



Software Service Engineering

Prof. Dr.-Ing. Martin Gaedke

Technische Universität Chemnitz

Fakultät für Informatik

Professur Verteilte und selbstorganisierende
Rechnersysteme

<http://vsr.informatik.tu-chemnitz.de>



Chapter 4

UDDI



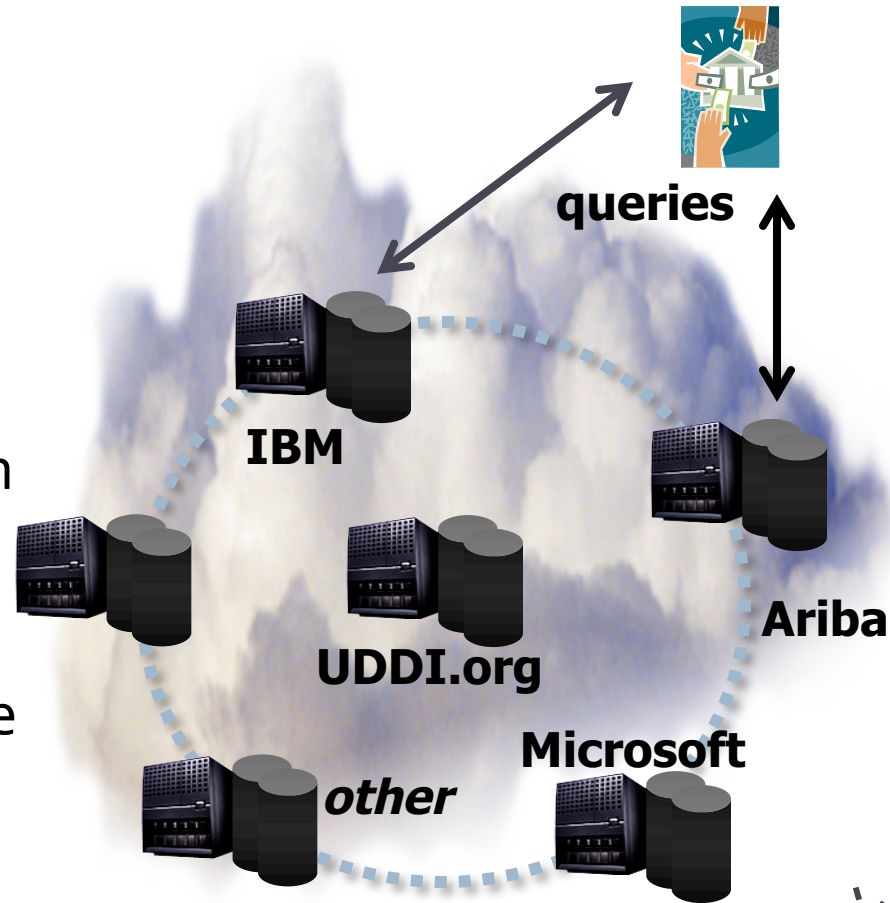
Einführung

- **Universal Description, Discovery, and Integration (UDDI)** – UDDI beschreibt Ansätze um eine interoperable Plattform zu etablieren, die es Firmen und Applikationen ermöglicht, schnell, einfach und dynamisch Web-basierte Dienste zu finden und über das Internet zu benutzen
 - OASIS-Standard, UDDI Version 3, Februar 2005
 - UDDI spezifiziert das API für Web-basierte Registries
 - API fokussiert Veröffentlichen und Auffinden von Diensten im Web (d.h. nicht nur Web Services)
 - Ursprünglich auch Beschreibung wie UDDI Business Registry betrieben wird (Geschäftsmodell, Bezahlung etc.)
 - Weitere Informationen unter <http://uddi.org>
- **UDDI Registry** – vereint drei Kategorien
 - **White Pages:** Namensregister, Kontakt zum Anbieter, Klassifikation (z.B. Bank, Großhandel, Ländercodes etc.) – vgl. Telefonbuch
 - **Yellow Pages:** Entspricht der Idee des Branchenbuchs, d.h. Einordnung von Diensten in Geschäftskategorien
 - **Green Pages:** Technisches Informationsmodell (tModel)



UDDI Business Registry (UBR) - Idee

- UBR agieren als Peer Nodes (Websites)
- Firmen (Kunden) können bei irgendeiner Firma bzw. Deren UBR ihre Daten registrieren
- Registrierungen werden mit den anderen UBRs repliziert (Registrierungen sind daher jederzeit über alle UBRs verfügbar)
- Standard SOAP API muss von allen UBRs unterstützt werden
- Vertragliche Regelung



Achtung! UBR-Idee wird derzeit nicht mehr verfolgt.

UBR Beispiel

Microsoft UDDI Business Registry (UBR) node - Microsoft Internet Explorer

Address: <http://uddi.microsoft.com/search/frames.aspx?frames=true&search=f920ab11-7ebf-429a-a43f-e610c2acd2c>

Microsoft uddi Business Registry Node

Home Search Quick Help UDDI Help

Search Results Explorer

WebEngineering.org Community

Dr.-Ing. Martin Gaedke

WebEngineering.org

<http://www.webengineering.org>

uddi-org:homepage

uddi-org:http

A provider is a party, such as a person, group, or business, that offers and supports services and additional information published for this provider using the tabs information can only be modified by the owner (publisher) listed.

Details Services Contacts Identifiers Categories Discovery URL

A categorization scheme is a predefined set of categories, derived from an external hierarchy, that can be used to classify a provider.

Categorizations

Categorization Scheme: ntis-gov:naics:1997
Key Name: Scientific Research and Development Services
Key Value: 5417

Categorization Scheme: microsoft-com:geoweb:2000
Key Name: Karlsruhe
Key Value: 504970

Categorization Scheme: unspsc-org:unspsc:3-1
Key Name: Software engineering
Key Value: 811115

Categorization Scheme: unspsc-org:unspsc:3-1
Key Name: Internet services
Key Value: 811121

Categorization Scheme: uddi-org:iso-ch:3166:1999
Key Name: Baden-Württemberg
Key Value: DE-BW

Categorization Scheme: microsoft-com:geoweb:2000
Key Name: Europe
Key Value: 100010

Categorization Scheme: microsoft-com:geoweb:2000
Key Name: Germany
Key Value: 200088

Categorization Scheme: microsoft-com:geoweb:2000
Key Name: Baden-Württemberg
Key Value: 300429

8 record(s) found.

© 2000-2003 Microsoft Corporation. All rights reserved.

Business Details - Microsoft Internet Explorer

Address: <https://uddi.ibm.com/ubr/findbusiness?action=details&businesskey=C134764E-77C3-4BDE-88C5-7CC8C037B434>

IBM Corporation > Services/UDDI > Find

UDDI Business Registry

UDDI Business Registry Version 2
Universal Description, Discovery, and Integration

Find

Related Links:

Web Services and UDDI

IBM UDDI Business Test Registry

IBM partners:

Business Details

The details of the selected business are shown below. Please use your browser's Back button to return to the previous page OR Press the New Search button to search again.

Business Information

Key
C134764E-77C3-4BDE-88C5-7CC8C037B434

Discovery URL
<http://uddi.microsoft.com/discovery?businesskey=C134764E-77C3-4BDE-88C5-7CC8C037B434>

Usage
businessEntity

Business Name(s)

Name
WebEngineering.org Community

Language
en

Business Description(s)

Description
WebEngineering.org Community provides a forum for researchers, teachers, practitioners, managers and others interested in latest information on the young discipline Web Engineering.

Language
en

Business Contact(s)

Name
Dr.-Ing. Martin Gaedke

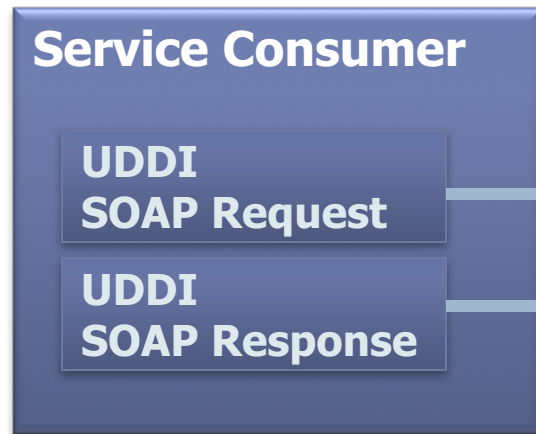
Role
None

Business Locator(s)

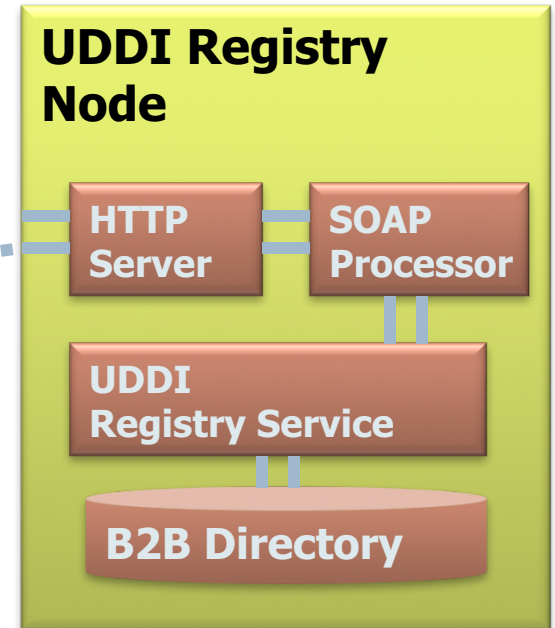
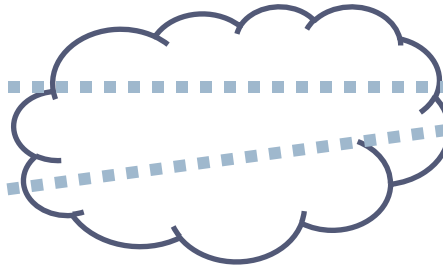
Code	Description	Type
5417	Scientific Research and Development Services	NAICS
504970	Karlsruhe	NAICS
811115	Software engineering	NAICS
811121	Internet services	NAICS
DE-BW	Baden-Württemberg	NAICS
100010	Europe	NAICS
200088	Germany	NAICS
300429	Baden-Württemberg	NAICS

Achtung! UBR-Idee wird derzeit nicht mehr verfolgt!

UDDI im Einsatz (1) - Szenario



**Create, View,
Update, Delete
Registrierungen**



**Black-Box-Prinzip
(Implementierungs-
Unabhängig)**

UDDI im Einsatz (2) - Ablauf



UDDI im Einsatz (3) - APIs

■ APIs in Verbindung mit SOAP

- Inquiry API für Finden von Diensten
- Publisher API für die Verwaltung von Dienst-Informationen des Anbieters (Service Providers)

■ Inquiry API

- Find things
 - find_business
 - find_service
 - find_binding
 - find_tModel
- Get Details about things
 - get_businessDetail
 - get_serviceDetail
 - get_bindingDetail
 - get_tModelDetail

■ Publishers API

- Save things
 - save_business
 - save_service
 - save_binding
 - save_tModel
- Delete things
 - delete_business
 - delete_service
 - delete_binding
 - delete_tModel
- security...
 - get_authToken
 - discard_authToken



UDDI Registry (1) – Wer/Was/Wo/Wie

- **Wer** – Information über Anbieter des Web-basierten Dienst
- **Was** – Typ im Sinne Geschäftskategorie
- **Wo** – Wo kann der Dienst abgerufen werden
- **Wie** – Beschreibung der Dienstschnittstelle

The screenshot displays the UDDI Registry web interface. On the left, there is a sidebar with sections: **Search** (by business name, Advanced Search), **LINKS** (Home, News), **TOOLS** (Register, Administer, Search), **DEVELOPERS** (For Developers, uddi.org Resources, Solutions), and **HELP** (Help, Frequently Asked Questions, Policies, About UDDI, Contact Us). The main content area has a **Browse** section with links to classification systems. Below this, the **Business detail** section shows information for 'Oxford University Libraries'. The **Contacts** section lists contact details for Matthew J. Dovey and Reg Carr. The **Services** section lists services like OLIS, InterLibrary Loans, and Copyright Deposit.

Search by business name
Advanced Search

Browse by category
North American Industry Classification System
Universal Standard Products and Services Codes
ISO 3166 Geographic Taxonomy
Standard Industrial Classification
GeoWeb Geographical Names

LINKS
[Home](#)
[News](#)

TOOLS
[Register](#)
[Administer](#)
[Search](#)

DEVELOPERS
[For Developers](#)
[uddi.org Resources](#)
[Solutions](#)

HELP
[Help](#)
[Frequently Asked Questions](#)
[Policies](#)
[About UDDI](#)
[Contact Us](#)

Business detail
Oxford University Libraries - 6188F999-0D47-4E98-9CC9-68055C280B30

Contacts
The following contact details have been provided:

Name	Usage notes	Contact details
Matthew J. Dovey R&D Manager	Technical Contact	matthe... +44 181 65 St G Oxford
Reg Carr Director of University Library Services and Bodley Librarian		

Services
Click on service name to see the full service details:

Name	Description
OLIS	Union Library Catalogue
InterLibrary Loans	
Copyright Deposit	

UDDI Registry (2) - Anbietersicht

UDDI Services - Microsoft Internet Explorer

Adresse <http://uddi.tm.uni-karlsruhe.de/uddi/edit/frames.aspx>

User: MW SERVICES\ga
Role: Administ

Home Search Publish Coordinate Quick Help UDDI Services Help

Publish

My UDDI

- Providers
 - ProviderXYZ
 - NewsService
 - http://ws.edu/ws.asmx**
 - sip://123
- tModels
 - Universität Karlsruhe (TH)

My UDDI | ProviderXYZ | NewsService
<http://ws.edu/ws.asmx>

A binding represents an access point and one or more instances of the service that can be accessed at that point. You can publish technical descriptions of an instance, such as an interface specification, by adding an Instance Info.

Details Instance Info

Type and briefly describe an access point for this service. Specify the protocol (URL type) this binding supports. To publish technical information for an interface, such as parameters or a WSDL file, add an Instance Info.

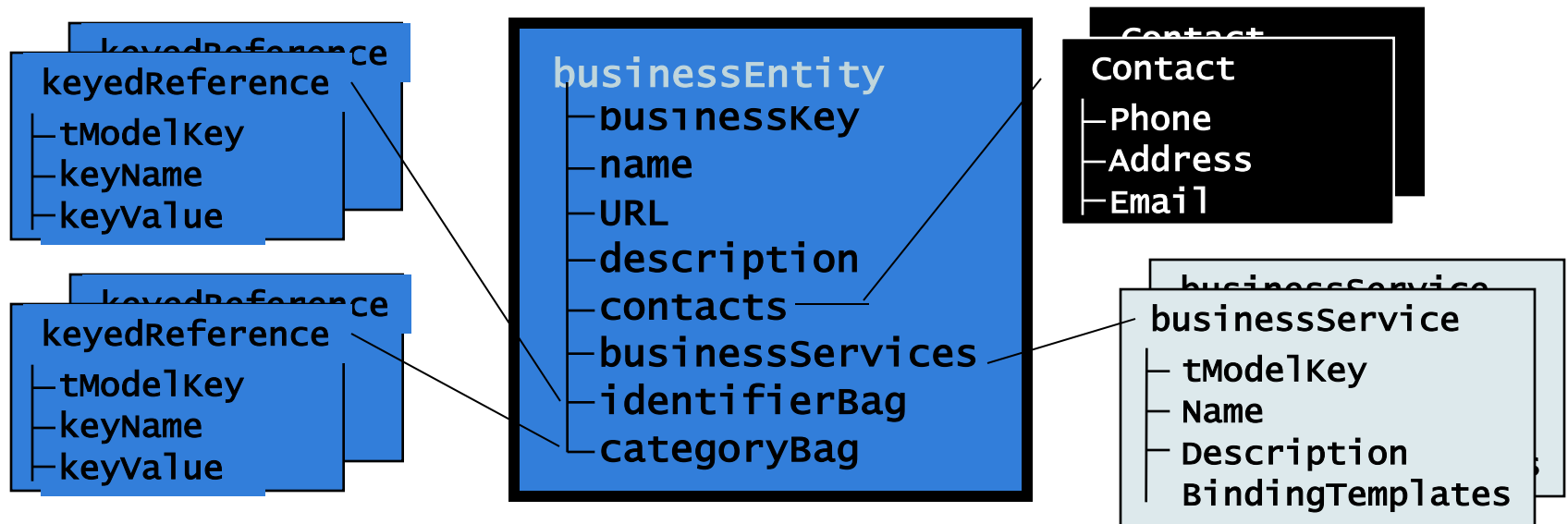
Binding Key:
041034f2-6659-423b-b568-ab26cbdb1483

Access Point	Actions
http://ws.edu/ws.asmx (http)	<input type="button" value="Edit"/>

Descriptions	Actions
0 record(s) found.	<input type="button" value="Add Description"/>

UDDI Registry (3) - Datenmodell

- **businessEntity** – Ist der zentrale Datentyp, der alle Informationen zu einer Organisation oder Geschäftseinheit in UDDI registriert
 - Dient als “master container” für alle Beziehungen
 - Kann keine organisatorischen Hierarchien abbilden (in V1)
 - Nutzt **tModel** Konzept



- **tModel** – (UUID, Name, Beschreibung)

UDDI Registry (4) – Bsp-Business (1)



The screenshot shows a Microsoft Internet Explorer browser window with the address bar displaying the URL: `http://uddi.microsoft.com/discovery?businesskey=c134764e-77c3-4bde-88c5-7cc8c037b434`. The browser's menu bar includes File, Edit, View, Favorites, Tools, and Help. The main content area displays an XML document. The XML starts with a declaration: `<?xml version="1.0" encoding="utf-8" ?>`. The root element is `<businessEntity>`, which contains several attributes: `xmlns:xsd="http://www.w3.org/2001/XMLSchema"`, `xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"`, `businessKey="c134764e-77c3-4bde-88c5-7cc8c037b434"`, `operator="Microsoft Corporation"`, and `authorizedName="Dr.-Ing. Martin Gaedke"`. The `xmlns` attribute is set to `urn:uddi-org:api_v2`. Following the `<businessEntity>` opening tag, there are several child elements: `<discoveryURLs>`, `<name>` (with `xml:lang="en"`), `<description>` (with `xml:lang="en"`), `<contacts>`, `<businessServices>`, and `<categoryBag>`. The XML document ends with the closing tag `</businessEntity>`.

```
<?xml version="1.0" encoding="utf-8" ?>
- <businessEntity
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  businessKey="c134764e-77c3-4bde-88c5-7cc8c037b434"
  operator="Microsoft Corporation" authorizedName="Dr.-Ing.
  Martin Gaedke" xmlns="urn:uddi-org:api_v2">
+ <discoveryURLs>
  <name xml:lang="en">WebEngineering.org
    Community</name>
  <description xml:lang="en">WebEngineering.org Community
    provides a forum for researchers, teachers, practitioners,
    managers and others interested in latest information on
    the young discipline Web Engineering.</description>
+ <contacts>
+ <businessServices>
+ <categoryBag>
  </businessEntity>
```

UDDI Registry (4) – Bsp-Business (2)

```
+ <discoveryUKLS>
  <name xml:lang="en">WebEngineering.org
    Community</name>
  <description xml:lang="en">WebEngineering.org Community
    provides a forum for researchers, teachers, practitioners,
    managers and others interested in latest information on
    the young discipline Web Engineering.</description>
- <contacts>
  - <contact useType="">
    <description xml:lang="en" />
    <personName>Dr.-Ing. Martin Gaedke</personName>
    <phone useType="">+49 (721) 6088076</phone>
    <email
      useType="">gaedke@webengineering.org</email>
  - <address sortCode="" useType="General mailing
    address">
    <addressLine>University of
      Karlsruhe</addressLine>
    <addressLine>Institute of
      Telematics</addressLine>
    <addressLine>Postfach 6980, Zirkel
      2</addressLine>
    <addressLine>D-76128 Karlsruhe</addressLine>
    <addressLine>Germany</addressLine>
    </address>
  </contact>
</contacts>
+ <businessServices>
+ <categoryBag>
```



UDDI Registry (4) – Bsp-Business (3)

```
us">WebEngineering.org</name>
- <bindingTemplates>
- <bindingTemplate bindingKey="5cec6be9-0fb9-
45bb-b186-ec172fb1e2f6"
serviceKey="536d76f6-47a3-4faa-baa6-
7b3464bfc253">
<description xml:lang="en">The
WebEngineering.org Site assists you in
endeavouring, teaching, or applying the
Web Engineering discipline by providing a
constantly updated suite of system
software, development tools, software
architectures, process models, research
approaches, liter</description>
<accessPoint
URLType="http">http://www.webengineering.org<
- <tModelInstanceDetails>
<tModelInstanceInfo
tModelKey="uuid:4cec1cef-1f68-4b23-
8cb7-8baa763aeb89" />
<tModelInstanceInfo
tModelKey="uuid:a9618442-aec4-4b0d-
9715-826b733e25da" />
</tModelInstanceDetails>
</bindingTemplate>
</bindingTemplates>
</businessService>
```

UDDI Registry (4) – Bsp-Business (4)

```
+ <businessServices>
- <categoryBag>
  <keyedReference tModelKey="uuid:c0b9fe13-179f-413d-
    8a5b-5004db8e5bb2" keyName="Scientific Research
    and Development Services" keyValue="5417" />
  <keyedReference tModelKey="uuid:297aaa47-2de3-4454-
    a04a-cf38e889d0c4" keyName="Karlsruhe"
    keyValue="504970" />
  <keyedReference tModelKey="uuid:db77450d-9fa8-45d4-
    a7bc-04411d14e384" keyName="Software engineering"
    keyValue="811115" />
  <keyedReference tModelKey="uuid:db77450d-9fa8-45d4-
    a7bc-04411d14e384" keyName="Internet services"
    keyValue="811121" />
  <keyedReference tModelKey="uuid:4e49a8d6-d5a2-4fc2-
    93a0-0411d8d19e88" keyName="Baden-Württemberg"
    keyValue="DE-BW" />
  <keyedReference tModelKey="uuid:297aaa47-2de3-4454-
    a04a-cf38e889d0c4" keyName="Europe"
    keyValue="100010" />
  <keyedReference tModelKey="uuid:297aaa47-2de3-4454-
    a04a-cf38e889d0c4" keyName="Germany"
    keyValue="200088" />
  <keyedReference tModelKey="uuid:297aaa47-2de3-4454-
    a04a-cf38e889d0c4" keyName="Baden-Württemberg"
    keyValue="300429" />
</categoryBag>
</businessEntity>
```

Chapter 5

THE CLIENT



Clients in SOA

- Important properties:
 - Cooperation
 - Negotiation
 - Autonomy
 - Optimisation
 - Adapted to the presence and availability of many different services over the internet



Client vs. Service

- **Service** is about modeling actions and interactions.
 - **Client** is to use those actions to achieve the users goals.
-
- **Service** provides the tools (i.e. functionality)
 - **Client** decides what to do and how to do it (i.e. which service will be called for executing the functionality needed)



Requirements

- **Service-based:** The basic means of interaction between entities residing on the platform will be the concept of service. This means that each action has to be annotated with a service description and needs to be executable via a service-protocol.
- **Distributed:** In order to make full use of the advantages of service-based communication, the entities need to be spreaded in a system over different physical locations.
- **Knowledge-based:** To make Agents be able to make intelligent decisions, they need a powerful representation for knowledge within them which allows the evaluation of a situation and reasoning about it. Furthermore the service descriptions need to contain enough useful information for the agents to deliberate about them.



Chapter 6

UNIFORM ADDRESSING



Addressing Resources

- As with DNS and IP-Addresses...
- Goals:
 - It must be possible to identify resources
 - By Name
 - By Address resp. Location
 - Any resource in the Internet should be identified
 - Web pages, FTP-Resources, Mailboxes, Directories, interactive services
- Requirements: Identification mechanism should be
 - Extensible
 - Complete
 - Printable (to be represented as string of 7-bit characters)



Uniform Resource Identifier

- Uniform Resource Identifier (**URI**)
 - Generic term for all textual names/addresses
 - URI is URL or URN or URC
- Uniform Resource Locator (**URL**)
 - The set of URI schemes that have explicit instructions on how to access the resource over the Internet
- Uniform Resource Name (**URN**)
 - A URI that has an institutional commitment to availability, etc.
 - A particular scheme intended to identify resources
- Uniform Resource Characteristic (**URC**)
 - A URC provides Meta Information



Uniform Resource Identifier

- URI – Syntax for identifiers [RFC3986] **<URI>**
::= <scheme>":"<scheme-specific-part>
- **<scheme>**
name of the scheme
- **<scheme-specific-part>**
identifier in a format that is according to the scheme



Reserved Characters

- For all types of URIs the following Rules apply:
- The percent sign ("% ", ASCII 25 hex)
 - Escape character
- Hierarchical forms ("/", ASCII 2F hex)
 - Delimiting of substrings whose relationship is hierarchical
- Hash - fragment delimiter ("#", ASCII 23 hex)
 - Identifies a fragment in a resource
- Query Delimiter ("?", ASCII 3F hex)
 - To delimit the boundary between the URI of a query able object



Uniform Resource Locator

- URL – Scheme definition [RFC1738,3986]
 - explicit instructions on how to access ...
 - `<scheme> ::= "http" | "https" | "ftp" | "news" | "mailto" | "nntp" ...`
- Specific Part defined in a general format
 - `<scheme-specific-part> ::= ["//"] [user [":"password] "@" host [":"port] ["/"url-path]`
- Definitions are maintained by the Internet Assigned Numbers Authority (IANA)
- URLs can also be relative [RFC 1808,3986]



Example – HTTP URL

- HTTP URL
- `<scheme> ::= "http"`
- `<scheme-specific-part> ::= "://"<host>[":"<port>][<abs_path>]`
- `<host> = "[" (IPv6address | IPvFuture) "]" | <IPv4address> | <reg-name>`
- `<port> = *DIGIT`
- `<abs_path> = "/"[<path>][";"<params>]["?"<query>]["#"<fragment>]`
- `<path> = <fsegment> *("/" <segment>)`
- Example
 - `http://vsr.informatik.tu-chemnitz.de:8080/a/b?x=1#2345`
 - `http://www.secret.xyz/account/euro?add=100#FragId`



Uniform Resource Name

- URN – Scheme definition [RFC 1737, RFC 2141]
 - URNs serve as persistent, location-independent, resource identifiers
- `<scheme> ::= "urn"`
- `<scheme-specific-part> ::= <nid> ":" <nss>`
 - nid = Namespace Identifier
 - nss = Namespace Specific String
- E.g. `urn:schemas:httpmail:subject`



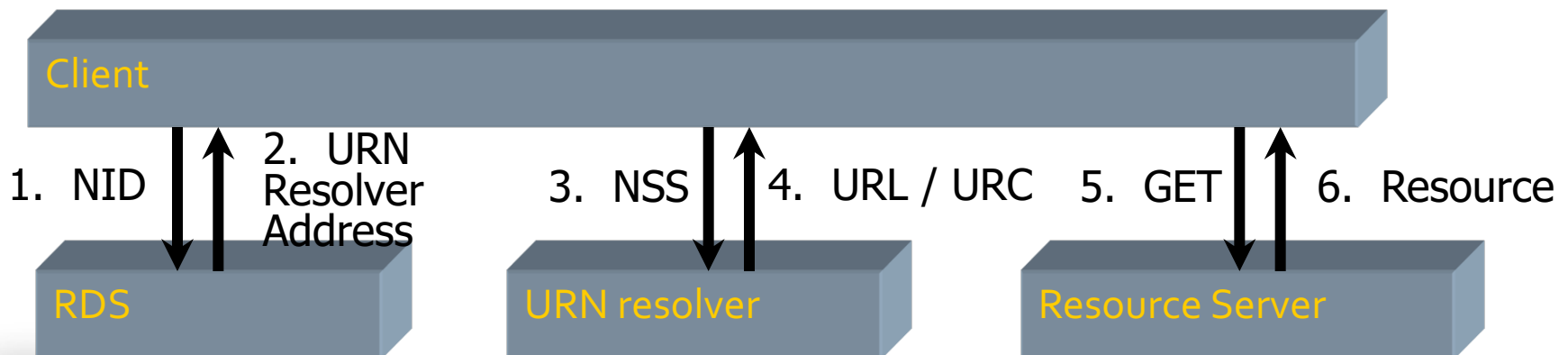
URN Properties

- Global scope and uniqueness
- Persistence
- Scalable
- Legacy support
- Extensible
- Independent
- Resolvable



URN - Resolution

- Infrastructure for URNs is still experimental
 - Resolver Discovery Service (RDS)
 - Name service, name resolution (URN resolver)
 - Result of the resolution is a URL or a URC
 - Cf. RFC 1737, 2276
 - $\langle \text{urn} \rangle ::= \text{"urn:"} \langle \text{nid} \rangle \text{" : " } \langle \text{nss} \rangle$



Comparison URN vs. URL

	URN	URL
Scope	Global	Global (abs. URL) Local (rel. URL)
Globally Unique	Yes	Yes (abs. URL) No (rel. URL)
Persistent	Yes	No
Scalable	Yes	Yes
Legacy Support	Yes	Limited
Resolution	Not yet determined	Partly using DNS



The core of all **bindings**
in the web

Chapter 7

HYPertext TRANSFER PROTOCOL (HTTP)



Section 1

THE BASICS



Hypertext Transfer Protocol

- **Hypertext Transfer Protocol (HTTP)** is used to exchange resources (such as websites, pictures, JavaScript, other MIME-types resources) between a user agent and a server following the Request/Response model.
 - Basic web resource transfer mechanism
 - IETF-Standard: RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1
 - Development: World Wide Web Consortium (W3C)
 - Is a transfer protocol, not a transport one
- **Protocol properties**
 - Exchange of Request/Response data based on TCP/IP
 - Stateless communication between user agent and server
 - Two message types: Request and Response
 - Messages are ASCII-encoded
 - Messages are used to realize methods: GET, POST, HEAD, etc.



HTTP Versions

- Early onset - Version 0.9
 - Only one command is supported: "GET" (no other approaches, like, for example, attributes)
 - Is not extensible, no versioning support
- Ideas of versioning and robustness of the Web
 - Use of version numbers for HTTP (RFC 2145, May 1997)
 - Robustness principle (backward compatibility support)
- May 1996 - Version HTTP/1.0 (RFC 1945)
 - Introduction of version numbers
 - Stateless protocol (TCP-Slow-Start problematics)
 - Support for extensions
 - HTTP/1.0 is still widespread
- June 1999 Version HTTP/1.1 (RFC 2616), supplemented by TLS (RFC 2817)
 - Enables persistent connections and proxy use
 - Many improvements in terms of performance, i.e. Request Pipelining or Multihomed Server
 - Addition of a secure connection possibility by means of Transport Layer Security (TLS)



Generic Approach

- Generic message structure enables extensions
- Important: The concept is used in all protocols that build upon HTTP

**Generic-Message = Start-Line
 *Header
 CRLF
 [Message-Body]**

Start-Line ::= Request-Line | Response-Line

Header ::= field-name ":" [field-value] CRLF
 field-name = token
 field-value = *(field-content | LWS)
 LWS = Linear White Space

Message-Body

If exists MUST be encoded

Presence signaled by header field Content-Length or Transfer-Encoding



HTTP-Request Message

- Message structure

<Method> " "<URI> " "<Protocol>

<Headers>

CRLF

[<Data>]

- *Method* ::= "GET" | "POST" | "HEAD" | ...
- *Protocol* ::= "HTTP/1.0" | "HTTP/1.1" | ...
- *Headers* ::= <hName> ":" <hValue>
hName – header name h (Attribut-Name)
hValue – Value of the value space of header h (Attribut-Wert)
- *Data* ::= <TEXT>

HTTP-Response Message

- Message structure

**<Protocol> " "<Status-Code> " "<Reason-Phrase>
<Headers>
CRLF
[<Data>]**

- *Protocol* ::= "HTTP/1.0" | "HTTP/1.1"
- *Status-Code* ::= DIGIT+ ; for use by automata
- *Reason-Phrase* ::= <TEXT> ; for use by human user
- *Headers* ::= <hName> ":" <hValue>
hName – Specific Header Name h (Attribut-Name)
hValue – Value of the value space of header h (Attribut-Wert)
- *Data* ::= <TEXT>

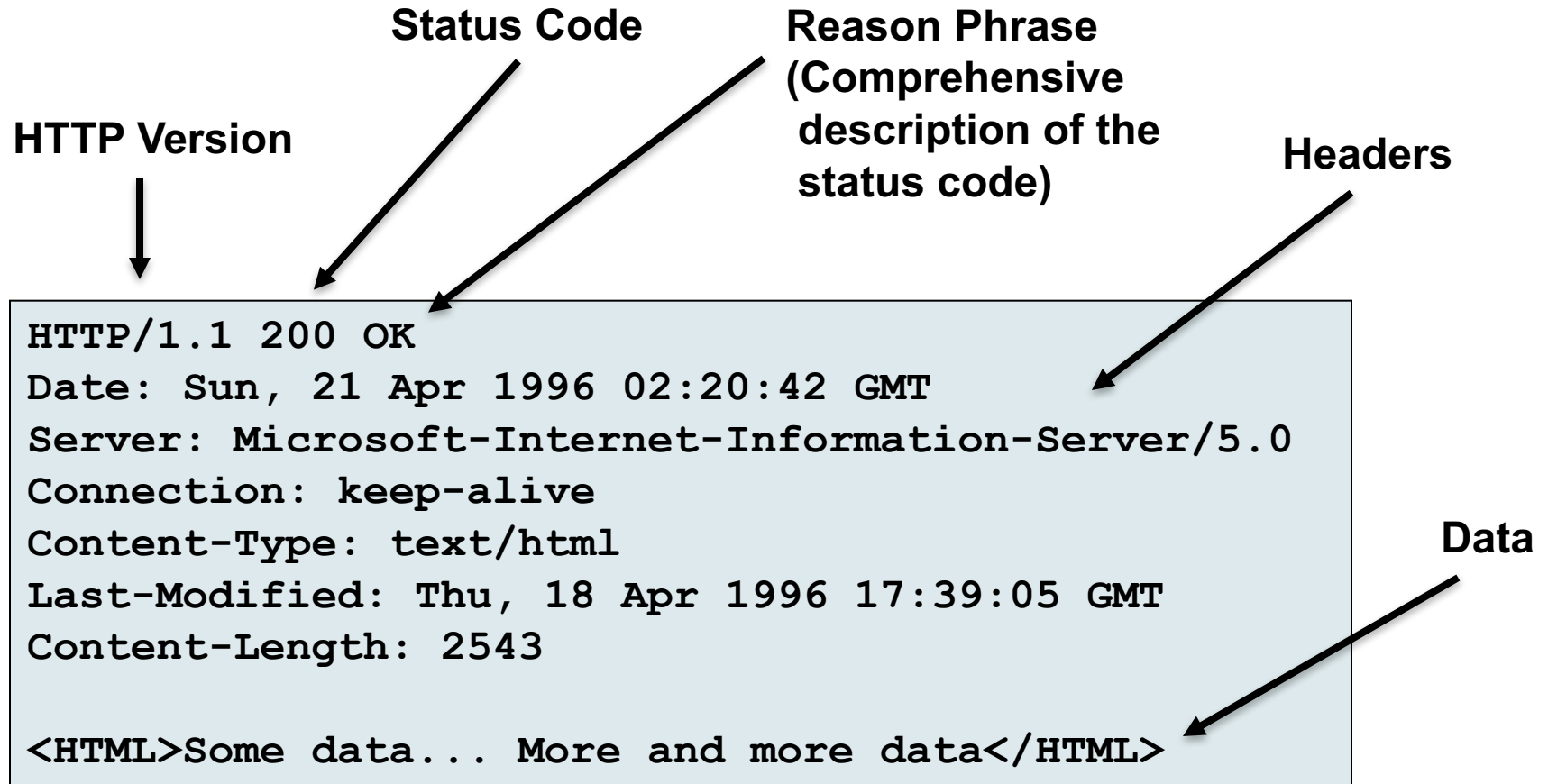
HTTP Request

Method	URI	HTTP Version	Headers
↓	↓	↓	↘
<pre>GET /default.asp HTTP/1.1 Accept: image/gif, image/x-bitmap, image/jpeg, */* Host: www.example.org User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95) Connection: Keep-Alive If-Modified-Since: Sunday, 17-Apr-96 04:32:58 GMT</pre>			
			↙
			↙

Blank line

Data – (if GET method, no data!)

HTTP Response



Typical HTTP Methods (1)

- Methods are applied in the context of HTTP Requests
 - For further detail on methods see RFC 2616
- **GET**
 - Deliver the resource addressed by the URI
- **POST**
 - Request to the server with respect to processing of encoded message body data (processing wrt. the URI provided in POST)
 - Enables among others:
 - Annotation of existing resources
 - Send (POST) data blocks to an application, like comment, user ID, pictures
 - Available (send data via messages) from a form
- **HEAD**
 - Like GET but without the Response Body
- **OPTIONS**
 - Request on information submission on communication options



Typical HTTP Methods (2)

■ PUT

- Resources encoded in the Body should be saved at the Request URI

■ DELETE

- Server should remove the resources connected to the Request URI

■ TRACE

- Methods for development support of the so-called application layer request loop-back
- All requests of user agents that the server gets should return to the user agent



Typical HTTP Headers (1)

- Repetition: Header entry (name-value pair) *hName:hValue*
 - The following examples are often used hNames
- **Content-Type**
 - Media type used
- **Expires**
 - Date/time from which the response is considered invalid
 - Important for caching!
- **Host**
 - Specifies Internet host and port number of the requested resource
 - Required for "multi-homed server" (HTTP/1.1)!
- **Last-Modified**
 - Date and time when the "variant" (object referenced by the Request-URI) was last modified
 - Important for caching!



Typical HTTP Headers (2)

■ Location

- Is set in the HTTP Response to notify the user agent of the new location of the requested Request-URI
- Very important concept in different protocols which build up on HTTP, for example, in the security area
- Is used for implementation of the so-called "Redirects"

■ Referrer

- Reference to the URI from which the user agent has posted the current Request URI
- Useful for maintenance/service tasks
 - Where do the users come from?
 - Logging, optimization, caching, user types
- Not used as a security mechanism

■ User-Agent

- Information about the user agent
- Important for personalization and internationalization

■ Numerous further attributes exist for use for different tasks



Typical HTTP Headers (3)

Code	Description
1XX	Information as intermediate response
2XX	Successful operations
200	OK
201	Created
3XX	Redirects
301	Moved Permanently
302	Moved Temporarily
4XX	Client Error
400	Bad Request – not understood
401	Unauthorized
403	Forbidden – not authorized
404	Not Found
5XX	Server Error