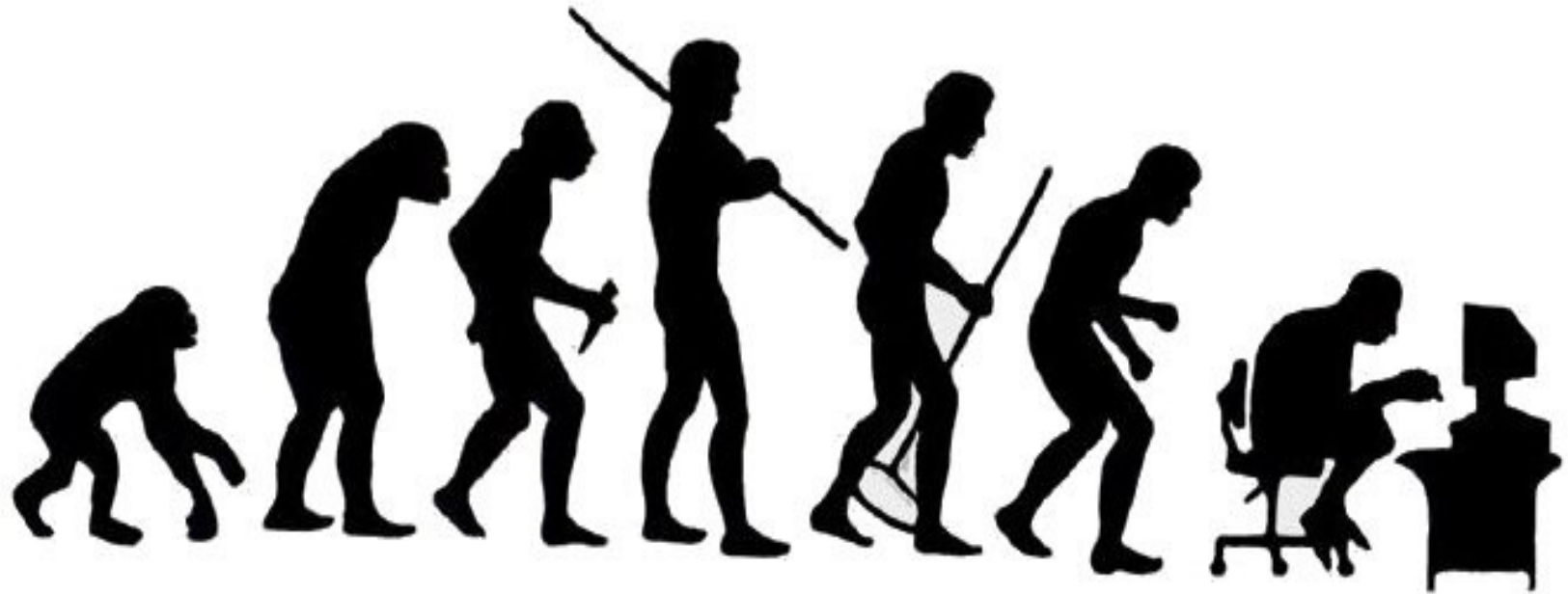


SVG, layered user interfaces and end to end models

Dave Raggett, W3C & JustSystems

Web Development



Something, somewhere went terribly wrong

In the beginning

- HTML was very simple
- No client-side scripting
- Everyone used the Mosaic browser
- Simple, but very boring ...

Then the trouble set in

- Browser wars and its legacy
- Variations in CSS support
- Variations in Scripting APIs
- Attractive, functional web pages
- But hellish to develop and maintain
- Lack of decent professional authoring tools

And it isn't getting any better

- HTML5 is codifying current practice
- WebForms proposal helps with trivial forms
- But developers are left with lots of challenges
 - Designing for old and new browsers
 - Interoperability problems in the details
 - Designing for browsers with scripting turned off
 - Supporting assistive technology through ARIA
- Mobile devices just add to the mess
 - Huge variations across devices

Over reliance on Scripting

- Expensive to develop, test, and maintain
- Burying the design in the code hinders communication between designers/coders
- Whatever happened to Computer Science?
- Can we do any better than this?

Separating out different concerns with a layered architecture

Each layer embodies a progressively finer model of behaviour with mappings between objects and events in adjoining layers

1. UI independent application models
2. Device independent abstract interface
3. Device dependent concrete interface
4. Policy guided generation of final UI

Agile Processes

Model-based design is great for agile processes

– to paraphrase Scott Ambler

- Defining requirements up front is inherently risky
- Agile modellers travel light and create models which are just barely good enough.
- Agile developers solve today's problem today and trust that they can solve tomorrow's problem tomorrow.
- Agile modelling is both evolutionary and collaborative.
- Requirements change over time, so embrace this concept and adopt techniques which allow you to react effectively.

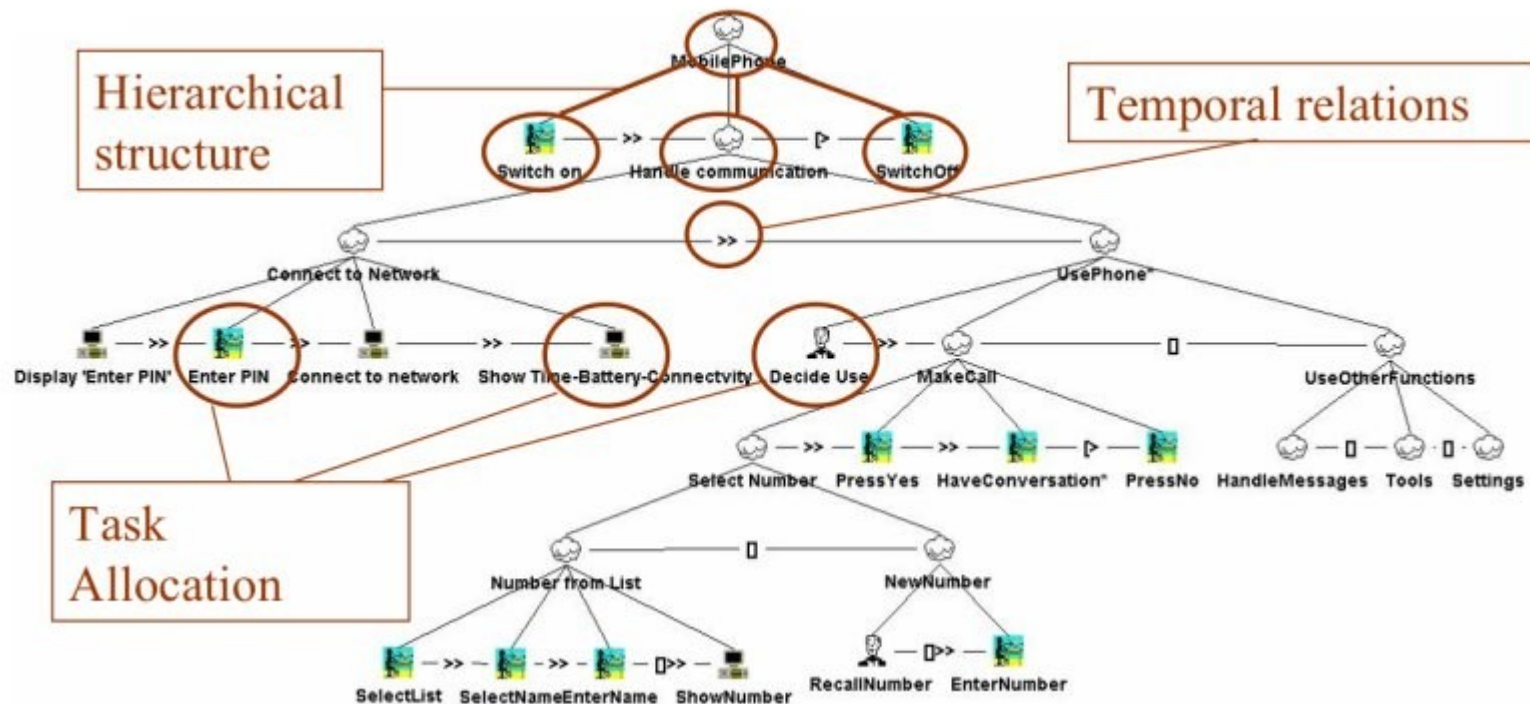
Application Models

- Independent of user interface details
- Use of ontologies for data models
- Service composition
- A diagram is worth a thousand words
 - Concur Task Trees
 - Unified Modelling Language (UML)
 - Business Process Modelling Notation (BPMN)

Task Models

Relationships between named tasks

- sub-tasks, temporal ordering, pre/post conditions

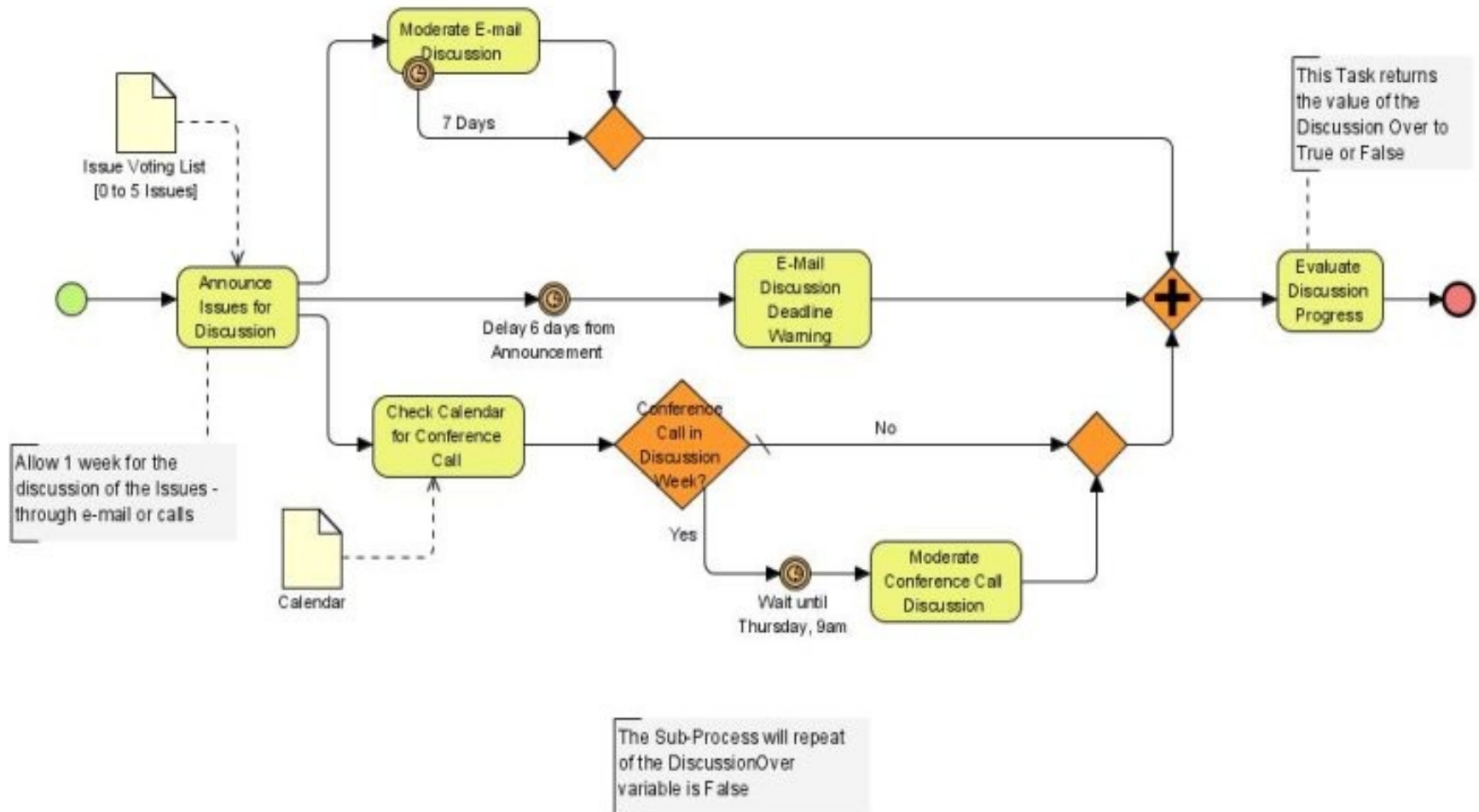


UML

- Widely used for modelling requirements, UML2 has 13 kinds of diagrams:
 - Structure diagrams
 - Classes, components and objects
 - Behaviour diagrams
 - Activity, state and use cases
 - Interaction diagrams
 - Subset of behaviour diagrams emphasising flow of control and data
 - Communication, interaction, sequence and timing
- Used to generate Java stubs for coding

BPMN

Non-executable models of business processes



Abstract User Interface

- Independent of choice of device or modality
- Deals with concepts like
 - Selection from a set of alternatives
 - Number of alternatives influences UI
 - Open ended input, e.g. text
 - Hypermedia links
 - Grouping and ordering of things
 - Hierarchical groups
 - Order by expected frequency of use
 - When a thing or group is deemed relevant

Concrete User Interface

- Dependent on choice of device and modality
 - Provide alternatives for different delivery contexts

Cardinality	Desktop	Mobile
Low	Radio button	Radio button
Medium	List box	Drop Down List
High	List with scrollbars	Drop Down List
Huge	Search box	Search box

- Temporal ordering may be imposed by display and memory limitations (e.g. mobile)
- Concrete UI defines some, but not all aspects of the final presentation

Final User Interface

- Automatic generation of final user interface guided by author's themes and policies
- Generate UI to match the delivery context
 - user preferences, device capabilities, environment
- Flexible choice of software platforms, e.g.
 - HTML, SVG and JavaScript
 - Flash, Java, or Silverlight
 - Automatically deal with browser quirks

How does SVG fit in?

- Diagram notations for modelling data and behaviour
 - Round-trip semantics and presentation
 - Combine both in same document
 - Return to early aims of SVG WG
 - New standards are needed to avoid vendor lock-in
- Presenting application data, e.g. charts
- Theming user interface controls
- Purely decorative purposes
- Together with scripting for implementing final UI

High level authoring tools

- Model-based design inevitably requires complex internal representations
- Authors work on diagrams, rules and UI components
 - Shield authors from direct editing of XML
 - Manage mappings between abstraction layers
 - Automatic support for ARIA for accessibility
- Markup standards for authoring tools
 - No need for direct browser support
 - But implementable via Flash, Java etc.

How is this being addressed?

- Proposed new W3C Incubator Group on Model-based User Interfaces
 - Evaluate research and advise on standardization
- If successful this will
 - Allow developers to switch vendors through the use of standards for authoring tools
 - Avoid the need for developers to battle with HTML, JavaScript, and browser interoperability woes
 - Make it easier to apply agile processes
 - Reduce the cost for developing, testing and maintaining Web applications

SVG, layered user interfaces and end to end models

Questions