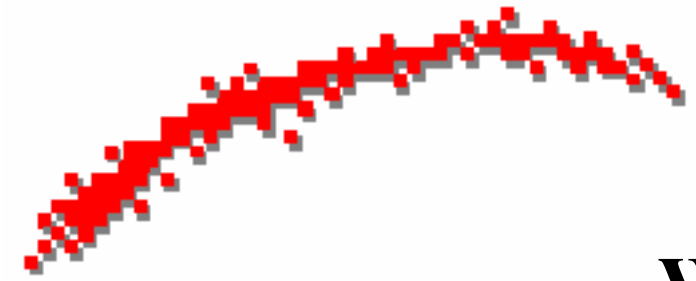
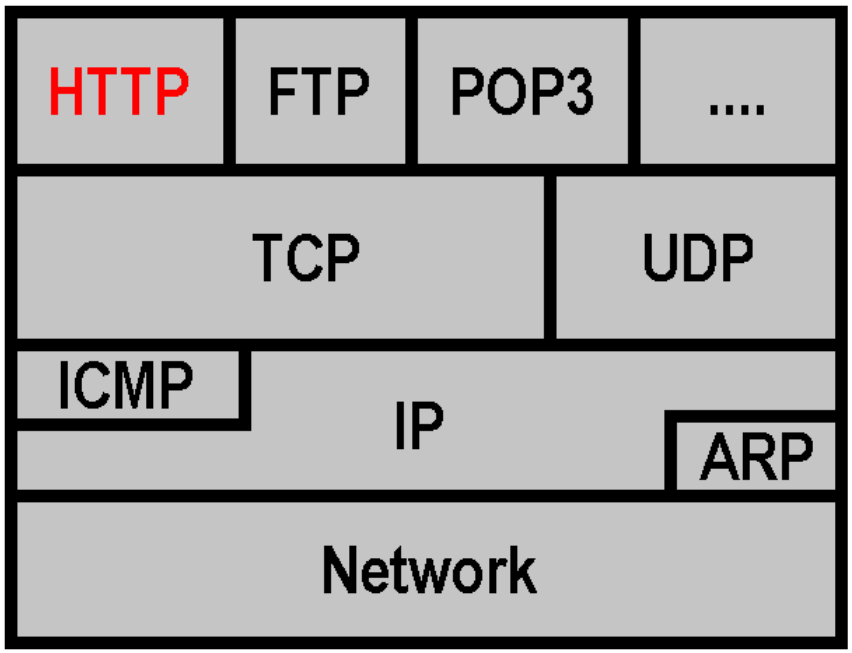


Web Server for Embedded Systems

KLAUS-D. WALTER
SSV EMBEDDED SYSTEMS
HEISTERBERGALLEE 72
D-30453 HANNOVER
WWW.SSV-EMBEDDED.DE

TCP/IP Protocol Basics

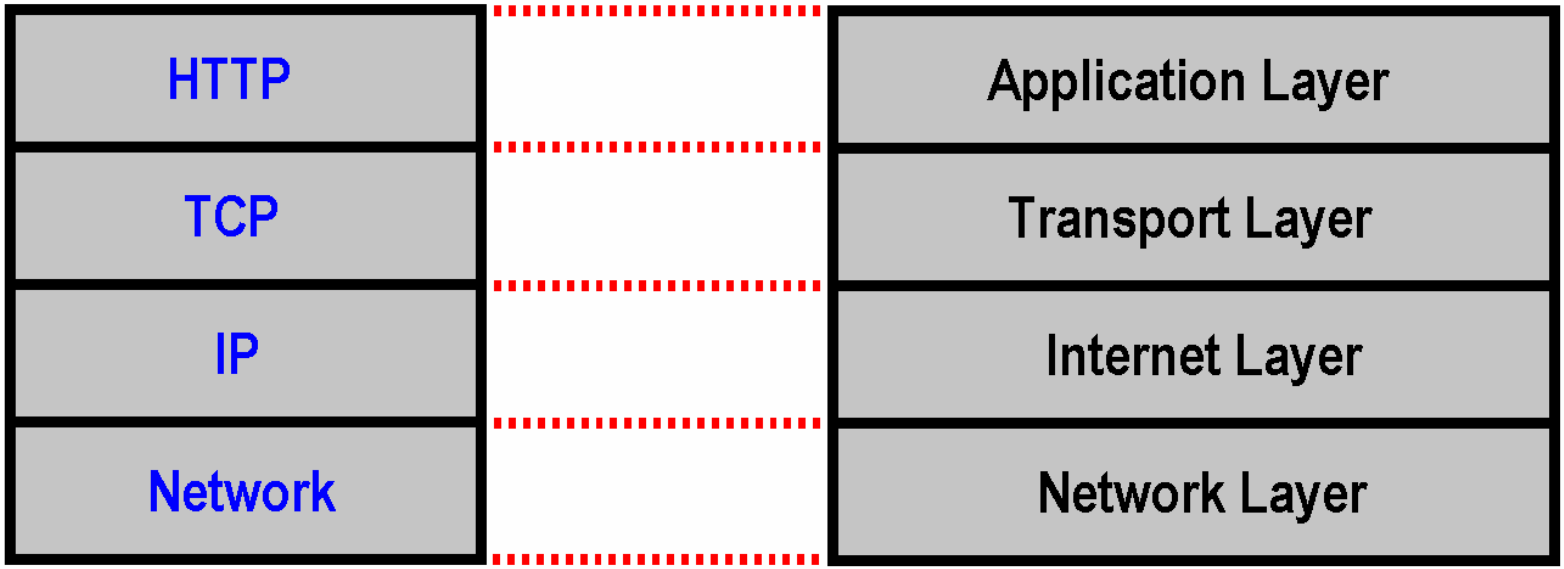
* Networking technology is organized in layers. The base is the OSI Reference Model. Each layer only communicates with the layers immediately above or below it.



* HTTP define the rules by which Web browers, Web servers, proxies, and other Web systems establish and maintain communications with each other.

TCP/IP Protocol Basics

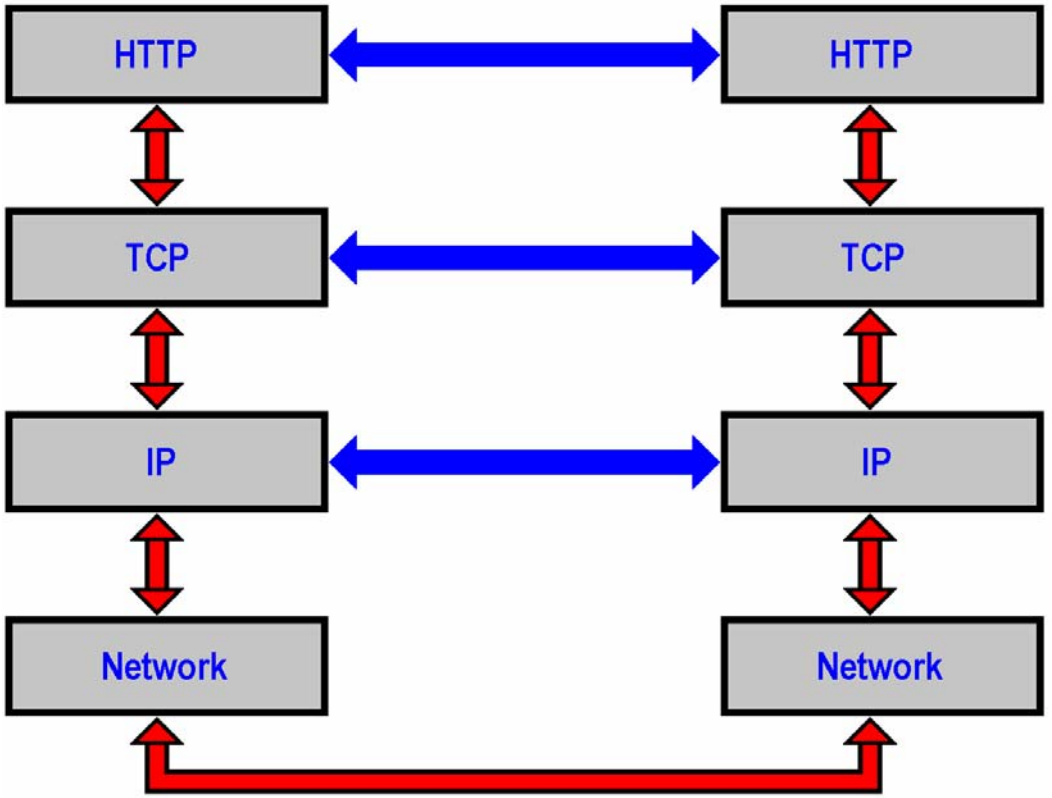
* The OSI Reference Model is based on seven layers. HTTP (and a TCP/IP protocol stack) is using only four protocol layers within a computer system.



* The lowest layer is the **network layer**. The protocol layer above the network layer is the **Internet Protocol** (IP). Next is the **Transmission Control Protocol** (TCP). The final protocol layer is **HTTP**, a TCP/IP application protocol.

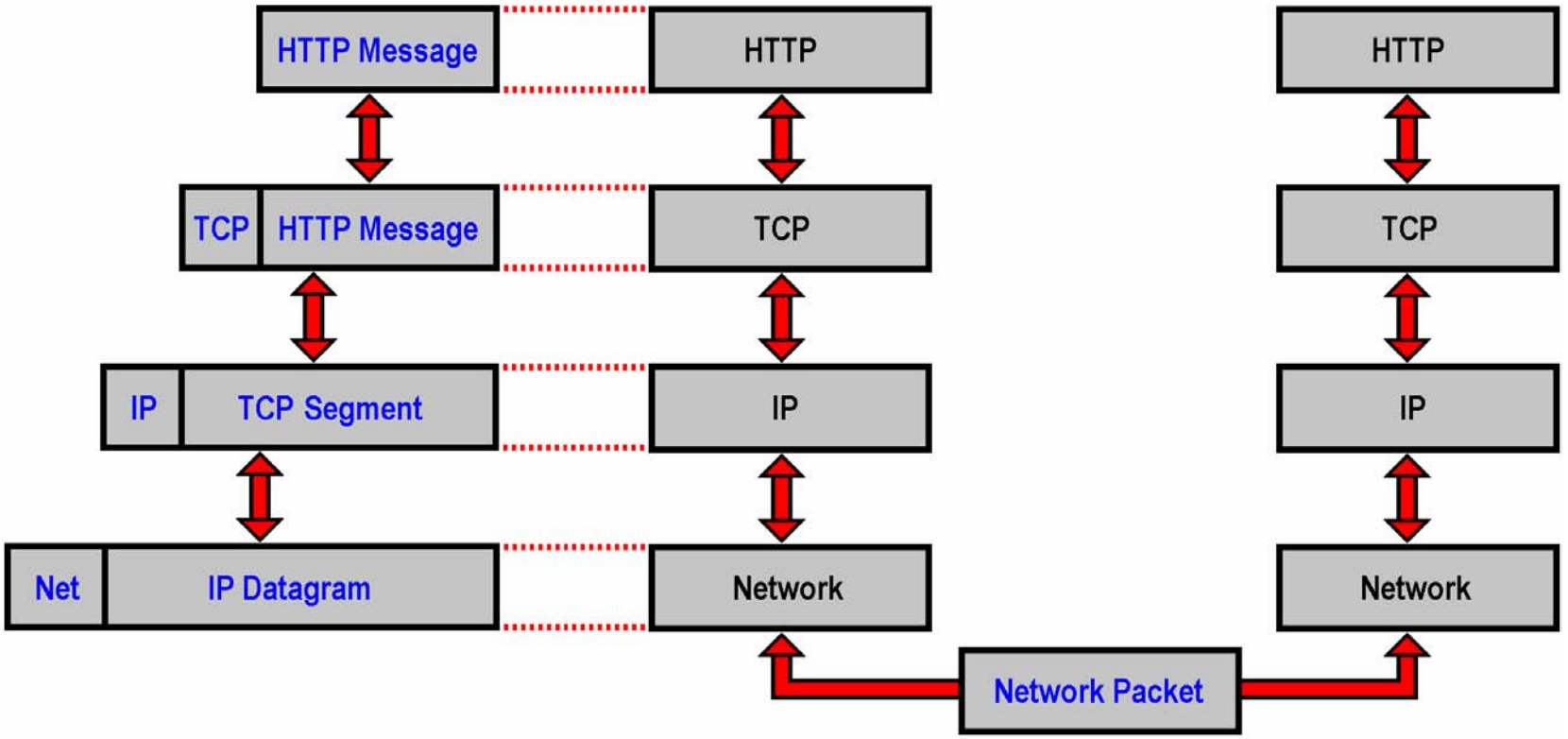
TCP/IP Protocol Basics

* There is a logical and physical communication between each protocol layer. HTTP reads or writes data from/to TCP. That protocol interacts directly with IP. IP interacts with the protocol controlling the network layer (i.e. Ethernet or PPP).



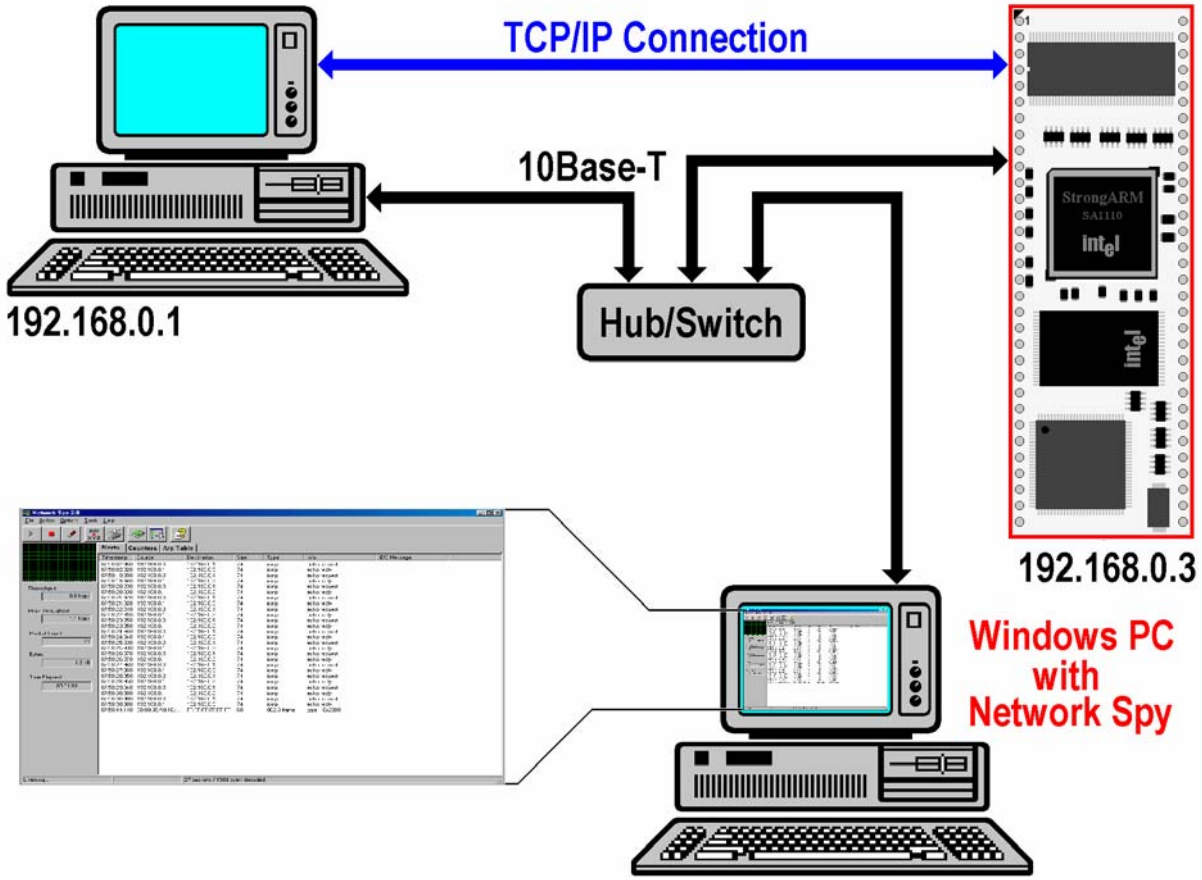
TCP/IP Protocol Basics

* A protocol is using a own header and a own name for the unit of data it sends and receives. Each protocol layer adds and removes it´s own specific information.



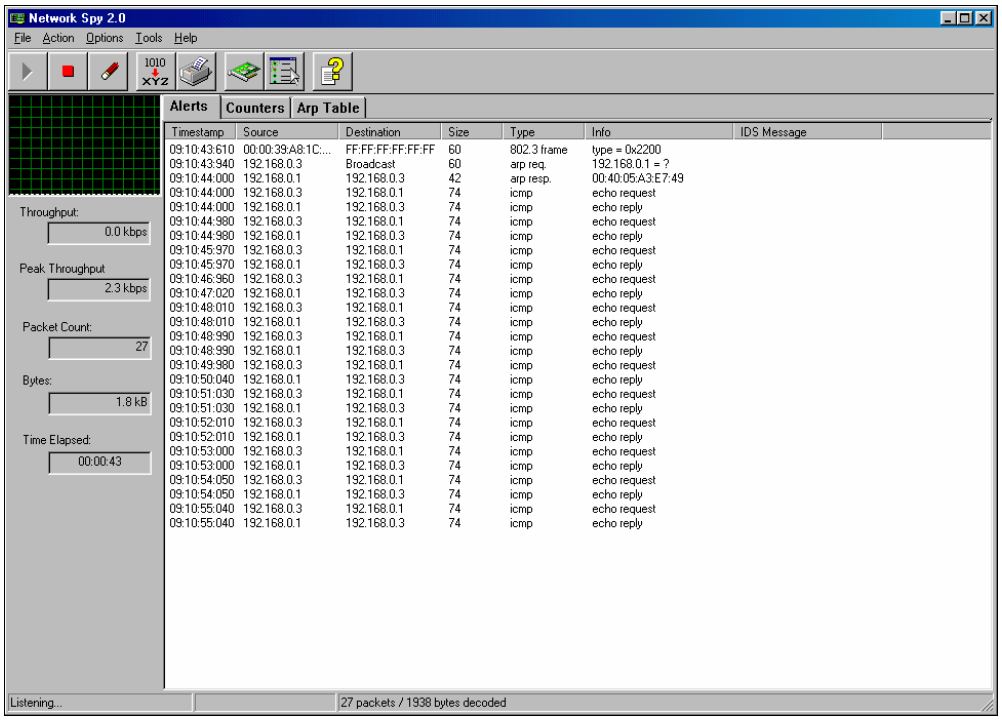
Debugging TCP/IP Connections

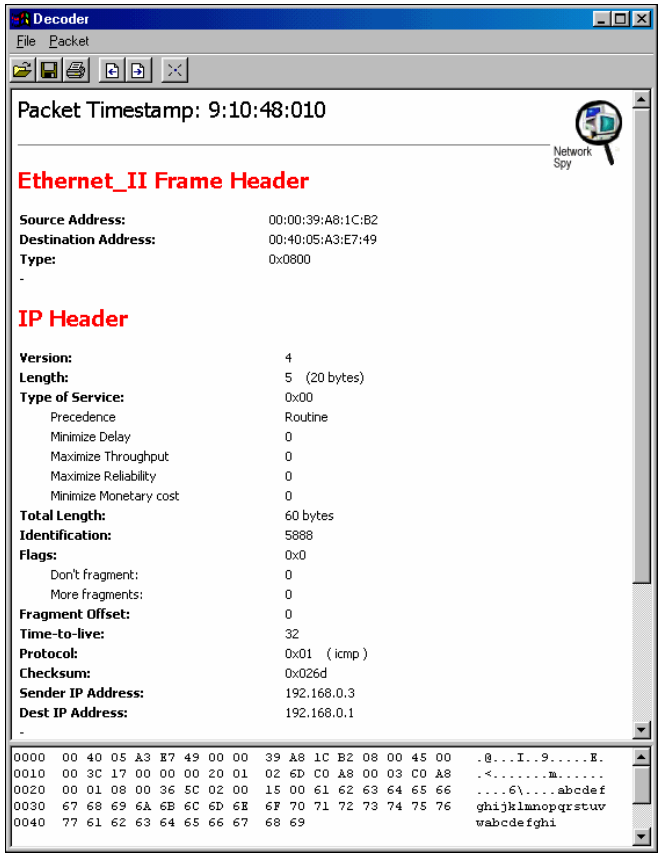
* Use a network monitor program (i.e. Etherrnet sniffers) . This kind of programs enables you to capture and examine network packets.



Debugging TCP/IP Connections

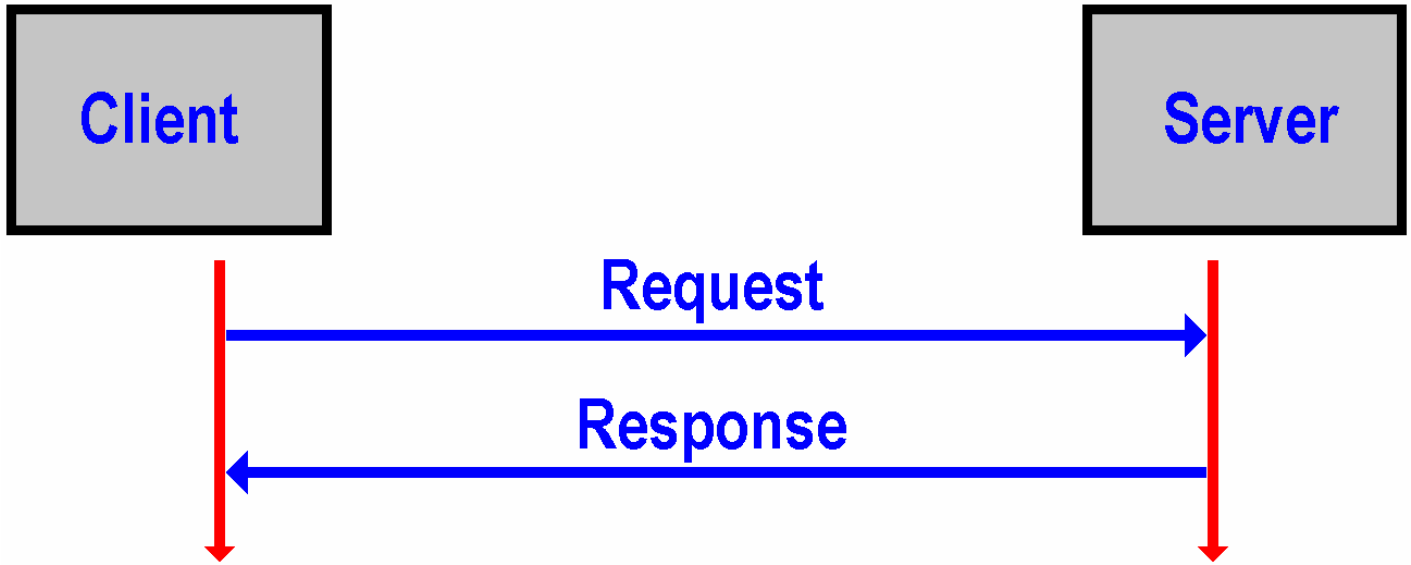
* Most network monitor programs comes complete with a suite of TCP/IP protocol parsers and decoders. With the aid of these tools, captured traffic can be analyzed.





HTTP Protocol Basics

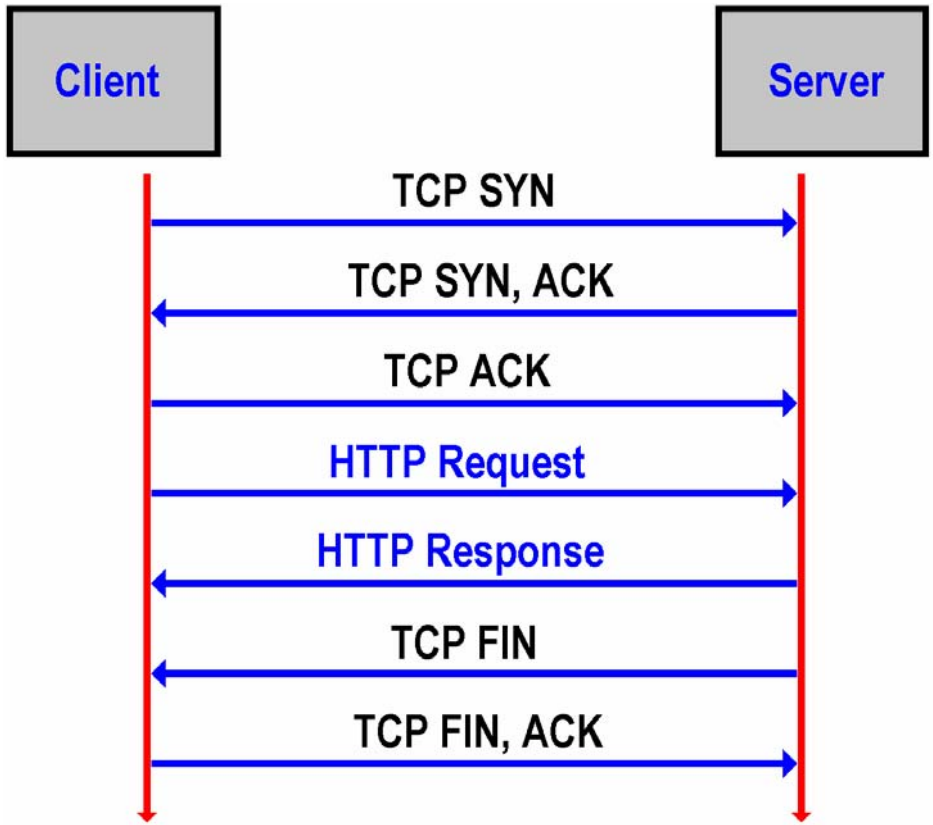
* HTTP follows client/server rules and procedures. The main difference between HTTP clients and servers is the responsibility for initiating communication. Only a client can do that.



* The server waits for a client request and reacts with a response. Typical the Web browser, in it´s rule of client, send a HTTP request to a server. The server returns a HTTP response.

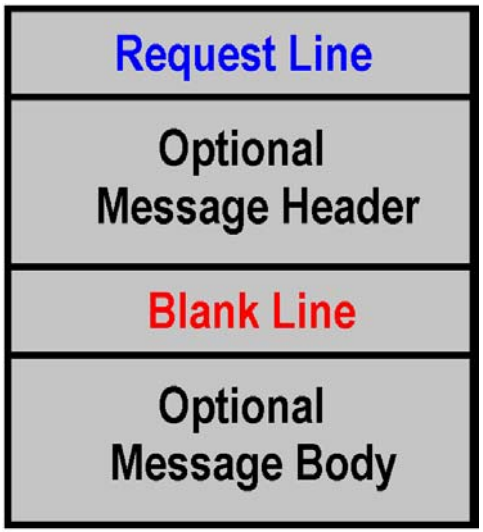
HTTP Protocol Basics

* HTTP requires a TCP connection. The client is responsible for initiating the HTTP communication to the server. After the TCP connection is establish the client can send a HTTP request.

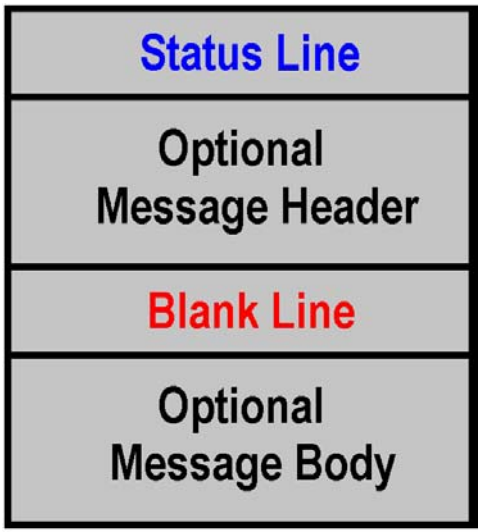


HTTP Protocol Basics

* The structure of HTTP messages is very simple. Each request begins with a request line. This text line indicates the HTTP method, the resource, and the HTTP version. Optional message header and body follows.



HTTP Request



HTTP Response

* The response begins with a status line. This text line starts with the HTTP version that the server supports. Then a status code follows within the status line. A optional header and body follows with the response.

HTTP Protocol Basics (HTTP Methods)

* **CONNECT**: Asks (proxy) server to establish a tunnel.

* **DELETE**: Asks server to delete the indicated resource.

* **GET**: Asks server to return requested resource.

* **HEAD**: Asks server for header information about a special resource.

* **OPTIONS**: Asks server to indicate the options it supports for a resource.

* **POST**: Asks server to pass the message body to a indicated resource.

* **PUT**: Asks server to accept the message body as the indicated resource.

* **TRACE**: Asks server simply to respond to the request.

HTTP Protocol Basics (HTTP Status Codes)

* **100 - 199**: Informational. The server received the client request but the final result is not yet available.

* **200 - 299**: Success. The server was able to act on the client request successfully.

* **300 - 399**: Redirection. The client should redirect the request to a different server or resource.

* **400 - 499**: Client Error. The request contained an error that prevented the server from acting on it successfully.

* **500 - 599**: Server Error. The server failed to act on a request even though the request appears to be valid.

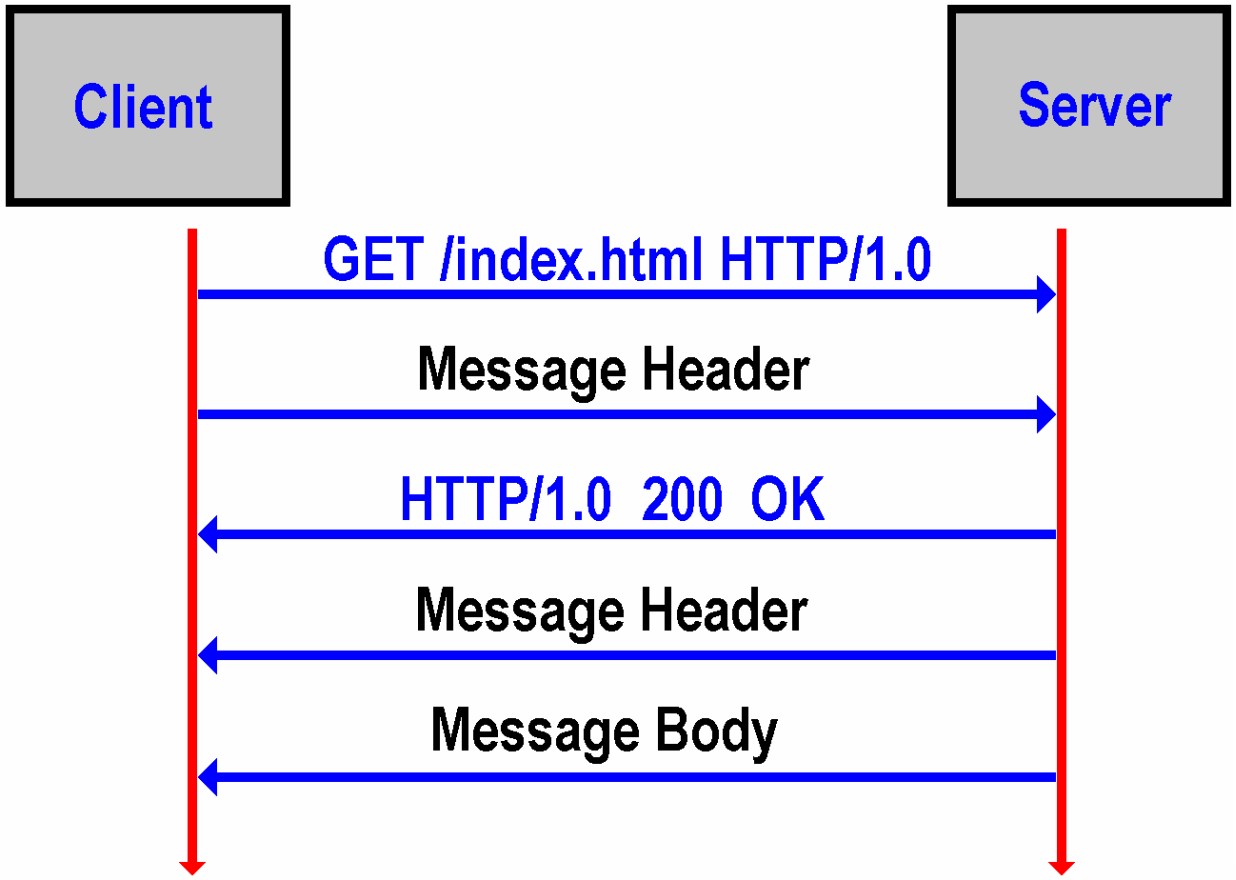
HTTP Protocol Basics

* The message header contains with **Content-type** the MIME (Multipurpose Internet Mail Extension) type of the message body.

`text/plain`	ASCII text
`text/html`	HTML formatted text
`application/pdf`	Document formatted in Adobe PDF
`image/gif`	Image encoded in GIF format
`image/jpeg`	Image encoded in JPEG format
`audio/basic`	Sound file
`video/mpeg`	Video clip coded in MPEG format

HTTP Protocol Basics

* A typical GET request consist of a request line and a message header. The server returns the status line, a message header and the requested resource.



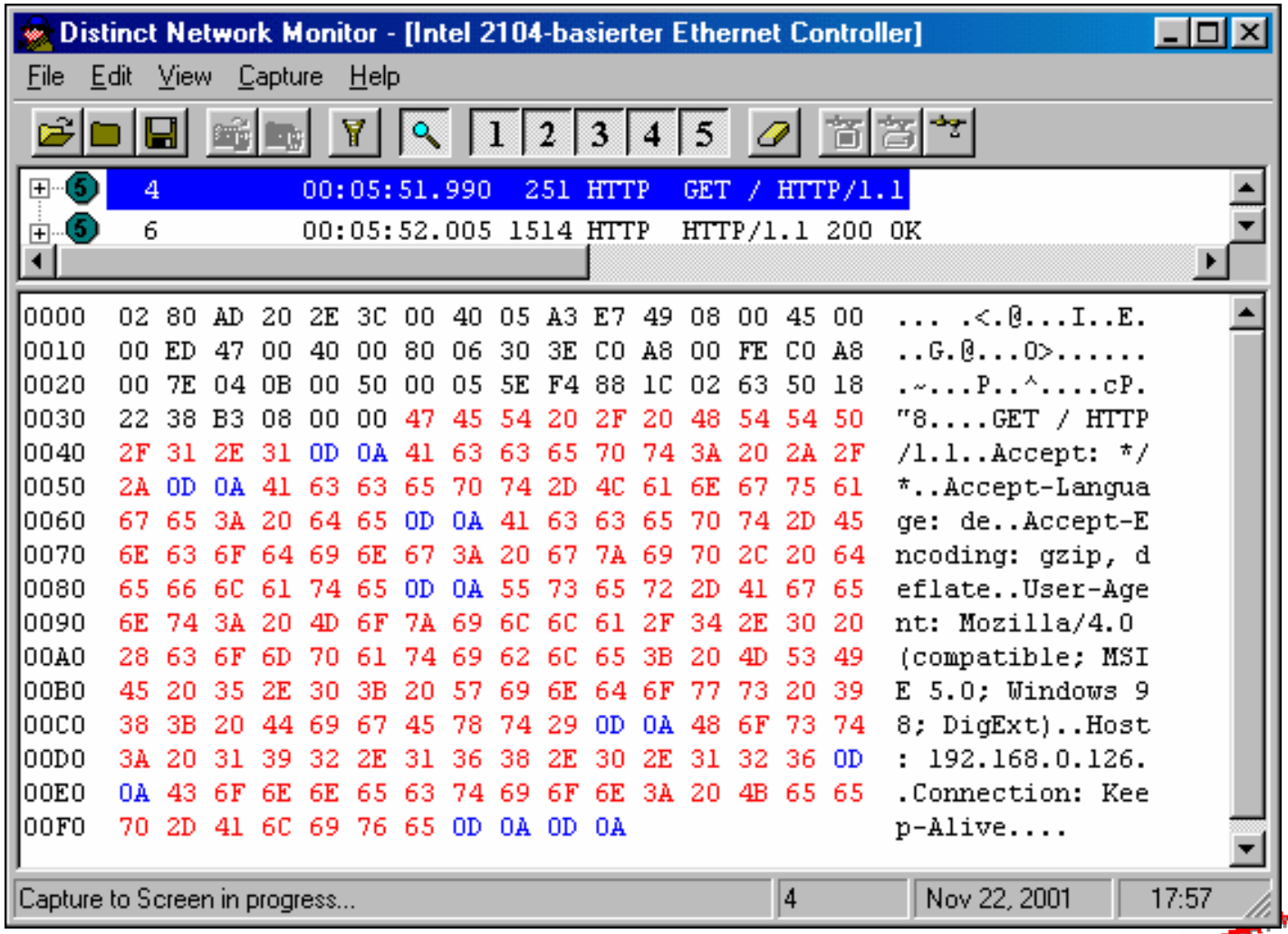
HTTP Protocol Basics

```
GET /test.htm HTTP/1.1
Accept: image/gif, image/jpeg, */*
User selling agent: Mozilla/4.0
Host: 192.168.0.1
```

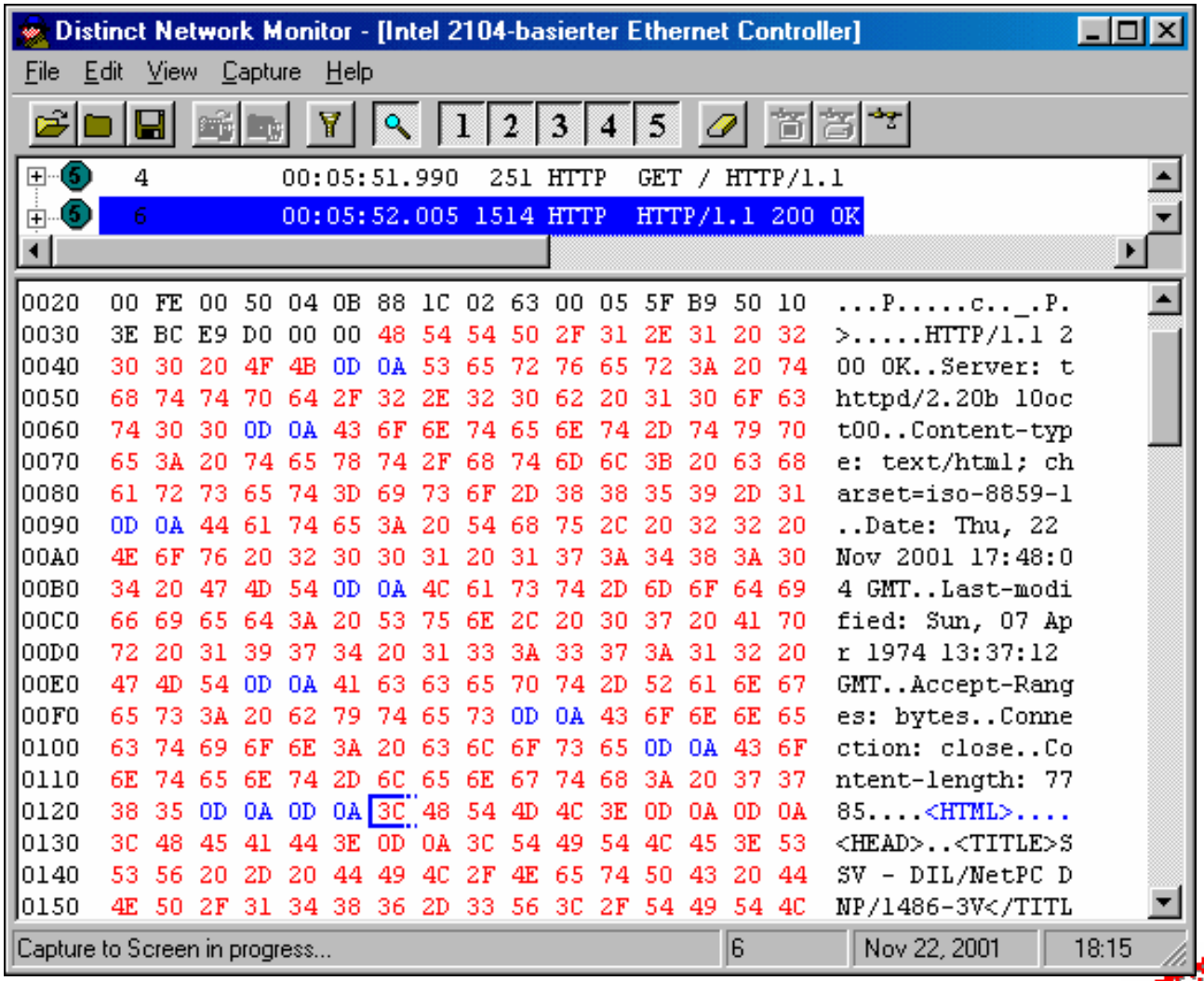
```
HTTP/1.1 200 OK
Date: Mon, 06 Dec 1999 20:55:12 GMT
Server: Apache/1.3.6 (Linux)
Content-length: 82
Content-type: text/html

<html>
<head>
<title>HTML Test Page</title>
</head>
<body>
HTML Test Page
</body>
</html>
```

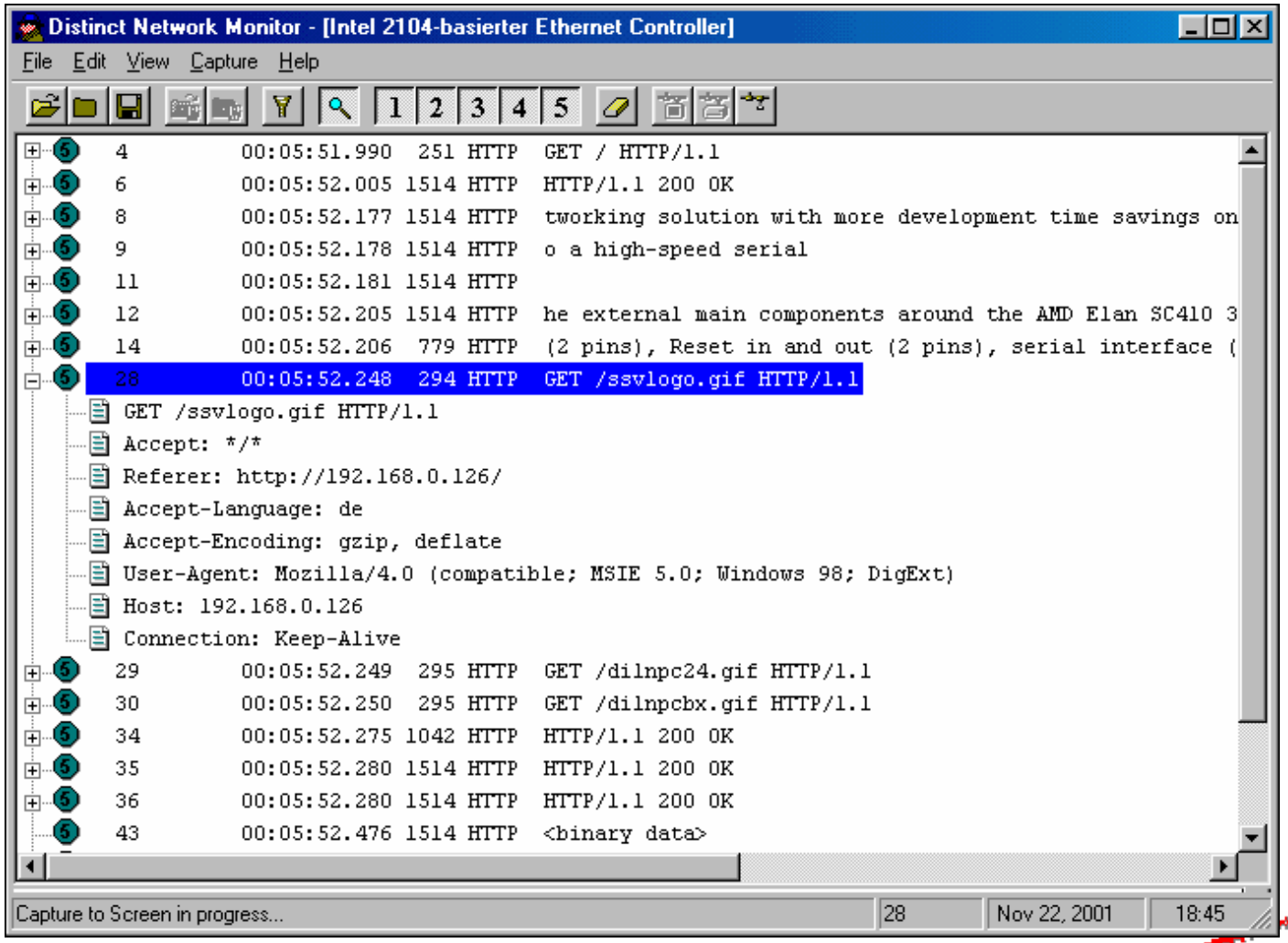
HTTP Protocol Basics



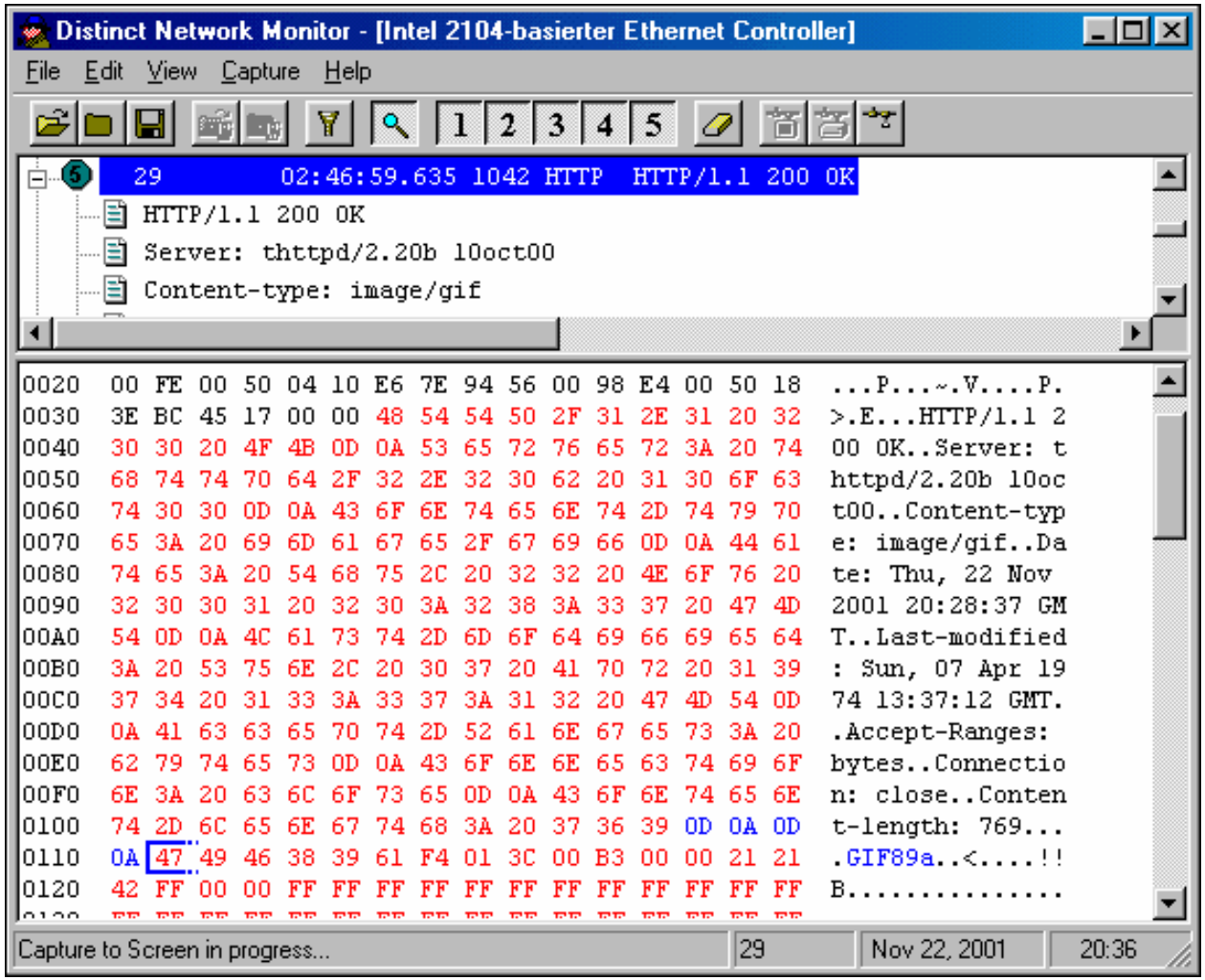
HTTP Protocol Basics



HTTP Protocol Basics

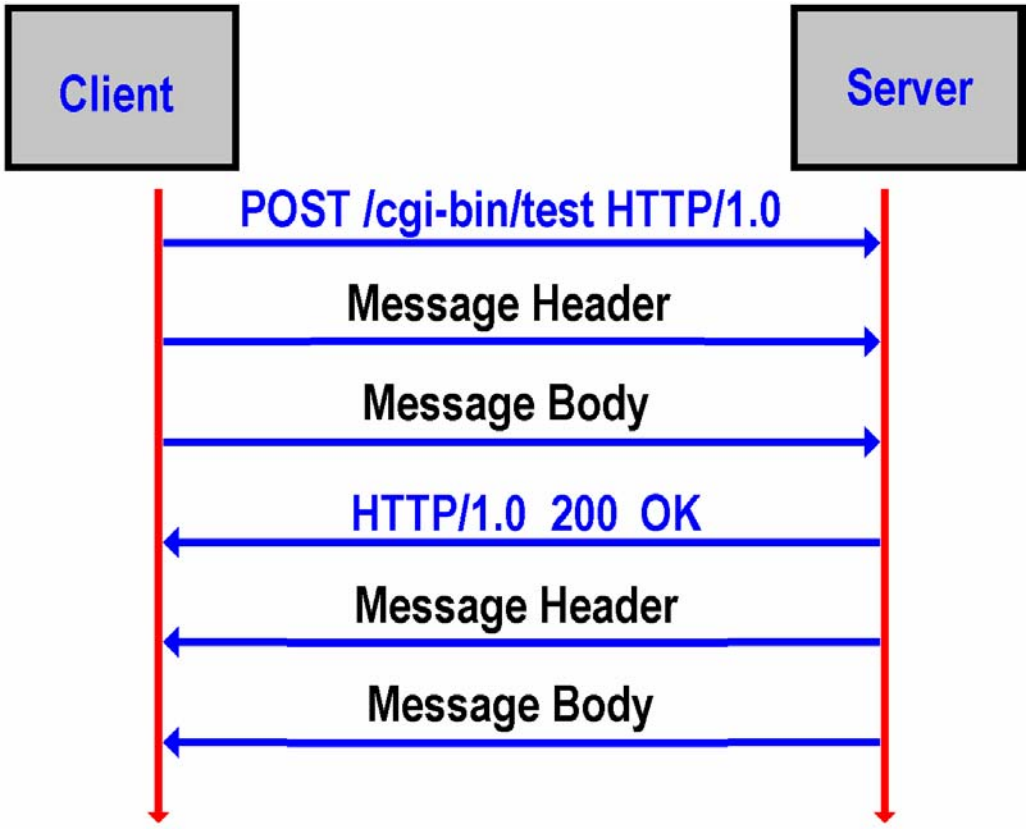


HTTP Protocol Basics



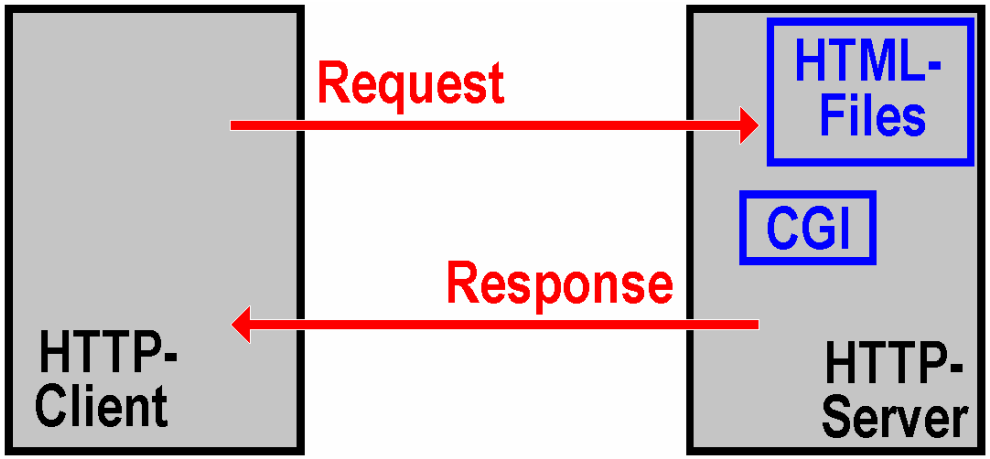
HTTP Protocol Basics

* A POST request consist of a request line, a message header, and a message body. The server returns status line, message header, and the output of the CGI program within the message body.



Web Server Basics

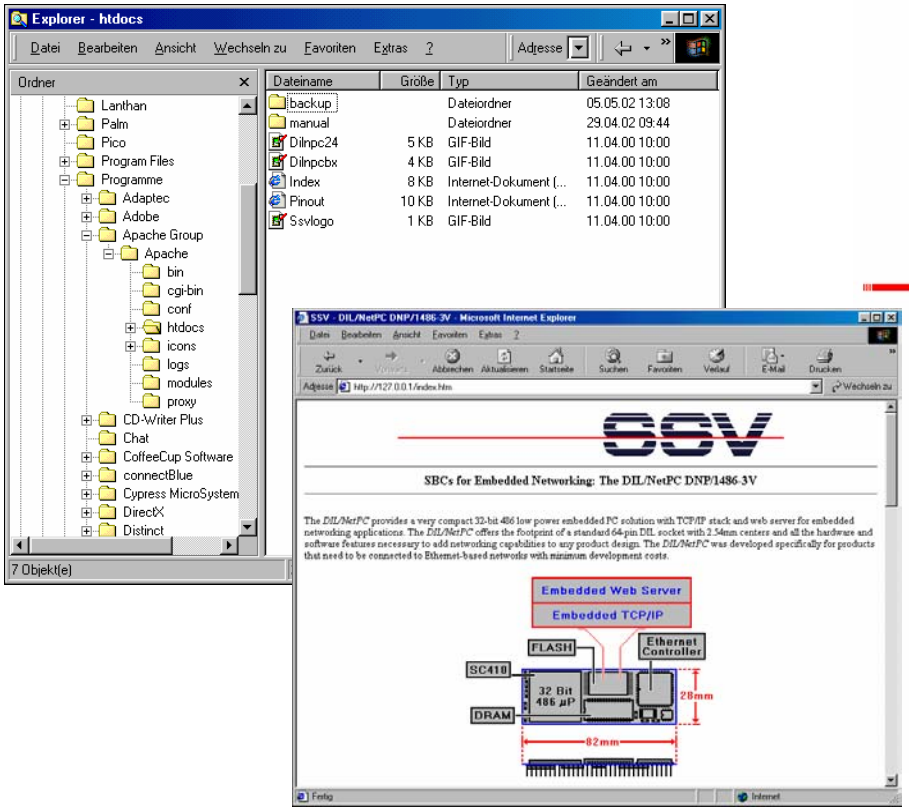
* Simplified a Web server can be imagined like a special kind of a file server. The “files” are the resources (static content: HTML files, GIF and JPEG pictures ...). The Web server deliver these resources with a very special communication protocol (HTTP).



* A Web server can not only deliver static content. There are different technologies (CGI, Server-Side-Includes) for generate dynamic content.

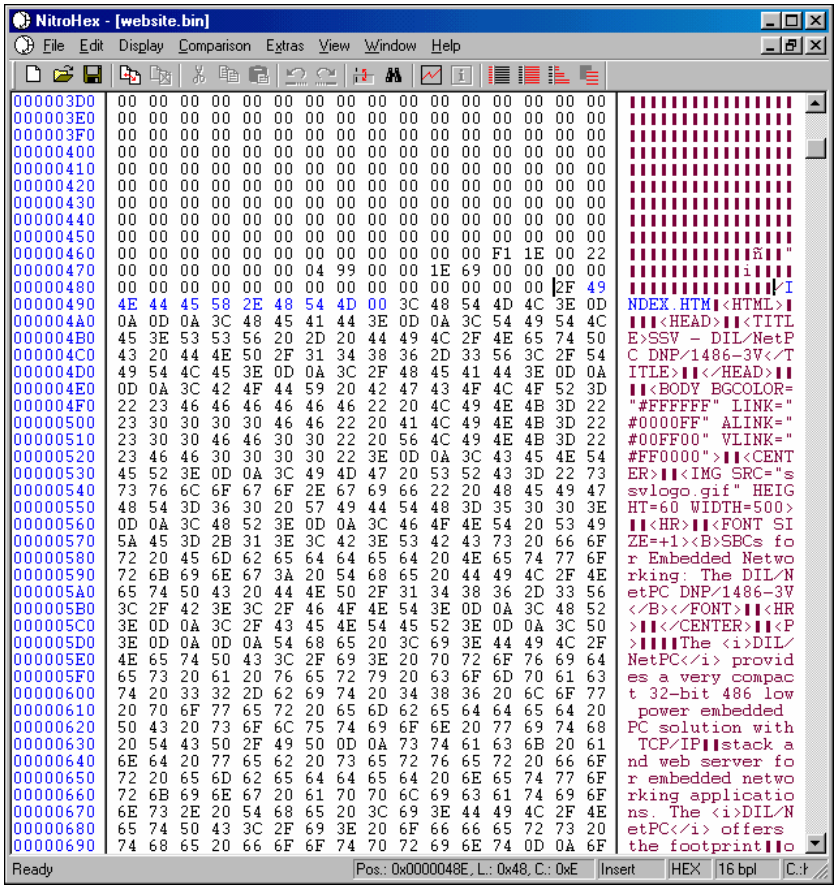
Web Server Basics

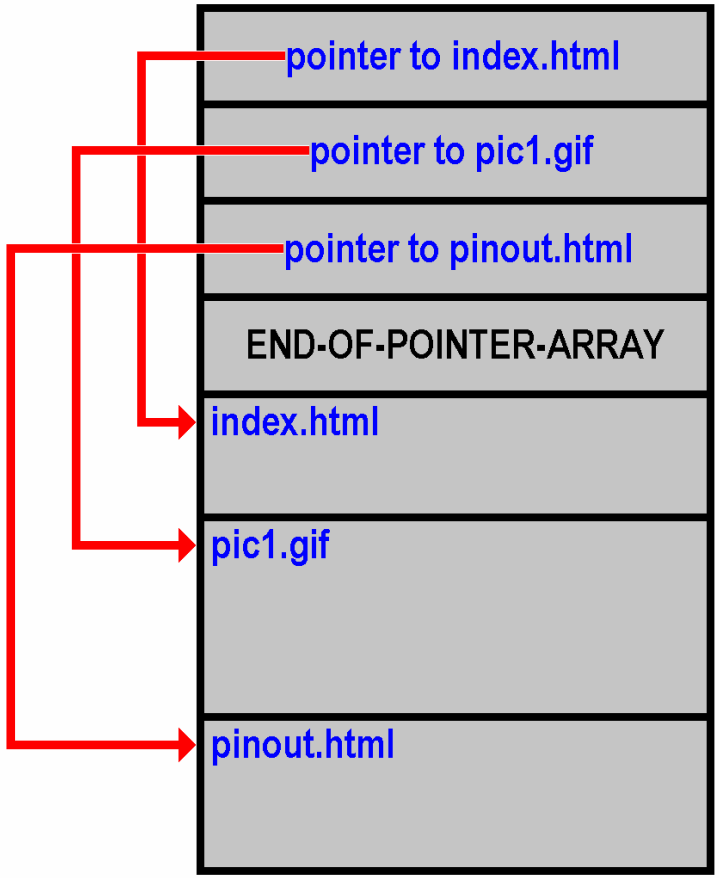
* Web server needs storage space for the resources (HTML files, GIF and JPEG pictures ...). Typical these resources are stored within special directories of a file system.



Web Server Basics

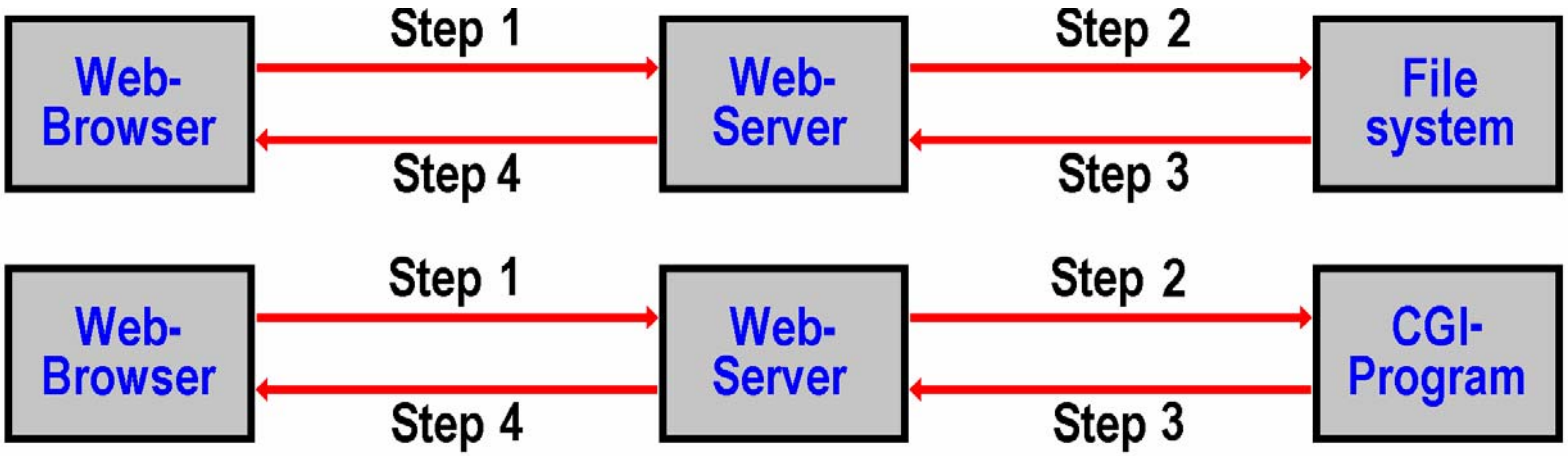
* Microcontroller-based (O/S-free) embedded Web server are using a **Resource Image File** within the firmware memory instead of a file system.





Web Server Basics

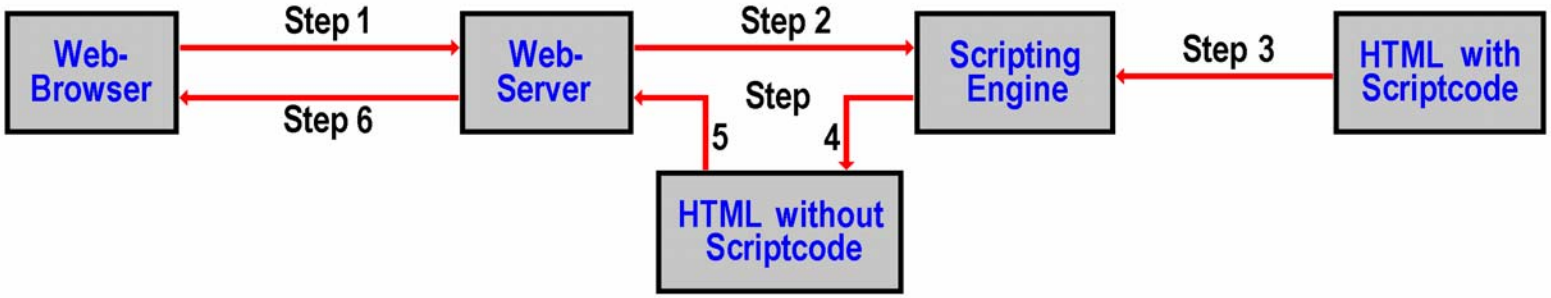
* With step 1 the browser sends a GET request for a specific resource to the Web server. Then (step 2) the server try to read this resource. If the resource accessible, the server reads the resource (step 3) and deliver the content to the Web browser (step 4).



* If the GET request points to **cgi-bin**, then the server runs the indicated program (step 2) and receives some output from this program (step 3). The outputs goes with the HTTP response to the Web browser.

Web Server Basics

* A other way for generating dynamic content is SSI (Server-Side-Include). The basic idea of this technology is to include script language statements to HTML pages. For interpreting the script statements the Web server needs the help of a **Scripting Engine** (i.e. PHP interpreter).

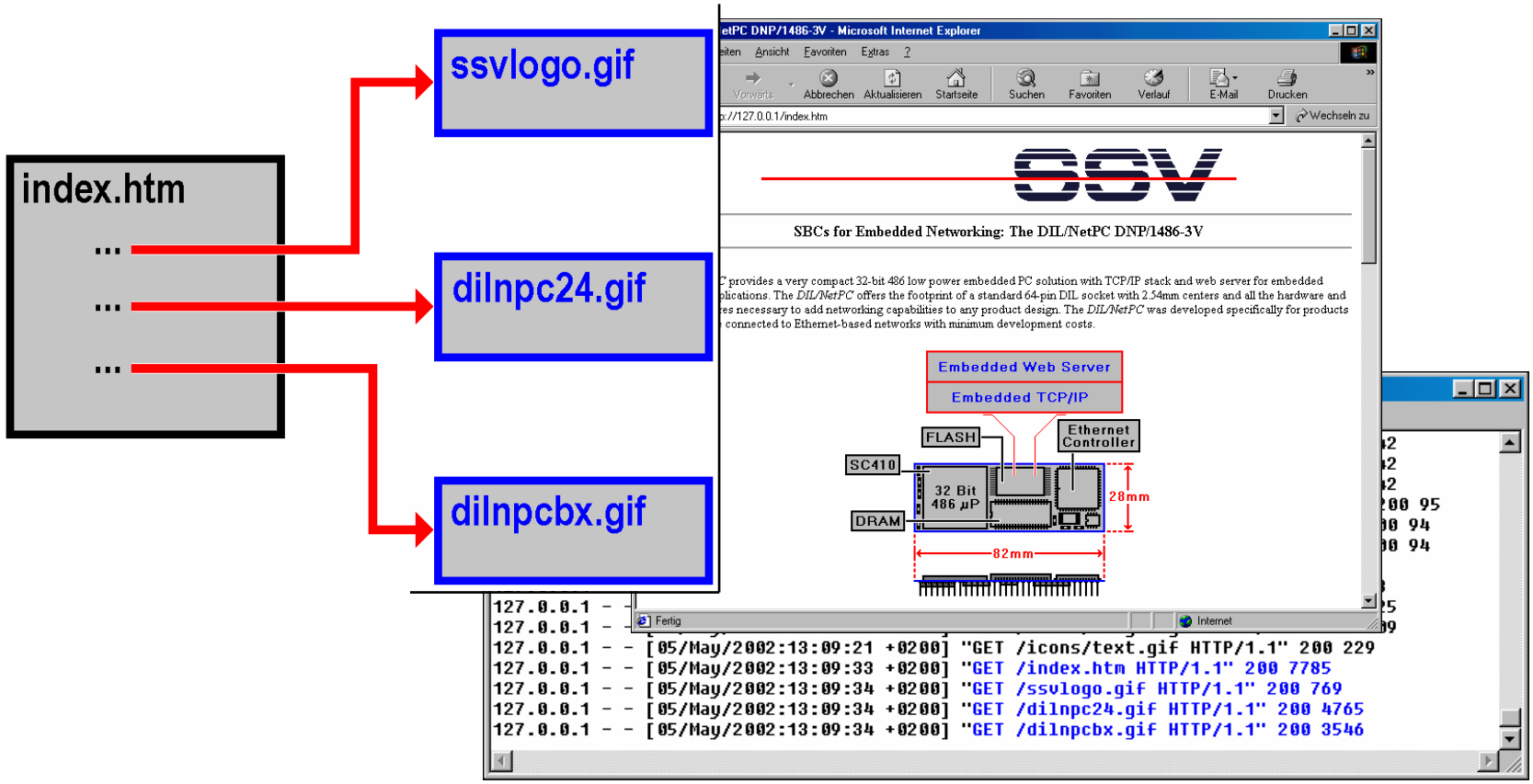


* Step 1 delivers the HTTP request to the Web server. Then the server starts the Scripting Engine (step 2). This program reads the requested HTML page and parses and executes the embedded script code (step 3).

* The output of a Scripting Engine is HTML without script code (step 4) direct to the Web server (step 5). With step 6 the Web server deliver the script-free HTML page to the Web browser.

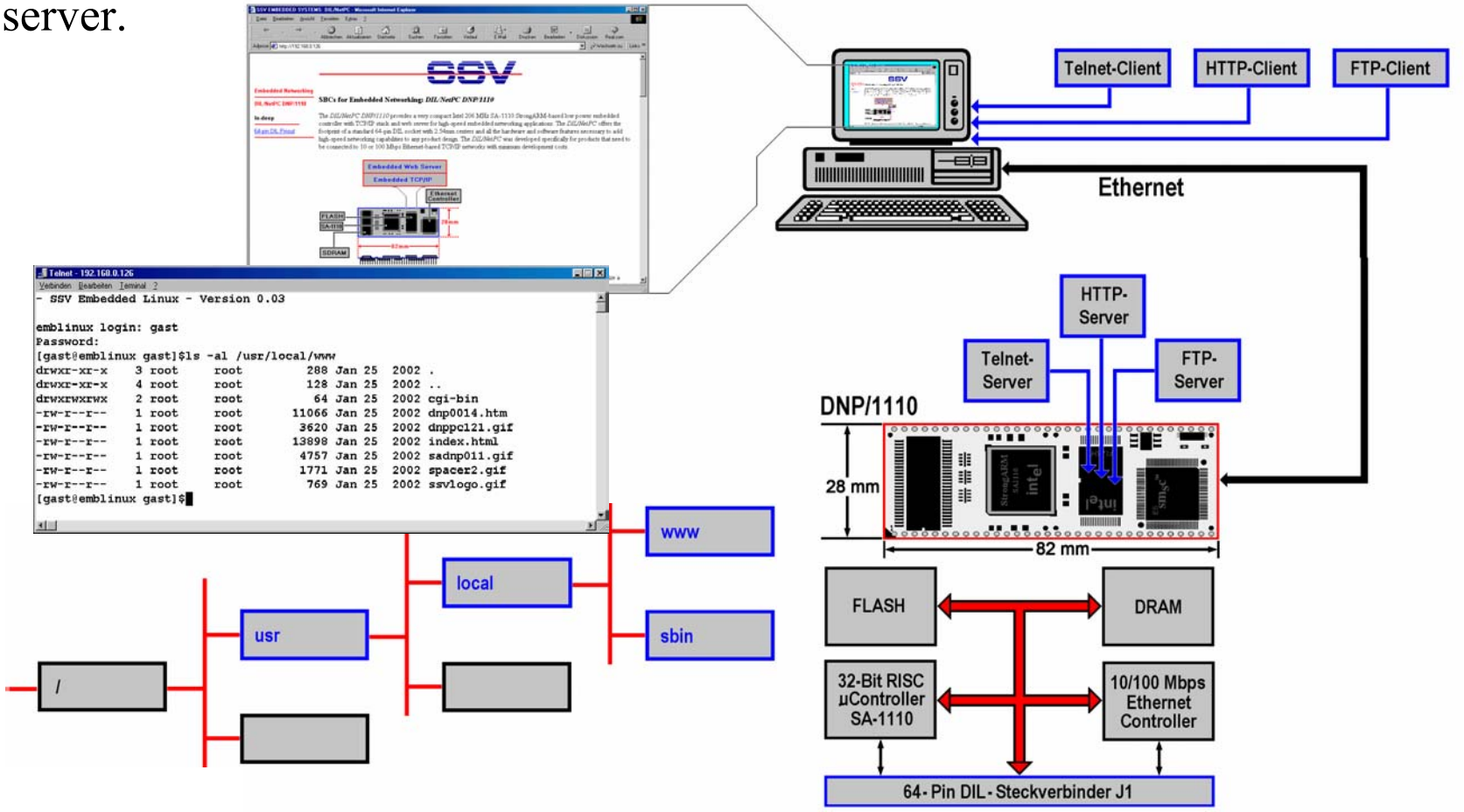
Web Server Basics

* Web resources are linked together. A browser parses the HTML file direct after the first GET. Then the browser issues a additional GET for each (hyper) link.



Solution Demonstration

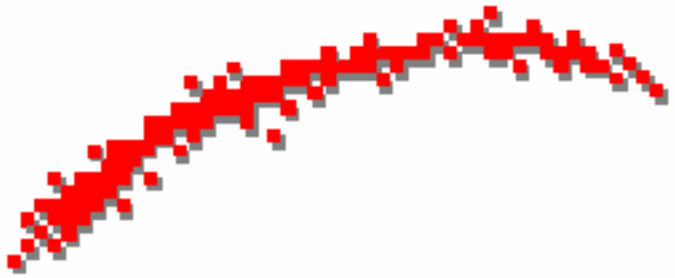
* DIL/NetPC DNP/1110 with embedded Linux and **thttpd** (Open Source) Web server.



Trademarks

Disclaimer

Stay Connected



SSV EMBEDDED SYSTEMS
HEISTERBERGALLEE 72
D-30453 HANNOVER
TEL.: +49-(0)511-40000-0
FAX.: +49-(0)511-4000040
WEB: WWW.SSV-EMBEDDED.DE
EMAIL: INFO@IST1.DE

File: WsforES1.ppt Revision: 1.1 - 11.05.2002