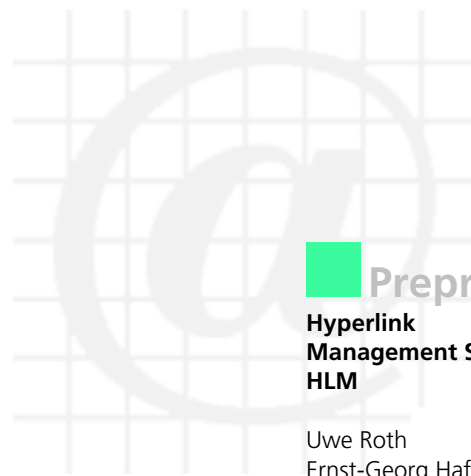




**Institut für Telematik**

unter Betreuung der  
**Fraunhofer Management GmbH**



**Preprint 99-05**  
**Hyperlink**  
**Management System**  
**HLM**

Uwe Roth  
Ernst-Georg Haffner  
Andreas Heuer  
Thomas Engel  
Christoph Meinel



Authors	Uwe Roth, Dipl. Inform. Ernst-Georg Haffner, Dipl. Inform. Andreas Heuer, Dipl. Phys. Thomas Engel, Dr. rer. nat. Dipl.-Phys. Christoph Meinel, Univ.-Prof. Dr. sc.
Copyright	© 1999 Institut für Telematik e.V., Trier
Trademarks	All terms that are mentioned in this paper that are known to be trademarks or service marks have been appropriately capitalised. Use of a term in this paper should not be regarded as affecting the validity of any trademark and service mark. The product or brand names are trademarks of their respective owners.
Printing	12/1999, Version 1.3 Printed in Germany All rights reserved
	The documentation was accomplished through the Institut für Telematik. The information contained in this document represents the current view of the authors on the issues discussed as of the date of publication. Because the mentioned enterprises must respond to changing market conditions, the results of this paper should not be interpreted to be a commitment on the part of the authors. Any information presented after the date of publication are subject to change. The right to copy this documentation is limited by copyright law. Making unauthorised copies, adaptations or compilation works without permission of the authors or institutions mentioned above is prohibited and constitutes a punishable violation of the law.



# Inhalt

<b>INHALT</b> .....	<b>3</b>
<b>1 EINLEITUNG</b> .....	<b>4</b>
<b>2 GRUNDLAGEN</b> .....	<b>4</b>
<b>3 DER MULTI-DOCUMENT-POOL</b> .....	<b>6</b>
<b>4 DIE IMPLEMENTIERUNG DES MDP</b> .....	<b>8</b>
<b>5 DER HYPER-DATA-POOL</b> .....	<b>9</b>
<b>6 DER LINK-MANAGER</b> .....	<b>10</b>
<b>7 ZUSÄTZLICHE MANAGER</b> .....	<b>10</b>
<b>8 DIE IMPLEMENTIERUNG DES HDP</b> .....	<b>11</b>
<b>9 REFERENZEN</b> .....	<b>12</b>

# 1 Einleitung

Die Verwaltung von Internet-Präsenzen (Web-Sites) gestaltet sich bei zunehmender Größe immer schwieriger. Ein klassisches Problem hierbei ist es, die Konsistenz von Links beim Verschieben oder Löschen von Dokumenten zu erhalten, sowie Broken Links, also Verweise zu Dokumenten, die nicht (mehr) existieren zu erkennen. Bei zunehmendem Umfang von Web-Sites ist aber auch die automatische bzw. halbautomatische Generierung von sinnvollen Links eine schwierige Aufgabe. Verstärkt wird die Problematik im Hinblick auf die mögliche Multilingualität von Web-Sites. Hier bieten existierende Tools meist keine Unterstützung.

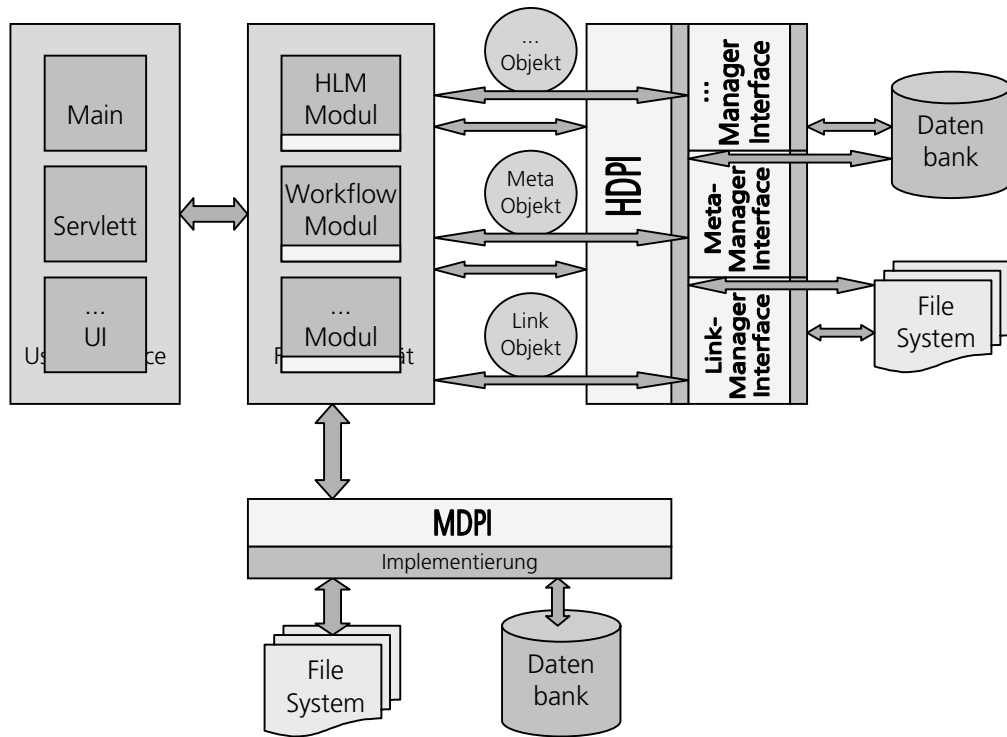
Am Institut für Telematik wurde im Rahmen des Forschungsprojektes "Konzeption und Realisierung eines Internet-basierten, multilingualen Dokumentenmanagers", das durch die Stiftung für Innovation in Rheinland-Pfalz gefördert wurde, ein Hyperlink-Management-System (HLM) entwickelt, das diese Probleme schon im Vorfeld berücksichtigt. Bei der Implementierung wurde darüberhinaus das Konzept eines modularen und erweiterbaren Frameworks verfolgt.

# 2 Grundlagen

Kerngedanke des am Institut für Telematik entwickelten Hyperlink-Management-Systems (HLM) ist es, die Hyperlink-Management-Funktionalität in einem besonderen HLM-Modul vom Dokumenten-Speicher und der Speicherung von Meta-Daten (zu diesen Dokumenten) zu trennen. Diese Trennung wurde durch die Definition zweier Interfaces, eines für den Dokumenten-Pool und eines für den Meta-Daten-Pool, vollzogen. Die Entscheidung über die Art der Implementation jedes dieser Interfaces (Datei-System, verschiedene Datenbanken, o.ä.) kann somit unabhängig voneinander getroffen werden, ohne dass dies die Implementierung der jeweils anderen Module berührt. Der Dokumentenpool (Multi-Document-Pool=MDP) implementiert das Multi-Document-Pool-Interface (MDPI), wohingegen die Meta-Daten-Haltung (Hyper-Data-Pool=HDP) das Hyper-Data-Pool-Interface (HDPI) implementiert. Dabei setzt sich das HDPI aus einem Core-Interface und den sogenannten Manager-Interfaces zusammen. Während das MDP-Interface und das Core-HDP-Interface Funktionen allgemeiner Art zur Verfügung stellen und bezüglich den Dokumenten nur die Basis-Funktionalitäten bietet, die sich auch in Zukunft nicht ändern werden, repräsentieren die Manager einen jeweils in sich abgeschlossenen Bereich der Meta-Daten. Diese Manager sind eng mit der gewünschten Funktionalität des HLM-Moduls verknüpft; Beispiele hierfür sind Link-Manager, Ressort-Manager, Meta-Manager. Zukünftige Funktionalitäten des HLM-Moduls oder andere Funktionsmodule können es erforderlich machen, weitere Manager zu implementieren. Jeder Manager operiert gegenüber dem Nutzer des Interface mittels Objekten (Link-Objekt, Ressort-Objekt, ...), wodurch die Benutzung des Interface erleichtert wird.

Eine Implementierung des HDPI beinhaltet das Core-Interface sowie die Manager-Interfaces, was sich aus den möglichen Abhängigkeiten der Daten in den darunterliegenden Datenbanken ergibt. In späteren Ausbaustufen ist eine Implementierung der Interfaces mittels RMI geplant.

Das Schema auf der folgenden Seite veranschaulicht nochmals die hier beschriebene Struktur des Systems.



Schema des Hyperlink-Management-Systems

### 3 Der Multi-Document-Pool

Der Dokumentenpool ist der Ort, an dem Dokumente (beliebige Dateien) abgelegt werden können. Bei der Definition der Eigenschaften des Pools, d.h. des entsprechenden Interfaces, wurde berücksichtigt, dass es sich bei den Dokumenten um Dokumente *für* eine oder *von* einer Internetpräsenz handelt. Somit findet eine Referenzierung der Dokumente im Pool über einen URI (Unified Resource Identifier) statt, die entweder dem URL des Ursprungsdokuments oder dem Ziel entspricht. Der URI ist dabei der absolute Pfad eines URL ohne Protokoll-Angabe bis zum ersten Erscheinen eines "?" oder "#" (z.B. "//www.mywebsite.de/bla/blub/doc.de.html"). URI's verweisen auf einen Entry (Eintrag), der ein Container (Ordner) oder ein Document (Dokument) sein kann. Wie auch in der hierarchischen Struktur eines Dateisystems, sind Dokumente und Container immer Teil eines Ober-Containers; einzige Ausnahme von dieser Regel ist der Wurzel-Container "/".

Bestandteile einer URI sind :

- Location: //www.mywebsite.de:8080
- Container : /bla/blub/
- Dokumentname doc
- Dokument-Suffix .de.html

Auf ein Dokument mit einer Location im Internet zuzugreifen bedeutet nicht, dass ein Zugriff auf das Original-Dokument im Internet durchgeführt wird. Die Location dient nur dazu die Dokumente der Präsenz zuzuordnen, von der die Dokumente stammen oder zu der sie exportiert werden sollen.

Der Dokumentenpool berücksichtigt weiterhin, dass Dokumenten in verschiedenen Ausprägungen unter dem selben URI existieren können, z.B. mit unterschiedlichem Exportstatus, Version o.ä. Diese Ausprägungen, die als Peculiarities bezeichnet werden, sind Paare von Peculiarity-Eigenschaft und Peculiarity-Ausprägung : z.B. (Version = 1.0), (ExportStatus = gesperrt). Jedem Dokument ist eine eindeutige Menge von Peculiarities zugewiesen. Durch diese Zuweisung wird somit ein Dokumenten-Raum aufgespannt, dessen Raumachsen die eindeutige Mengen von Peculiarities darstellen.

Die Referenzierung eines Entry im Dokumentenpool erfolgt durch die eindeutige Angabe eines Paares aus URI und der Peculiarities-Menge. Echte Teilmengenbeziehungen werden somit als unterschiedlich aufgefasst. Das Paar aus URI und Peculiarities-Menge wird als PURI bezeichnet. Der PURI alleine sagt jedoch nicht aus, ob es sich bei der referenzierten Ressource um ein Dokument oder einen Container handelt. Dies hängt damit zusammen, dass Web-Browser bei der Angabe eines Container-URL ein Default-Dokument, oder ein Dokument, dass den Inhalt dieses Containers darstellt, zurückliefern können.

Aus diesem Grund wird der PURI weiter differenziert und zwar in den CPURI, der einen Container referenziert und den DPURI, der ein Dokument referenziert.

Die Navigation in einem Dokumentenpool gestaltet sich ähnlich der Navigation in einem Dateisystem. Folgende Befehle stehen zur Verfügung.



void	<b>init</b> (MDMProperties properties) inits the module with the given properites
PURI []	<b>locations</b> () shows the root-puris of all locations
Container	<b>dir</b> (Entry entry) shows the contens of a directory, referenced by an entry every part of the peculiarity has to match
Container	<b>dir</b> (PURI puri) shows the contens of a directory, referenced by a puri every part of the peculiarity has to match
Container	<b>dirByFilter</b> (PURI puri) shows the contens of a directory, referenced by a puri only the necessary parts of the peculiarity are part of the puri
CPURI	<b>mkCont</b> (CPURI cpuri) creates a container
void	<b>rmCont</b> (CPURI cpuri) deletes a container and all its contens
Boolean	<b>exists</b> (CPURI cpuri) proofes, if the container exists
DPURI	<b>put</b> (Document document) puts a document
Document	<b>get</b> (DPURI dpuri) get a document, referenced by a document-puri
Void	<b>delete</b> (DPURI dpuri) deletes a document
Void	<b>copy</b> (DPURI dpuriFrom, DPURI dpuriTo) copies a document
Void	<b>move</b> (DPURI dpuriFrom, DPURI dpuriTo) moves a document
Boolean	<b>exists</b> (DPURI dpuri) proofs, if the document exist
Long	<b>length</b> (DPURI Dpuri) return the length

## 4 Die Implementierung des MDP

Die aktuelle Implementierung setzt auf dem bestehenden Dateisystem auf. Auf die Möglichkeit, Dokumentenroots unverändert in einen Dokumentenpool zu übertragen zu können musste jedoch leider verzichtet werden. Dies bedeutet, dass Teilverzeichnisse existierender Dokumentenroots nicht am MDP vorbei in den Dokumentenpool kopiert werden können.

Drei Komponenten erweitern das Dateisystem:

- ".pecuDir<x>":  
Unterverzeichnis, das die Dokumente einer Pecularity beinhaltet;
- ".pecus"  
Serialisiertes Objekt, das Informationen darüber enthält, welche Pecularity zu welchem ".pecuDir<x>" gehört;
- ".pecu"  
Text-Dokument innerhalb des ".pecuDir", das die Peculiarities in Textform darstellt;

Zur Verwaltung des Pools werden nur die Komponenten ".pecus" und ".pecuDir<x>" benötigt. Die Komponente ".pecu" dient einzig und allein der Orientierung, um im Pool außerhalb des MDP zu navigieren.

Der Pfad, in den ein Dokument abgelegt wird, setzt sich folgendermaßen zusammen:

`<pool-root><transformed-location><path>.pecuDir<x>/`

Dabei gilt, dass die Zuordnung von Peculiarities zu `pecuDir<x>` im Objekt `<pool-root><transformed-location><path>.pecus` gespeichert ist und diese Information als Text noch einmal in `<pool-root><transformed-location><path>.pecuDir<x>.pecu` protokolliert wird.

Die Transformation der Location entfernt die vorangestellten "//" und ersetzt ":" durch "\_" und "\_" durch "\_\_".

Damit sowohl "dir" als auch "dirByFilter" sinnvoll funktionieren können, muss für einen Pfad der allgemeinen Form `<path>=/<container1>/<container2>/.../<containerN>` jeder Teilpfad `/, /<container1>, /<container1>/<container2>, ... /<container1>/<container2>/.../<containerN>` ein ".pecuDir<x>" beherbergen.

Nicht nur aus Gründen der Performance gestaltet sich die zentrale Haltung eines Dokumentenpools in einem Filesystem als äußerst schwierig, vor allem, wenn das System von mehreren Benutzern genutzt werden soll. Deshalb wurde entschieden, den Dokumentenpool auch in einer Datenbank-Version zu implementieren.

## 5 Der Hyper-Data-Pool

Das Core-Interface besitzt ähnliche Funktionen wie das MDP, also Methoden, um Dokumente aufzunehmen, zu löschen, zu suchen, sowie Container anzulegen oder zu löschen. Neu im HDP ist dabei die *getInstance*-Methode, die die Instanz eines Manager zurückliefert. Der Zugriff erfolgt dann über das dazugehörige Manager-Interface. Der nutzenden Klasse muss also schon über die Information, welchen Manager sie benutzen muss, verfügen. An dieser Stelle wird allein die Implementierung ausgewählt.

Method Summary	
void	<b><u>init</u></b> (MDMProperties p) initializes the interface
ManagerInterface	<b><u>getInstance</u></b> (java.lang.String manager) returns an instance of a specified manager
Container	<b><u>getByFilter</u></b> (PURI puri) get a container, defined by a puri(-filter)
CPURI	<b><u>mkCont</u></b> (CPURI cpuri) creates a container
void	<b><u>rmCont</u></b> (CPURI cpuri) deletes a container and all its contents
void	<b><u>put</u></b> (DPURI dpuri) adds a document
void	<b><u>delete</u></b> (DPURI dpuri) deletes a document

Aus diesem Grund beschränkt sich der Funktionsumfang des Manager-Interface zur Zeit auch nur auf die *init*-Methode.

Method Summary	
void	<b><u>init</u></b> (MDMProperties p)

## 6 Der Link-Manager

Der Link-Manager liefert Methoden zur Behandlung von Links, z.B., welche Links ein Dokument beinhaltet, welche Dokumente auf ein anderes Dokument zeigen u.s.w..

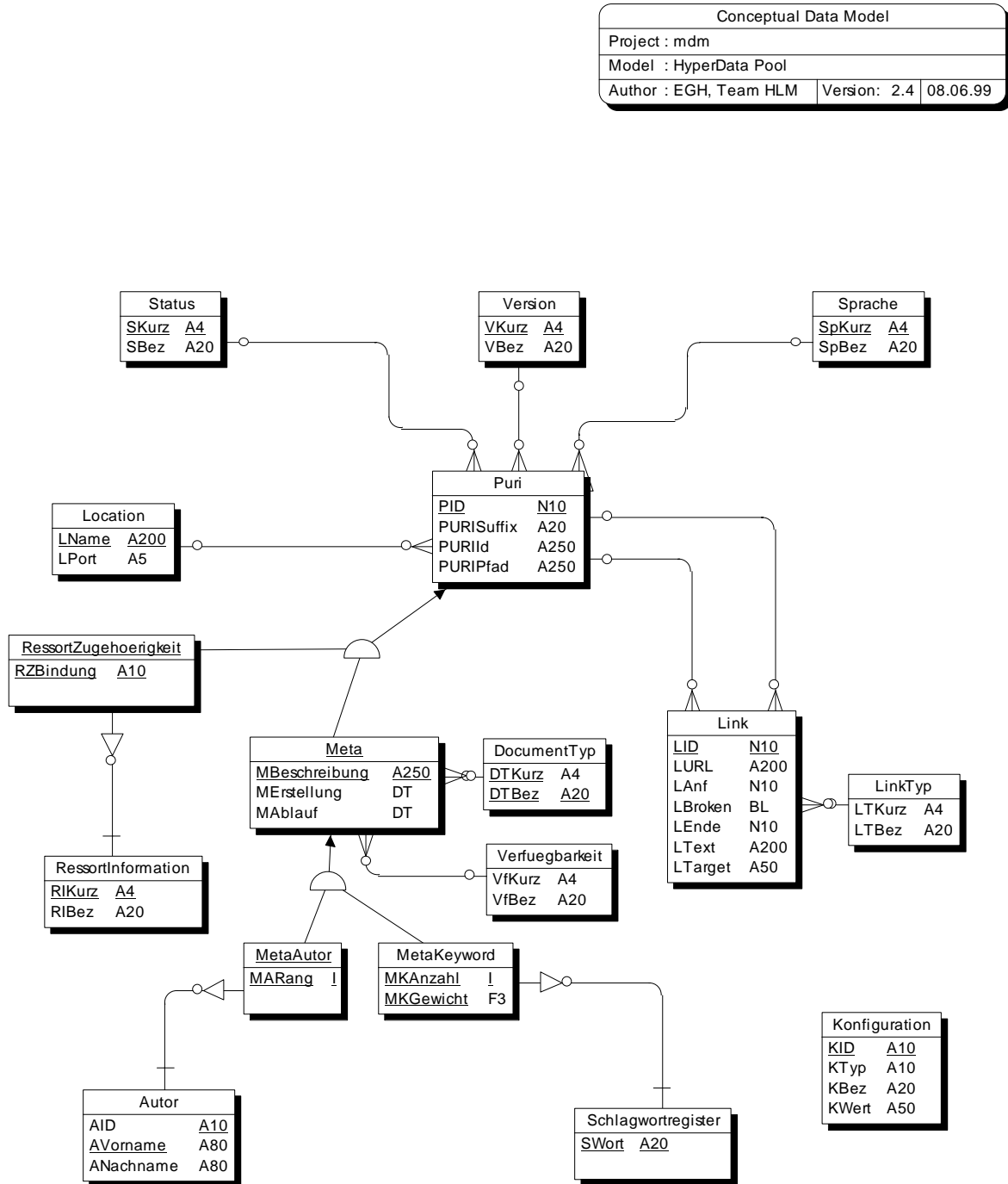
Method Summary	
<u>LinkList</u>	<b><u>getByFilter</u></b> ( <u>Link</u> l) returns all links that match the filter criteria given in <i>Link l</i> that is the parameter of this method.
<u>LinkList</u>	<b><u>getInverseByFilter</u></b> ( <u>Link</u> l) returns all links that do <b>not</b> match the filter criteria given in <i>Link l</i> that is the parameter of this method.
<u>LinkList</u>	<b><u>getLinksFrom</u></b> ( <u>PURI</u> puri_ref) returns a list of links containing all links that have their source in the document specified in the given PURI.
<u>LinkList</u>	<b><u>getLinksTo</u></b> ( <u>PURI</u> puri_ref) returns a list of links containing all links that point to the specified document.
<u>LinkList</u>	<b><u>updateByFilter</u></b> ( <u>Link</u> filter, <u>Link</u> destination) updates all links that match the criteria given in <i>Link l destination</i> .
<u>Link</u>	<b><u>verify</u></b> ( <u>Link</u> l) verifies <i>Link l</i> attributes like being broken, etc. by use of the hyper data pool.
<u>int</u>	<b><u>delete</u></b> ( <u>Link</u> l, boolean allowFilter)
<u>Link</u>	<b><u>put</u></b> ( <u>Link</u> l) stores the given <i>Link l</i> in the hyper data pool. If the <i>Link l</i> already exists, it will be updated.
<u>URI</u>	<b><u>PURI2URI</u></b> ( <u>PURI</u> puri) converts the PURI into an URI.
<u>PURI</u>	<b><u>URI2PURI</u></b> ( <u>PURI</u> puri_ref, <u>URI</u> url_ref) converts a URI to an PURI using the Peculiarities of the given PURI

## 7 Zusätzliche Manager

- Meta-Manager:  
Meta-Daten zu Dokumenten.
- Similarity-Manager  
Stellt Ähnlichkeiten von Dokumenten zueinander her um Link-Vorschläge zu generieren.

## 8 Die Implementierung des HDP

Der HDP existiert zur Zeit in Form einer Datenbank-Implementierung für Oracle und MySQL. Das zugrundeliegende Datenbankmodell ist in der folgenden Abbildung dargestellt.



## 9 Referenzen

- C.Meinel, T.Engel, E.-G. Haffner, A. Heuer, U. Roth, Z. Zhang, S. Müller, K. Siemeonsen  
*Konzeption und Realisierung eines Internet-basierten, multilingualen Dokumentenmanagers*  
Abschlussbericht für die Stiftung Rheinland-Pfalz für Innovation, 1999.
- A. Heuer, E. G. Haffner, U. Roth, Z. Zhang, T. Engel, Ch. Meinel  
*Hyperlink Management System for Multilingual Web Sites*  
Asia Pacific International Web Conference, APWEB'99, Hong Kong, 1999.
- Z. Zhang, E. Haffner, A. Heuer, T. Engel, Ch. Meinel  
*Role-based Access Control in Online Authoring and Publishing Systems vs. Documentation Hierarchy*  
17th International Conference on Systems Documentation, ACM SIGDOC'99, New Orleans, USA, 1999.
- Z. Zhang, A. Heuer, T. Engel, Ch. Meinel  
*DAPHNE - A Distributed Tool for Web Authoring and Publishing*  
Annual Conference of American Society for Information Science, ASIS'99, Washington, USA, 1999.
- Heuer, Z. Zhang, T. Engel, Ch. Meinel  
*DAPHNE - Distributed Authoring and Publishing in a Hypertext and Network Environment*  
luK'99, Jena, 1999.