Recommendations for Improving Device Independent Presentation Authoring

Krishna Vedati
Agenda

- Device-Independent Presentation Authoring
- Current Approaches to Device-Independent Authoring
  - Pros and Cons
- Recommendations for Improving W3C’s Device-Independent Presentation Authoring Standards (XHTML, XSLT, CC/PP, CSS...)
  - Standard Multi-Channel XHTML Hints Module
  - Distributed, Personalized Assembly
  - Multi-Channel, Multi-Modal Interaction Models
- Q/A
The Problem:
Single Application, Multiple User Experiences

Applications consist of tasks...

But on different devices...

The same task has different navigation flows and displayed information.

Task 3
Task 2
Task 1

Login
Main Menu
News
PIM

PDA
Voice
WAP
# Today’s Solutions: Device Independent Presentation Authoring Paradigms

<table>
<thead>
<tr>
<th>Multiple Sources Design “native” HTML, WML, or VoiceXML user interfaces separately</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Authoring Design using a proprietary markup language (First generation mobile platforms offer this today)</td>
</tr>
<tr>
<td>Style Sheet/Transformation Design pages that will be transformed at runtime, using W3C Standards (XHTML + XSLT + CSS)</td>
</tr>
</tbody>
</table>

## Implementation and Management Effort

<table>
<thead>
<tr>
<th>More Implementation and Management Effort</th>
<th>Less</th>
</tr>
</thead>
</table>

## Scale of Automation

<table>
<thead>
<tr>
<th>Less Scale of Automation</th>
<th>More</th>
</tr>
</thead>
</table>
How They Stack Up
Presentation Quality vs. Cost of Ownership

Multiple Sources

High Presentation Quality

Style Sheet Transformation & Multi-Channel Authoring Standards

Style Sheet Transformation

Low Cost

Low Presentation Quality

Single Authoring
Recommendation #1: A Standard “Hints” Module for Optimal Device Independent Authoring
Example: Widget Transformation Hint

Instance-specific transformation of a “widget”
Overrides the default widget for a given class of devices

Common Use Cases:
Change widget to best fit a class of devices based on
• Browser capabilities
• Screen dimensions
• Connection speed

Implementation Suggestions:
Standardize set of most common widget transformations
Provide hints for identifying widget and desired transformation output

Others Transformations
List - Paragraph
Select List – Hyperlinked List
Table – Paragraph

...
Example: Layout Hint

Modification of layout based on screen capabilities

Common Use Cases:
Reorder form elements on a PDA
- Personalization
- Device form factor

Implementation Suggestions:
Provide device independent layout hints during design
Possibly even separate layout specification from modular UI
Encourage designers to build modular UIs
Example: Content Optimization Hint

Reordering, inclusion, or omission of content

Common Use Cases:
Content inclusion exclusion, modification applied based on
- Browser capabilities
- Screen dimensions
- Connection speed

Implementation Suggestions:
Provide hints to prioritize content
Provide hints to select or exclude content for a specific device context

Others transformations
Include only “high priority” columns on PDA
Omit optional form elements in WAP

Include 1st, 3rd, and 4th table columns
Standard Device Independent Authoring Paradigm

Design-time

XHTML Module
- Content
- Layout
- Style
- Hints

CSS
- Style

XHTML Module
- Hints

Device-Independent Presentation Architecture

Run-time

Transformation Engine
- XSLT

Device Detection
- XSL

Profile-Specific Style Sheet Library

Device Context

Request

Response
Recommendation #2: A Standard for Distributed Presentation Assembly
Today’s Solutions:
Layout and Assembly of Distributed Presentation Content

- How do we design the MyYahoo’s of the world?

- There is no easy, device independent layout design paradigm:
  Developers use JSP includes to code the assembly of distributed content sources
    - Not standards-based
    - No clean separation of business logic and presentation

- Today’s portal server architectures use layout descriptor systems
  - No standard
Typical Presentation Design Issues

Information Architecture
Different for each terminal type

Page
Web page, card in a deck

Page Components
Includes, Portlets

Pocket PC IA
WAP IA
PocketPC Page
WAP Pages

Page Component  Page  Page-Specific Markup  Terminal-specific IA  Multi-Channel IA
Example: How would you build this system?

Designer needs a high-level assembly language to specify system of pages based on delivery context.
Design Goals for a Device Independent Layout Standard

- Build and assemble heterogeneous presentation components into individual pages
  - The presentation components use distributed data sources

- Support device profile-depandant assembly such as:
  - Include/exclude presentation component in each channel
  - 2x2 grid assembly for a PDA
  - 4-column layout for a web page
Recommendation #3:
A Standard for Interaction Models
Just Presentation Design is Not Sufficient...

- **What is an interaction model?**
  - An interaction model is the workflow of interactions between a user and the system.
  - It is typically the controller of the M-V-C paradigm.

- **Why is this necessary for multi-channel design?**
  - Different modalities and channels require different information architectures and page design
    - Ex: A voice application may require disambiguation states
  - Dealing with dropped calls; continue on a different channel!
    - Multi-channel state and session management
  - Handling out-of-sequence client requests during a transaction
Example: A Multi-Channel, Multi-Modal Currency Converter

- More end-user help
- Extra disambiguation states
Example:
A Multi-Channel, Multi-Modal Interaction Model

Typically a well-designed voice application needs more end user help, error checking, and some disambiguation states

- **Red** indicates voice-specific interaction states
- **Green** indicates WAP-specific interaction states
Recommendations for Further Work

Improve Current W3C’s Device Independent Authoring Standards

- Standard for markup to markup transcoding hints module
- Standard for layout and assembly
- Standard for interaction model
  - Investigate emerging Web Services standards, such as WSXL & WSFL