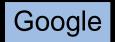
# **Cloud Computing** W4A Keynote: Equal Access For All

T. V. Raman





### Outline

- Creating Web Applications
- Tangible User Interfaces
- Consuming Web Applications
- Usable UI Patterns
- Web APIs And Specialized Browsing
- Conclusion

Goal: Ubiquitous access.



# **Separate UI From Application**



#### Web Architecture

Basic Web building blocks

URI Universal means for addressing content.HTTP Protocol for client/server communication.HTML A language for hypertext documents.

Web Browser — a lens for viewing the Web



# **Discovering Web Applications**

Web —Global hypertext system

HTML Presentation-independent information.
Forms Interactive Web hypertext.
CSS Style content.
DOM Programmable Web.
JavaScript Custom behaviors.



# **Distributed Web Applications**

Application Logic and Data separate from UI!

Data Resides in the Web cloud.Application Logic runs on the server.Presentation Delivered as HTML to the client.UI Augmented by DOM-based interaction.

Facilitates multiple UI to a single application.



### **Google Calendar**

Data UI-independent, lives in the cloud.UI Delivered via the Web.Clients Manipulate underlying representation.Sync Multiple clients manipulate same data.

Specific UI used is no longer significant!



# **Creating Web Applications**



# **Anatomy Of A Web Application**

Server Manage data, application logic.Client Presentation, interaction.Bind Connect the dots.

**Opportunity:** Separation of UI!



# **Application Data**

- Resides in the network cloud.
- Enables ubiquitous access.
- Is independent of any specific UI.
- Ranges from the simple to the complex:
  - Maps
  - Spreadsheets



# **User Operations**

User operations manipulate application data

Create Add new data — *PUT*.
Read Retrieve existing data — *GET*.
Update Modify/edit data — *POST*.
Delete Delete data — *DELETE*.

User operations mapped to HTTP verbs.



# Examples

	Maps	Calendar
Model	Lat/Long	Data hierarchy
MetaData	Geo-coding	Dependencies
Operations	View, Zoom	Edit, View
Request	Name values	ATOM Feeds
Protocol	HTTP	HTTP+APP
Response	Maps	Tables



# **Tangible User Interfaces**



### **Tangible User Interfaces**

UI realized as a dynamic hypertext document!

- Connect application model to desired UI.
- Instantiate by creating an HTML DOM.
- DOM holds presentation content.
- Encapsulate content, style and interaction.

Web Applications come alive!



# **Document Is The Interface**

User interface delivered as interactive hypertext.

HTML Serialization of the HTML DOM.
DOM Encapsulates content.
CSS Style rules.
Handlers JavaScript event handlers for behavior.

Result is a UI, not a document.



# **User Interface Is Not A Document!**

Documents	User Interfaces	
Pure content	Includes interaction	
Consistent structure	Highly customizable	
Mostly static	Mostly dynamic	
User reads	User interacts	



# **Consuming Web Applications**



#### Web Browser

Web application model discovered not designed.

- Web UI rendered by the browser.
- Browsers require augmentation via AT.
- AT treats Web pages as documents.
- Web pages are now live user interfaces.

Transition causes impedance mismatch.



# **Eliminating Feature Gap**

W3C ARIA: enable AT regain lost ground.

DOM Live properties expose metadata.Role Identifies widget type.State Reflects current interaction state.

Live Regions Observer-observable relations.

Web user interfaces gain parity with desktop GUI.



### **Usable UI Patterns**



### **Usable UI**

From accessible widgets to usable applications!

- ARIA makes UI controls visible to AT.
- Web applications are more than UI controls.
- Task completion is the final determiner.

ARIA is necessary but not sufficient!



# **End-To-End Usability**

#### Steps in UI augmentation

- Automatically speak relevant updates.
- Augment icons with relevant metadata.
- Add navigation keys for random access.
- Allow user to query for information.
- Produce automatic feedback for user actions.

Not all accessibility gaps are due to bugs.



### Examples

Augmenting UI for visually impaired users

Emacspeak Extensions and Web wizards.JAWS Application-specific scripts.ORCA Application-specific Python extensions.Window Eyes User set files.

Augmentation happens at multiple levels.



# Web Applications

Web Applications present unique challenges

- Large number of small Web applications.
- Applications updated continuously.
- New features delivered incrementally.
- Enables ubiquitous access.

Web-2.0 benefits for all users?



# **Evolving Web Accessibility**

Mainstream benefits for users with special needs.

- Extend platform AT via the Web.
- Deliver augmentation via the Web.
- Distribute augmentation at Web scale.
- Expose relevant APIs to Web developers.



### **Web-Scale Augmentation**

Injection AT-neutral application augmentation.AT Scripts AT-specific augmentation.Metadata Wire-formats like ARIA in HTML DOM.Web Distributing scripts via the Web.

Approaches are not mutually exclusive.



### **Examples Of Augmentation**

Browsers Implement W3C ARIA.Screenreaders Bundle application scripts.Community Open Source projects.AxsJAX Inject AT-neutral augmentation.



# **Specialized Browsing**



#### Web APIs

Enable custom access to Webformation!

Task Task-specific gadgets, *e.g.*, weather.Environment Specialized access, *e.g.*, mobile.User Special needs, *e.g.*, AT.

Custom Web access liberates end-users!



### Conclusions

- Web applications are here to stay.
- Desktop AT has found transition challenging.
- W3C ARIA goes a long way in helping.
- Web access creates new opportunities.

Profound impact on how we work and play!



### Watch The Web Take Off!



