Ubiquitous Computing

UBIQUITOUS WEB SYSTEM

Ms. Lavanya G.V. / Mrs. Pushpa P.V.
Protocol Engineering & Technology (PET) Lab
Electrical Communication Engineering,
Indian Institute of Science,
Bangalore – 560012, India
What is a web application?

• A Web application is an application that was designed from the beginning to be executed in a Web-based environment.

• Definition reveals two very important aspects of such an application:
  • A Web application is designed from the start to run in a Web-based environment.
  • A Web application is an application, not just a set of Web pages. In particular, this implies that it enforces the notion of a session, which differentiates it from the ordinary request-response Web paradigm.
What is an ubiquitous web application?

• It is a Web application that suffers from the anytime / anywhere / anymedia syndrome.
• It should be designed from the start taking into account not only its hypermedia nature, but also the fact that it must run as is on a variety of platforms, including mobile phones, PDAs, full-fledged desktop computers, and so on.
• It must take into account the different capabilities of devices comprising display size, local storage size, method of input, network capacity, etc.
• It must be, on the one hand, context-aware i.e., aware of the environment it is running in, and on the other hand it must support personalisation.
Goal-Oriented Requirements Engineering

• The key achievement of this new approach is that it makes explicit the why of requirements.
• The requirements on data and operations were just there; one could not capture why they were there and whether they were sufficient.
• van Lamsweerde's approach provides three levels of modelling that allow to represent the ultimate objectives of the system (the goals), and shorter-term, more concrete objectives (the requirements), that operationalise the goals.
Web Application Engineering

• **Levels: content, hyperbase, and presentation.**
  • The first dimension of web application modelling comprises three different levels in terms of
    • content level
      • refers to domain-dependent data used by the web application and is often managed by means of a database system.
    • hyperbase level
      • denotes the logical composition of web pages and the navigation structure.
    • presentation level.
      • concerned with the presentation of the hyperbase level i.e., the layout of each page and user interaction.
  • Note that the emphasis of each of these levels depends on the kind of web application being modelled.
Web Application Engineering

- **Aspects: structure and behaviour.**
  - Comprises the aspects of structure and behaviour, which are orthogonal to the three levels of the first dimension.
  - at the content level, besides structuring the domain by means of standard abstraction mechanisms such as classification, aggregation and generalisation, the behavioural aspect in terms of domain-dependent application logic has to be considered.
  - at the hypertext level, structure in terms of page compositions and navigational relationships between them, as well as behaviour like computing the endpoint of a certain link at runtime have to be modelled.
  - At the presentation level, user interface elements and their hierarchical composition have to be modelled concerning the structural aspect.
Web Application Engineering

• **Phases: analysis, design and implementation.**
  • Comprises the different phases of a software life cycle, ranging from analysis via design to implementation.
  • This dimension is orthogonal to the two previously presented ones, meaning that structure and behaviour of content, navigation and presentation has to be addressed in each phase of the development process.
  • Currently, there is no consensus on a general process for web application development.
  • However, the influence of technological aspects tailoring the model towards the implementation environment, such as distribution, heterogeneity and database aspects, should certainly increase within the later phases of the modelling process.
Shortcomings of Existing Methods.

- **Behavioural Modelling**
  - Modelling the behavioural aspect of web applications at all levels is often neglected by existing methods.
  - If behaviour is considered then mainly at the presentation level.
  - Only those methods that are based on object-oriented modelling formalisms partly deal with behaviour modelling at all levels.

- **No Uniform Modelling Formalism.**
  - With the exception of those few approaches which fully rely on UML, the majority of modelling methods is based on a mix of mainly proprietary modelling formalisms.

- **No Process Support.**
  - Most modelling methods do not follow a process for guiding the activities throughout the development of a web application.
Shortcomings of Existing Methods.

- **Presentation Level not Captured by Conceptual and Logical Modelling Concepts.**
  - Most of the modelling methods do not support the presentation level with appropriate analysis and design concepts.
  - Authoring tools are often suggested for capturing the presentation level, thus losing the benefit of technology independence.

- **Lack of Customisation Support.**
  - One of the most severe drawbacks is the lack of concepts for customisation as needed by ubiquitous web applications.
  - The various approaches focusing on customisation are mainly implementation-oriented but do not provide proper concepts for the analysis phase and the design phases.
The Frame Work

1) Goal
2) Service
3) Environment
4) Context
5) Profiles
6) Requirements
7) Application
8) Customization Rules
9) The Meta level

Figure 1. The Overall Framework
Introduction to Customization
Motivation

• Ubiquitous Web Applications
  • have to be "aware" of their context
  • device, network, user, location, time
  • have to adapt accordingly

Adaptation to Context = Customisation

• Enables to exploit the possibilities of ubiquity as such personalised services
  - location-based services
  - Device – optimised services
Customization Characteristics

- Personalisation
  - Adaptive User Interfaces [Carrol88]
  - Adaptive Hypermedia [Good84]
  - Intelligent/Advisory Help and Tutoring Systems [Loeb92]
- Location-based services [Badrinath00]
- Network Adaptation [Want01]
- Multi-Channel Delivery [Eisenstein01]
- Information Filtering/Recommender Systems [Brusilowsky96]

Customisation

Anytime, anywhere, anymedia

User
Customization Characteristics

Core Web Application

Context

Mapping

Adaption

Ubiquitous Web Application

Customization
Customization Characteristics

• Context means the of “Situation of Use”
• Sensed from the environment
  • temperature, pressure, GPS-position
• Explicitly given information in terms of “profiles”
  • user profile, device profile, network profile
• Adaptation is responsible for appropriate changes of the web application to reflect the context
  • Consider current location in a car navigation system when directing a driver
  • Personalize the explanations in an eLearning application to reflect the pre-knowledge of a student
  • Delete pictures from a web page when displayed on a mobile phone or even:
  • Design a on-site personalized guided tour for tourists using various devices (combining a series of different adaptations)
Customization Characteristics

- External (proxy-based) Customisation
- most approaches e.g., IBM Transcoding Publisher Oracle Wireless Edition

Inter-Level Customisation
- e.g., WebML

Intra-Level Customisation
- e.g., OOHDM, WUML

Diagram:
- Presentation Level
- Hypertext Level
- Content Level
- HTTP-Request
- HTTP-Response
Context - Scope

- Property: aspects of the context which are relevant
  - Natural context: location, time
  - Technical context: device, browser, network bandwidth, application itself
  - Social context: user, user groups
- Extensibility: possibility to extend built-in by additional ones
  - certain applications may require additional context properties which are not built-in
  - current outside temperature or heartbeat rate in
  - it is not possible to foresee what kind of context properties that might be since the list of context properties is virtually unlimited
Context - Scope

• **Chronology:**
  - for which time span the context is considered
  - **Current context:** context at the time of the request, e.g.: current user
  - **Historical context:** previous context, e.g.: previously visited locations
  - **Future context:** prediction of future context, e.g.: prediction of throughput

• **Validity:** validity period during which a context is valid
  - Context properties may not be valid during the complete period until they are updated on basis of the environment
  - If e.g., location information doesn't change within certain period, the device might not be online any longer, thus, the location information might no longer be valid
Context - Representation

*Reusability:*

* A customisation approach should allow to explicitly represent context within the system, not just to intermingle context with adaptation or the application itself
* An explicit representation would allow for reusability of already defined context across several applications
* Another requirement for achieving reusability is that context provided by the customisation approach should be generic, that is application-independent
* Application independent context may even be reused from external sources, e.g., third-party providers
* An example would be a Location Service converting a physical cellID to an XY-position in space and vice versa or a GIS (Geographical Information System)
Context - Acquisition

- Automation: who is in charge of context acquisition
  - Automatic: context information is collected automatically, e.g.: usage analysis
  - Manual: explicitly given information, e.g.: user specifies his/her preferences
  - Semi-automatic: combination of automatic and manual

- It is desirable to automatically gather as much context information as possible to reduce user interaction

- Physical context properties may be sensed directly from the environment, logical context may be automatically computed on the basis of other context information available
  - Examples: Bandwidth available in the future constructed on basis of history of bandwidth

- Building user categories on the basis of interaction patterns
Modelling Ubiquitous Web Applications

Problem

- E-commerce and m-commerce heavily demand for UWAs in terms of the anytime/anywhere/anymedia paradigm
- Existing software engineering methods not sufficient for developing high UWAs

Goal

- Development of methods, notations and tools for designing and prototyping UWAs
- Explicit support for requirement elicitation, hypermedia modelling, transactional aspects and ubiquitous access
Modelling Ubiquitous Web Applications

• Approach

• Customisation model in terms of a UML profile

• Event/ Condition/Action rules as a means to deal with ubiquity

• Object-oriented framework to support pre-defined context models and to separate the core functionality of a web application (stable part) from its context-dependent part (variable part)

• Tool support to facilitate an integrated development process
Modelling Ubiquitous Web Applications

UML Profile

Requirements

Customisation Model

Context

Logical Context

Physical Context

Variable part

Stable part

Web Application

ECA Rules

Object-Oriented Framework

Environment

Service

Goals

Adaptation

maschine

world

reify

realises

satisfies

goals

trigger

actuates

determine

influence

operationalise

trigger

reify
Transaction design

- Transactions in web applications are critical for businesses.
- Web transactions can be complex. They may be,
  - composed of several sub-transactions,
  - accessing many different resources including existing legacy systems and
  - have complex semantics.
- Web transaction design needs to be very flexible allowing both
  - developing web applications from scratch by decomposing user level goals into sub-goals that (top-down design),
  - Using already existing systems or services to compose new applications offering added value services (bottom-up design).
Transaction Model

- Transaction models that provide for transactions with complex internal structure are known as extended transaction models (ETM).

  - Although the ETMs are valuable in many application domains relaxing some of the ACID transactional properties, they can't always deal with the full complexity that some modern ubiquitous web applications have.

  - Their limitations come mainly from their inflexibility to incorporate different transactional semantics in one (structured) transaction or to describe different behavioral.
AN EXAMPLE

- It is an example for an interactive, Web-based tourist guide, designed with the given methodology.
- Within requirements design, requirements are expressed as a directed acyclic graph going from higher-level goals to requirements proper (the leaves of the graph).

Figure shows a partial derivation graph including functional (e.g. Guide Tourists) and non-functional (e.g. Maximise Ubiquity) requirements of ubiquity and customisability.
• Figure shows a small fraction of the tourist guide's Navigation Design, modeling the Navigation Cluster for a tourist site.

• The Navigation Cluster contains the available nodes, holding information about the site's description, i.e.,
  • a short description (cf. TouristSight),
  • a full description of the tourist sight (cf. FullDescription), and
  • a route information (cf. RoutelInfo), as well as the links in between.

Partial Navigation and Customization Design
Ubiquitous Web Application Development - A Framework for Understanding
Profile Modelling

ProfileModel

Profile

1..*

UserAgent

getUserType()

getUserType()

getUserType()

getDeviceType()

getBrowserType()

graphicEnabled()

getUserName()

getStreet()

getLocalTime()

getBandwidth()

Figure 4. The Profile Model
Customisation Rules

Figure 5. The Customisation Rule Model
Customisation Rules

Figure 6. The Event Model
A Framework For Ubiquitous Web Service Discovery
Introduction

• The proliferation of computer-controlled devices into the physical environment and of electronic services into our everyday lives creates a demand for bridges between the physical world in which we live and work, and the virtual world of the computing services that assist us.

• The ubiquitous services that facilitate our actions in the real world should then have tangible, physical anchors that provide a means of discovery, creating a link between the physical space and information space.

• A framework is developed for the real-world discovery and invocation of ubiquitous services through physical, detectable tokens and a Web services approach to service invocation.

• The project includes implementation of a tool for generating bar-code based Web service tokens and a working platform for invoking a Web service on detection of such tokens.
Scenario

• A traveller arriving at a bus stop finds a barcode service token.
• Her mobile device reads the token through its inbuilt camera and decodes the service identifier.

The service interface description is retrieved and she is presented with travel services relevant for her journey from that location, such as finding out the time until the next bus arrives, or a search for the bus routes taking her to a particular location.
System Design

• This model are XML-based messaging, a service description, and service discovery according to a publish, find and bind paradigm,

• The service providers publish descriptions to a central registry,

• From Central Registry, descriptions are independently retrieved by service consumers

• Which is then bind directly with the provider to invoke the service.

• The service description is a Web Services Description Language (WSDL) document that specifies the service endpoints, protocols, operations and message formats.

• Service discovery involves the Universal Description, Discovery and Integration (UDDI) registry.

  • It is an open, industry-wide standard for distributed public registries exposing information about organisations and the Web services they provide.
System Design

Deployment diagram
System Design

• Here, token authoring tool is implemented.
• It accepts a WSDL file, such as that generated automatically by standard tools,
• Then generates suitable UDDI structures to publish the service description to the registry according to the latest guidelines.
• A barcode is then returned that confers the unique UDDI service identifier in a standard encoding, and is distributed in the real-world. e.g., on display at the bus stop.
• On sensing and decoding the barcode with a suitably equipped mobile device, a look-up call to a UDDI registry node returns an up-to-date reference to the service description that is retrieved
• It then invokes with a client interface that dynamically binds to the service endpoint according to the protocols and interfaces specified in the WSDL.
Conclusion

• The Web services model proved a powerful but lightweight framework for exposing service functionality in ubiquitous computing applications.
• Services can be deployed with ease and discovered in the physical environment via a barcode token and the UDDI registries.
• Though barcode tokens have been implemented, it has been envisioned that other mechanisms for conferring a token could be used.
• In particular, RFID tags, infrared and even coordinate location systems might be utilised. In addition, a security model may be implemented to establish support for commercial applications.