Web Information Systems Design
in the Era of Web 2.0 and Beyond
Part I: Web 1.0, 2.0, 3.0

iiWAS 2009
Kuala Lumpur, Tutorial
Dec. 14, 2009

(Hui Ma\textsuperscript{1}) & Klaus-Dieter Schewe\textsuperscript{2} & (Bernhard Thalheim\textsuperscript{3})

\textsuperscript{1}Victoria University, Wellington, New Zealand
\textsuperscript{2}Information Science Research Centre, Palmerston North, New Zealand
\textsuperscript{3}Christian-Albrechts-University Kiel, Germany

hui.ma@ecs.vuw.ac.nz \hspace{1cm} kdschewe@acm.org \hspace{1cm} thalheim@is.informatik.uni-kiel.de
Part I. Web Information Systems

From Simple Web to Web Information Systems

- Web Information System (WIS) is a database-backed information system that is realised and distributed over the web with user access via web browsers.
- A WIS is open to (almost) every user, but access in a particular role may be restricted.
- Websites have gone through three stages: Web1.0, Web2.0, Web3.0
  - Web1.0 is mainly author-driven
  - Web2.0 is user driven and content centered
  - Web3.0 is characterised by (4C + P + VS)
Web 1.0

Our past, present and also future

- Web 1.0 has mainly been oriented towards content provision to deliver content with a rudimentary functionality, e.g., navigation, acquisition information, linking, search and browse.
Achievements of Web 1.0

- Resulting websites are simple to use, without any learning effort
- It is based on an application development according to application scenarios
- It models story space as schemes for utilization
- It uses security techniques provided by server-sided logic and aiming being robust
- It uses taxonomies for classification and systematisation
Limitations of Web 1.0

- It is often resulted in high time exposure for stepwise page buildup
- It is based on fixed page content
- It uses information dissemination from a central source
- It pushes information to users
- It is based on a unidirectional communication
- It does not providing relation to desktop organization
Web 2.0 after Web 1.0

Web 2.0: Involving the user, enriching infrastructure, new models and services

Web 1.0
“the mostly read-only Web”
250,000 sites
1996
published content
user generated content
46 million global users

Web 2.0
“the wiki-ly read-write Web”
80,000,000 sites
2006
published content
collective intelligence
user generated content
1 billion+ global users

Web x.y
Applications 2.0
Online Communities
Mashup Containers
Web Identity
Wikis
Pitfalls
WIS Co-Design

Media types

Story
Actor
# Top 20 Web 2.0 Webseiten (US)

<table>
<thead>
<tr>
<th>Rank</th>
<th>Service</th>
<th>Market Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MySpace</td>
<td>44.14%</td>
</tr>
<tr>
<td>2</td>
<td>eBay</td>
<td>16.22%</td>
</tr>
<tr>
<td>3</td>
<td>Facebook</td>
<td>13.03%</td>
</tr>
<tr>
<td>4</td>
<td>YouTube</td>
<td>9.88%</td>
</tr>
<tr>
<td>5</td>
<td>Wikipedia</td>
<td>4.76%</td>
</tr>
<tr>
<td>6</td>
<td>Craigslist</td>
<td>3.14%</td>
</tr>
<tr>
<td>7</td>
<td>Yahoo Answers</td>
<td>1.41%</td>
</tr>
<tr>
<td>8</td>
<td>myYearbook</td>
<td>1.05%</td>
</tr>
<tr>
<td>9</td>
<td>Tagged</td>
<td>.95%</td>
</tr>
<tr>
<td>10</td>
<td>Flickr</td>
<td>.86%</td>
</tr>
<tr>
<td>11</td>
<td>Bebo</td>
<td>.71%</td>
</tr>
<tr>
<td>12</td>
<td>Meebo</td>
<td>.59%</td>
</tr>
<tr>
<td>13</td>
<td>BlackPlanet.com</td>
<td>.52%</td>
</tr>
<tr>
<td>14</td>
<td>GaiaOnline.com</td>
<td>.45%</td>
</tr>
<tr>
<td>15</td>
<td>Blogger</td>
<td>.40%</td>
</tr>
<tr>
<td>16</td>
<td>Adam4Adam</td>
<td>.35%</td>
</tr>
<tr>
<td>17</td>
<td>hi5</td>
<td>.35%</td>
</tr>
<tr>
<td>18</td>
<td>WikiAnswers</td>
<td>.33%</td>
</tr>
<tr>
<td>19</td>
<td>IMEEM</td>
<td>.31%</td>
</tr>
<tr>
<td>20</td>
<td>LiveJournal</td>
<td>.30%</td>
</tr>
</tbody>
</table>

Source: weblogs.hitwise.com/us-heather-hopkins, August 05, 2008
Web 2.0 and Trends, e.g., E-Business

Main trends:

- Emergent collaboration approaches
- New production and distribution strategies
- New products, e.g., turning products into open platforms
- Story-driven depending on the refinement of story pattern
- Context-sensitivity depending on user, culture, product, activity
Promises of Web 2.0

- Web 2.0 extends Web 1.0 by collaboration of user communities and based on sophisticated story spaces
- Typical applications: GoogleAdSense, Flickr, Wikipedia
Promises of Web 2.0 (con.)

- It improves operability of websites because everyone is an author, producers, evaluator and user
- It is based on new design patterns, intelligent forms and validation
- It supports dynamic user guidance and actualisations
- It provides drag and drop functionality using the desktop metaphor
- It is based on replication of data to reduce network load
Towards Enterprise 2.0: Extending Core Pattern of Web 2.0

Harnessing Collective Intelligence
Data is the Next “Intel Inside”
Innovation in Assembly
Rich User Experiences
Software Above the Level of a Single Device
Perpetual Beta
Leveraging the Long Tail
Lightweight Software/Business Models and Cost Effective Scalability
Towards Enterprise 2.0

- SOA finally goes pragmatic, Web-oriented, and lightweight.
- Enterprise search will remain broken or highly limited in most organizations.
- Security will become a major concern as Web 2.0 apps and SaaS make the edge of enterprises increasingly porous.
- Unstructured information from blogs and wikis will proliferate, driving demand for solutions to extract and consolidate business information.
- A large number of enterprise intranets will get social networking capabilities.
- Enterprise mashups will make their appearance in a widespread way but won't take off big, yet.
- Budget for next generation governance infrastructure will be demanded by IT to manage the proliferation of SaaS, new SOA, and mashups.
- Collective intelligence platforms and enterprise decision management support applications will see significant early adoption this year.
- New rich user experience platforms such as Adobe AIR and Microsoft's Silverlight will see widespread use in the enterprise in 2008.
- Mobile devices, such as the iPhone and especially the BlackBerry, will see significant uptake for embedding workers into business processes and Web-based collaboration.
- Web 2.0 versions of traditional enterprise applications will struggle for marketshare amongst the software giants.
- Enterprise 2.0 will become a standard feature in most organizations, while views on the outcomes of deployments in organizations will be viewed through the lens of predisposition.
Elements and Drivers of Enterprise 2.0

New platforms change the shape of Enterprise 2.0

- **Search** – Discoverability of information drives reuse, leverage, and ROI.
- **Links** – Using URIs to forge thousands of deep interconnections between enterprise content 24/7.
- **Authorship** – Ensuring every worker has easy access to Enterprise 2.0 platforms.
- **Tags** – Allowing natural, organic, on-the-fly organization of data from every point of view.
- **Extensions** – Extend knowledge by mining patterns and user activity.
- **Signs** – Make information consumption efficient by pushing out changes.

Media types

**WIS**

Story  Actor
Reasons, challenges, opportunities:
- Ease-of-use, first and foremost
- Providing motivation by changing the environment
- Discoverability is a core and not an afterthought
- It’s OK to fear loss of control and misuse
- Dynamic, effective advocates are a key enabler
- The problems will be with the business culture, not the technology
- Triggering an Enterprise 2.0 ecosystem quickly is likely an early activity driver
- Allow the tools to access enterprise services
The New Challenge: Office 2.0

1996: concept for a Network Computer did not really make sense
Escaping the tyranny of behemoths for installing software and updates
Novel visual technologies meeting everyday office life
Driven by AJAX and at least one killer application Gmail

Three challenges:
- Security beyond sysadmin;
- Confidentiality without breaches from outside instead of inside;
- Connectivity to the real needs
Cloud computing: Rewards and Risks

Types of cloud computing services

- Compute clouds e.g. Amazons EC2, Google App Engine, and Berkeleys BOINC

- Cloud storage
- Cloud applications

Cloud computing significantly changes many aspects of enterprise computing acquisition, operations, and governance

- Usually though not always for the better. These aspects are:
  - Reduced capital expenditures
  - Low barrier to entry
  - Multitenancy
  - Security
  - Scalability and performance
  - Centralization vs. federation
  - Service-oriented
Cloud Computing @Google and @Amazon

Differences:

- Operational capability of the platform hosting
- Implications to both the business models of PaaS vendors as well as their customers
- Good for the startup community; but what about the enterprise?
A rich web of REST resources.

Simple tools to weave the Web of resources into new applications.

Highly consumable and reusable WOA “parts” including widgets, gadgets, and embedded social applications.

Open Web APIs exposed on the Internet to ad hoc partners.

Maintaining core ideas behind SOA and enhancing by Web 2.0.

The enterprise environment is much different from the consumer space.
Merging SOA and Cloud Computing into Web-Oriented Architectures

Big difference results in these two very different environments because of:

- Control
- Cost structure
- Mandate vs. necessity

The last major difference between enterprises and Internet companies: Service models!

WOA: Radically distributed, granular, web-oriented, open, highly consumable
Web-Oriented Architectures (WOA)

- Information is represented in the form of resources on the network and are accessed and manipulated via the protocol specified in the URI, typically HTTP.
- Every resource on the network can be located via a URI. Access to resources must be layered and not require more than local knowledge of the network.
- Resources are manipulated by HTTP verbs (GET, PUT, POST, DELETE) using REST (Representational State Transfer).
- Manipulation of network resources is performed solely by components on the network (essentially browsers and other Web servers).
- It is the responsibility of the components to understand the representations and valid state transitions of the resources they manipulate.
- The service contract of WOA resources is implicit; it’s the representation that is received. WOA embodies Thomas Erl’s essential Principles of SOA.
- WOA resources contain embedded URIs that build a larger network of granular representative state (i.e. order resources contain URLs to inventory resources).
Wide Variety of Stories for Online Communities

Power law of participation

Activities:
- Tagging (URL is the hardest part) and commenting
- Subscribing requires a commitment
- Sharing is the principal activity
- Network not only to connect, but leverage as a filter to fend off information overload
- Write (e.g., blog) and have conversations
- Refactor, Collaborate, Moderate and Lead
Online Communities: Towards Pull-Based Stories

Online communities are changing to become more active.

Infocult: Information, culture, policy, education

Information: Its culture, history, and role in teaching and learning.
Consumer Communities: Towards Pull-Based Stories

Consumer communities must be considered together with their context.

Web services as mediator between customer and provider

Customer requests a service
Best Practices for Online Customer Communities

- Put the needs of the community first.
- Community is mostly not a technology problem. e.g., OpenSocial, DataPortability.org, Drupal, Joomla, or DotNetNuke
- Active community management is essential.
- Measuring success with community requires new yardsticks.
- Consumer social networks, grassroots customer communities, and business-initiated customer communities are closely related yet very different creatures.
- Customer communities do work as a marketing channel, just not in the traditional way.
- The more the business is integrated, the better the community will work.
- Growth will come, but not until a community finds its identity.
- Mutual ownership and control of communities enables trust and involvement.
- Going to the community, instead of making it come to you, is a risky but increasingly viable strategy.
- Connect the community with the other CRM-related aspects of the organization.
Standards Support for Mashups Emerge

Mashups turn into an industry as offerings mature

DreamFaces new Mashup Kits, Intels Mash Maker, JackBes new Presto 2.0 platform

Enterprise mashups get ready for prime-time

Mashups aren’t just browser-based integration, but can be server-side too

Web-based applications created as easily as using a spreadsheet by assembling
Organic Versus Tool-Assisted Mashups

The opposite end of mashup technology: From natural and emergent to tools-assisted and highly-structured

- Effective leverage of Web parts and the Global SOA.
- Simple, lightweight software models and services.
- A focus on self-service and DIY.
- Providing effective “enterprise context”.
- Distribution and consumption.
- SEO, analytics, page views are all challenged by the mashup model.
Merging Technologies: Enterprise 2.0 + Mashups

- No commonly accepted assembly model.
- An immature services landscape.
- The splintering of widgets.
- Management and support of end-user mashup apps.
- Deep support for security and identity.
- Data quality and accuracy.
- Version management.
- Awareness and realization of the potential of mashups by the businesses community.
- Low levels of support by major software firms.
- Few killer demo mashups.
openid: The once and Future Enterprise
Single Sign-On?

(1) Step today: choose their preferred user account provider for logging into other Web sites.

(2) Step tomorrow: choose your preferred service for your identity management

(3) Step intended: choose your environment wherever you are and whatever you are using
Enterprise ID and OpenID

- OpenID is a free and easy way to use a single digital identity across the Internet.
- With one OpenID you can login to all your favorite websites and forget about online paperwork!
- Now, you get to choose the login that’s right for you.
- Challenges that openid faces to increase adoption and use; security and usability.
- Future work: Authentication is out of scope for OpenID. Information Cards solve a different problem than OpenIDs. Nobody is currently really adopting OpenID.
Wikis for Everything?

Wiki:

piece of content
edited by a community
for a certain purpose
presented in a variety of forms

E.G., enterprise context for wikis:

- Integrated enterprise security (SSO, LDAP, Identity 2.0, etc.)
- Content auditing and policy support (automated)
- Open and extensible (inbound and outbound integration points)
- Support for enterprise search
- Internal and external hosting options
- Integration with standard workflow products and standards (BPEL, for instance)
Wikis and Inherent Quality Problems

- Web desktops
- Badges
- Blogs
- Wikis
- Feeds
- Enterprise 2.0

- Simple mashups
- Situational apps
- Widgets
- WOA

- Web apps
- Enterprise mashups
- Composite apps
- Office 2.0
- SOA

- Anyone
- Expert users
- Developers

- Media types
- Story
- Actor

- Wikis
- Pitfalls
- WIS Co-Design
The Quality Dimension of Content

Wikipedia accessed May 28, 2008

Dresden: (etymologically from Old Sorbian Drež’d’any, meaning people of the riverside forest ?, Upper Sorbian: Drježdz’any?) is the capital city[4] of the German Federal Free State of Saxony. It is situated in a valley on the River Elbe. The Dresden conurbation is part of the Saxon Triangle metropolitan area.[5] ?

Dresden has a long history as the capital and royal residence for the Electors and Kings of Saxony, who for centuries furnished the city with cultural and artistic splendour. The controversial bombing of Dresden in World War II by the British Royal Air Force ! and the United States Air Force !, plus 40 years in the Soviet bloc state of the German Democratic Republic as well as contemporary city development has changed the face of the city broadly !. Considerable restoration work has settled the damage ?.

Since German reunification in 1990, Dresden has emerged as a cultural, political, and economic centre in the eastern part of Germany. ?

Location

Dresden lies on both banks of the river Elbe, mostly in the Dresden Elbe Valley Basin, with the further reaches of the eastern Ore Mountains to the south, the steep slope of the Lusatian granitic crust to the north?, and the Elbe Sandstone Mountains to the east ? at an altitude of about 113 ? meters. The highest point of Dresden is about 384 meters in altitude.[6]

With a pleasant location and a mild climate on the Elbe, as well as Baroque-style architecture ? and numerous world-renowned museums and art collections, Dresden has been called “Elbflorenz” (Florence of ? the Elbe). The incorporation of neighboring rural communities over the past 60 years ? has made Dresden the fourth largest urban district by area in Germany after Berlin, Hamburg, and Cologne.[7] ? ...
The Quality Dimension of Content

Encyclopedia Britannica 2003 edition edited quality

Dresden: city and capital of Saxony Land (state), eastern Germany. Dresden is the traditional capital of Saxony and the third largest city in eastern Germany after Berlin and Leipzig. It lies in the broad basin of the Elbe River between Meissen and Pirna, 19 miles (30 km) north of the Czech border and 100 miles (160 km) south of Berlin. Sheltering hills north and south of the Elbe valley contribute to the mild climate enjoyed by Dresden.

It originated as the Slav village of Drezdzany, meaning “Forest Dwellers on the Plain,” on the Elbe’s north bank. The town on the south bank was founded at a ford by Margrave Dietrich of Meissen as a German colony, first recorded in 1216. The Slav settlement on the north bank, although older, was known as Neustadt (“New Town”) and the later German town on the south bank as Altstadt (“Old Town”).

In 1270 Dresden became the capital of Margrave Henry the Illustrious, and after his death it belonged to the king of Bohemia and the Margrave of Brandenburg until it was restored about 1319 to the margraves of Meissen, who chartered it in 1403. On the division of Saxony in 1485 it became the residence and capital of the Albertine line of Wettin rulers, later electors and kings of Saxony. Dresden accepted the Protestant Reformation in 1539. After a disastrous fire in 1491, the city was rebuilt and fortified. The electors Augustus I and Augustus II modernized the city in the Baroque and Rococo styles in the late 17th and 18th centuries, rebuilding Neustadt (burned in 1685) and founding Friedrichstadt, northwest of Altstadt. The Treaty of Dresden (1745), among Prussia, Saxony, and Austria, ended the second Silesian War and confirmed Silesia as Prussian. Two-thirds destroyed in the Seven Years’ War (175663), Dresden’s fortifications were later dismantled. In 1813 Napoleon I made the town a centre of military operations and there won his last great battle on August 26 and 27. Dresden’s prosperity grew rapidly...
Who is better? Wikipedia or Encyclopedia?

Essentially both are right!!!

Information demand of the user for

(1) content ("what"),
(2) formatting and presenting ("how"),
(3) functionality ("how to use"),
(4) timeliness ("when"),
(5) quality ("personal or objective measure"),
  controlled completeness, partial correctness, relative truth
(6) price/value ratio ("who pays for it")

Story of the user / user community / content evolution

Biases and intensions based on the culture of people involved
  providers, suppliers, editors, politics
groups

Meta-data for the user
The Risks of Web 2.0

- The web becomes more fault afflicted because content are often produced by users
- It needs more investments for tests
- It increases the manipulation risk of sites
- It results data mining attacks
- It extends the area of spear phishing and sophisticated spam
- It does not provide copyright protection
Towards Web 3.0 Beyond Web 2.0

Avoiding already existing pitfalls

- Web 3.0 is characterised by the formula \((4C + P + VS)\), where:
  - 4C denotes Content, Commerce, Community and Context,
  - P denotes Personalization and
  - VS denotes Vertical Search.

- The evolution of Web x.y compared with its 'semantification'

Problems in Web Information Systems Development

- Major difference to Enterprise Information Systems: a WIS is open to (almost) every user
  - no user involvement in system development, no pre-determined skill set, no usage training
  - giving up, looking for alternatives in case of non-satisfaction

- Challenge: a WIS must be comprehensible, acceptable and customisable to all kinds of users, their preferences, goals and preferred technical environment and access channels, integral with respect to requirements and consistent with respect to (static, dynamic, deontic) constraints
Experiences with Web Information Systems

- Many WISs are still developed in an ad-hoc way, not using appropriate methodology:
  - page not found or does not contain what was expected, page overloaded
  - link points to general entry point ignoring context
  - search produces heaps of pages, in one of which the desired information may be found
  - repeated information, repeated requests for data entry
  - etc.
Need for WIS Development Methods

- In general, the need for methodological development arises, if systems tend to become large, if content changes frequently, if complex tasks are to be supported, if multiple users use the system.

- WIS often serve purposes such as booking, selling, edutainment, etc., i.e. users are customers: giving up and turning away is highly undesirable.

- The web is an attractive (and fast) business channel for communication, advertisement, commerce: systems have to serve the business goals.
Existing WIS Development Methods

- Most WIS development methods (OOHDM, ARANEUS, HERA, WISM, WebML, UWE, Co-Design, etc.) concentrate on the data-intensive aspect: views on some underlying database schema.

- All methods support the generation of pages and layout.

- Some (Co-Design) emphasize also the modelling of users/actors, tasks and action schemes/plots.

- Some (WISM, Co-Design) emphasize personalisation.

- Strategic modelling is rarely addressed (except Co-Design).
Web 1.0 and Co-Design

Web 1.0: Established web engineering

Web 1.0

Content

Functionality

Data objects knowledge

Presentation

Interfaces depending on the environment

Navigation search work

Web 1.0 is oriented towards content

Web 1.0 provides some functionality

Web 1.0 provides a variety of presentation
From Web 1.0 to Web 2.0

**Web 1.0:**
- **Content:**
  - Data objects
  - Knowledge
- **Functionality:**
  - Navigation
  - Search
- **Presentation:**
  - Interfaces depending on the environment

**Web 2.0:**
- **Content:**
  - Data objects
  - Knowledge
- **Functionality:**
  - Navigation
  - Search
- **Presentation:**
  - Interfaces depending on the environment

**Goal, application area**
- Profile, information demand

**(((Context)))**
- Technics
- Organisation
- WIS context

**(((User and intention)))**
- Web 2.0 collaboration
- Group content
- Collective identity

**WIS Co-Design**

**Web x.y**
- Applications 2.0
- Online Communities
- Mashup Containers
- Web Identity
- Wikis
- Pitfalls

**Media types**
- WIS
- Story
- Actor

**Stories**
- Tasks

**Web 2.0:**
- User driven, content centered, GoogleAdSense, Flickr, Wikipedia, blogs, optimised search engines, pay per click, web services, participate instead
- Be attracted, tagging, syndication
- Common usage of bookmarks, clicks; communities; tracking goals of usage; data ownership, portability, economics, transparency; architectures of participation

**Aal/LOW principle:** let others work
Web 2.0: Return of Engagement

Beobachtungen

(1) Cost reduction and improvement of customer service
   customer needs understanding, customer participation

(2) Cross selling and upsell

(3) 60-60-24-7 thermometer for R&D innovation
   customer insight, feedback

(4) Brand, reputation and social management
   web brand: product brand + community brand
   trusted advice, embrace and leverage instead of control, customer
   loyalty, emotional brand building

(5) Virality, traction
   value cloud with shared information, help, trust, recommend, herd
   instinct
**From Web 1.0 to Web 3.0**

Web 1.0:

- **Content**
  - Data objects
  - Knowledge

- **Functionality**
  - Navigation
  - Search
  - Work

- **Presentation**
  - Interfaces depending on the environment

Web 3.0:

- **Content**
  - Data objects
  - Knowledge

- **Functionality**
  - Interfaces depending on the environment

- **Presentation**

**Web 1.0**

- Author driven, publish/provide story/support or advertise/wait/attract/react/retain for users:
  - Inform/subscribe/obtain/answer/come back

**Web 3.0**

- Asset driven, content-asset centered, provides additionally linguistic semantics,
  - Technology combines: artificial intelligence, automated reasoning, cognitive architecture, composite applications, distributed computing, knowledge representation, ontology (computer science), recombinant text, scalable vector graphics, semantic Web, semantic Wiki, software agents
Web Information Systems Design in the Era of Web 2.0 and Beyond
Part II. Service Orientation / WIS Co-Design

iiWAS 2009
Kuala Lumpur, Tutorial
Dec. 14, 2009

(Hui Ma\textsuperscript{1}) & Klaus-Dieter Schewe\textsuperscript{2} & (Bernhard Thalheim\textsuperscript{3})
\textsuperscript{1}Victoria University, Wellington, New Zealand
\textsuperscript{2}Information Science Research Centre, Palmerston North, New Zealand
\textsuperscript{3}Christian-Albrechts-University Kiel, Germany

hui.ma@ecs.vuw.ac.nz \quad kdschewe@acm.org \quad thalheim@is.informatik.uni-kiel.de
Abstraction Layer

Application domain layer
- Scoping
- Requirements acquisition layer
- Structuring specification
- Distribution specification

Business user layer
- Variating
- Designing
- Conceptual layer
- Structuring specification
- Dialog specification

Implementation layer
- Implementing
- Conceiving
- Functionality specification

Conceptual layer
- Story
- Actor

Media types
- WIS
Abstraction Layer Models

Separation by Level of Detail

Application domain layer concerned with description of the application

Requirements acquisition layer concerned with prescription of system requirements

Business user layer concerned with behaviour or users, their demands to the system

Conceptual layer concerned with specifications (schemata) that describe the system

Implementation layer concerned with logical and physical (specifications and) programs

Deployment layer concerned with introduction, usage, maintenance, evolution of the system
Classical dichotomy: Human-computer systems and information systems

Description/prescription layer

Design Refinement

Application area description

Requirements prescriptions

WIS description and prescription

Information system specification

Presentation system specification

WIS specification

Information system

Web information system

Conceptual layer

Implementation layer

Media types

WIS

Story

Actor

Media Types

Screenography

Pragmatics

Strategic Modelling

Storyboarding
Dichotomy of human-computer systems and the software systems

Description/prescription layer

Conceptual layer

Implementation layer

Design
Refinement

Application area description

Requirements prescriptions

WIS description and prescription

Presentation system specification

Information systems specification

WIS specification

Presentation system

Information system

Web information system

Design

Transformation

Implementation

Description/prescription

Conceptual

Implementation
Application Domain Modelling

- Strategic modelling clarifies the purpose of the system and expectations associated with it
- Strategic modelling identifies target users, their intentions and tasks to be executed by them
- Strategic modelling identifies the content and functionality items in a rough way
- Strategic modelling specifies ambience and progression model for the system
- Strategic modelling reflects the business model: excite Attention, arouse Interest, stimulate Desire, provoke Action
Elements of Application Domain WIS Modelling

- The application domain model consists of:
  - a *mission statement and brand* describing in general terms what the WIS is about
  - a *utilisation space* describing content, functionality and context
  - a *utilisation portfolio* describing actors, goals and tasks
  - and *general principles* describing the ambience and desired atmosphere of the WIS
  - based on the description of *life cases*
    - characterising the user, the desire, the demands
    - reflecting tasks (portfolio) of the user and
    - considering the context of the application domain with filters for stereotypes of usages
  - The last part will guide the layout and playout design of the WIS
Mission Statement and Brand

- The *brand* of a WIS adds details to the classification scheme $P^W_{2U}A$ for WISs giving a rough picture of content, functionality and users using only descriptive keywords, e.g. bank\textsuperscript{loan}, mortgage\textsuperscript{2customer}apply\textsuperscript{for\_loan}, apply\textsuperscript{for\_mortgage}

- The brand is usually the result of a brainstorming activity discussing the what, whom and for whom of the WIS

- The *mission statement* complements the brand by an informal, textual description taking each of the actions in the brand as the major tasks, adding which types of users are involved, which activities are supposed to be executed, which content will be provided for them and requested from them, what will be the results of these activities

- In addition, the mission statement will describe what the major and minor purpose of the system is, how each task will contribute to these purposes, what the benefits of the system for the provider and the users will be
Example (cont.)

- Using a tabular semi-formal description, the mission statement for the brand is as the following

<table>
<thead>
<tr>
<th>Content</th>
<th>loan, mortgage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>customer</td>
</tr>
<tr>
<td>Tasks</td>
<td>apply_for_loan, apply_for_mortgage</td>
</tr>
<tr>
<td>Major Purpose</td>
<td>open an additional sales channel</td>
</tr>
<tr>
<td></td>
<td>address technology-experienced customers</td>
</tr>
<tr>
<td></td>
<td>address informed, goal-oriented customers</td>
</tr>
<tr>
<td>Minor Purpose</td>
<td>improve banking efficiency in loan sector</td>
</tr>
<tr>
<td>Benefits</td>
<td>closer binding of customers</td>
</tr>
<tr>
<td></td>
<td>attraction of new customers</td>
</tr>
<tr>
<td></td>
<td>improvement of cost efficiency</td>
</tr>
<tr>
<td></td>
<td>increased availability of bank services</td>
</tr>
</tbody>
</table>
Example (cont.)

Furthermore, we obtain the following informal descriptions for the tasks apply_for_loan:

<table>
<thead>
<tr>
<th>Task: apply_for_loan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description: The system will provide information about the available types of loans including conditions for repayment, i.e. principal and interest, conditions for creditworthiness, and intended purposes of the loans. The information about loans will be complemented by easy loan examples. Then the system will allow customers to enter their personal data, select the appropriate loan, and check, whether their personal finances are in accordance with the rules for repayment.</td>
</tr>
<tr>
<td>Participants: customer</td>
</tr>
<tr>
<td>Required Content: list_of_loans, loan_conditions, loan_purpose</td>
</tr>
<tr>
<td>Produced Content: loan_application, customer_data</td>
</tr>
<tr>
<td>Result: confirmed_loan_application</td>
</tr>
</tbody>
</table>

the description of the task apply_for_mortgage will look similar
Utilisation Space

- “Utilisation space” is used as a metaphor to characterise the WIS as a space, through which a user can navigate, thus enabling optimal orientation in the utilisation space.

- The what-part of the brand uses nouns describing the content in coarse terms; the what_for-part of the brand gives verbs describing the functionality, i.e. what to do with the content.

- The utilisation space adds details to content and functionality, sets the nouns and verbs used in the brand into relation, places both into a utilisation context.

- Refine the content/functionality keywords and place them in semantic relationships: specialisation, part-of relationships, or associations of global context with details.

- Relate the functionality with the content, i.e. specify which content is needed by which activity, which content is produced by which activity, in which order (if any) the content will be used by an activity.
Utilisation Portfolio

- The utilisation portfolio emphasises the whom-part of the brand: the WIS users (actors), their goals and tasks
  - Tasks correspond to the actions in the brand and their refinement in the utilisation space
  - Users used in the brand and mission statement will be roughly classified according to roles
  - A goal corresponds to task that is meant to achieve this goal
  - Tasks are broken down into subtasks to a level that elementary tasks can be associated with a single role; subtasks should refer to subgoals
  - Furthermore, we obtain dependencies between goals
Ambience of a WIS

- The ambience addresses which impression the WIS as a whole should convey to its users
- **Ambience categories** can be energetic, romantic, elegant, refreshing, harmonic, or stimulating, etc., each having implications on the **layout**, in particular choices of form and ground colouring schemes
- In addition, the ambience of a WIS is concerned with the progression patterns for the tasks reflecting the logical connection of information revealed to the user
- **Progression patterns** can be *circular* (centered around a particular content item), *incremental* (emphasising the development of several content items over time), etc.
- Progression patterns impact on the **playout**, i.e. the arrangement of available functionality
Methods Applied in Strategic Modelling

- **Brainstorming** for branding and definition of mission
- Linguistic analyses of **word fields**
- Noam Chomsky’s theory of **governance and binding** (theoretical linguistic)
- Communication analysis, in particular with respect to exchanged messages and possible communication barriers
- Knowledge of **cognitive psychology** addressing the effect of ambiguity on users
- Documentation and discussion means: text, templates, semantic trees, task-goal graphs
Pragmatics of Storyboarding

- **Usage analysis:**
  - Life Cases: How the stories match the users’ intentions and expectations
  - User Models: Which user/actor profiles and portfolios have to be considered
  - Contexts: Everything that surrounds and thus impacts on a utilisation situation

- **WIS portfolio:**
  - content and utilisation chunks

- **Starting point: Analysis of intentions**
  - Intentions are coarsely formulated as part of a strategic WIS model (mission, purpose, goals)
  - Utilisation scenarios are developed on the basis of intentions
  - Therefore, address the description of intentions:
    - Clear understanding of aims and targets of the WIS
    - This includes long-range and short-term targets, expected visitors, characteristics of this audience, tasks performed by the users, necessary and nice-to-have content, restrictions of usage, etc.
Storyboarding

- For WIS development conceptual data and interface modelling has to be embedded in an analysis process: *usage model*

- This leads to *storyboarding*, the main activity on the business layer

- A storyboard specifies a WIS on a high level of abstraction of a WIS: who will be using the system, in which way and for which goals

- A *storyboard* consists mainly of story space, plot, actors, and tasks:
  - A *story space* consists of a hierarchy of labelled directed graphs called *scenarios* describing scenes, scene transitions, and actions
  - A *plot* is an assignment-free process, in which the basic actions correspond to the labels of edges in the scenarios
Storyboarding (con.)

- An **actor** is an abstraction of a user group defined by **roles**, which determine obligations and rights, and **user profiles**, which determine user preferences.

- A **task** is associated with a **goal** of a user.

- In addition, there are many constraints comprising static, dynamic and deontic constraints for pre- and postconditions, triggering and enabling events, rights and obligations of roles, preference rules for user types, and other dependencies on the plot.


- The pragmatics of storyboarding will be discussed later on.
Syntax of Storyboarding

- For story spaces: labelled directed graphs (or matrices, semi-formal templates)
- For plots: story algebra
- For constraints: propositional deontic logic
- For user profiles: dimension/value sets, Kiviat graphs
- For preference rules: conditional equations on the plot
- For tasks: task-goal graphs, labelled directed graphs
- ...
Semantics of Storyboarding

- Stories: Paths through the story space
- Role-oriented or task-oriented views
- Kleene algebras with tests (KATs) for plots
- Rewriting in KATs for personalisation and other reasoning tasks
- Task satisfaction/enforcement through deontic inferencing
- ...
The Story Space

- Regard a location together with local actions, i.e. actions that do not change the location, as a unit called scene.
- Thus, describe the story space (or parts of it) by a labelled directed graph called scenario:
  - The vertices represent the scenes.
  - The edges represent transitions between scenes.
- A story is a path in the story space.
- At a finer level of details we may add events, a precondition and a postcondition to each action.
Adding Details to Actions and Scenes

- **Pre- and Postconditions:**
  - associate a *precondition* with each action $\alpha \in A$ to specify exactly, under which conditions the action can be executed.
  - $\mathcal{P}re(\alpha)$ is a propositional formula in $\mathcal{F}$:
    - Each atomic proposition $\varphi_i$ is a proposition in $\mathcal{F}$.
    - If $\varphi, \psi \in \mathcal{F}$, then also $\neg \varphi$, $\varphi \land \psi$, $\varphi \lor \psi$ and $\varphi \rightarrow \psi$ are propositions in $\mathcal{F}$.
  - associate a *postcondition* $\mathcal{P}ost(\alpha) \in \mathcal{F}$ with each action $\alpha \in A$ to specify exactly, which effects the action will have.
Example

- take $\alpha = \text{select\_mortgage}$ in the loan application example
- a user can only execute this action, if s/he has received the necessary information about available mortgages, i.e. mortgages must be known
- this gives
  $\mathcal{P}_{\text{re}}(\text{select\_mortgage}) = \text{mortgages\_known}$
  using an atomic proposition
- analogously, set
  $\mathcal{P}_{\text{post}}(\text{select\_mortgage}) = \text{mortgage\_selected}$
  which does not state more than the after executing $\alpha$ a mortgage has been selected
Adding Details / 2

- **Enabling and Triggering Events:**
  - the availability of an action to a user may also depend on an event
  - associate an *enabling event* with each action \( \alpha \in A \)
  - this clarifies the conditions, under which an action can be executed
  - the enabling event can be specified by a list of actions that must have been executed already, and a list of actions that must not have been executed
  - analogously, associate a *triggering event* with each action \( \alpha \in A \), which specifies, whether a user is obliged to execute the action
Example

- consider the action \( \alpha = \text{proceed\_with\_securities} \)
- in this case the actions \( \text{select\_mortgage} \) and \( \text{provide\_applicant\_details} \) must have been already executed
- so these two action will appear on the positive list specifying the enabling event of \( \alpha \), while the negative list is empty
- thus, a user who has selected a mortgage and entered the details of at least one applicant is permitted to enter securities for the mortgage
- the positive list of the triggering event for the action \( \beta = \text{set\_up\_budget} \) contains the action \( \text{select\_mortgage} \), while the negative list is again empty
- thus, if a user has selected a mortgage, s/he must set up a budget for it
Adding Details / 3

- **Associated Actors:**
  - for each scene $s \in S$ and each action $\alpha \in A$ with $\sigma(\alpha) = s$ we would like to indicate, which actors are likely to appear in that scene and execute that action
  - associate with scenes and actions a set of *roles* of actors indicating that only actors in these roles have access to the scene and can execute the action
  - roles lead us to rights and obligations that can be expressed in a propositional deontic logic
  - associate with scenes a set of *user types*, each of which can be described by certain characteristics
  - user types capture which kind of actors are expected to appear at the scene and thus give information on how the scene should be designed
Adding Details / 4

• add details regarding the data that is processed in a scene:
  • with each scene we associate the data content that is presented to the user: *data consumption*
  • the data consumption of a scene $s \in S$ should lead to a view $V_s$ on some underlying database, but first restrict to names of the data items
  • associate *data production* to each action $\alpha \in A$
  • express the *manipulation requests* by indicating, which data items will be written or updated
• each scene transition $(s_1, s_2, \alpha) \in \tau$ gives rise to *data communication* between scene $s_1$ and $s_2$
• add a name $c_\alpha$ for this data communication, i.e. extend the scene transition to a quadruple $(s_1, s_2, \alpha, c_\alpha)$
Plots and Story Algebras

- Look at the flow of action, which defines the *action scheme* or *plot*:
  - actions can be executed sequentially or in parallel
  - action can be iterated
  - actions can have pre- and postconditions
  - users may choose between actions
- Model plots by a *story algebra*, which gives rise to Kleene algebras with tests.
Example

```
enter_loan_system;

( ( {φ₀} (specify_purpose {φ₁₃} □ skip) ; look_at_loans_at_a_glance
  □ ( {φ₁} request_personal_loan_details ;
      ( look_at_personal_loan_samples □ skip ) {φ₃} ) □
  ( {φ₂} request_mortgage_details ;
      ( look_at_mortgage_samples □ skip ) {φ₄} ) ) ) {φ₅} ) ;
( select_personal_loan {φ₆} □ select_mortgage {φ₇} ) ;
( ( {φ₆} ( provide_applicant_details ;
      ( provide_applicant_details □ skip ) ;
      ( describe_loan_purpose || enter_amount_requested || enter_income_details ) ;
      select_pl_terms_and_conditions ) {φ₈} ) □
( {φ₇} ( provide_applicant_details ; provide_applicant_details* ;
      ( describe_object || enter_mortgage_amount || describe_securities* ) ;
      ( enter_income_details || enter_obligations* ) ;
      ( ( {¬φ₁₂} select_m_terms_and_conditions ; calculate_payments )* ;
        {φ₁₂} select_m_terms_and_conditions ) ) {φ₉} ) ) ) ;
confirm_application {φ₁₀ ∨ φ₁₁}
```
the Boolean conditions are:

\[ \varphi_0 \equiv \text{information}_\text{about}_\text{loan}_\text{types}_\text{needed} \]
\[ \varphi_1 \equiv \text{information}_\text{about}_\text{personal}_\text{loans}_\text{needed} \]
\[ \varphi_2 \equiv \text{information}_\text{about}_\text{mortgages}_\text{needed} \]
\[ \varphi_3 \equiv \text{personal}_\text{loans}_\text{known} \]
\[ \varphi_4 \equiv \text{mortgages}_\text{known} \]
\[ \varphi_5 \equiv \text{available}_\text{loans}_\text{known} \]
\[ \varphi_6 \equiv \text{personal}_\text{loan}_\text{selected} \]
\[ \varphi_7 \equiv \text{mortgage}_\text{selected} \]
\[ \varphi_8 \equiv \text{personal}_\text{loan}_\text{application}_\text{completed} \]
\[ \varphi_9 \equiv \text{mortgage}_\text{application}_\text{completed} \]
\[ \varphi_{10} \equiv \text{applied}_\text{for}_\text{personal}_\text{loan} \]
\[ \varphi_{11} \equiv \text{applied}_\text{for}_\text{mortgage} \]
\[ \varphi_{12} \equiv \text{payment}_\text{options}_\text{clear} \]
\[ \varphi_{13} \equiv \text{loans}_\text{recommended} \]
Media Types

- The concept of *media type* is central to the codesign approach to WISs:
  - “Media” as means for mass communication suggesting this applies to the web
  - “Types” for classification and formal abstraction from content and functionality of such communication means
- A media type is first an interface abstraction extending the notion of “dialogue type”:
  - The global data content of a WIS is captured by means of an instance of a conceptual database schema
  - Local content needed at the interface is modelled by means of views
- It further includes the navigation structure, and operations expressing the functionality
Media Types (cont.)

- Media types are more than just interface abstractions:
  - Hierarchical versions: switch between representations at various levels of granularity
  - Cohesion: automatic split of information adhering to restrictions imposed by channels, end-devices and user preferences (adaptivity)
  - Consistency and personalisation with respect to content and functionality by logical reasoning about media types
- Media types are linked to scenes and actions of the storyboard modelling the WIS usage
- Not bound to elementary scenes, thus supporting sessions, collaboration and context
Containers

- Media types allow content and functionality to be tailored to the needs of a user
  - The user, channels and end-devices specify the frame, the required container
  - Media types, on the service-side, provide mechanisms to pack instances into the containers
  - A tailor-made instance can be created to fit into the restrictions from users, channels and end-devices by the means of cohesion, hierarchies, session support, context, and presentation operation,
  - The instance is then packed into the container and shipped to user
Interface Abstraction

• Let $S$ denote some conceptual database schema

• Assume a type system comprising base types and various type constructors, e.g.

\[ t = b | \{t\} | [t] | \langle t \rangle | (a_1 : t_1, \ldots, a_n : t_n) | (a_1 : t_1) \oplus \cdots \oplus (a_n : t_n) \]

• Assume that among the base types there is a type $URI$ representing abstract surrogates for URIs

• With each type $t$ associate a set of values $dom(t)$
Interaction Types

- **Content type expressions**: extend the type system permitting $r : M$ to appear in lieu of a base type, with $r$ and $M$ being names for a reference and a media type.

- Replacing $r : M$ by URI results in a proper type, called *content representation type*.

- An **interaction type** comprises
  - a content type expression $ct(M)$ (with corresponding content representation type $t_M$)
  - a query $q_M$ defined on $S$ with output type $(id : URI, value : t_M)$
  - a finite set $Op_M$ of operations

- An **interaction schema** $I$ is a finite set of interaction types that is closed under references.
Semantics

- Take an instance \( db \) of the database schema \( S \)
- Execute the queries \( q_M \) for all \( M \in I \) resulting in sets \( db(M) \) of pairs \((u, v)\) with a URL \( u \) and a value \( v \) of type \( t_M \) subject to the conditions:
  - Whenever a URI value \( u' \) appears in \( v \) corresponding to the reference \( r : M' \), then \( db(M') \) must contain a value \((u', v')\)
  - URI values must be globally unique
- The value \((u, v) \in db(M)\) represents the content \( v \) at the abstract URI \( u \) including references to other units of content
- Interaction types require query languages that create abstract identifiers (fixed-point construction)
Operations

- Operations associated with an interaction type permit updating the underlying database and to open and close interaction objects:
  - Require a signature describing input- and output-parameters as well as a selection type, and an operation body
  - The selection type specifies, which part of the content value must be selected as a precondition for the operation to be executed
- This captures form processing on a reasonably high level of abstraction, but it is not limited to this
Higher-Level Interface Abstraction

- The interface abstraction mechanism described so far associates a media type with an elementary scene leading to abstraction of (classes of) pages
- There is no formal reason not to associate media types also with non-elementary scenes
- This leads to interface abstraction on a higher-level of abstraction – benefits to be discussed soon
Adaptivity

- Two extensions of interaction types (turning them into *media types*) address
  - the granularity of the represented data
  - the adaptivity to users, channels and end-devices
- The former one adopts ideas from on-line analytical processing (OLAP)
- The latter one addresses splitting and condensation of information
- Both extensions exploit the type lattice.
Hierarchical Versions

- Take the lattice defined by the supertypes $t'$ of $t_M$: let $t' \leq t_M$
- A set of hierarchical versions is a subset $\mathcal{H}(M)$ of types in the lattice that defines a tree with root $t_M$
- By means of canonical projections from $\text{dom}(t_M)$ to $\text{dom}(t)$ for each $t \in \mathcal{H}(M)$, we obtain for each value $v$ of type $t_M$ and each $t \in \mathcal{H}(M)$ a version $v_t$ of type $t$
- Due to the tree structure we also get canonical up and down navigation operations between these versions
- As $t_M$ itself represents very condensed information, the hierarchical versions $v_t$ represent lighter versions
Cohesion

- **Cohesion** permits a controlled splitting of a value $v$ of type $t_M$ into a sequence of values $v_1, \ldots, v_\ell$ of types $t_1, \ldots, t_\ell$ such that
  - $\bigcup_{i=1}^{\ell} t_i = t_M$
  - $v_i$ is the canonical projection of $v$, i.e. the decomposition is lossless
  - each $v_i$ contains a reference to $v_{i+1}$
  - each $v_i$ is of “higher importance” than $v_{i+1}$
Cohesion / 2

• This split can be achieved in basically two different ways:
  • by means of a cohesion pre-order, which extends the order defined by the type lattice to a total pre-order
  • by means of proximity values for each pair of types in a specified maximal antichain in the type lattice

• The main difference between the two approaches to cohesion is that proximity values use an a priori decomposition of $t_M$, whereas cohesion pre-orders do not

• Cohesion, hierarchies and style options are orthogonal extensions that can be mutually combined
Session and Context Support

- Media types enable interface abstraction also for non-elementary scenes in the storyboard.
- A **session** comprises all scenes visited by a user from entering the systems until leaving it.
- Exploiting the hierarchies of scenes, a session can be represented by a non-elementary scene.
- Entering a session creates the corresponding media object, which is linked to media objects associated with subscenes.
- We model this by hidden links from media types of subscenes to the overarching session scene.
- A session object may represent data that is carried around through the session.
- Deletion of session objects can be done by garbage collection.
Navigation Context

• A context $C$ is a finite set of triples $(u, n, r)$:
  • $u$ is an abstract URI value of a media object, i.e. a value of the base type $URI$
  • $n$ is a value of some type
  • $r$ is either a reference $\rightarrow C'$ to a context $C'$ or $nil$ In the former case $C'$ is a context for the media objects, represented by media objects for paths, i.e. non-elementary scenes
Reasoning about Media Types

Formalise the operations associated on media types using abstract programs:

- $1$ and $0$ are abstract programs meaning skip and fail
- An assignment $x := exp$ with a variable $x$ and an expression of the same type as $x$ is an abstract program:
  - The possible expressions are defined by the type system
  - In addition, we permit expressions $\{P\}$ with a logic program $P$, assuming that $P$ contains a variable $ans$ – the expression $\{P\}$ is interpreted as the result of the logic program bound to $ans$
Abstract Programs (cont.)

- If \( p, p_1 \) and \( p_2 \) are abstract programs, the same holds for the *iteration* \( p^* \), the *choice* \( p_1 + p_2 \) and the *sequence* \( p_1 \cdot p_2 = p_1 p_2 \).

- If \( p \) is an abstract program and \( \varphi \) is a condition, then the *guarded program* \( \varphi p \) and the *postguarded program* \( p \varphi \) are also abstract programs.

- If \( x \) is a variable and \( p \) is an abstract program, then the *selection* \( @x \bullet p \) is also an abstract program.
Dynamic Logic

- We may consider a logic program \( \mathcal{P} \) as a representative of a higher-order logical formula, say \( \varphi_\mathcal{P} \).
- If \( \{\mathcal{P}\} \) is used as the right-hand side of an assignment, then it will correspond to a term \( I_{\text{ans}}.\varphi_\mathcal{P} \) denoting the unique \( \text{ans} \) satisfying formula \( \varphi_\mathcal{P} \).
- All conditions are formulae of a logic \( \mathcal{L} \), which happens to be a higher-order logic with a fixed-point operator.
- Extend \( \mathcal{L} \) to a dynamic logic by adding formulae of the form \( [p]\varphi \) with an abstract program \( p \) and a formula \( \varphi \) of \( \mathcal{L} \).
- We can formulate proof obligations for the operations that result from the specification of the story space, in particular for consistency with respect to static and dynamic integrity constraints.
- We can extend WIS personalisation in the light of dynamic logic.
The Screenography Pentagon for WIS Presentation

Generic Generation of Layout and Playout

- intention
- context
- content
- functionality
- storyboard

The description of the kind or the specification of the general grid or pattern
Principles of Visual Design for Screenography 1/2

Principles of visual communication: clear and well-defined design of a screen layout

- **Vision**
- **Cognition**
- **Processing and memorizing characteristics**

Specific visual features:

- **Contrast**
- **Visual analogies**
- The **picture dramaturgy**
- The **reading direction**
- **Visual closeness**
- **Symmetric presentation**
- **Space and movement**
Principles of Visual Design for Screenography 2/2

Principles of visual cognition: ordering, effect delivery, and visualisation.

- principle of organisation
- principle of economy
- principle of communication
- Screen design standards

Principles of visual design:

- optical vicinity
- similarity
- Closeness
- Symmetry
- Conciseness
- Reading direction
Generic Refinement Starting with Pattern Towards Grids and Finally Interfaces

Screen pattern

Transformer @ Strategic Layer

Control rules

compile(source_lang,target_lang)

variable_type := type_expression

apply_restriction

Screen grid

Transformer @ Tactical Layer

Control rules

connection := database
database := 

view_consumed := 

view_produced := 

view_communicated := 

support_function := 

import_function := 

export_function := 

storage_function := 

session_media_object := 

apply_restriction

Interface code

Information

Inject

Media types

WIS

Story

Actor
**Screenography Playout, e.g. Search Functionality**

Intelligent Search and Navigation

**Forms of search and navigation:** well-formed query based on known databases structuring and infrastructure, search in known data massives, instantiated query and answer forms, menus with search terms, digging and hunting, dwelling based on associations, browsing jumping, zapping

**Support for search and navigation:** guidance, surveyance, hyperspace exploration with landscaping (Hänsel & Gretel), drilling into browsing, zapping, memorizers, update recharge, work place, work space

**Support for formulation:** formulation support, support for selection of available data, refinement and selection of content chunks, transformation of data, linguistical support (synonyms, troponyms, ...)
Result of User-Oriented CMS

Not trapped in the SQL trap

Tina Musterfrau, casual user

Search request

topic welt concepts

search concept

result concept

query form

answer form

SQL query

answer for search

parametric HERM expressions

relational database schema

query interface

DBS

data base

DBMS query representation
The Cottbus Intelligent NL Request Transformer

- **Ontology**
- **WordNet**
- **RADD DB Design Tool**
- **DB Thesaurus Manager**
- **DB Schema Manager**
- **(e)ER Schema**
- **ER2R Translation Style**
- **ISL**
- **DBMain DB Design Tool**
- **Database Managem. System**
- **DB2Web System**
- **Web Presenter**

**NL query**

**Web Input**

**Query Liquefaction**
- **Syntactical Analysis**
- **Syntax tree**
- **Intelligent Path Extractor**
- **Paths in ER Schema**
- **Relational Query Melting-Pot**
- **Paths and SQL queries**

**Database**

**Media types**
- **WIS**
- **Story**
- **Actor**

**Strategic Modelling**
**Storyboarding**
**Media Types**
**Screenography**
**Pragmatics**
Screenography: Search Pattern

- querying data sets (by providing query expressions in the informed search approach),
- seeking for information on data (by browsing, understanding and compiling),
- questing data formally (by providing appropriate search terms during step-wise refinement),
- ferreting out data necessary (by discovering the information requested by searching out or browsing through the data),
- searching by associations and drilling down (by appropriate refinement of the search terms),
- casting about and digging into the data (with a transformation of the query and the data to a common form), and
- zapping through data sets (by jumping through data provided, e.g., by uninformed search).

see Düsterhöft, DKE
Pragmatics

- Semiotic Triangle: Syntax, Semantics, Pragmatics
  - Syntax: Construction of Language
  - Semantics: Interpretation of the words of the language
  - Pragmatics: Use of utterances by users and context of words

- Distinguish pragmatics from **pragmatism**, the balance between principles and practical usage
Pragmatics of Storyboarding

- **Usage Analysis:**
  - Life Cases: How the stories match the users’ intentions and expectations (in WebIST 2007)
  - User Models: Which user/actor profiles and portfolios have to be considered (in WISE 2006)
  - Contexts: Everything that surrounds and thus impacts on a utilisation situation (in APCCM 2008)

- **WIS Portfolio:**
  - content and utilisation chunks (in preparation)
Syntax versus Pragmatics

- Information
- Story
- User
- Data
- Functions
- Content
- Functionality
- Technical environment of the user
- Information metaphors
- Story and functionality metaphors

Media Types
- WIS
- Strategy Modelling
- Storyboarding
- Screenography
- Pragmatics

Layout
- Web utilisation space
- Playout

Syntax
- Media types
- WIS
- Actor
Starting Point: Analysis of Intentions

- Intentions are coarsely formulated as part of a strategic WIS model (mission, purpose, goals)
- Utilisation scenarios are developed on the basis of intentions
- Therefore, address the description of intentions:
  - Clear understanding of aims and targets of the WIS
  - This includes long-range and short-term targets, expected visitors, characteristics of this audience, tasks performed by the users, necessary and nice-to-have content, restrictions of usage, etc.
Facets of Intention / 1

- **Purpose facet:**
  - The purpose specifies anticipated outcomes that are intended or guide the planned actions (part of strategic model)
  - The *aims* specify what is intended to be attained and what is believed to be attainable
  - The *objectives* specify something toward which an effort is directed, e.g. goals or ends of an action
  - The purpose depends on the intended audience, and is influenced by the mission

- The *time facet* is used for specification of the general time restrictions:
  - The *design* specifies a more carefully calculated plan
  - The *end* stresses the intended effect of an action, often in distinction or contrast to the action or means as such
Facets of Intention / 2

- **Intent facet:**
  - The intents suggest a clearer formulation or greater deliberate-ness:
    - The *targets* of stakeholders specify the steps of a plan and its termination or satisfaction conditions
    - The *object* of stakeholders is related to wishes or needs of users, and specifies an effort or activity of a user in order to satisfy the need
  - The intent facet is related to the tasks
  - The intents may be ordered and generalized to *themes* that represent classes of intents
  - The *representation facet* supports intentions using descriptions or annotations by utterances, words or pictures
Example

- An **aim** of an edutainment user may be to obtain a certificate that proofs the success of learning.

- Typical **objectives** of edutainment systems are greater ease of learning, greater pleasure or satisfaction during learning, etc.

- Typical **objects** of using edutainment systems are faster task completion, successful completion of more tasks, commission of fewer errors, etc.

- **Targets** of using data mining edutainment can be knowledge of data mining methods, and the ability to capture the achievements and the disadvantages and pitfalls of such methods.

- **Themes** can be learning association methods, preprocessing of data, prediction methods, etc.
Representation of Intentions

Intentions can be represented textually, semi-formally using a template, or graphically:

[Diagram showing a flowchart with nodes and arrows representing the preparation for a visit, involving actions such as getting informed, making a visit plan, and printing resources.]
Capturing Intention through Life Cases

- Analysis of intentions structures the WIS: actors and tasks become apparent
- Life cases shed more light into the tasks and the way they are to be executed
- Life cases give rise to the development of task scenarios
- Life cases focus on observations in real life situations, thus takes a user-centered approach that goes beyond task analysis
- This will help identifying and resolving conflicting or competing intentions, before abstracting and mapping life cases to scenarios
Characterisation of Life Cases 1/2

- **Observations:**
  - collect and assess behaviour relating to the specific application
  - include a background check that relates usage to intentions, goals or tasks

- **Processes:**
  - arrange all the actions observed in an application into a main logical and coherent pattern
  - this may involve exceptions through exceptions and parallel execution of independent actions

- **Assessment:**
  - reconstruct the sequence of actions and specific behaviour of users
  - understand the role each individual has within the story
  - develop the user profile
Characterisation of Life Cases 2/2

- **Individual profiles:**
  - clarify background including physical, and behavioural characteristics of individuals

- **Interpersonal coherence:**
  - a variation in the activity will relate to variations of other life cases

- **Significance of time and place:**
  - choices depend on mobility, surrounding, and schedules of events of interest

- **Experience and skills:**
  - individuals may show different behavioural patterns of service employment
Capturing Intention through Life Cases

Modelling of normal succeeding behaviour

Mapping users behaviour with all possible options
Intelligent support by knowledge structures
Adaptation to users profile and portfolio
Logistics for content
Motivating Example

- Consider the relocation of a person:
  - change of basic relocation data
  - change of official documents such as the passport
  - optional change of relation enhancements such as the registration of pets, relocation of cars
  - change of specific personal data such as family enhancements, or relationships to religious bodies
  - change of data for additional relocation announcements such as tax, insurance changes
  - specific additional tasks such as applications for housing allowances
Integration of the Description of the Application Domain

Life and application cases

WIS Design
@ Web 2.0, …
Dec. 14, 2009
© Ma/Schewe/Thalheim

Strategic Modelling
Storyboarding
Media Types
Screenography
Pragmatics

Media types

WIS

Story
Actor
Motivating Example (cont.)

- We make several observations:
  - Persons act in the role of an **issuer**
  - A relocation is enhanced by the profile of the issuer, specific tasks, specific laws and regulations, etc.
  - The life case consists of several steps: change of address data, change of registration data, etc.
Another Example

- Consider the support of hotel search within an information service
- Observe the behaviour of individuals in travel agencies while seeking for hotels
- Search based on associations is preferred over search by main properties such as name, address or facilities
- Associations show a large variety: convenience to reach a hotel, location, places of interest, events, etc.
- Hotel search is combined with other intentions of users such as visiting cultural institutions
Representation of Life Cases

- Life cases represented textually or semi-formally by the template:

  Life case: \(\langle\text{life case name}\rangle\)
  Characterisation: \(\langle\text{outcome description}\rangle\)
    - Tasks: \(\langle\text{list of user tasks}\rangle\)
    - Problems: \(\langle\text{list of problems}\rangle\)
  Background: \(\langle\text{general characterisation}\rangle\)
  Objectives: \(\langle\text{list of objectives}\rangle\)
  Life case flow: \(\langle\text{general graphical description}\rangle\)
  Milestones: \(\langle\text{graph of milestones}\rangle\)
  Content consumed: \(\langle\text{consumed content items}\rangle\)
  Content produced: \(\langle\text{produced content items}\rangle\)
  Themes: \(\langle\text{class of intents}\rangle\)
  Figures: \(\langle\text{actors list}\rangle\)
  Expected profile: \(\langle\text{general profile description}\rangle\)
  Collaboration: \(\langle\text{general collaboration description}\rangle\)
  Context: \(\langle\text{general context description}\rangle\)
    - Time: \(\langle\text{temporality limitations}\rangle\)
    - Place: \(\langle\text{assignment of places}\rangle\)
  Legacy: \(\langle\text{names of documents}\rangle\)
  WIS: \(\langle\text{general WIS context}\rangle\)
  Representation: \(\langle\text{general behavior}\rangle\)
  Approaches: \(\langle\text{general description of approaches}\rangle\)
Development of Life Cases / 1

- Life cases may be developed and interpreted step by step:
  
  (1) Survey of possible application cases:
      The observations could have more than one meaning and may follow a variety of user-related goals
  
  (2) Deep assessment of the life cases:
      Extract the different execution orders, the relationship among actions, and the roles individuals play these actions
  
  (3) Extract life case characteristics that are distinguishing features and relevant for time and location
      Search for similarities within these life cases
  
  (4) Characterization of potential users, their behavioural patterns, and their roles in various stages of the life case
Development of Life Cases / 2

- Various interview techniques can be used to develop life cases
- Life cases can be mapped onto scenarios that are then integrated to define the story space
- Observed life cases can be completely different from life cases supported by the envisioned WIS
- It may be necessary to reorganise processes completely
Mapping Life Cases to Business Use Cases

The natural language representation of application domain elements

**Application domain**

- is characterised by the life case that characterises typical application situations for groups of users (actors) within a context,
- can be used to derive business use cases depending on the task portfolio and on the story under consideration, and
- is described through narrative descriptions based on word fields.
Mapping the Business Use Cases to Requirements

The use of application domain information for requirements elicitation and analysis

The application domain description can be used to deduct the requirements prescription

- through functional requirement,
- through non-functional requirements, and additionally
- through the storyboard of the application.
User Models (Profiles and Portfolios)

- The objective is to tailor systems to the users, their needs, abilities, and preferences
- User modelling is based on:
  - *user profile* specification, i.e. the characterization of the user
  - *user portfolio* specification, i.e. tasks, involvement, and collaboration on the basis of the WIS mission
- User profiles are specified through
  - *education profile* addressing knowledge, skills, and abilities
  - *work profile* capturing specific work approaches
  - *personality profile* representing specific properties
- User profiles can be abstracted to give *actor profiles*
Education Profile

- Characterized by properties obtained during education, training, and other educational activities: education, capabilities, application knowledge
  - *Education*: technical and professional training resulting in erudition, knowledge, and literacy
  - *Capabilities*: understanding the problem area, reasoning capabilities, communication ability, ability to explain results and solutions, etc.
  - *Application knowledge*: type, domain, structure, description, context, functions, etc.
Work Profile

- Characterized by the task solving knowledge and skills in the application area: task expertise and experience, system experience, information and interaction profiles
  - *Task expertise*: knowledge of data, procedures, algorithms, functions, constraints, associations, etc.
  - *Task experience*: identifies both positive experience, e.g. applicable knowledge, strategies, etc. and experience in coping with errors
  - *System experience*: depends on the systems to be explored and used
  - *Information profile*: based on information needs, i.e. intentions in approaching the system, maximum amount and complexity of information
  - *Interaction profile*: frequency, intensity, style of WIS utilization
Personality Profile

- Characterizes general properties and preferences in dealing with the WIS
  - General properties: status, context for performing a task, psychological and sensory properties, background and personality factors, behavioural patterns, required guidance, etc.
  - User preferences: input, output, dialogues
    - Input and output: handling of types, preference of specific forms, required guidance, command preferences, etc.
    - Dialogues: information load, flexibility, complexity, dialogue forms and styles, dialogue structuring, dialogue control, etc.
Template for User Profiles

User profile: \{user profile name\}
Education profile: \{general description\}
  Education: \{list of degrees\}
  Capabilities: \{list of skills\}
  Knowledge: \{description of knowledge in the application\}
Work profile: \{general description\}
  Task expertise: \{description of knowledge\}
  Task experience: \{positive and negative experience\}
  System experience: \{experience with infrastructure planned\}
Information profile: \{information need\}
Interaction profile: \{interaction properties\}
Personality profile: \{general description\}
  General properties: \{list of user properties\}
  Preferences: \{list of input/output/dialogue preferences\}
Based On: \{user goals\}
Based On: \{user types\}
Actor Portfolios

- Characterization through responsibilities and targets:
  - Set of tasks an actor has or intends to complete, and for which the actor has the authority and control
  - Description of involvement within the task solution
- Task modelling involves understanding user goals
- Task analysis may lead to a reorganisation of the work processes
- Task support can be tailored depending on the profile and the context of the actors
Tasks

- Piece of work that has to be finished within a certain time by a user or a set of users
- It may consist of subtasks implying a construction on the basis of elementary tasks
  - Problem statement: tasks are usually associated with problems
  - Target states: effect after successfully completing a task
  - Initial states: specify task enactment conditions
  - Profile required for task completion: skills, experience and knowledge
  - Instruments for task completion: required actions and data
  - Collaboration profile: in case of a group of collaborating users subtasks for each user, obligations and restrictions must be specified
  - Auxiliary conditions: any other condition
Example (E-Learning)

- Task: understand the essentials of data mining and seek for a suitable algorithm
  - First search for algorithms that might be applicable for the application
  - Next understand the algorithm that seems to the most appropriate:
    - Learn how to interpret sample results
    - Look at demonstrations and illustrations for the chosen algorithm
  - Experiment with the data: select data and configure them for the algorithm
  - Explain the results
  - Continue by selecting another method or algorithm and obtain additional insight into the data to be analysed
Task Execution Model

- Task Execution Model:
  - Task activities define how the work is actually done
  - Flow control defines the sequence of activities
  - Data flow specifies how data flows through the task

- Involvement of actors within task execution:
  - the *role* an actor plays
  - the *part* the actor plays within a scenario
  - the *rights* and *obligations* an actor has within the given role
Example (E-Business)

- Roles: worker, customer, and roles within the distribution scenario
  - Worker role: contractor, contact, employee, clerk, sponsor, etc.
  - Customer role: billing and shipping, end-user
  - Within the distribution scenario we may distinguish active and pro-active roles: distributor and agent
  - These roles can again be further refined
- Several roles are played by people or organizations
# Representation of Actor Portfolios

- **Actor portfolio:** \(\langle\text{actor portfolio name}\rangle\)
- **Task:** \(\langle\text{general description}\rangle\)
  - **Extension of:** General characterisation of tasks
  - **Characterisation:** \(\langle\text{general description}\rangle\)
    - **Initial state:** \(\langle\text{characterisation of the initial state}\rangle\)
    - **Target state:** \(\langle\text{characterisation of the target state}\rangle\)
  - **Profile:** \(\langle\text{profile presupposed for solution}\rangle\)
  - **Instruments:** \(\langle\text{list of instruments for solution}\rangle\)
  - **Collaboration:** \(\langle\text{specification of collaboration required}\rangle\)
  - **Auxiliary:** \(\langle\text{list of auxiliary conditions}\rangle\)
  - **Execution:** \(\langle\text{list of activities, control, data}\rangle\)
  - **Result:** \(\langle\text{final state, satisfied target conditions}\rangle\)
- **Actors involvement:** \(\langle\text{general description}\rangle\)
  - **Role:** \(\langle\text{description of role}\rangle\)
  - **Part:** \(\langle\text{behavioural categories and stereotypes}\rangle\)
  - **Collaboration:** \(\langle\text{general description}\rangle\)
    - **Communication:** \(\langle\text{protocols and exchange}\rangle\)
    - **Coordination:** \(\langle\text{contracts and enforcement}\rangle\)
    - **Cooperation:** \(\langle\text{flow of work}\rangle\)
  - **Restrictions:** \(\langle\text{general description}\rangle\)
    - **Actor restrictions:** \(\langle\text{general description}\rangle\)
    - **Environment:** \(\langle\text{general description}\rangle\)
  - **Based On:** \(\langle\text{life cases, intentions, general tasks, audience, mission, goal}\rangle\)
Applying User Models

- Observe real life situations and the data and action flow in them
- This leads to a collection of *life cases*
- The life case description allows to deduct the demands for data, functions, performance, and supporting aids such as *workspace* and *workplace support*
- The workspace is determined by the portfolio
- Its support is determined by the profiles of the actors involved
- Life cases can then be abstracted to *scenarios* of the storyboard
Example

- Consider a life case *relocation*

- Selected tasks in the portfolio:
  - change basic data of issuer
  - provide a view on the data to all other interested parties
  - change data in official documents

- Selected subtasks: checking completeness of supplied documents, need for special handling, etc.

- Actors involved: issuer, civil servant, data protection official, official bodies, etc.

- For this life case we get tasks related to basic changes, changes of associated parties, to change of data ownership, etc.
Context

• A context characterises the situation in which a user finds him/herself at a certain time in a particular location.
• More generally: context is everything that surrounds a utilisation situation of a WIS by a user and can throw light on its meaning.
• A context is characterised by interrelated conditions for the existence and occurrence of the utilisation situation such as the external environment, the internal state, location, time, history, etc.
• For WISs we need to handle the mental context that is based on the profile of the actor or user, the storyboard context that is based on the story leading to a situation, the data context that is based on the available data, the stakeholder context, and the collaboration context.
• These different kinds of contexts have an influence on the development of the storyboard.
Context Space

• Assume we know already the major life cases we would like to support, the intentions associated with the WIS, the user and actor characterisation on the basis of profiles and portfolios, and the technical environment

• Then distinguish the following facets of context (forming the context space):
  
  • **Actor context**: assumptions and restrictions for the involvement and collaboration of actors
  
  • **Storyboard context**: stories that describe what an actor needs to understand to effectively and efficiently solve his tasks
  
  • **System context**: decisions concerning WIS architecture and technical environment
  
  • **Temporal context**: history of utilisation as relevant for an actor in a scene
Context Space / 2

- A context is associated with desirable properties of a WIS: quality, privacy, security, etc.
- Quality criteria can be rather fuzzy such as suitability for users, comprehensibility, etc., but nevertheless imply implementation obligations
- E.g., comprehensibility requires clarity of the visual representation, predictability, directness and intuitiveness
- Other properties that may be associated with context refer to the potential utilisation for other tasks outside the scope of the storyboard
Additional Context Facets

- We may consider additional facets:
  - **Provider context**: captures mission, intentions, specific policies, etc. of the WIS provider
  - **Developer context**: captures hard- and software, implementation and embedded information systems
  - **Organisational context**: organisation for task solution, collaboration, goals
  - **Social context**: interactive and reactive pressures
Actor Context

- The context of an actor is based on his/her intentions and profile, leading to the following refinement:
  - **Actor projection context**: refers to content and functionality that is dropped in accordance with expectations of the actor
  - **Actor approximation context**: refers to condensed or approximated information that is refined later
  - **Actor ambiguity context**: refers to ambiguity in a larger scope, while within a narrow scope the actor encounters non-ambiguity
  - **Actor mental context**: captures attitude, knowledge, expectations, etc. of the actor
Example: Relocation

- An issuer of the relocation expects that his personal and identification data are already sufficient for providing all necessary details.
- Thus, the context in which the issuer reacts is based on projection and ambiguity.
- If we use the information the passport office provides as public information for the city office, then we can adapt the corresponding life case directly to the current one.
- At the same time, the visit of the issuer might be not the first such in his/her life, so we can now use the information on previous life cases for scaling the life case to the expectations the issuer has.
Storyboard Context

• The storyboard context refers to scenes in the story space, their past and future leading to the following refinement:
  • **Pre-scene context:** refers to the content that has already been delivered to the actor, i.e. the (condensed) content encountered on the path leading to the current scene
  • **Scene context:** refers to superimposed meta-data for the scene, e.g. collaborators, log details, etc.
  • **Post-scene context:** refers to potential playout of scenes that may be reached from the current scene
System Context

- The system context refers to content and functionality of the WIS, thus comprises the following (major) parts:
  - **Source and acquisition**: associate with each scenario the subschema of the (global) content database schema that is used for the generation of consumed data and integration of produced data
  - **Associated content**: refers to data that is effected by consumed/produced data by means of constraints, triggers, etc.
  - **Supported functionality**: refers to functions supporting the user actions in scenarios, i.e. transaction management
  - **Security**: refers to enciphering and encryption
Web Information Systems Design
in the Era of Web 2.0 and Beyond
Part III. Service Composition and Collaboration

iiWAS 2009
Kuala Lumpur, Tutorial
Dec. 14, 2009

(Hui Ma\textsuperscript{1}) \& Klaus-Dieter Schewe\textsuperscript{2} \& (Bernhard Thalheim\textsuperscript{3})

\textsuperscript{1}Victoria University, Wellington, New Zealand
\textsuperscript{2}Information Science Research Centre, Palmerston North, New Zealand
\textsuperscript{3}Christian-Albrechts-University Kiel, Germany

hui.ma@ecs.vuw.ac.nz \hspace{1cm} kdschewe@acm.org \hspace{1cm} thalheim@is.informatik.uni-kiel.de
The Web Utilization Space Based On the Characteristics of WIS
Web 2.0 and the Co-Design Hexagonal Dimensions

Web 2.0: Towards sophisticated web engineering

User and intention

Context
Technics
organisation

Storyboard

Functionality

Presentation

Content
Data
objects
knowledge

Web 2.0
collaboration
group content
collective identity

Goal, application area
profile,
information demand

Stories
tasks

Interfaces
depending on the environment

Navigation
search
work

Data

Text

Web 2.0 allows context injection and is user-centered and story-centered.
Abstraction Layer Models

Separation by Level of Detail

Application domain layer concerned with description of the application, e.g., user space, context space, life cases

Requirements acquisition layer concerned with prescription of system requirements, e.g., business use cases, raw stories, main features, presentation pattern, architecture

Business user layer concerned with behaviour or users, their demands to the system, e.g., usage pattern, usage stories, usage screens, infrastructure coupling

Conceptual layer concerned with specifications (schemata) that describe the system, e.g., conceptual media type schemata (content with functionality), storyboard, distribution and collaboration schemata

Implementation layer concerned with logical and physical (specifications and) programs

Deployment layer concerned with introduction, usage, maintenance, evolution of the system
The Co-Design Abstraction Layer Model

Abstraction Layer

Application domain layer

Scoping

Requirements acquisition layer

Variating

Business user layer

Designing

Conceptual layer

Implementing

Implementation layer

Structuring specification

Distribution specification

Dialogue specification

Functionality specification
Reminder: Media Types, Media Object Suite

Theoretical Basis

- Interaction types $= (ct(M), q_M, Op_M)$
  - content type $cont(M)$, defining query $q_M$
  - generic functions $Op_M$ for changing the database
- Attached operations: (signature, selection type, body)
  - selection type - supertype of $ct(M)$
  - e.g. generalization/specialization, reordering, browsing, linking, surveying, searching, join
- Media type: interaction type + unit extension
  + order extension + cohesion/adhesion + hierarchical versions
- Usage modelling: usage dimensions, scales, user profiles, user kind, context, session
- Container $= (ct(C), layout(C), kind(C))$
  for shipping and representation
Web 2.0 “Innovations” - Already Solved Through Our Technology?

Challenges:

- E-Business: adaptation, user context, user history, CRM, variable stories
- Enterprise 2.0: enhancing typical database applications by sophisticated functionality
  see next slide
- Office 2.0: complex management of documents, their orchestration, syndication, proliferation
- Cloud computing: container technology with various source databases
- Web-Oriented Architectures: may-layered view facilities
- Online communities: flexible playout of story spaces with life cases
- Mashups: see slide after next one
- Wikis: see collaboration wikis
- Changing presentations: presentation pattern and grids (see our references)
- User and identity models for singleton users, users in groups, societies
- Semantification notion by J. Pokorny and P. Vojtas of semantic web: see database and knowledge base conferences such as FoIKS’08 and SDKB’08

Our technology can easily be adapted for a solution to challenges.
Elements and Drivers of Enterprise 2.0

New platforms change the shape of Enterprise 2.0

**Search**: sophisticated generic search pattern for websites and their automatic instantiation

**Links**: link management facility by media types

**Authorship** and other metadata by enhancement of web content

**Tags** by injecting tagging into media types or folding tag media types into media types

**Extensions** by automatic behaviour tracking and analysis in the supporting databases

**Signals** by storing usage history and management of changes
Mashups as Collecting Media or Container Types

Mashups are special containers
Mashups are based on media types that use different sources
Container types have been introduced in 1999
  enhanced by playout functionality
  enhanced by special integration functionality
  enhanced by usage stories
  enhanced by session media types
Micro, Meso and Macro Levels of Users Work

- User and identity management management depending on whether singleton users, abstract users, groups or collaborations of users or entire societies are going to be supported
- Separation of support into support for the formation, collaboration or proliferation phases
6 Dimensions
Solutions
Media Types Applied
Collaboration
Realisation
Finally

Micro, Meso and Macro Levels of Users
Work

Collaborative work

Formation of collaboration
Working in collaboration
Consequences of collaboration

Micro-, meso- and macro-societies

Individual self-realisation
Group
Society

Collaborative work

Formation of collaboration
Working in collaboration
Consequences of collaboration

Micro-, meso- and macro-societies

Individual well-being
Group cohesiveness
Society social integration

Information demand
Participation

Web Information System Chunks

WIS chunk: content + concepts + topics + context

- **What**: Structural (data, syntactic) representation, media types
- **Assumptions and thoughts**: Intentional (semantic) description
- **In which way**: Topic (annotation) description
- **By whom**: Context (origin) description

Assets: Structural media types with annotation media types
Semantical units: Structural media types with concept media types
User units: Structural media types combined with context media types
The Syntactic Dimension

Syntactic languages do not matter

Explicitly describing the choices made:

- Representation ((supported) modelling) languages: $\mathcal{L}$
- Selected media types (schema) used for encoding media objects: $S_L$ for $L \in \mathcal{L}$
- Set of all possible states $\mathcal{DB}_S$ of a certain schema $S \in S_L$

Structural component $D$ of a WIS chunk $C$

defined by choosing $(L, S, I)$

- modelling language $L \in \mathcal{L}$,
- schema $S \in S_L$,
- $I \in \mathcal{DB}_S$

based on a theory of “small” theories

typical for any application domain
The Semantic Dimension

WIS Transparent Intensional Logic

- Different types of individuals in the objectual base over semantic types $\iota_1, \ldots, \iota_n$
- Different types of modalities
- All first order types $\alpha_i$ defined in TIL

Intensional function $f : \omega_{i_1} \times \omega_{i_k} \rightarrow \alpha$

mapping possible worlds $(w_{i_1}, \ldots, w_{i_k})$ to instances of a first order type $\alpha$

Non-trivial intensional function: there are two possible worlds

$$(w_1, \ldots, w_k), (v_1, \ldots, v_k) \text{ with } f(w_1, \ldots, w_k) \neq f(v_1, \ldots, v_k)$$

Given $\mathcal{F}$ facets

$$(6W1H: \text{who, what, when, where, which intention, at what quality, how})$$

Content intension: set of functions $i : \mathcal{F} \rightarrow \mathcal{B}$

mapping facets from $\mathcal{F}$
to concepts from $\mathcal{B} = \{\mathcal{B}_1, \ldots, \mathcal{B}_n\}$
in the current context
The Pragmatic Dimension

- Media types transformed to local vocabulary
- Embedding into the usage context by supplementary content:
  - static content, e.g., logo, statically linked elements,
  - decorative content,
  - additionally delivered content, and
  - navigational events.
- Multiple topic maps may be merged for multi-modal applications.
The Context Dimension

Towards this century WIS

Given a finite set $C$ from a universe $\mathcal{U}_C$ of contexts for a set $A = \{A_1, \ldots, A_n\}$ of content chunks

Actor context dimension

Storyboard context dimension

System context dimension

Temporal context dimension

Additional context dimensions e.g. provider, developer, organisational, social

expressible through mind mapping and annotation techniques
Dimensions of Understanding WIS Chunks

Generalising Speech Act Theory

can be generalised to communication acts

sender-receiver relationship
Collaboration by Wikis: Separation of Concern Into 3C-C

Collaboration Triangle Relating Communication, Coordination, and Cooperation

- Communication
  - supports
  - generates commitments that are managed by

- Collaboration
  - demands in
  - creates opportunities for
  - requires

- Coordination
  - arranges tasks for

Wiki Collaboration Acts

Communication act view based on sending and receiving

Concurrency view based on commonly used data, functions, and tools

Cooperation context view combines the context of cooperation, i.e. portfolio to be fulfilled, the cooperation story and the resources that are used
Layers of a Typical Collaboration System

- **Cooperation Layer**: workspace control, awareness, notifications, security over user functions.
- **Coordination Layer**: operation management, session management, shared resources management, users management.
- **Communication Layer**: (a)synchronous, multicast/broadcast, protocols, standard.

**Media object unit manager**

**Coordination and contracting system**

**Communication support system**

Generalising Ochoa, Guerrero, Fuller, Herrera: Infrastructure of Groupware Systems
Explicit Collaboration

Communication via exchange of messages and information as only one of the perspectives demands cooperation and generates commitments that are managed by coordination choice of media, transmission modes, meta-information, conversation structure and paths, restriction policy

Coordination via management of individuals, their activities and resources as the dominating perspective generates communication and arranges tasks for cooperation pre-/post-articulation of tasks; management of tasks, objects, and time; loosely ... tightly integrated activities, enabled, forced, blocked

Cooperation as production taking place on a shared space as the workflow or life case perspective creates opportunities for coordination and demands communication storyboard-based interaction, mapped to (generic, structured) workflows production, manipulation, organization of contributions through media objects

Awareness is fostered by each of the aspects and mediates each of the aspects
Typical Wiki Collaboration Pattern

Proxy collaboration: partial system copies (remote proxy, protection proxy, cache proxy, synchronization proxy, etc.)

Broker collaboration: coordination of communication either directly, through message passing, based on trading paradigms, by adapter-broker systems, or callback-broker systems

Master/slave collaboration: tight replication in various application scenarios (fault tolerance, parallel execution, precision improvement; as processes, threads; with(out) coordination)

Client/dispatcher collaboration: for name spaces and mappings

Publisher/subscriber collaboration (observer-dependents paradigm) active/passive subscribers or passive with their subscription profile

Model/view/controller collaboration: e.g. three-layer architecture of database systems
Collaboration Support for Wikis

Wiki collaboration operations
for intensions $\mathcal{I}$, topic maps $\mathcal{T}$, structure descriptions $\mathcal{D}$, contexts $\mathcal{C}$

**extraction function** $\text{generate} : \mathcal{T} \times \mathcal{C} \rightarrow 2^{\mathcal{D} \times \mathcal{I}}$,

**delivery function** $\text{deliver} : \mathcal{I} \times \mathcal{D} \times \mathcal{C} \rightarrow \mathcal{T}$,

**exchange function** $\text{exchange} : \mathcal{D} \times \mathcal{I} \times \mathcal{T} \times \mathcal{C} \times \mathcal{C} \rightarrow \mathcal{D} \times \mathcal{I} \times \mathcal{T} \times \mathcal{C}$

Adaptation functions

- derive context
- apply intentions based on concepts
- associate with content
Wiki Storyboards

forming a Wiki team with three roles (evaluator, member, supporter)
modelling the Wiki collaboration story
modelling the intended result
Wiki Specification Based on Content Chunks

Wiki $\mathcal{W} = (\mathcal{I}, \mathcal{S}, \Sigma_{\mathcal{W}})$ specifying

Content chunk processes $\mathcal{I} = (\mathcal{V}, \mathcal{M}, \Sigma_{\mathcal{I}})$ specifying
the content chunks $\mathcal{V}$ based on media types (“what”),
the wiki manager $\mathcal{M}$ (“how”), and
the competence $\Sigma_{\mathcal{I}}$ through a set of tasks (“for what”)

Wiki storyboard $\mathcal{S}$ specifying the organization frame (“how”),
the parties (“who”) and the context (“whereby”)

Quality of wiki $\Sigma_{\mathcal{W}}$ agreeing on the quality and motivation (“why”)

Exchange frame specifying

Architecture drafting the general engine (“where”)
Wiki collaboration style drafting the flow (“when”)
Wiki collaboration pattern describing the functionality (“how”, “whereby”)

as a generalization of distributed systems, communication systems,
groupware systems, and collaboration architectures
Support for Wikis

content chunks based on media types: views + functionality + adaptation + presentation

Active media types: in parallel
open(content); inform(proprietor, usage)

Self-protecting media types: content based on queries and supported view protocol: contact(proprietor,possessor, usage);
obtain(proprietor,token); provide(media type, token)

Communication protocols based on service (distributed ADT), signals, shared variables, sender/reponder ASM (signature (e.g., signal), phases (via small ASM)), and timer represented through SDL or message sequence chart or other protocols

Security techniques against passive/active sniffing, trust exploitation, viruses, downloadables, OS holes, hacking

Control techniques for focusing, access, user authorization, (password) protection, biometrics, content/concept/topic security, fire walls, hiding/anonymizing/translating, identity management

Privacy enhancing techniques based on virtual private networks, key encryption, secure transaction, and corporate policy on security, privacy and control, cookie cooker
An Architecture of User-Oriented Web Content Management Systems

Towards this century CMS based on WIS

Web Playout System
- Story Space
- Stories
- Actors
- Scenarios
- Context

User Management System
- Profile manager
- Portfolio manager
- Association generator / Natural language engine

Privacy Protection System
- Content base

Content Management System
- Content types
  - Structure
  - Functionality
- Container
  - Structuring
  - Process
  - Dynamic IC
- (Pragmatics)

Topic Management System
- Topic manager
- Community manager
- Asset manager / Infon representer

Concept Management System
- Concept manager
- Derivation engine
- Unit manager / Infon representer

Media types
- WIS
- Story
- Actor
Web 1.0: City information

Information sites

z.B. www.Cottbus.de

- simple
- fast
- actual
- simple management
- manyfold of tasks
Web 1.0: Collaboration depending on needs

Collaboration of groups

Association Cottbusnet

depending on environment

depending on information

depending on partner

depending on rights,...

depending on group
Summarising

- Web information systems cover e-business applications, community services, edutainment services, information services and identity services. The only difference is what intention is supported, which story space is used, which context must be considered, which content is provided, which functionality is provided, and which presentation has been chosen.

- WIS theory and development methodology for specification of Web x.y WIS based on the theory and technology of media types, database systems.

- Not reported here: Sophisticated playout and layout based on automatic generation of presentation and function support depending on user, environment, history, usage.

- Applied for the development of three-score WIS applications in the past, e.g., e-business, edutainment, community and service WIS.
Exploring the Potential of Freeform, Social Collaboration for E-Business

- Support for external and internal collaborators both.
- Reputation systems.
- Enterprise class feed readers and management.
- Exploiting collective intelligence in the enterprise.
Supporting the trinity of Web 2.0: People, Information, and Software. E-business story space, intentions, context, content and functionality.
Further Reading /1


(4) **Binemann-Zdanowicz, A., Schulz-Brünken, B., Thalheim, B., and Tschiedel, B.** *Flexible e-payment based on content and profile in the e-learning system DaMiT.* In *Proceedings of E-Learn 2003* (Phoenix (Arizona), USA, 2003), Association for the Advancement of Computing in Education.


Further Reading /2


(10) Fiedler, G., and Thalheim, B. An Approach to Conceptual Schema Evolution Preprint 0701, Department of Computer Science, Kiel University, Januar 2007


(12) Hui Ma, Schewe, K.-D., and Thalheim, B. Integration and Cooperation of Media Types. ISTA 2005: 139-153


(15) Hui Ma, Schewe, K.-D., and Thalheim, B. Conceptual application domain modelling. APCCM 2009, 49-57


Further Reading /3


(22) Noack, R. and Thalheim, B. Patterns for screenography. LNBIB5, Springer, 2008, 484-495


Further Reading /4


(32) Schewe K.D. and Thalheim B. Usage-Based Storyboarding for Web Information Systems Preprint 0613, Department of Computer Science, Kiel University, November 2006

Further Reading /5

(34) **Schewe K.D. and Thalheim B.** Life Cases: A Kernel Element for Web Information Systems Engineering. WEBIST (Selected Papers) 2007: 139-156


(38) **Schewe, K.-D., and Thalheim, B.** Term Rewriting for Web Information Systems - Termination and Church-Rosser Property. WISE 2007: 261-272


(40) **Schewe, K.-D., and Thalheim, B.** Conceptual modelling of web information systems. *Data and Knowledge Engineering*, 54 (2005), 147-188

(41) **Schewe, K.-D., Zhao, J., and Thalheim, B.** Quality Assurance in Web Information Systems Development. QUSIC 2007: 219-224

(42) **Schewe, K.-D., Thalheim, B., and Zlatkin, S.** Modelling Actors and Stories in Web Information Systems. ISTA 2004: 13-23

(43) **Schewe, K.-D., and Thalheim, B.** Pragmatics of storyboarding for web information systems: Usage analysis International Journal Web and Grid Services, 128-169 (2007)
Further Reading /6


(45) Schewe, K.-D., and Thalheim, B. Facets of Media Types. LNBIB5, Springer, 2008, 296-305

(46) Schewe, K.-D., and Thalheim, B. Pragmatics of Storyboarding Web Information Systems Portfolios iiWAS 2009


