispiel: Wäre der Artikel X bekannt gewesen, dann wäre der lange verfolgte Software-Design-Ansatz niemals in Erwägung gezogen worden.
- **Zuordnungskosten**: Kosten für die Identifikation möglicher Informationsquellen.

  ... Auch wenn bekannt ist, dass zum Thema relevante Beiträge existieren, muss festgestellt werden, welche (hier: Artikel X) und wo sie verfügbar gemacht wurden.
- **Lokalisierungskosten**: Kosten für die Lokalisierung möglicher Informationsquellen.

  ... Mit dem Wissen, dass Artikel X in der Zeitschrift Y von 1972 veröffentlicht wurde, muss festgestellt werden, wo auf die Zeitschrift Y zugegriffen werden kann.
- **Verfügungskosten**: Kosten für die Beschaffung von und den Zugang zu Informationsquellen.

  ... Dieser Ort muss ggf. aufgesucht werden, es müssen Lizenzen bezahlt und/oder auf eine Ausleihe gewartet werden.
- **Selektionskosten**: Kosten für die Lokalisierung der benötigten Informationen in einer Informationsquelle.

  ... Der Artikel X ist sehr lang. Explizit zum Thema wird etwas nur kurz in der Mitte erwähnt (was von außen nicht zu erkennen war). Eine verständlichere Detaillierung wurde in einem anderen, bisher nicht in Betracht gezogenen Artikel vorgenommen.
- **Interpretationskosten**: Kosten für die Verarbeitung der erhaltenen Information.

  ... Die benötigten Informationen sind zusammengetragen. Sie müssen verstanden und korrekt in den Kontext der eigenen Information übertragen werden.

1.2 Desinformationskosten

Die Kosten für die Nichtbeschaffung von Informationen entstehen im Wesentlichen durch die Unterschiede von vermuteten und tatsächlichen Informationen. Die Folgekosten solcher Fehlinformationen können beliebige Ausmaße annehmen. So fokussiert z.B. die RTCA DO-178B als Richtlinie zur Zulassung von Softwareteilen in Luftfahrzeugen auf eine lückenlose Rückverfolgbarkeit von Entscheidungen (Traceability), insbesondere dann, wenn Menschenleben durch Fehler gefährdet werden.

2 Der Lösungsansatz

Das **Link-System** soll einen **universellen Hyperlink-Dialog** zur Verfügung stellen, der das Navigieren zwischen beliebigen Informationsquellen erlaubt. In diesem (nicht modalen) Dialog soll ein Autor beliebige Ziele in beliebigen Datensätzen von Applikationen auswählen können. Diese sollen in seine Information in verständlicher und lesbarer Form eingetragen werden, sodass sie einem Leser dieser Information einen Hinweis darauf geben, welche Hintergrundinformation er erwarten sollte.

Wird ein solcher Hyperlink ausgewählt, soll er punktgenau an die vom Autor spezifizierte Stelle innerhalb der Information navigieren. Selbst dann noch möglichst präzise, wenn diese Information zwischenzeitlich verändert wurde oder die Stelle nicht gesondert markiert ist (z.B. als Marke in Microsoft Word).

2.1 Einschätzung der Lösung in einem Rohde & Schwarz Entwickler-Workshop

Als die wesentlichen Risiken eines solchen Systems wurden benannt: 1. der Projekt-Umfang; 2. der Verlust der Akzeptanz durch häufige ‚Broken Links' (z.B. wegen sich ändernder referenzierter Inhalte); 3. das Finden des richtigen Maßes für eine optimale Verlinkungsdichte oder –Häufigkeit in Projekten. Letzteres kann nach Einschätzung der Mitglieder nur durch Erfahrung mit dem System erlernt werden.

Wesentliche Nutzen waren die Einschätzung der Verringerung der Entwicklungsaufwendungen um ca. 7%, sowie die Verminderung des Arbeitsstresses durch bessere Entscheidungsgrundlagen. Es sollte eine funktionale Überfrachtung vermieden werden, die Links sollten lesbar sein, um nicht zu Störinformationen zu werden, und das System sollte einfach, ohne Handlungszwänge zu bedienen sein. Zuletzt sollte dieses System nicht als Wissensmanagement-System interpretiert und in dieser Hinsicht ausgearbeitet werden. Dies meint, eine Datenbank als Container aller Informationen, wie z.B. in einem Wiki sollte vermieden werden, um zu starke Eingriffe in die Applikationen zu vermeiden und im Arbeitskontext durch die Applikationen selber alle Informationen erreichbar und modifizierbar zu haben.

Aus diesem Workshop ergab sich für Rohde & Schwarz, dass mit der Integration von (nur) 17 Applikationen mehr als 65% aller Entwicklungs-Aufgaben durch dieses System unterstützt werden.

2.2 Architektur der Lösung in Microsoft Windows Betriebssystemen

Die Applikation, in deren Daten der Link eingetragen werden soll, öffnet einen **Hyperlink-Dialog**, der optisch dem von Microsoft Office ähnlich ist. Im Unterschied zu diesem Link-Dialog können sich hier Adapter als Datenquellen anmelden und über den Button „Textmarken..." die Link-Ziele ihrer Informationsquellen zum Browsen anbieten. Dies geschieht über einen **Adapter** für jede Applikation, der deren Daten standardisiert im System repräsentiert. Der Adapter wandelt oder erzeugt, falls nötig, einen für den Leser verständlichen Link (~> Abbildung 1).

Wird der Link ausgelöst, so wird ein **Moniker** [MSDN08] des Link-Systems aufgerufen. Dieser übernimmt entweder mit Hilfe des spezialisierten Adpaters die Navigation der Applikation (falls diese keine Hyperlinks unterstützt), oder bestimmt die von ihr benötigte URL und leitet diese an das Standard Link Management im Betriebsystem weiter. Unterstützt die Applikation keine Navigation innerhalb der Informationen, kann der Moniker nach dem Start der Applikation durch das Standard Link Management zusätzlich mit Hilfe des Adpaters in der Applikation punktgenau navigieren.

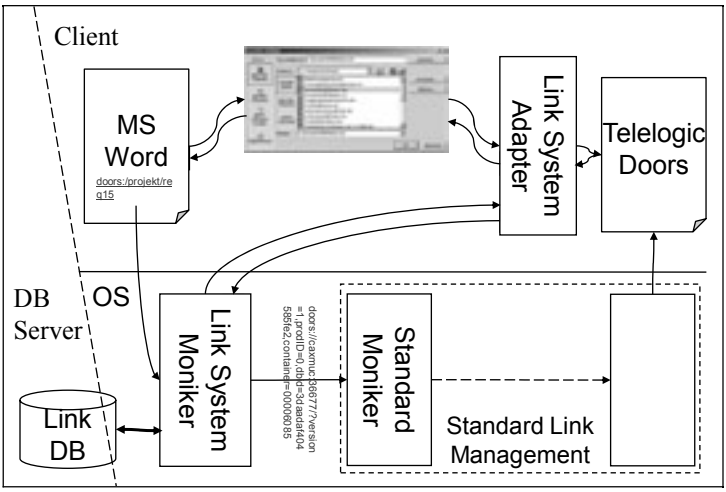

Abbildung 1: Architektur des Link-System
(Bsp: Microsoft Word / Telelogic Doors und Microsoft Windows)

Die **Link-Datenbank** wird im Hintergrund bei Speichervorgängen mit den aktuellen, in einer (versionierten) Informationsquelle verfügbaren Zielen aktualisiert, sowie mit den von dieser Quelle referenzierten Zielen. Sie erlaubt es dem Moniker, bei einer ungültig gewordenen URL die jeweils neue gültige zu ermitteln und über ein Service-Tool diese ggf. in den Informationen zu aktualisieren. Hauptzweck der Link-Datenbank ist somit die Vermeidung von Broken Links und die Möglichkeit zu einer performanten Forward- und Backward Traceability-Analyse.

Es kann eine zentrale Link-Datenbank für alle Client-Systeme verwendet werden, wenn die verknüpften Informationen ebenfalls z.B. durch ein Konfigurations-Management-Werkzeug oder gemeinsame Datenserver zentralisiert sind. Soll das Link-System auch bei rein lokalen Informationsquellen verwendet werden, muss die zentralisierte Link-Datenbank mit einer auf dem jeweiligen Client-System für lokale Quellen verwendeten Link-Datenbank im Verbund arbeiten.

Mit dieser Integration bleiben alle Funktionalitäten der Applikationen ohne Modifikation erhalten. Neuerungen in den Applikationen können genutzt und in das Link-System integriert werden, da es eine reine **Ergänzung** des Link Managements im Betriebsystem ist. Die Integration des Monikers erfordert allerdings einen erheblichen Eingriff in das Betriebssystem (Hooking). Ändern sich Applikationsschnittstellen müssen im schlimmsten Fall die Integration des Hyperlink-Dialogs und die Implementierung des spezialisierten Adapters angepasst werden.

Integriert werden können all jene Applikationen, die bereits die Navigation mit klassischen Hyperlinks in ihren Daten unterstützen, da sie in diesem Fall den Aufruf in das Standard Link Management des Betriebssystems implementiert haben. Der Aufwand zur Integration einer Applikation hängt dann wesentlich von der Qualität und Reife ihrer Programmierschnittstellen und der Anpassbarkeit ihrer Menüleisten ab.

3 Sensitives Hyperlinking

Sollte die zu referenzierende Informationsquelle keine Marken oder Strukturen in ihren Informationen unterstützen (z. B. Microsoft Notepad), oder sollte die zu referenzierende Stelle in der Information nicht durch deren Autor geeignet markiert worden sein, soll im Link-System das Konzept der **sensitiven Hyperlinks** [Fi06] eingesetzt werden, um auch in diesen Fällen navigieren zu können. Derartige Links sollen (möglichst) robust gegen Änderungen in den von ihnen referenzierten Informationen sein (vgl. 2.1).

Sensitive Hyperlinks werden innerhalb einer allen Adaptern gemeinsamen Adapter-Basis implementiert, die auch eine einheitliche Schnittstelle ins System sicherstellt (vgl. 2.2).

3.1 Die Erstellungs- und Wirkungsweise sensitiver Hyperlinks

Möchte ein Autor sensitiv auf eine Information in einem Dokument verweisen, soll er in diesem nur die gewünschte Passage markieren. Mit Hilfe unterlegter Wörterbücher und Thesauren wird dann von dem gesamten Dokument ein Profil erstellt, aus dem die Vorkommenshäufigkeiten aller Begriffe der markierten Passage einschließlich ihrer Flexionen und ggf. auch Synonyme passagenweise hervorgehen. Damit kann ermittelt werden, welche Begriffe ausschließlich in der vom Autor markierten Passage vorkommen. Diese werden in einem ersten Schritt ignoriert. Stattdessen wird eine (möglichst kleine) Menge von Begriffen gebildet, die ansonsten im Dokument nur selten vorkommen, die jedoch durch den Algorithmus zum Auffinden der Passage im Dokument (auf Basis des zuvor extrahierten Profils) zuverlässig gefunden werden. Begriffe zum Auffinden der Ränder der Passage müssen entsprechend dem obigen Verfahren gesondert der Begriffsmenge zugewiesen werden. Zuletzt werden die nur in der Markierung aufgetretenen Begriffe der Menge zugeordnet. Diese Begriffe einschließlich einer Information zum Umfang der Passage bilden die Indikator-Information im sensitiven Hyperlink, die zum erneuten Auffinden der Passage in diesem Dokument verwandt werden kann[1].

Der Algorithmus zum Auffinden der Passage bildet mit den im Indikator eingetragenen Begriffen ebenfalls ein Profil der Vorkommen dieser Begriffe und ihrer Flexionen und ggf. Synonyme im gesamten Dokument. Dann vergleicht er entsprechend der Umfang-Information des Indikators und einer einfachen Existenz-Vergleichs-Metrik die Anzahlen in den Passagen des Profils mit denen der Mengen. Die optimalen Fundstellen markieren die wahrscheinlichste vom Autor ursprünglich markierte Passage.

Für dessen Akzeptanz muss dieses Konzept performant umgesetzt werden. Zudem sollte es bzgl. Rechtschreibkorrekturen und Änderungen in Silbentrennungen etc. stabil sein. Daher bietet es sich an, den fehlertoleranten, bitparallelen Such-Algorithmus von Myers [My99] entsprechend der hier benötigten Anforderungen wie folgt zu ergänzen.

[1] Die Indikator-Information eines sensitiven Hyperlinks kann als Skizze der bei seiner Navigation zu identifizierenden Textpassage interpretiert werden. Diesen Gedanken weiterführend, lässt sich dieses Konzept auch auf Bild-Informationen erweitern.

3.2 Der bitparallele Such-Algorithmus von Myers [My99] mit Filter-Konzept

Das **k-Differenz-Problem** [St02] bezeichnet die Suche eines (Pattern-)Suchstrings $s=s_{[1]}...s_{[m]}$ in einem Textstring $t=t_{[1]}...t_{[n]}$ mit $m\equiv|s|\leq n\equiv|t|$, die einen **Levenshtein Abstand** [Le65] (entspricht der minimalen Anzahl von Einfügen-, Löschen- und Ersetzen-Operationen zur Überführung eines Strings x in einen String y) von höchstens k (Fehlstellen mit den Teilstrings aus t) haben. Die **Distanz-Matrix** $D := D_{i,j}$ mit:

$$D_{i,0} = i \qquad 0\leq i\leq|s|=m \qquad \textbf{[1]}$$

$$D_{0,j} = 0 \qquad 1\leq j\leq|t|=n \qquad \textbf{[2]}$$

$$D_{i,j} = \min\begin{cases} D_{i-1,j} + 1 \\ D_{i,j-1} + 1 \\ D_{i-1,j-1} + Eq(i,j) \end{cases} \quad 1\leq i\leq|s|=m,\ 1\leq j\leq|t|=n \quad \textbf{[3]}$$

$$\text{mit } Eq(i,j) = 0 \text{ falls } s_{[i]}=t_{[j]}, \text{ sonst } 1$$

löst dieses Problem für jedes $1\leq j\leq|t|$, für das $F(s)(t_i)\equiv D_{|s|,j}\leq k$ gilt [WMM92]. Dies bedeutet, dass der mit dem Zeichen t[j] endende Teilstring $t_{[j-|s|+1]}...t_{[j]}$ von t, mit höchstens k Fehlern, mit s übereinstimmt.

Myers [My99] konnte mit seinem Algorithmus die Rekursionsformel der Distanz-Matrix auflösen und aufgrund einiger Eigenschaften der Matrix deren Berechnung ausschließlich als inkrementelle Veränderung herleiten. In seiner Herleitung wird der Übergang von einer Spalte j zu der Spalte j+1 der Matrix für jedes i durch 2 Bits (+1, 0, -1) repräsentiert. Ihm ist es gelungen, alle einzelnen Bitoperationen als (logische) Wortoperationen eines Prozessors mit Bitbreite w zu formulieren (**Meyers-Funktion**). Damit erreichte er erstmals eine Laufzeit der Ordnung $O(|\Sigma| + |s| + |t|*\lceil|s|/w\rceil)$ zur Lösung dieser Aufgabe, wobei $|\Sigma|$ die Anzahl der Zeichen des Alphabets Σ beschreibt. Hyyrö [HN02] gelang es, die Anzahl der Wortoperationen zu reduzieren und mit einem dem folgenden Ansatz ähnelnden Verfahren, diese Laufzeit für lange Strings mit $|s|>w$ zu verbessern.

Die für die sensitiven Hyperlinks zweckmäßige Veränderung dieses Algorithmus verfolgt ein Filter-Konzept und basiert auf der folgenden Eigenschaft der $D_{i,j}$ der Distanz-Matrix D [Uk85]: $D_{i,j}\in\{ D_{i-1,j-1}, D_{i-1,j-1}+1\}$ **[4]**. Wird der zu suchende String s als aus den Strings s_1 und s_2 zusammengesetzt betrachtet ($s=s_1+s_2$), gilt wegen [*4*] für den Fehler an einer Stelle i: $F(s_1+s_2)(t_{i+|s_2|}) \geq F(s_1)(t_i) \ \forall\ 1\leq i\leq|t|-|s_2|$ **[5]**. Außerdem folgt aus [*1*]: $F(s_1+s_2)(t_i) \geq |s| - |s_2| \ \forall\ 1\leq i\leq|s_2|$ **[6]**.

Die Bedingung [*5*] rechtfertigt, dass der zweite Wortteil s_2 nur dann gesucht wird, wenn $F(s_1)(t_i) \leq k$ **[7]** gilt. Ebenfalls wegen [*5*] muss dies dann notwendig die nächsten $|s_2|$ Zeichen geschehen. Im Falle von [*7*] bedeutet dies, dass innerhalb der Ablaufschleife über die Zeichen von t und der Berechnung der Meyers-Funktion für s_1 ein Zonen-Zähler z auf $|s_2|$ gesetzt werden muss, der für $z>0$ zeichenweise dekrementiert wird und die Berechnungen der Meyers-Funktion für s_2 ebenfalls durchführt. Immer, wenn z von Null auf einen Wert größer Null wechselt, müssen die Parameter der Meyers-Funktion für s_2 neu initialisiert werden. Die generelle Algorithmus-Initialisierung der Ordnung $O(|\Sigma| + |s|)$ muss jedoch nur einmalig erfolgen. Wegen [*6*] ist es notwendig, den Algorithmus mit einem Zählerwert von 1 zu starten. Diese (iterativ für weitere Unterteilungen von |s|) fortsetzbare Modifikation, hat für w=3 und k=1 das in Abbildung 2 dargestellte

Ablaufverhalten und eine Ordnung kleiner gleich $O(|\Sigma| + |s| + |t|*\lceil |s|/w \rceil)$. Für Suchstrings mit $|s| > w$ kann dies den Algorithmus von Myers beschleunigen.

Neben dieser Veränderung kann der Algorithmus außerdem für die Suche mehrerer Strings $s^1,...,s^p$ modifiziert werden. Dazu kann der Suchstring s des Algorithmus als aus den $s^1...s^p$ zusammengesetzt aufgefasst werden. Innerhalb der Meyers-Funktion muss nur ein Übersprechen der Wortoperationen über die einzelnen Begrenzungen hinweg mit Hilfe einer einfachen Bitmaske verhindert werden, sodass der Algorithmus gleichzeitig diese Strings suchen kann. Er muss dann allerdings alle $F(s^q)(t_i)= D_{\sigma,j}$ mit $\sigma = \Sigma_{r=1...q}|s^r|$ und $1 \leq q \leq p$ bezüglich der Bedingung [7] überprüfen.

Im Falle sensitiver Hyperlinks werden mehrere Begriffe einschließlich ihrer Flexionen gesucht. Flexionen haben häufig die Eigenschaft, ein mit dem ursprünglichen Wort gemeinsames Präfix zu haben („*Tante*“, „*Tanten*“, „*Tantchen*“). Werden nun diese Präfixe der unterschiedlichen Begriffe gemeinsam in den Algorithmus von Myers eingespeist und wird entsprechend einer ‚Kreuzung' der obigen Konzepte nur ‚wenn nötig' nach den unterschiedlichen Suffixen gesucht, kann das Erstellen und das Auffinden des Indikators auch in großen Dokumenten ausreichend schnell geschehen [Fi06].

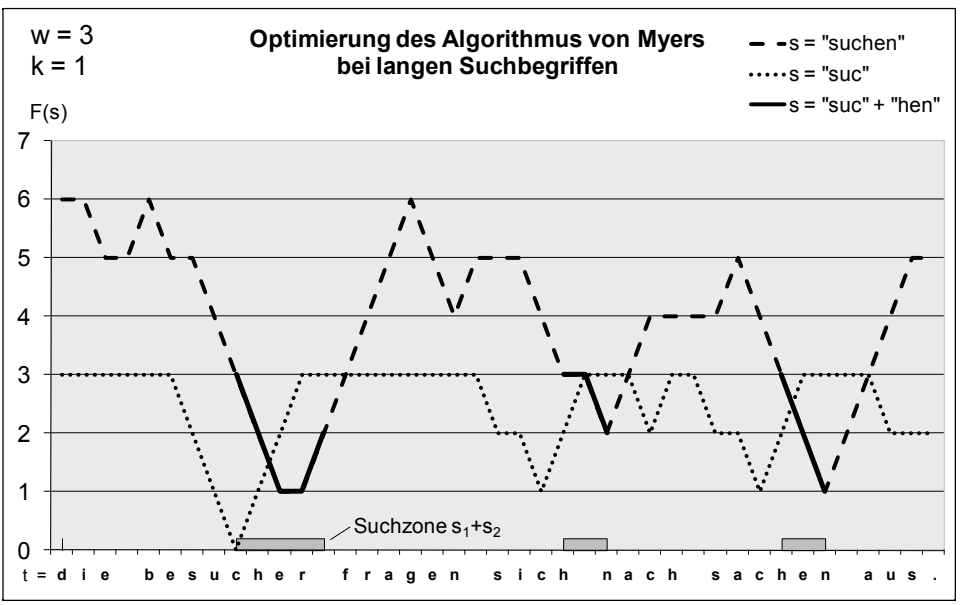

Abbildung 2: Optimierung des Algorithmus von Myers mit dem Filter-Konzept

4 Zusammenfassung

Im Gegensatz zu bestehenden Systemen vermeidet das hier vorgestellte Konzept die Zusammenführung und damit Kopie von Daten in eine große gemeinsame Datenbank.

Desweiteren verzichtet dieses Konzept auf die automatische Extraktion von Links, wie es in vielen Information-Retrieval-Applikationen praktiziert wird. Damit soll bewirkt werden, dass Autoren bewusst Hintergründe zu ihren Informationen benennen und so gezielt Störinformationen auf ein Minimum begrenzen können. Dies ermöglicht die in Kapitel 2.1 erwähnte Notwendigkeit zur bewussten Steuerung der Link-Dichte.

Zuletzt zeichnet sich das hier vorgestellte System dadurch aus, dass es den Anwender im Arbeitskontext einer Aufgabe innerhalb der dazu aktuell verwendeten Applikation unterstützt. Es stellt die Verknüpfung nicht in einem nachgeschalteten, im schlimmsten Fall zeitlich verzögerten und von den Arbeitsabläufen und den dazu verwendeten Applikationen entkoppelten System bereit. Die Fokussierung auf die Unterstützung während der Durchführung einer Aufgabe wird insbesondere durch die sensitiven Hyperlinks unterstrichen. Mit diesen kann auch dann innerhalb eines Dokuments referenziert werden, wenn ein Dokument nicht mit (klassischen) Markierungen als Hyperlinkziele versehen wurde.

Das hier vorgestellte System ist eine querschnittliche und schlanke Integration der Werkzeuge, die eingesetzt werden, ohne selber als Werkzeug in Erscheinung zu treten.

Literaturverzeichnis

[Fi06] Fischer, Dirk: „Flexible Informationssuche", Diplomarbeit am Lehrstuhl für Betriebswirtschaftslehre, insbesondere Operations Research an der Fernuniversität in Hagen (2006), unveröffentlicht.

[HN02] Hyyrö, Heiki, Navarro, Gonzalo: „Faster Bit-parallel Approximate String Matching" in: Proceedings of the 13th Annual Symposium on Combinatorial Pattern Matching, July 03-05 (2002) S. 203-224, Web-Download: http://portal.acm.org/citation.cfm?id=736383&dl=GUIDE&coll=GUIDE&CFID=10952486&CFTOKEN=81722893

[Le65] Levenshtein, Vladimir I.: „Binary codes capable of correcting deletions, insertions, and reversals." in: Doklady Akademii Nauk SSSR, 163(4) S. 845-848, 1965 (Russisch). Englische Übersetzung in: Soviet Physics Doklady, 10(8) S. 707-710, 1966.

[MSDN08] Microsoft Developer Network Library, 2008, Web-Download: http://msdn2.microsoft.com/en-us/library/ms691261.aspx

[My99] Myers, Gene: „A Fast Bit-Vector Algorithm for Approximate String Matching Based on Dynamic Programming", in: Journal of the ACM, Volume 46 [3] (1999) 395-415, Web-Download: http://citeseer.ist.psu.edu/cache/papers/cs/3956/http:zSzzSzwww.cs.arizona.eduzSzpeoplezSzgenezSzPAPERSzSzbit.mat.pdf/myers99fast.pdf.

[St02] Stamme, Katrin: „Einsatz von bitparallelen Algorithmen und Filtern zur approximativen Zeichenkettensuche", Diplomarbeit am Institut für Informatik der Universität Hannover (2002).

[Uk85] Ukonen, E.: „Algorithms for approximate string matching", in Information and Control 64 (1985) 100-118.

[WMM92] Wu, Sun, Manber, Udi, Myers, Gene: „A Sub-quadratic Algorithm for Approximate Limited Expression Matching", in: Algorithmica 15 [1] (1992) 50-67,Web-Download: http://citeseer.ist.psu.edu/cache/papers/cs/3178/ftp:zSzzSzftp.cs.arizona.eduzSzreportszSz1992zSzTR92-36.pdf/wu92subquadratic.pdf.