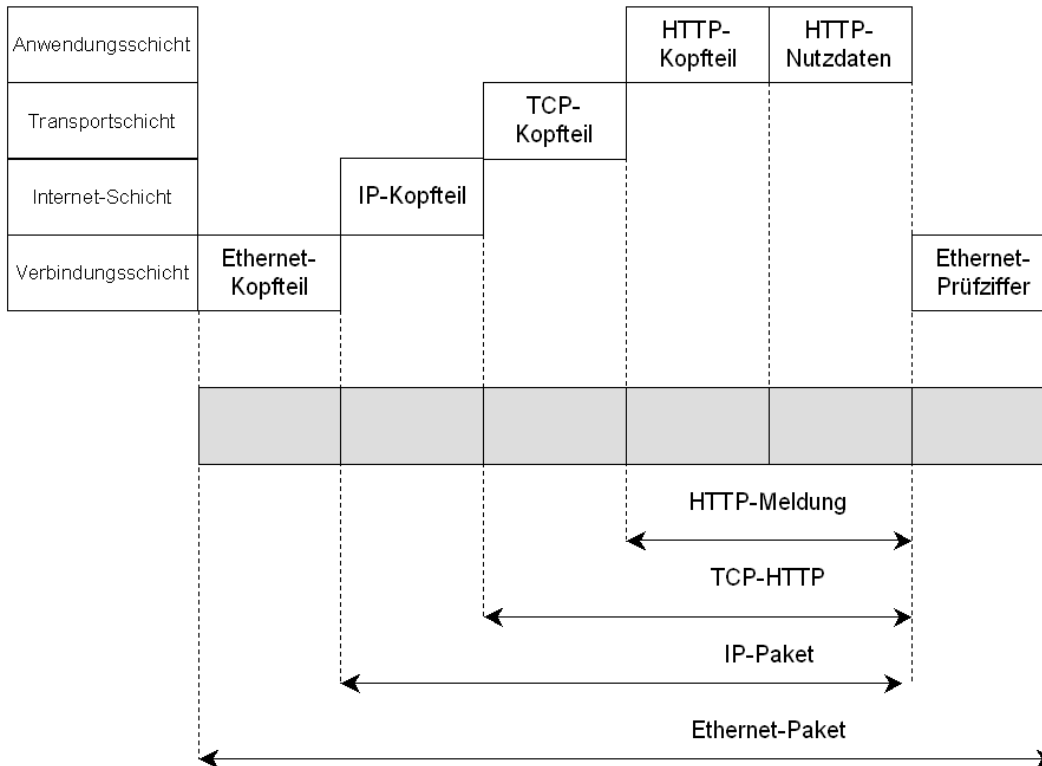


## 2 Grundlegende Funktionsweise eines HTTP-Servers

In diesem Abschnitt soll das Zusammenspiel zwischen der Transportschicht und der Anwendungsschicht am Beispiel des Protokolls HTTP erläutert werden. Im Laufe der zugehörigen Ausführungen wird beschrieben, wie dieses Protokoll arbeitet und welche Definitionen es für darauf aufsetzende Anwendungen vorgibt.



Das Protokoll HTTP (Abkürzung von engl.: hypertext transfer protocol) ist ein Protokoll der Anwendungsschicht und definiert die (grundlegende) Kommunikationsfunktionalität des WWW. HTTP verwendet auf der Transportschicht das Protokoll TCP. In der Praxis werden sowohl die Anwendungsprotokolle HTTP/1.0 (definiert durch den RFC 1945) als auch HTTP/1.1 definiert durch den RFC 2616) nebeneinander eingesetzt.

Exkurs: Vielfach werden die Begriffe WWW, Web und Internet mehr oder minder synonym verwendet. Dies ist jedoch nicht korrekt. Wie bereits beschrieben wurde, hat sich das Internet seit den 1960er Jahren kontinuierlich weiterentwickelt. Viele der Kernprotokolle sind seit der Frühzeit des Internets weitgehend unverändert im Einsatz (zum Beispiel: Telnet seit 1972, FTP erste offizielle Version 1973, TCP seit 1980, SMTP seit 1982).

Der Begriff World Wide Web (WWW) wurde 1989 von Tim Berners-Lee geprägt. Die erste Version der WWW-Spezifikation wurde 1991 vom CERN (Abkürzung von: Conseil Européen pour la Recherche Nucléaire, europäisches Forschungszentrum für Teilchenphysik) freigegeben. Die Kerndefinitionen des WWW sind die Spezifikationen von HTML als Präsentationssprache und HTTP als Anwendungsprotokoll. Das WWW (oder Web) ist somit eine Anwendung des Internets und bezeichnet jene Dienste, die in HTML (oder einer Weiterentwicklung von HTML) dargestellt und über HTTP abrufbar sind.

Wenn Sie beispielsweise von Ihrem Webbrowser auf die Startseite der Suchmaschine Google gelangen möchten, benötigen Sie zunächst eine Möglichkeit, diese zu adressieren. HTTP definiert für diesen Zweck einen eigenen Adressierungsmechanismus, der URI genannt wird.

Ein URI (Abkürzung von engl.: uniform resource identifier) ist entweder ein Verweis auf einen Ort, an dem ein bestimmtes Dokument gespeichert ist (engl.: uniform resource locator, abgekürzt: URL) oder ein symbolischer Name für eine prinzipiell beliebige Ressource (engl.: uniform resource name, abgekürzt: URN). Die meisten heute verwendeten URIs sind URLs.

Der URL für die Suchmaschine Google ist beispielsweise <http://www.goggle.com/index.html>. Der Name des angeforderten Dokuments „index.html“ ist für die Startseite optional. Ein Webbrowser kann aus diesem URL ermitteln, dass zu dem HTTP-Server, der auf dem Rechner [www.google.com](http://www.google.com) läuft, eine TCP-Verbindung geöffnet werden soll (alle HTTP-Verbindungen basieren auf TCP). Für eine TCP-Verbindung ist eine Dienstnummer notwendig. Da in dem URL keine Dienstnummer angegeben war, wird automatisch die standardmäßige Dienstnummer für HTTP-Server 80 angenommen.

Ein Beispiel für die explizite Angabe einer Dienstnummer in einem URL:

<http://www.google.com:80/index.html>

Bevor die Startseite angefordert werden kann, muss der Browser mithilfe von DNS allerdings zunächst noch aus dem Rechnernamen die zugehörige IP-Adresse ermitteln. Die ermittelte IP-Adresse ist beispielweise 216.239.39.100. Der Browser kann nun eine TCP-Verbindung zu der ermittelten IP-Adresse und der Dienstnummer 80 öffnen (Socket-Adressierung) und ruft zu diesem Zweck eine Funktion der Transportschicht auf.

Die Transportschicht und die darunter liegende Vermittlungsschicht kümmern sich um den Verbindungsaufbau. Hierzu wird unter anderem über ARP die Ethernet-Adresse des Rechners ermittelt, an den die Pakete weitergeleitet werden (in der Regel ein Router). Diese Vorgänge sind für die Anwendungsschicht jedoch unsichtbar; sie wird erst verständigt, sobald die TCP-Verbindung zum genannten Serverprogramm mit der Dienstnummer 80 hergestellt wurde.

## 2.1 Ablauf von HTTP-Anfragen

Nun kann im nächsten Schritt eine HTTP-Anfrage an diesen Server geschickt werden, um die genannte HTML-Seite (index.html) anzufordern. Diese Anfrage ist eine HTTP-Meldung, deren Struktur durch die Spezifikation von HTTP definiert wird.

Jede HTTP-Meldung (engl.: HTTP message) besteht aus einem Kopfteil (engl.: header), einer Trennzeile (engl.: separator line) und einem Nutzdatenteil (engl.: body). Der Kopfteil der Meldung enthält die Steuerinformation.

Eine HTTP-Meldung ist entweder eine HTTP-Anfrage oder eine HTTP-Antwort. Eine HTTP-Anfrage eines HTTP-Clients wird durch eine HTTP-Antwort eines HTTP-Servers beantwortet. Die Abb. zeigt ein Beispiel einer HTTP-Anfrage und einer HTTP-Antwort.

Eine HTTP-Anfrage (engl.: HTTP request) besteht aus einer Kopfzeile, optionalen Anfrageparametern (engl.: request header fields) und einem Nutzdatenteil, der auch leer sein kann. Die Kopfzeile der HTTP-Anfrage enthält die HTTP-Methode (engl.: HTTP method), einen Bezeichner für die angeforderte Ressource und die Bezeichnung der verwendeten Version des HTTP-Protokolls.

### Anfrage von Client an Server:

```
GET /index.html HTTP/1.0
host: www.google.com
```

### Antwort von Server an Client:

```
HTTP/1.0 200 OK
Date: Tue, 26 May 2005 13:47:53 GMT
Content-Length: 4255
Content-Type: text-html
```

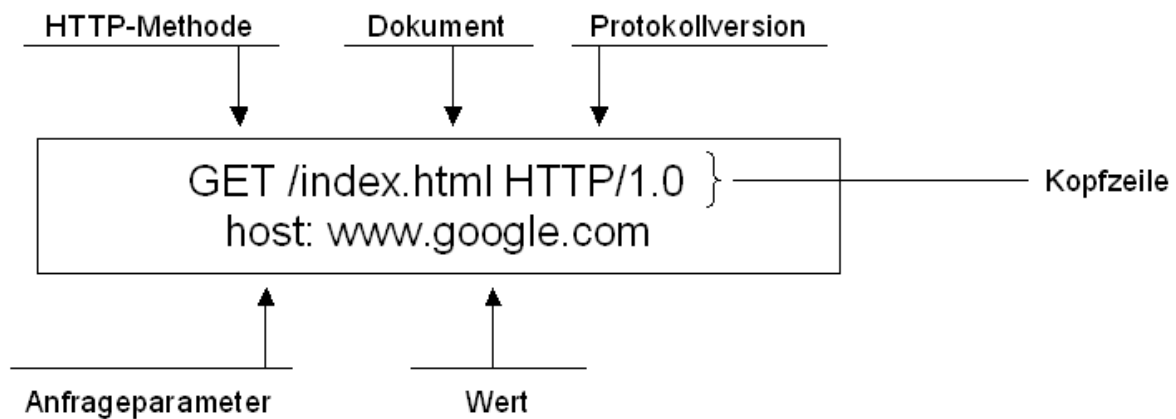
```
<HTML>
  <HEAD>
    <TITLE>Google</TITLE>
```

...

```
</HTML>
```

HTTP definiert mehrere „Methoden“, die den konkreten Diensten eines Webservers entsprechen. Die HTTP-Methode, um ein Dokument anzufordern, ist die Methode GET. Abhängig von der verwendeten HTTP-Methode können unterschiedliche Anfrageparameter angegeben werden. Jeder Anfrageparameter besteht aus einer Zeile, die mit der Bezeichnung des Anfrageparameters gefolgt von einem Doppelpunkt beginnt, und mit den Wert des Parameters abgeschlossen wird. Im weiteren

Verlauf dieses Abschnitts wird noch genauer auf HTTP-Methoden eingegangen. Die folgende Abbildung zeigt die HTTP-Anfrage des obigen Beispiels in größerem Detail.



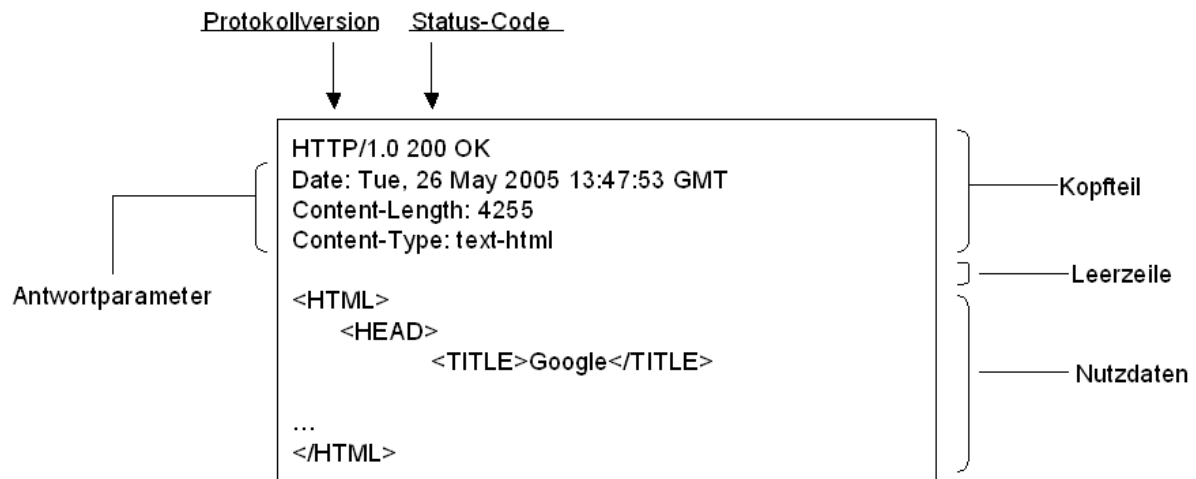
In diesem konkreten Beispiel ist der Parameter `host` (der den Zielrechner der Anfrage angibt) der einzige Anfrageparameter. Weitere Beispiele für Anfrageparameter sind `if-modified-since` oder `range`. Durch den erstgenannten Parameter kann ein Dokument unter der Bedingung transferiert werden, dass dieses nach dem im Parameterwert angegebenen Zeitpunkt verändert wurde. Durch den `range`-Parameter kann ein Teil (ein Ausschnitt) eines Dokuments transferiert werden.

Eine HTTP-Antwort (engl.: HTTP reply) besteht aus einer Kopfzeile, optionalen Antwortparametern (engl.: reply header field) und einem Nutzdatenteil, der auch leer sein kann. Die Kopfzeile der HTTP-Antwort enthält die Protokollversion, die der HTTP-Server unterstützt, gefolgt von einem Status-Code und einer Status-Meldung.

Der Status-Code der Antwort teilt dem Clienten mit, ob die gewünschte Operation durch den Server ausgeführt werden konnte, und gibt dem Clienten Aufschluss über die anschließend einzuleitenden Operationen. HTTP unterscheidet im Wesentlichen zwischen Erfolgsmeldungen, Warnungen, fehlerhaften Anfragen und serverseitigen Fehlern. Die nachfolgende Tabelle zeigt die wichtigsten im HTTP definierten Status-Codes mit einer Kurzbeschreibung.

	Status-Code	Status-Meldung	Beschreibung
Erfolgsmeldungen:	200	OK	Erfolgreiche Ausführung
Warnungen:	301	Moved Permanently	Ressource wurde auf Dauer an einen anderen Ort verschoben
	302	Moved Temporarily	Ressource wurde temporär an einen anderen Ort verschoben
	304	Not Modified	Ressource wurde nicht verändert (bei bedingter Anfrage)
Ungültige Anfragen:	400	Invalid Request	Ungültige Anfragesyntax
	401	Unauthorized	Benutzer ist nicht berechtigt
	402	Payment Required	Zahlung notwendig
	403	Forbidden	Zugriff ist nicht möglich
	404	Not found	Angefragte Ressource konnte nicht gefunden werden
Fehler des Servers:	500	Internal Server Error	Fehler auf Serverseite

In der HTTP-Antwort werden als Antwortparameter `Date`, `Content-Length` und `Content-Type` verwendet. Der erstgenannte Parameter enthält den Zeitpunkt, an dem die Anfrage beantwortet wurde. Der Antwortparameter `Content-Length` besagt, wie viele Bytes der Nutzdatenteil enthält und `Content-Type` teilt dem Clienten mit, welchen Inhalt der Nutzdatenteil besitzt. Im konkreten Beispiel ist dies ein HTML-Text (der Wert ist `text/html`).



Die gültigen Werte für den Content-Type-Parameter sind durch den MIME-Standard (Abkürzung von engl.: multipurpose internet mail extension) definiert, durch den beispielsweise Text-, Grafik- und anwendungsspezifische Datenformate beschrieben werden. Anhand der MIME-Formatangaben kann ein Webbrowser erkennen, wie die empfangenen Nutzdaten angezeigt werden sollen. Als wesentliche Optionen kommen hier die Anzeige durch den Webbrowser und die Anzeige durch ein browserexternes Hilfsprogramm (engl.: external viewer) in Betracht.

Beispiele für MIME-Formatangaben sind: text/html, text/plain, image/png, image/jpeg oder application/pdf.

## 2.2 HTTP-Methoden

Wie bereits erwähnt, definiert HTTP mehrere „Methoden“, die den konkreten Diensten eines Webserver entsprechen, also angeben, welche Aufgaben ein HTTP-Server erfüllen muss, um der Spezifikation von HTTP zu entsprechen.

Die mit Abstand am häufigsten verwendete HTTP-Methode ist die Methode GET, die zur Anforderung von Dokumenten dient und im obigen Beispiel bereits vorgestellt wurde. HTTP ist aber keinesfalls auf (Text-)Dokumente beschränkt und verwendet den allgemeinen Begriff der Ressource, um anzudeuten, dass Dateien aller Art sowie auch beliebige Internetdienste durch den gleichen Mechanismus angefordert werden können.

Die folgende Tabelle zeigt einen Überblick über die wichtigsten Methoden, die in HTTP/1.0 und HTTP/1.1 definiert sind.

HTTP-Methode	Kurzbeschreibung
GET	Anforderung einer Ressource
HEAD	Anfrage der Steuerinformation ohne Nutzdaten
PUT	Abspeichern einer Ressource auf dem Server
POST	Übertragung von Benutzerdaten vom Klienten zum Server
DELETE	Löschen einer Ressource auf dem Server

Aus dieser Übersicht ist erkennbar, dass HTTP nicht nur Methoden zur Abfrage von Ressourcen, sondern auch Methoden zum Abspeichern und Löschen anbietet. Diese Funktionen sind allerdings über die gängigen Browser nicht erreichbar beziehungsweise sind in den weitaus meisten HTTP-Servern abgestellt. HTTP spezifiziert für das Einspielen von Web-Ressourcen auf einen Server nur die Grundfunktionen. Die für gemeinschaftliches Arbeiten im Web notwendigen Erweiterungen von HTTP werden beispielsweise durch WebDAV (Abkürzung von engl.: web distributed authoring and versioning, Spezifikation in RFC 2518) definiert.

## **2.3 HTTP als verbindungsloses Protokoll**

HTTP/1.0 ist ein verbindungsloses Protokoll: Sobald der Client die Antwort auf die Anfrage empfängt, wird auch die Verbindung beendet. Bei einer neuerlichen Anfrage wird wiederum eine neue Verbindung hergestellt. Wenn beispielsweise eine HTML-Seite mit zehn Grafiken angefordert wird, so bedeutet dies, dass durch die erste Anforderung zunächst die HTML-Seite übertragen wird, in einem weiteren Schritt untersucht der Client diese Seite und extrahiert die URLs für die eingebetteten Grafiken. Für jede einzelne dieser Grafiken wird sodann eine neue Anfrage (mit Verbindungsaufbau und -abbau) an den Server gestellt, um diese Grafiken zum Clienten zu übertragen. Da in den meisten Fällen die eingebetteten Grafiken vom gleichen Server bezogen werden, wird im genannten Beispiel für das Anzeigen eines einzigen HTML-Dokuments mit Grafiken elfmal eine Verbindung zum gleichen Server auf- und abgebaut. Dies ist aus Sicht des Antwortzeitverhaltens und der Netzwerklast nicht wünschenswert.

In HTTP/1.1 ist der Verbindungsabbau nach Empfang der Antwortmeldung nicht mehr zwingend vorgeschrieben, da in dieser Version des Protokolls die Behandlung von so genannten persistenten Verbindungen (engl.: persistent connection) definiert wird. Dadurch werden Verbindungen erst nach einer gewissen Zeitspanne ohne weitere Anfragen abgebaut. Während die Verbindung besteht, können auf dem (virtuellen) Duplexkanal mehrere Anfragen und Antworten ausgetauscht werden, wodurch der Mehraufwand des wiederholten Verbindungsauf- und -abbaus entfällt.

Das Protokoll WAP (Abkürzung von engl.: wireless application protocoll) ist eine komprimierte Form von HTTP für mobile Endgeräte (wie beispielweise Mobiltelefone).



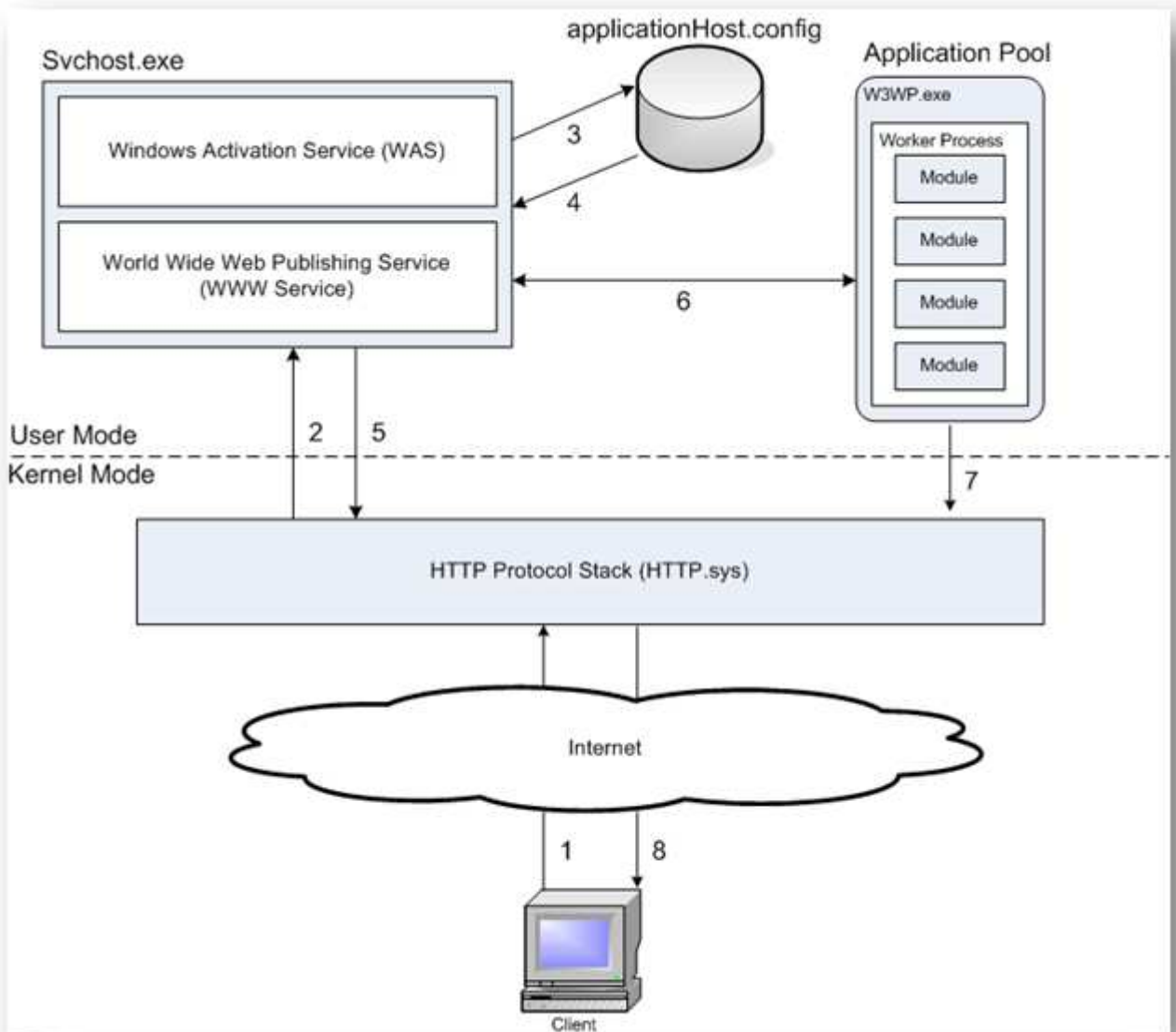
### 3 Architektur von IIS 7.5

IIS 7.5 ist hochmodular aufgebaut; allein 40 Installationskomponenten können im Setup unterschieden werden. Neu ist der „Windows Process Activation Service“.



Grafik: Microsoft

Statt bisher zentral (in einer XML-basierenden „**Metabase**“) werden nun die Konfigurationsdaten des IIS und der Anwendungen in einer hierarchisch gegliederten Struktur bestehend aus mehreren XML-Dateien gespeichert. Im Verzeichnis `%system%\inet_srv\config` findet man zunächst die IIS-Konfigurationsdatei auf Serverebene, `ApplicationHost.config`. Es kann dann noch für jede Webanwendung Web.config-Files geben, die in der Ordnerstruktur der Webanwendung gespeichert sind.



Im Kernel Mode läuft der **HTTP-Protokollstack (http.sys)**, ein Listener, der auf HTTP- und HTTPS-Anfragen hört. Das **World Wide Web Publishing Service (W3SVC)** ist ein http-Listener-Adapter, der die Kommunikation zwischen http.sys und dem **Windows Process Activation Service (WPAS, WAS)** verwaltet. WAS verwaltet die Arbeitsprozesse: Es startet und stoppt Anwendungspools und überwacht die Integrität aller Arbeitsprozesse. Die Konfiguration wird vom Configuration Store applicationHost.config bezogen.

IIS 7.5 unterstützt zwei Pipeline-Modi für ASP.NET-Anwendungen:

- Klassischer Modus: In diesem Modus wird .NET-Framework als ISAPI ausgeführt. So läuft .NET-Framework 1.1 ohnehin nur als ISAPI.
- Integrierter Modus: In diesem Modus greifen .NET-Module direkt auf die Kernkomponenten von IIS 7.5 zu; sie agieren damit genauso wie IIS-Komponenten.