One day meeting on

Model-based UI

Hosted by Fabio Paternò
and the HIIS Laboratory of the
Istituto di Scienze e Tecnologie
dell'Informazione

Dave Raggett, W3C/JustSystems

Rough Agenda

- 0915 Introductions
- 0930 Scene setting
- 1000 Work at ISTI
- 1030 Work at Telefónica
- 1100 Break
- 1130 Work at Siemens
- 1200 Work at JustSystems
- 1300 Lunch
- 1400 Pulling it all together
- 1600 Break
- 1630 Summing up and next steps
- 1700 Close

Scene Setting

- Model based user interfaces
- Contrast with current pratice
- XML, Semantic Web, Diagrams and Rules
- Relation to existing W3C Work
- W3C Incubator Process
- Learning from each others experiences

Model-based design

- Declarative versus Imperative approaches
- Describe what should happen rather than how
- Separate out different concerns, e.g.
  - Application data from user interface
  - Implementation details specific to platform choice
  - Different roles and skills of team members
    - analysts, designers, coders, testers, ...
- Greater flexibility and reduced costs
- But what hard evidence is there for these benefits over traditional approaches?

Building on years of research

- There has been a lot of research into how to build user interfaces over last 15 years
- Model-based
- Multiple layers of abstraction
- Each layer models behavior at a progressively finer level of detail
- Functional transformations between layers
- Use delivery context to select transformation

Layered UI

with mappings defined between each layer

1) Application task and domain models
   - supported via diagramming languages (e.g. UML)
2) Abstract User Interface
   - UI independent, e.g. select 1 from n
3) Concrete User Interface
   - UI specific, e.g. set of radio buttons
4) Realization on specific device context
   - may be generated via a compilation step
   - for delivery to HTML, SVG, Flash, Java, .NET

Contrast with Current Practice

- Web pages are hard to construct
  - HTML, JavaScript, CSS, Images, Flash
  - Variations across browsers
- Server-side is also complex
  - Emphasis on scripting and libraries
    - Java, Perl, Ruby on Rails, ...
- Lack of shared tools
  - Designers use Photoshop to mock up pages
  - Coders program in a variety of languages
  - No shared machine manipulable models

Consequences

- Web pages don't meet requirements
- Easy to break pages due to lack of enforced separation of data and user interface
- The details are in people's heads and lost when they move on to new jobs and companies
- Poor quality of websites due to shortage of really good people to develop them
- Not fulfilling the potential of the Web !

Benefits from using XML

- Reduced costs for development and maintenance
  - compared to non-declarative techniques
- Improved security, accessibility, usability
- Easier delivery to wide range of devices and platforms
  - through use of a layered architecture
- Facilitate people with different roles to work on different aspects as part of a distributed team
  - allow team members to focus on what they do best

Semantic Web

- Labelled links as building block for models
  - RDF triples (Subject-Predicate-Object)
  - Decoupled from syntax
  - Can be used to front-end legacy systems
- Makes it easier to combine multiple information sources
- Rich Ontologies using OWL
  - Delivery Context Ontology
    - User preferences, device capabilities, environment
    - Key to ambient intelligence (dynamic adaptation)

Diagrams

- Unified Modelling Language
  - Suite of diagram formats for different kinds of models
    - taxonomies, processes, state charts, ...
- BPMN business process modelling notation
- Diagrams as requirements
  - Compile into Java stubs for implementation
- But how to exploit diagrams throughout the application life cycle?

Rules

- Rules can be used to describe
  - Actions to be taken in response to events
  - Constraints that the application must conform to
- Rete algorithm for efficient rule interpretation
  - Forward chaining for large rule sets
  - Used in business rules engines
- High level rule languages for use with diagrams
  - Making it easier to describe behaviour
  - Compiled into lower level rules for execution

Existing W3C Work

- Data models
  - XML Schema, RDF Schema, OWL
- Query languages
  - XQuery for XML, SPARQL for RDF
  - XPath and XSLT for XML
- UI and Presentation
  - XHTML, SVG, MathML, XForms,
  - CSS, XFL-FO
- Adaptation
  - UWA Delivery Context Ontology, DISelect/XAF

Other W3C Work

- XML Binding Language (XBL)
  - Bind widget to XML data
  - Widget defined as a mix
    - SVG, JavaScript and images
- SCXML (State Chart XML)
  - Implements UML hierarchical state charts
  - Event handlers expressed in XML or JavaScript
    - But other rule languages are possible
- RIF (Rule Interchange Format)
  - Enable transfer of rules between rule systems

XML for UI

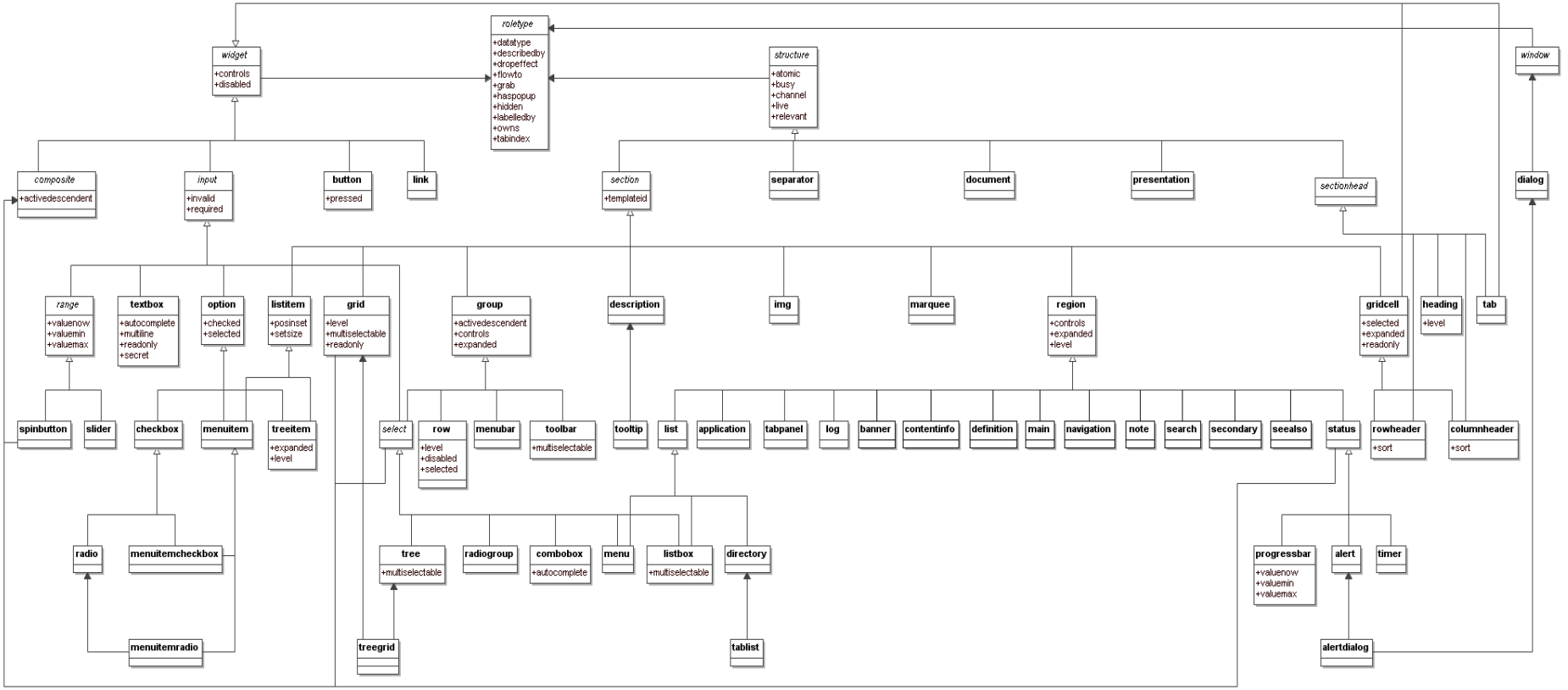
- Many examples of proprietary UI markup languages, e.g.
  - Microsoft (XAML)
  - Adobe (MXML)
  - Lazlo (OpenLazlo)
  - Nexaweb (XAL)
  - Mozilla (XUL)
- Time for W3C to define an open standard?
  - For authoring tools rather than run-time
  - Alignment with accessibility APIs

XML for Concrete UI?

- Use XML for defining UI layout and controls
  - vertical/horizontal/grid layout managers
  - full set of controls e.g. buttons, menus, text input, ...
  - associated concrete UI events
- Themes define details of appearance and behavior on target platforms
- Compile into final UI
  - HTML+JavaScript+CSS
  - Java for JVM (JAR)
  - ActionScript for Flash Player (SWF)

WAI-ARIA

- Ontology of UI controls, properties and states
  - Used to enable assistive technology



W3C Incubator (XG) Process

http://www.w3.org/2005/Incubator/about.html

- Intended as precursor to standards track work in a W3C Working Group
- Easy to set up, low administrative overhead
- New, potentially foundational technologies
  - Innovative/speculative ideas
  - Ideas requiring further work
  - Ideas for which there is insufficient consensus
- Work relating to Web-based applications
  - Testing the foundations
  - Supporting particular user communities

How to Form an XG

http://www.w3.org/2005/Incubator/how-to.html

- Three or more W3C Members draft charter
  - See charter generator
  - Choice between Member-only and public mailing list and web pages
  - Main product is an XG Report published by W3C
    - Initial charter for 1 year, may be extended to 2 years
- The corresponding AC Representatives then submit the charter to W3C Management
  - email xg-activity@w3.org (Member confidential)
- Reviewed by W3C Team
  - Approval process typically takes two weeks

Potential Goals for an XG

- Collect use cases
- Identify requirements
- Evaluate existing research work and current solutions
- Propose particular solutions
- Promote a shared vision
  - Demonstrable benefits over current practice

Model-based UI

Questions?