

Design and Implementation of an internet based Calendar System

(WML - Version)

WS 2000/2001

Ednan Masovic &
Reinhold Klapsing



University of Essen

Information systems and software
engineering

Univ. Prof. Dr. Rony G. Flatscher

Overview of this presentation

- ✍ Introduction
- ✍ Requirements Analysis
- ✍ System Architecture
- ✍ Class System
- ✍ Scripts (Rexx)
- ✍ Database „CalendarStore“
- ✍ Screenshots of the Prototype
- ✍ Future work

Introduction

How do I connect to WAP?

- You need a WAP phone
- Option is called 'Internet' or 'Services'
- information you need to know are
(available from your operator: T-D1, D2,...):
 - ✍ Dial-up number (set up for a certain gateway)
 - ✍ IP address (takes you through a gateway)
 - ✍ Log-in name (sometimes required)
 - ✍ Password (sometimes required)

Introduction

Connecting to WAP without a WAP device

✍ Can be done with an WAP emulator:

- "NOKIA WAP Toolkit version 2.0 "
<http://www.forum.nokia.com/>
- „WinWAP 3.0 Pro - Free trial version“
<http://www.winwap.org/WinWAP3P.exe>

Introduction

Aspects of this work

- ✍ PC-Terminal vs. WAP-Device
 - **costs**
 - **comfort**
- ✍ Combined use of PC and WAP-Device
 - **Designing**
a system for both device types
 - **Implementation**
of a system for both device types

Introduction

PC-Terminal vs. WAP-Device

Costs

- of providing informations for the devices
- of using devices to get informations

Comfort

- Inputmodes and displays
- mobility of the informations

Introduction





costs

- ✍ of providing information for the devices
 - Updating costs are similar
- ✍ of using the devices to get information
 - WAP device
 - ✍ 0,2 €/ min. for using a gateway of a provider
 - ✍ Further costs for infos from Content-Providers
 - PC – Terminal
 - ✍ 0.01 €/ min

Introduction

comfort

Inputmodes and displays

- PC-Terminal
 -  Input: Mouse, Keyboard (101/102 Keys)
 -  Display: ? 200 words in 20 lines
- WAP device
 -  Input: numpad ? 20 keys
 -  Display: ? 10 to 20 words in 2 to 5 lines

Mobility

- WAP: „mobile internet“
- PC-Terminal: „internet cafe“

Introduction

Combined use of PC and WAP-Device






- ✍ Designing a system for both device types
 - Supports HTML and WML
 - Priority has HTML
 - a common database is used
 - WML is an enhanced feature of the system

"mobile internet"

Introduction

Combined use of PC and WAP-Device

Combined implementation

- of static pages
 -  Problem when updating the web informations
 -  Both, HTML and WML must be updated separately
- or dynamicly generated pages?
 -  No problem when updating web informations
 -  Only the common database must be updated
 -  Scripts are needed!

Introduction Standards

- ✍ IETF – Internet Engineering Task Force
 - **Internet Calendaring and Scheduling Core Object Specification**” [RFC 2445]-November 1998
- ✍ WAPFORUM - an organization of several big internet and telecom companies
 - „**WAP Architecture Specification**“ – 30 Apr 1998
 - „**WAP WML**“ – 19 Februar 2000
- ✍ W3C – WWW Consortium
 - **“HTTP 1.1”** – [RFC 2616] – June 1999

Introduction



„Web-Calendar“

- ✍ A prototype for mobile devices:
 - <http://swt.wi-inf.uni-essen.de/~emasovic/index.wml>
 - only retrieve of appointments
- ✍ The HTML-prototype:
 - <http://swt.wi-inf.uni-essen.de/~tjungman/logon.html>
 - edit, retrieve,... of appointments
- ✍ For testing ✍ Username: **tt** Password: **tt**

Requirements Analysis

This work

Target of this work

- Retrieving of personal appointments via WWW
- User Agents:
 -  Web-browser
 -  WAP-device

Emphasis of this work

- Designing and implementation of the WAP part of the information system

Requirements Analysis Components (1/2)

Components to be designed

- Session-Management
- Common Database („CalendarStore“)
- Static WML
- Dynamic WML (scripted by „Rexx“)

Required Components

- „MySQL“ – Database
- „Apache“ – Server
- „Rexx“ – Interpreter
- „Nokia“ – WAP-Emulator

Requirements Analysis Components (2/2)

- ✍ Used interfaces of the Components
 - CGI ✍ INTERNET-DRAFT
 - HTTP ✍ [RFC 2616]
 - WAP ✍ WAPFORUM.ORG
 - “Rexx/SQL” ✍ interface to SQL databases

Requirements Analysis Session-Management (1/2)

- ✍ **Hidden-Form-Fields** specified in HTML 4.01
- ✍ **Post-Fields** specified in WML 1.1
- ✍ **Cookies** specified in [RFC2109]
- ✍ **PATH-Info** specified in CGI

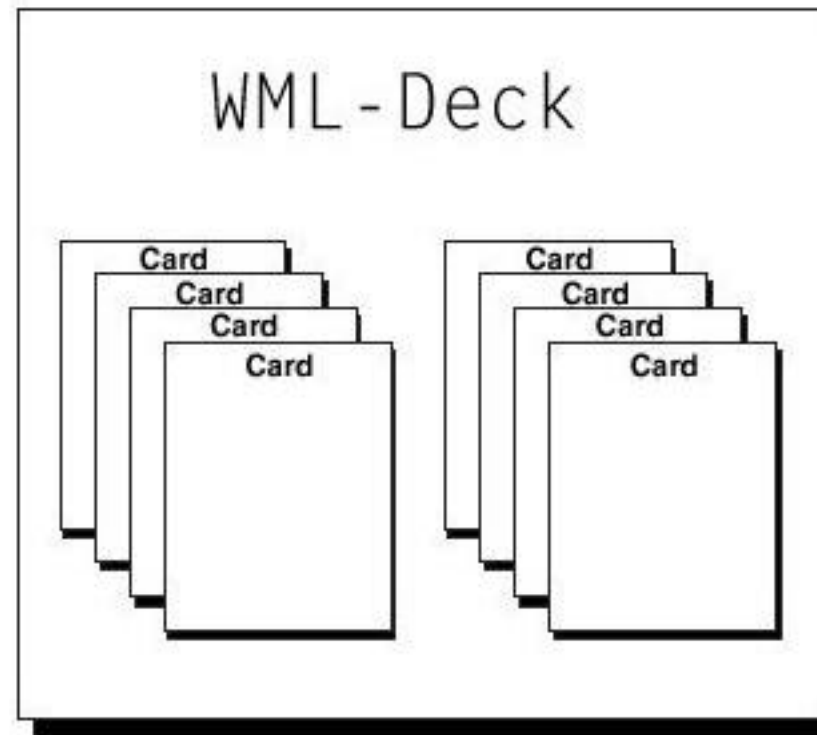
Requirements Analysis Session-Management (2/2)

Post Fields specified in WML1.1 [1]

```
<do type="accept" label="login">  
  <go href="cgi-bin/login.cgi" method="get">  
    <postfield name="user" value="$(user)" />  
    <postfield name="pw" value="$(pw)" />  
  </go>  
</do>
```

Requirements Analysis

WML-Decks (1/2)



Requirements Analysis

WML-Decks (2/2)

- ✍ WML-decks are defined in WML1.1
- ✍ Is used for specifying content and user interface for narrowband devices
- ✍ WML is an application of XML [2]

Requirements Analysis Database

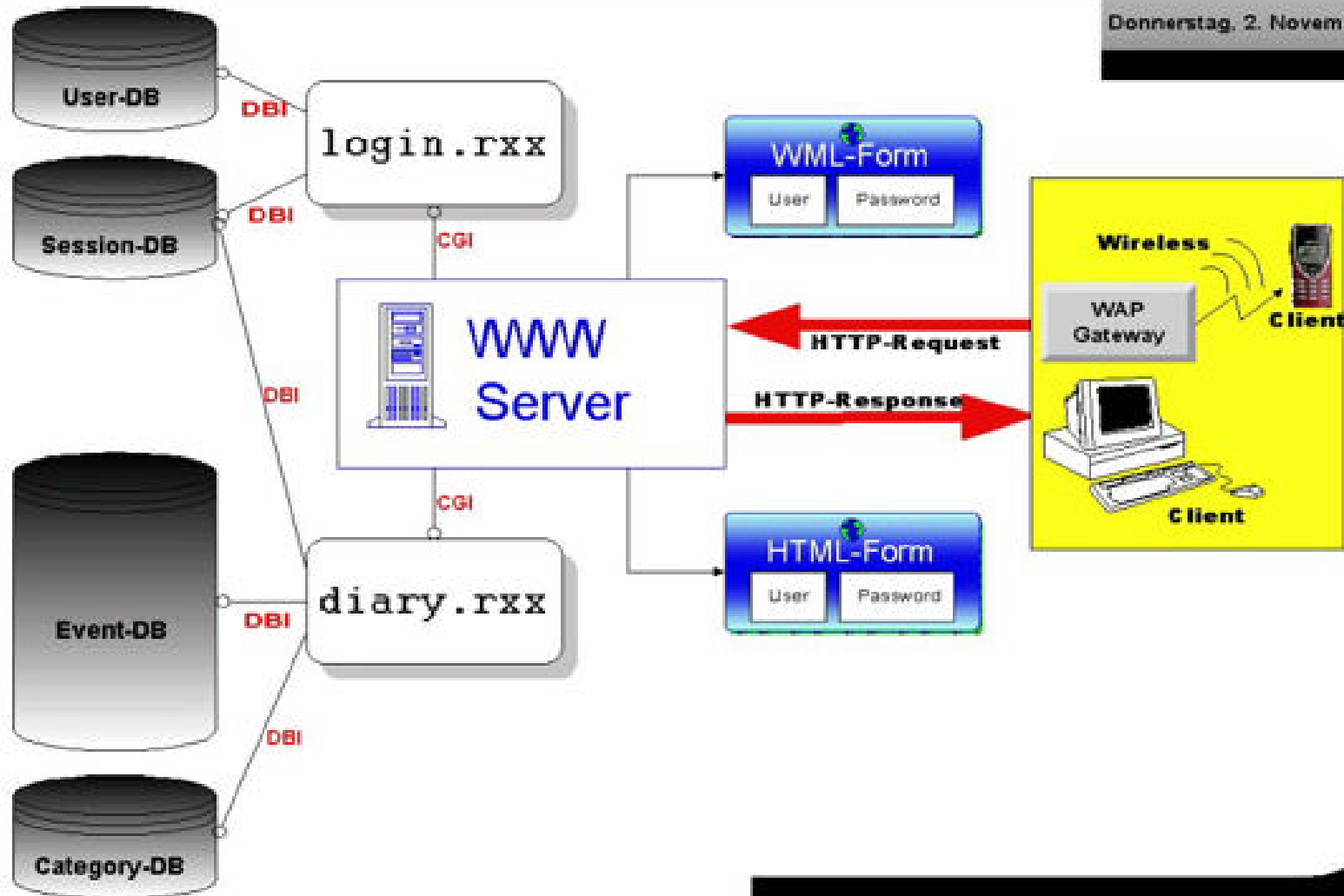
„CalendarStore“ is needed

- ✍ For administration of users and passwords
- ✍ and the session management temporarily,
- ✍ To categorize users appointments,
- ✍ To store the journals of the individual events.

System architecture

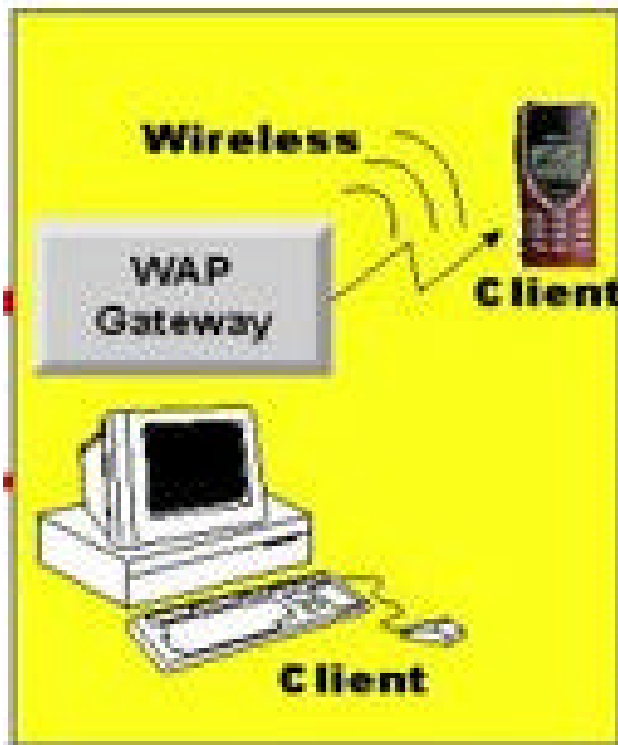
Systemarchitektur des web-basierten Terminkalenders

Donnerstag, 2. November 2000



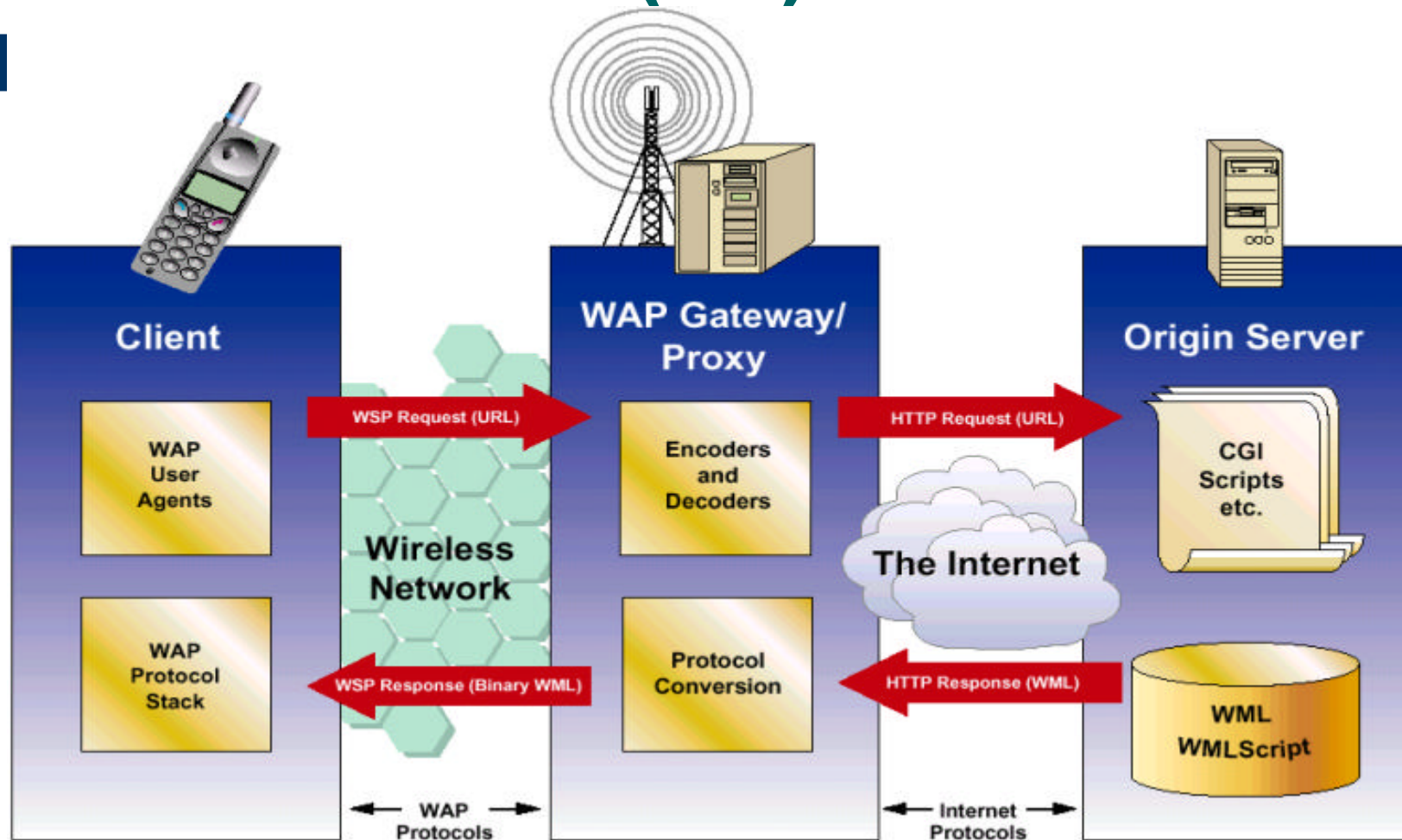
System architecture

Clients



- ✍ Clients/User Agents:
 - Web Browser
 - ✍ Communicates with the system via internet (HTTP)
 - WAP Device
 - ✍ Communicates with the Internet (HTTP) via a WAP-Gateway

System architecture HTTP and WAP (1/2)



System architecture

HTTP and WAP (2/2)

- ✍ WAP Devices communicates with the Gateway via WAP [3]
- ✍ WAP-Gateway communicates with the Web server via HTTP [4]

System architecture

WWW-Server

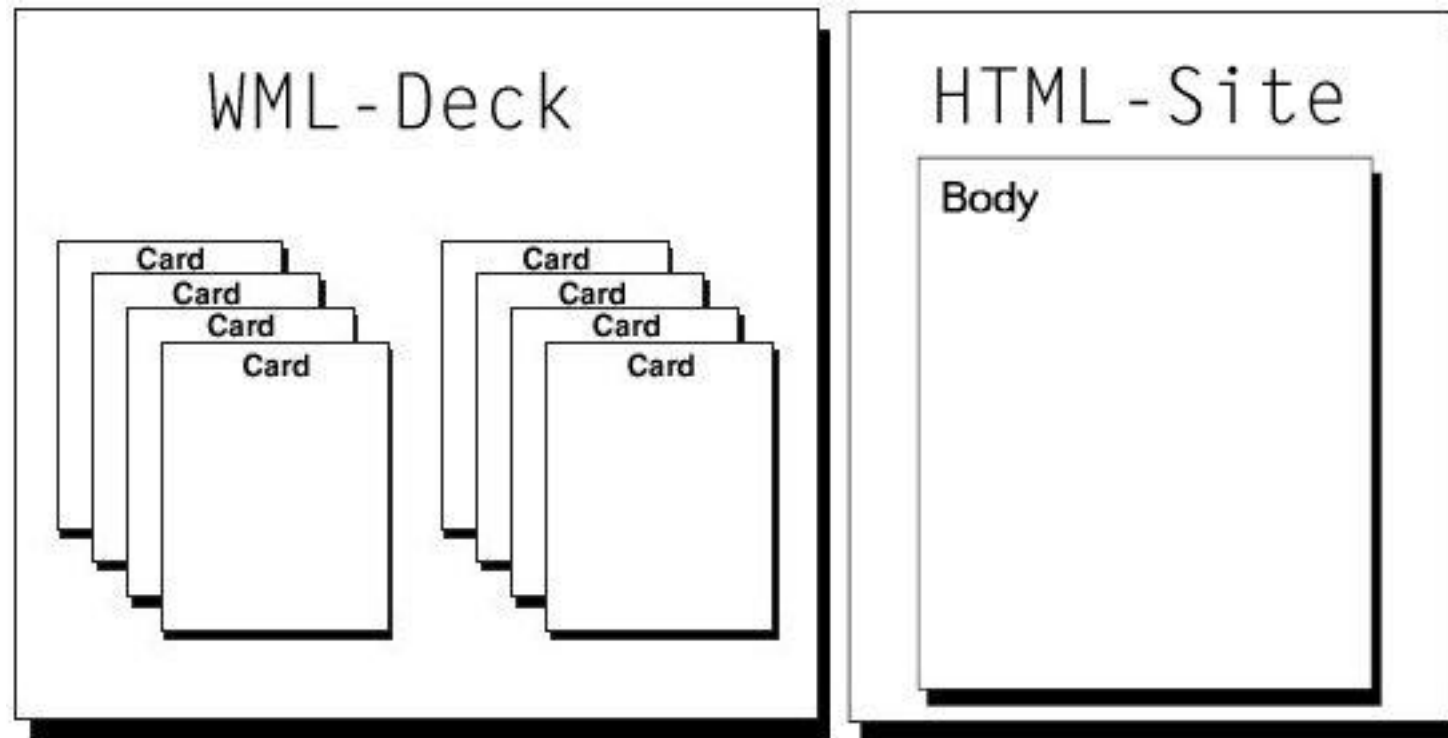
✍ The used Server is an Apache [5]



- Is a HTTP Daemon
- receives requests
- sends appropriate responses
- presented in HTML or WML
- Interfaces are CGI and HTTP
- Executes scripts

System architecture

HTML vs. WML (1/2)



System architecture

HTML vs. WML (2/2)

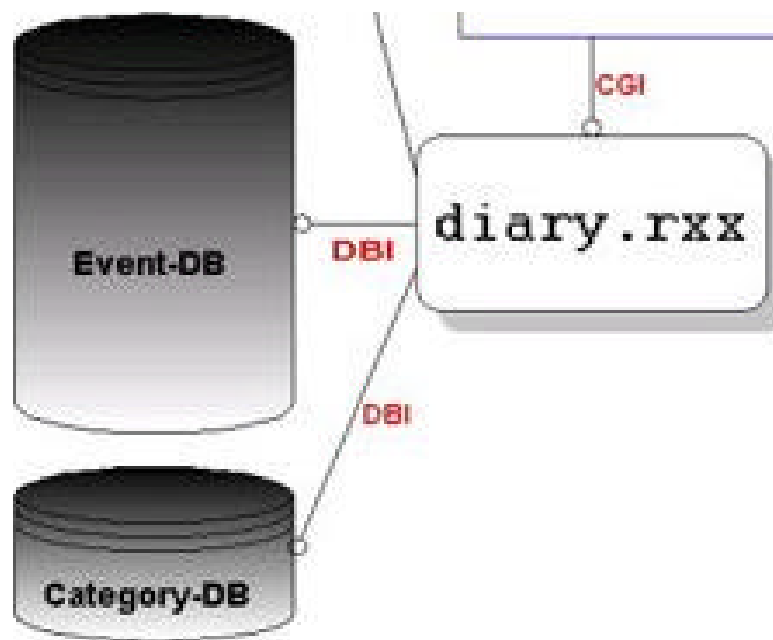
- ✍ HTML pages cannot be displayed on a WAP device
- ✍ A WML-deck contains several WML-cards
- ✍ Each WML-Deck has to start with the following directives
 - **<?xml version="1.0"?>**
 - ✍ directive for the XML Parser
 - **<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN" „http://www.wapforum.org/DTD/wml_1.1.xml">**
 - ✍ The URL of the DTD, to check if document is "valid"

System architecture login.rxx



- ✍ Suplies the login function
 - **Validates** the input data (name, password) with the entries in the database
 - In the case of succesfull validaton a **session ID is generated**
 - and **stored** in the session database temporarily
- ✍ the logout function is also supported

System architecture diary.rxx



✍ Shows the appointments of the user from the Event - database

– By selected view-type

- ✍ Day
- ✍ Week
- ✍ Month
- ✍ Year

– By selected category from the Category-database

- ✍ All
- ✍ Birthday
- ✍ Meeting
- ✍ Call

Class system

- ✍ All file names (almost all,;) correspond to the class name with the addition "...lib.rxx"

e.g.

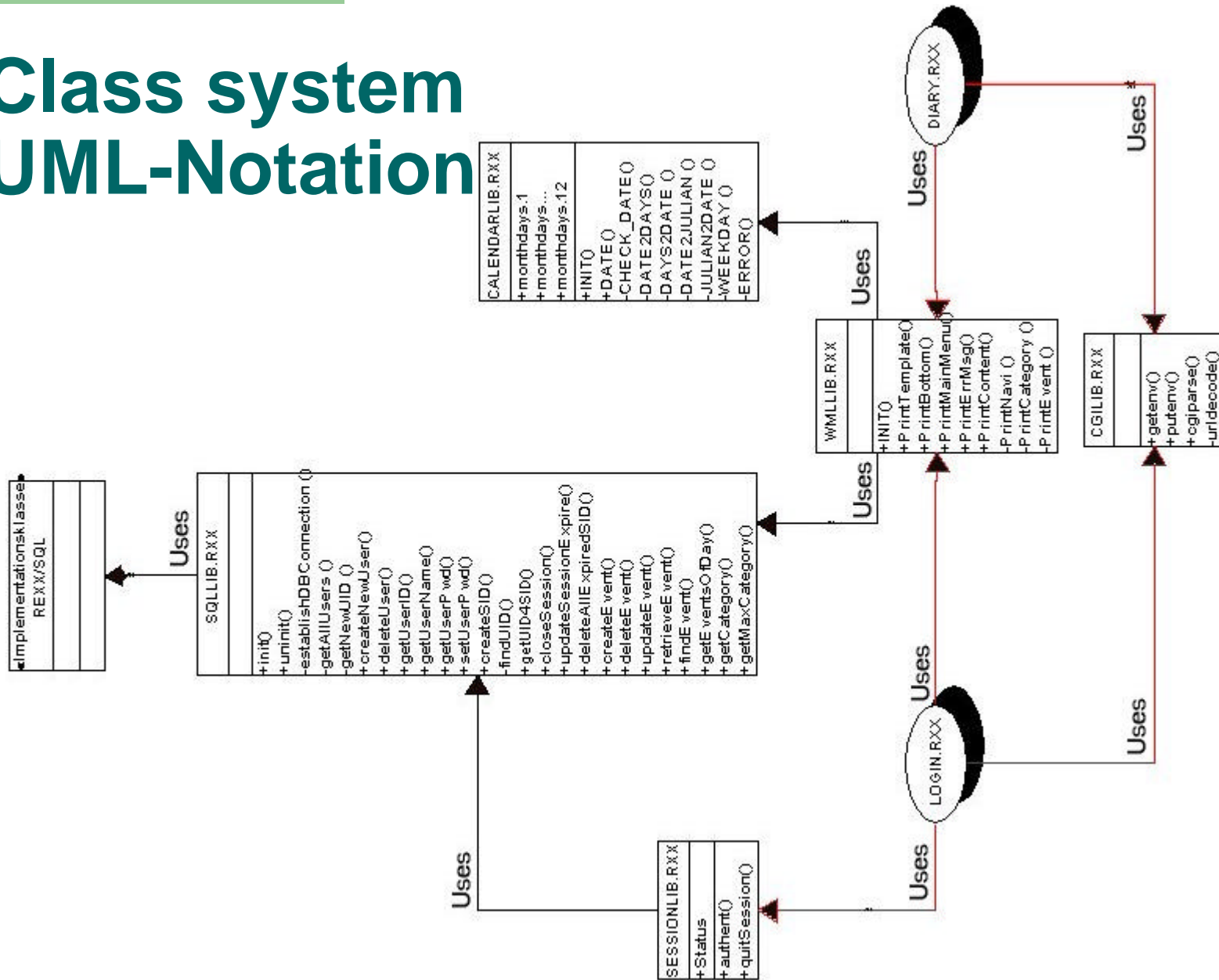
- File: "**cgilib.rxx**" contains only one class

✍ ::CLASS **cgi** PUBLIC

- File: "**sessionlib.rxx**" contains only one class

✍ ::CLASS **session** PUBLIC

Class system UML-Notation



Class system

UML-Notation (2/2)

- ✍ The figure shows the **class hierarchy**
 - **::REQUIRES** "[name_of_the_classlib]"
 - ✍ as well as all **methods**
 - **::METHOD** [name_of_the_method] **PUBLIC**
 - ✍ and **attributes**,
 - **::METHOD**[name_of_the_method] **ATTRIBUTE**
- the one class supplies

Class documentation

- ✍ The classes are documented separately
 - Individual method calls,
`[a_instance_of_the_class]~[method]()`
 - the expected arguments
`[a_instance]~[method](ARG1,...ARGn)`
 - and the return values (RESULT in Rexx)
`[a_instance_of_the_class]~[method]()`
RESULT

Class documentation

cgilib.rxx

- ✍ simplifies the access to environment variables
- ✍ suplies a method "cgiparse", which url-decodes and parses the QUERY_STRING [6]
- ✍ Based on CgiParse() from Sascha Prins

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/cgilib.rxx>

Class documentation

SQLLIB.rxx

- ✍ regulates the access to the database
- ✍ methods are supplied for the manipulation of database contents of the database "CalendarStore"
- ✍ This class uses the Rexx/SQL - INTERFACE from Mark Hessling

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/sqllib.rxx>

Class documentation

WMLLIB .rxx

- ✍ simplify the dynamic generating of the WML-decks
- ✍ e.g the processing instructions are generated "<?xml version='1.0'?>" in this class
- ✍ The DTD is referenced, too

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/wmllib.rxx>

Class documentation

SESSIONLIB .rxx

- ✍ supplies a method needed for authentication of users
- ✍ after a successful authentication a session ID is generated
- ✍ The outlog-function with the following deletion of the session ID from the database
“CalendarStore” is also supplied

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/sessionlib.rxx>

Class documentation

CALENDARLIB .rxx

- ✍ supplies methods that symplify operations with dates
- ✍ Only one method: **DATE(ARG1 , ARG2)**
 - ARG1 - a date [YYYYMMDD]
 - ARG2 - a operatione.g.
 - ✍ „nd“ - next day
 - ✍ „nw“ - next week
 - and so on
- ✍ Based on [datergf.cmd](#) from Rony G. Flatscher

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/calendarlib.rxx>

Class documentation

REXX/SQL version 2.3 [7]

- ✍ “consists of a number of external Rexx functions which provide the necessary capabilities to communicate with the SQL database „
- ✍ is used by "sqllib.rxx" to communicate with the SQL database "CalendarStore".

Scripts

login.cgi

-
- ✍ is a shell script
#!/bin/tcsh
 - ✍ sets the necessary environment variables
setenv LD_LIBRARY_PATH
/net/swt/u/students/tjungman/rmysql/lib
 - ✍ executes the "login.rxx" script with the appropriate
Rexx interpreter
/usr/bin/rexx login.rxx
-

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/login.txt>

Scripts

login.rxx

- ✍ is a Rexx script
 - ✍ logging in
 - After the successful authentication
 - a session identifier is generated
 - ✍ and logging out
 - session Identifier is deleted from the session DB
- are processed by this script

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/login.rxx>

Scripts

diary.cgi

-
- ✍ is a shell script
#!/bin/tcsh
 - ✍ sets the necessary environment variables
setenv LD_LIBRARY_PATH
/net/swt/u/students/tjungman/rmysql/lib
 - ✍ executes the “diary.rxx” script with the appropriate
Rexx interpreter
/usr/bin/rexx diary.rxx
-

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/diary.txt>

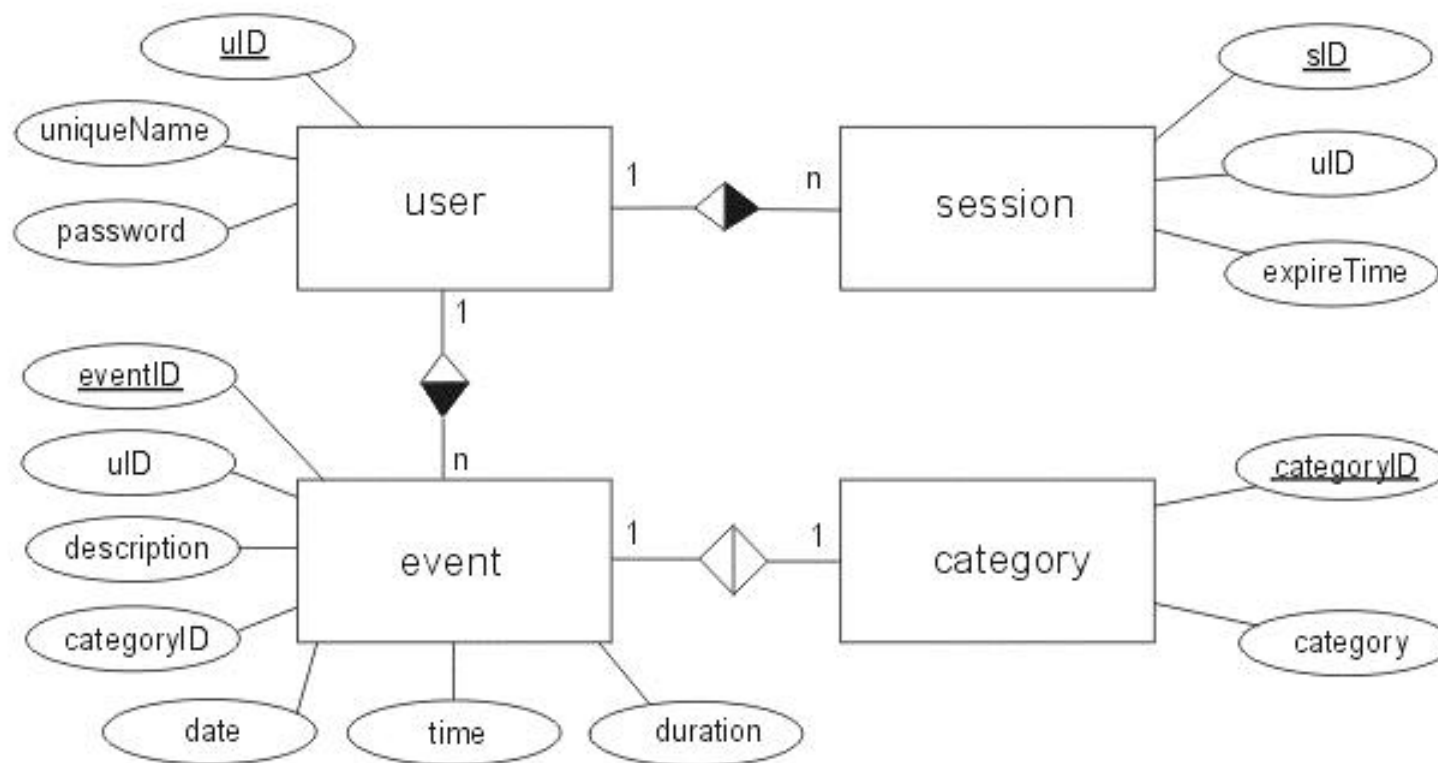
Scripts

diary.rxx

- ✍ is a Rexx script
- ✍ With each request the session ID is validated and the timer reseted
- ✍ the desired information (suitable to the session!) are displayed according to the selected display options
- ✍ modification of the appointments or the creation of new ones is not offered

Source-Code: <http://swt.wi-inf.uni-essen.de/~emasovic/src/diary.rxx>

Database „Calendarstore“ (1/2)



Database „Calendarstore“ (2/2)

User

- (uID,uniqueName,password)

Event

- (eventID,uID,categoryID,description,date,time,duration)

session

- (sID;uID,expireTime)

category

- (categoryID,category)

Screenshots of the prototype Login-Menu



- ✍ entrance page ("index.wml")
- ✍ only static page of the system
- ✍ is a WML-form
- ✍ expects the user name and the user password as input
- ✍ input data are passed with the GET method to the "login.rxx"-script

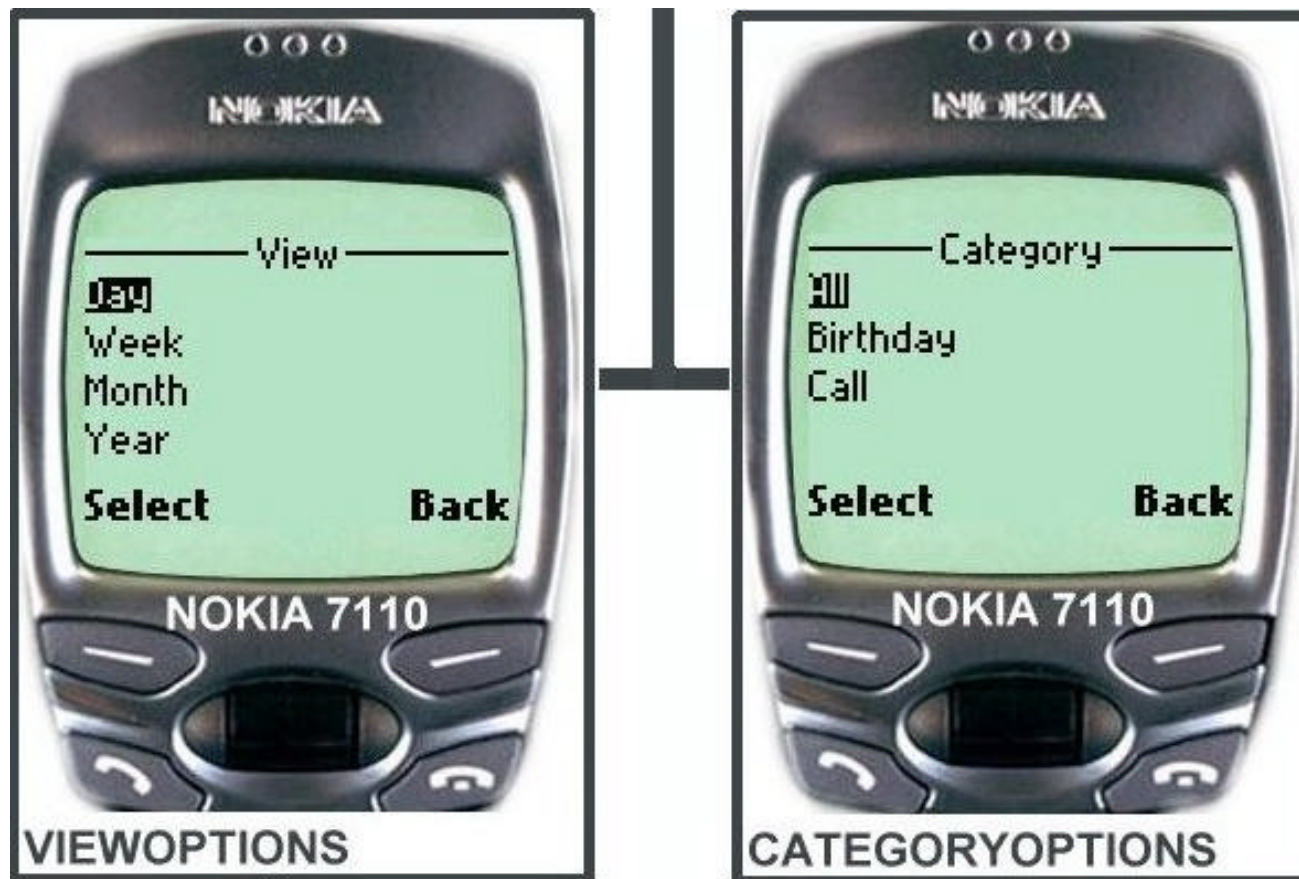
Screenshots of the prototype

Main-Menu (1/2)

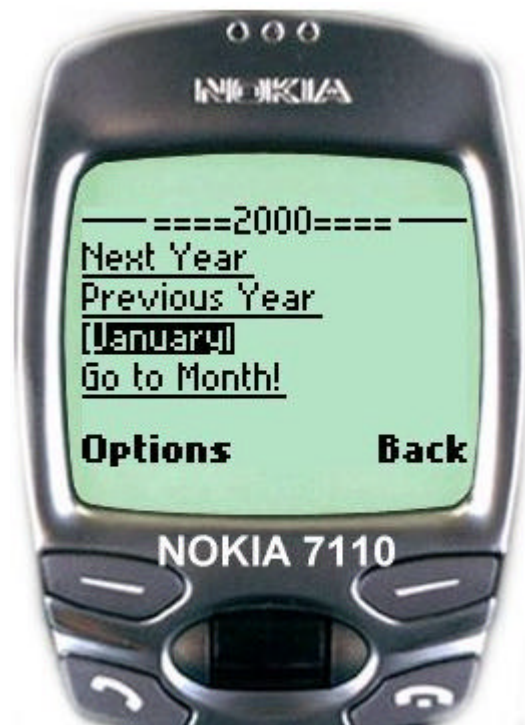


- ✍ After successful authentication the Main-Menu is displayed
 - ✍ „**View**“-type
 - day, week, month, year
 - ✍ Appointment-**“category”**
 - All, birthday, call, meeting
- can be selected by the user
- ✍ executes the “diary.rxx”- script
 - ✍ logout function is supplied in the option menu

Screenshots of the prototype Main-Menu (2/2)

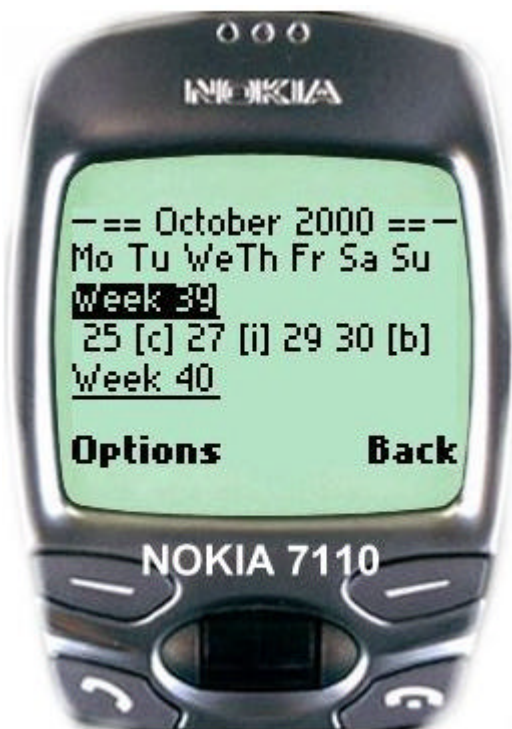


Screenshots of the prototype View-Types (1/5)



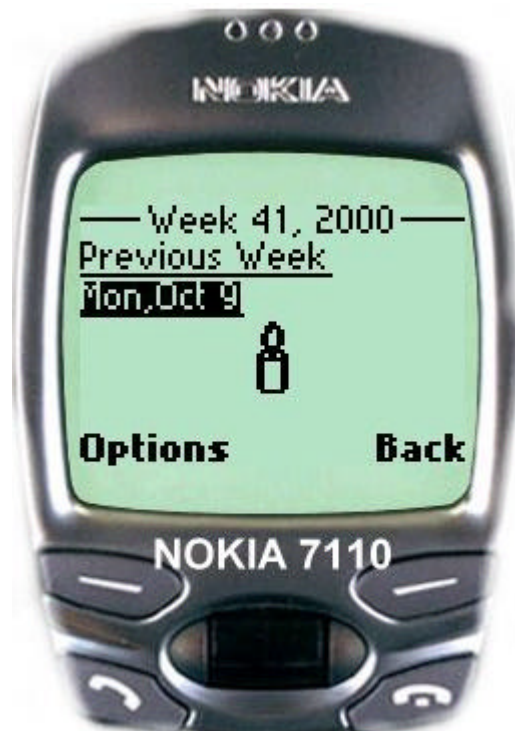
- ✍ „Yearview“
- ✍ Only scrolling between years available
- ✍ No direct access of e.g. the year 1436
- ✍ After selection of the desired year and month the **monthview** is displayed

Screenshots of the prototype View-Types (2/5)



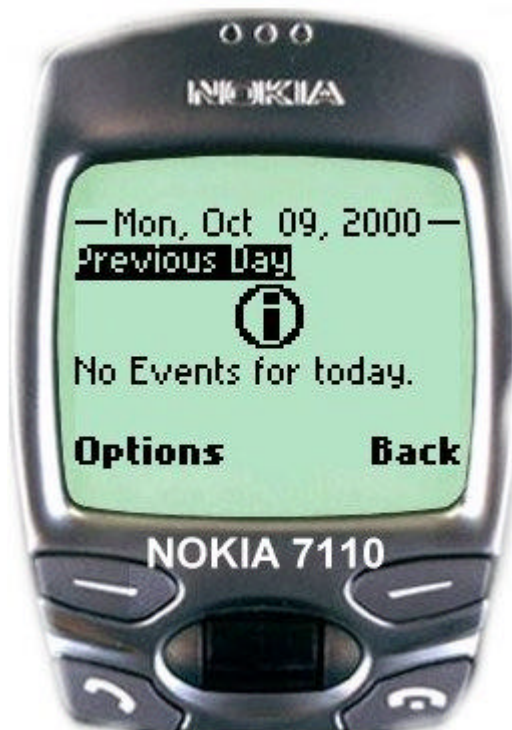
- ✍ „Monthview“
- ✍ Scrolling between month available
- ✍ different categories of appointments are differently signed in the week-overview
 - [c] – call
 - [b] – birthday
 - [i] – all
- ✍ After selection of the desired week the **weekview** is displayed

Screenshots of the prototype View-Types (3/5)



- ✍ „Weekview“
- ✍ Scrolling between days available
- ✍ Different categories of appointments are signed with different images
- ✍ When no events in database found image is not displayed
- ✍ After selection of the day the **dayview** is displayed

Screenshots of the prototype View-Types (4/5)



- ✍ „Dayview“
- ✍ Scrolling between days available
- ✍ if journal entries for the day exists they are displayed in this view-type
- ✍ In the options-menu is a link to the **Main-Menu**

Screenshots of the prototype View-Types (5/5)



Birthday



Call



Meeting



All

WBMP

- The image-type of WML
- WBMP (WML Bitmap)
- 2 Colors

```

```

Future work

- ✍ The system still offers no memory function by SMS or something similar. A memory function would complete the system as a commercial product and make it more competitive on the market.
- ✍ The data communication of the server to the Client could be made more secure with WTLS^[1] and protect by the way the system and the data users against attacks from outside

✍^[1]The WTLS Protocol – Wireless Transport Layer Security based on TLS 1.0