Overview

1. Company Profile O³-Software

2. Short introduction into HTTP/HTML
   - 2.1 HTTP / HTML
   - 2.2 The URL
   - 2.3 Dynamic Data / CGI
   - 2.4 Differences to the BlackBox programming style

3. The framework
   - 3.1 The Server (benefits / drawbacks)
   - 3.2 Architecture - The Handler-Concept
   - 3.3 HTML Generation and Parsing
   - 3.4 Session Management
   - 3.5 User Authentication / Language Support

4. The demo

5. Questions and Answers

Company Profile O³-Software

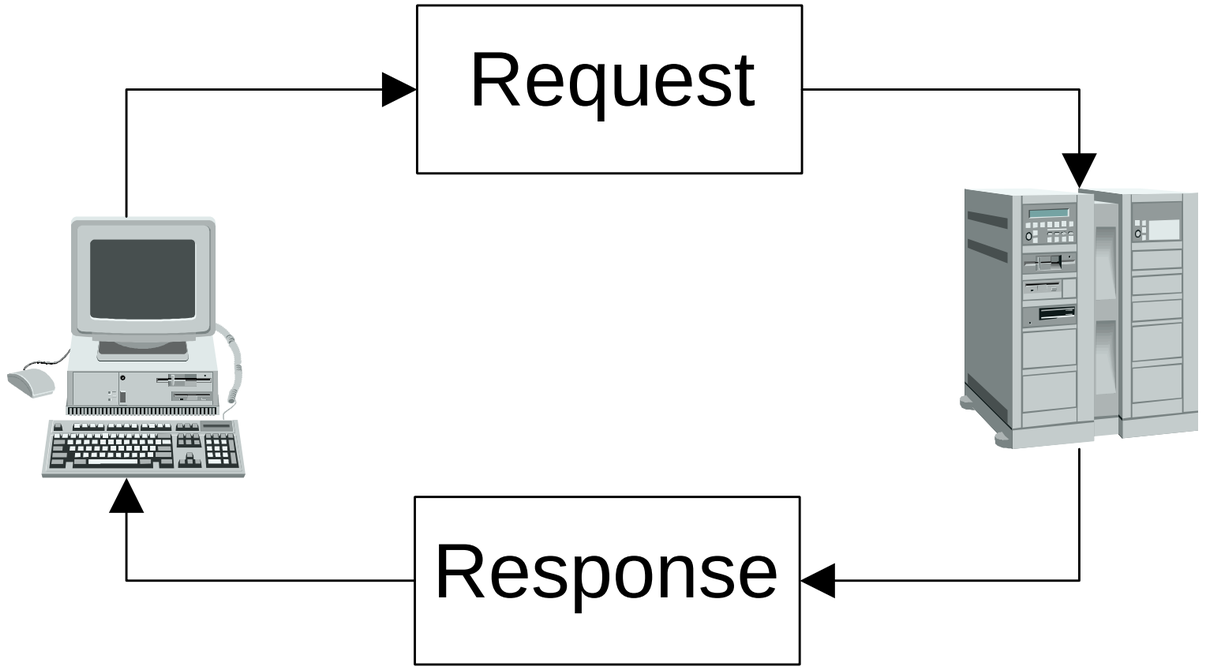
Founded in 1996

BlackBox-only

Development of custom and off-the-shelf software

Reference Customers
- Siemens AG
- Kienbaum Consulting

HTTP-Protocol Principle of operation



HTML
- based on human readable text
- most of the work done by the browser

The URL (more precise URI)

general form (simplified)

protocol://location

e.g. "http://www.oberon.ch"

location

server/path

e.g.
"http://www.oberon.ch/products/index.html"

form-data (GET-Method)

server/path?name=value+name2=value2...

e.g. "http ... search?topic=html+results=all"

Dynamic Data / CGI

special directory
- e.g."cgi-bin"
- execution of external commands
- input http-request
- output http-response (html-document)

PERL

often used for creation of dynamic data
scripting language
support for parsing the URL

Differences to the BlackBox programming style

No
- guards
- notifiers
- `Dialog.Update`

Complex (ordinary) forms have to be transformed to several simple (html) forms

Developer
- has to think more about processes and dialog-steps
- has to do less testing of complex forms
- does not have to bother about intermediate states

The Server

features

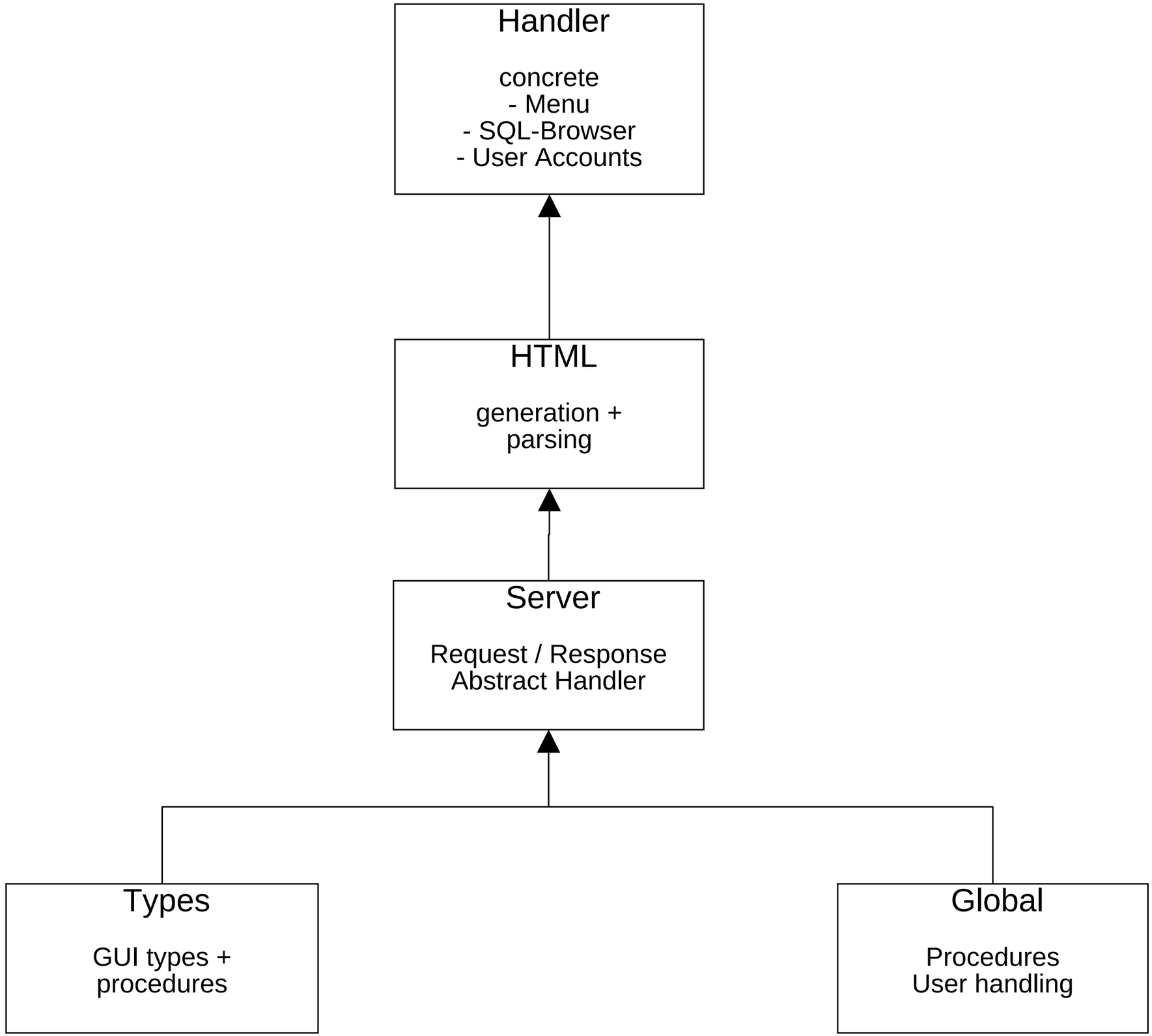
- runs as background task using `Services.Action`
- Automatically restarts itself if a trap occurs and beeps further on
- implements the HTTP 1.0 Protocol
- supports user authorization and account management
- support for keeping session-information (server-side)

benefits

- easy installation (compared to Apache/Perl)
- cgi is not done via scripting
- developers don't have to leave the BlackBox
- BlackBox way of mapping record fields to forms conserved
- intensive use of Meta-programming (super-genericity)

drawbacks

- no support for load-balancing
- rudimentary language support (will be extended)
- no SSL
- no POST-Method (soon)
- no logging/statistics (yet)
- rudimentary mime types support (will be extended)


Handler
concrete
- Menu
- SQL-Browser
- User Accounts
HTML
generation +
parsing
Server
Request / Response
Abstract Handler
Types
GUI types +
procedures
Global
Procedures
User handling

The Handler Concept

```
Handler* = POINTER TO ABSTRACT
RECORD
   ...

END;
```

During Startup the Web-Server:
- scans all subdirs for code files
- loads corresponding modules
- collects all Handler (Extensions)

Advantage
- Handlers are auto-registered
- Menus are automatically created
- features can be added / removed just by
  adding / removing the corresponding code file

Disadvantage
- all modules are loaded (memory usage)

```
Handler = POINTER TO ABSTRACT RECORD
      name-: ARRAY 256 OF CHAR;
      server-: Server;
      (h: Handler) Do- (m: Message), NEW, ABSTRACT;
      (h: Handler) GetSession (m: Message), NEW;
      (h: Handler) Info (language: INTEGER; OUT info:
HandlerInfo)

      , NEW, ABSTRACT;
      (h: Handler) QueryToRec (m: Message; VAR rec: ANYREC),
NEW;
      (h: Handler) SendResponse (m: Message), NEW
END;

Message = POINTER TO LIMITED RECORD
      rq: Request;
      rs: Response;
      sd-: SessionData
END;

SessionData = POINTER TO LIMITED RECORD
      id-: INTEGER;
      user-: O3webGlobal.User;
      keys-: SET;
      global-, local: ANYPTR
END;

HandlerInfo = RECORD
      supportedLanguages: SET;
      menu: ARRAY 64 OF CHAR;
      submenu: ARRAY 64 OF CHAR;
      name: ARRAY 64 OF CHAR;
      description: ARRAY 1024 OF CHAR;
      rights: ARRAY 32 OF ARRAY 128 OF CHAR
```

How are handlers called?

typical URL
http://127.0.0.1/cgi/O3webHandler.SqlBrowser/4711/execute?statement=...

Special directory cgi --> Web-Server calls a Handler.

Handler --> `Module.Type`

extraPath --> session-id followed by (possible) params

queryString --> form data

HTML generation and parsing

```
PROCEDURE
	Create (IN rec: ANYREC): TextModels.Model;

PROCEDURE
	GuiToForm (IN gui: ANYREC; IN param: O3webTypes.GuiParam;
				m: TextModels.Model);

PROCEDURE
	Load (path: ARRAY 260 OF CHAR): TextModels.Model;
```

2-step Form preparation

1. Create or Load HTML Document
2. Parse and fill Document from Record

`Create` works best on special Types defined in `O3webTypes` (similar to e.g. `Dialog.List`) but can
also handle „ordinary“ records.

Session Management

normal applications have
- global data
- local data

```
SessionData = POINTER TO LIMITED RECORD
      id-: INTEGER;
      user-: O3webGlobal.User;
      keys-: SET;
      global-, local: ANYPTR
END;
```

web-application adds
- per user local data
  (every handler can store its local per user data here)
- per user global data
  (all handlers can share the global per user data)

User Authentication

- only available for Database-Applications
- via http-authenticate
- generates Session-ID (numeric)
- checks IP-Adress and Session-ID
- 32 keys (SET) per Handler
- users can inherit rights from profiles

Language Support

- direct support by the Framework
- rudimentary right now