Ubiquitous Web Applications

Dave Raggett (W3C/Volantis)

April 2007

Contact: dsr@w3.org

Ubiquitous Web Applications

- Incorporates a very wide range of devices
- Physical world of sensors and effectors
- Access to local and remote capabilities
- Autonomous agents that act upon behalf of their users

Agent acts as both client and server

Ubiquitous Web Applications

Applications areas include:

- Home monitoring and control
  - security/surveillance, safety, energy efficiency, equipment failures and predicted failures
- Home entertainment
- Office equipment
- Mobile
- Automotive

Different Perspectives

- End Users
  - easy to use and compelling reason for purchase
- Application Developers
  - ability to add value whilst controlling costs
- Device Vendors
  - appeal to users, developers and network operators
- Network Operators
  - applications and devices that drive ARPU
- Browser Vendors
  - appeal to device vendors, developers, operators
- Website and content owners
  - attracting users and developers

Examples of Devices

- Security sensors for movement, pressure, windows/doors
- Door locks and security cameras
- Smoke, Carbon Monoxide and pollution detectors
- Lighting, heating and other environmental controls
- Household appliances (e.g. washing machine, freezer)
- Hand held remote controllers
- Flat screen display/television
- Media servers
- Home gateways
- Phones, Printers, Scanners, Cameras, Projectors, ...

Mix of networking technologies

- Multiplicity of rapidly evolving networking technologies
- Challenge of how to create applications involving a changing mix of technologies
- Wireless
  - WiFi, BlueTooth, Infrared, Ultrasound, ...
- Wired
  - Twisted pair, Optical Fibre, Powerline, ...
- Home gateways as bridges between different technologies
  - telephone, stereo system, kitchen appliances and other network-enabled devices of the future

Mix of vendors and generations

- People expect devices to work for many years
  - varies with kind of device, e.g. phone vs television
  - technology is changing on a faster time scale
- Mix of vendors
  - people will buy devices from a range of vendors and expect applications to work across vendors and device generations
- Standards are needed to meet user expectations
- This will lead to an ecosystem for device vendors and application developers

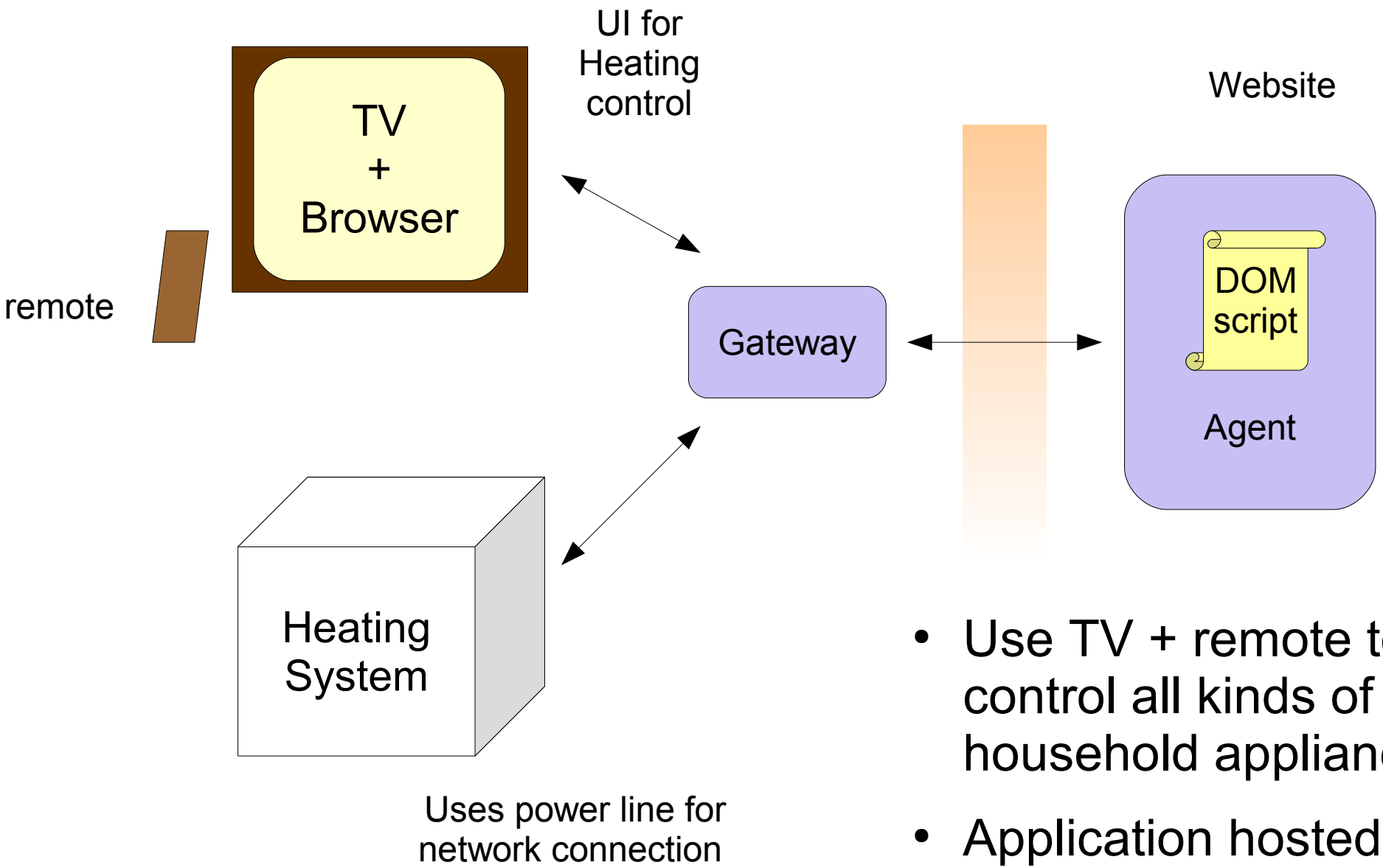
Combining local & remote services

- The Web makes it easy to exploit remote services
  - Remote servers can support streaming media, news, guides, social networks, games, photo sharing, etc.
- These can add value to native capabilities of devices
  - e.g. turn mobile phone into language translator via sending photo of Japanese menu to server for OCR and translation to English
- Web servers
  - May be used to install and manage applications as an easier alternative to traditional shrink-wrapped software
  - May provide rich descriptions of device capabilities
  - May be involved in service discovery as brokers
- Critical to motivating Network operators and Websites

Usability, Security and Privacy

- Effective security is essential
  - Don't give the front door key to intruders!
  - Protect users privacy
- Must be usable if security is to work in practice
  - Lessons from phishing
  - Don't put the burden on users
- Practical considerations
  - how to verify that this application should be granted access to a particular set of device capabilities
  - registering wireless devices with network

Home network example



- Use TV + remote to control all kinds of household appliance
- Application hosted by website

Realizing the Potential

- Initially, just proprietary solutions
  - end user purchases complete solution
  - single vendor and single product generation
- Followed by narrowly focused industry standards
  - e.g. Pictbridge as solution for printing direct from camera when printer and camera from different vendors
- Broader standards follow later, enabling new applications
  - Traditional programming languages like C++ and Java offer low level control but are costly to develop with
  - Web technologies will make applications easier and cheaper to develop, enabling a much bigger ecosystem

What's needed to achieve this?

- Standard-based architecture that decouples application authoring from the details of networking technologies and device platforms
- Standards for groups of devices with similar functions so that applications are not tied to specific devices
  - Bringing together interested parties to work on ontologies of device capabilities and exposure as APIs for markup and scripts to access these capabilities
  - Careful consideration for versioning to ensure that new devices will work with existing applications, and that new applications will work with older devices

How is W3C addressing this?

- New Ubiquitous Web Applications Working Group
  - Launched 30 March 2007
  - Successor to former Device Independence WG
  - Broadened focus on Ubiquitous Web Applications
- Support for regional subgroups
  - can hold meetings in local language, e.g. Japanese
  - greater convenience for meeting times and locations
  - meeting summaries and technical specs in English
- Balance between openness and confidentiality
  - publish approved meeting summaries and approved editorial drafts of technical documents

Why become a member?

- To get a head start on future standards
  - some competitors will be involved but not all
- To drive the direction of those standards
  - based upon your own experience and needs
- To reduce costs of development through shared test suites
  - Voice Browser WG members benefited from pooling tests, resulting in stronger test suite for lower cost
- Specifications benefit from scrutiny by people from different companies and backgrounds
  - reduces future costs by spotting problems early

UWA Approach

- Define user interface, data models and behaviour as combination of markup and event-driven scripting
  - XML + Events + RDF + Object Model
- Device coordination framework
  - descriptions, binding and use of capabilities
- Logical support for passing events between devices over different networking technologies
  - coupling devices and support for remote user interfaces
- Distinction between authoring and execution
  - policy-based content adaptation to match the delivery context (user preferences, device capabilities, etc.)

Device Behaviour

How to “program” device behaviour?

- Simple devices with fixed behaviour
- XML + scripted event handlers
  - e.g. XHTML/SVG + ECMAScript
- Pure XML with language defined event handlers
  - e.g. SCXML (StateChartXML)
    - event driven state machines as in UML
- Pure script with event handlers
  - Device has script engine + library of objects

Device Coordination Framework

Finding and binding to services
in the context of an application session

Examples of Services

- Device capabilities, e.g.
  - audio capture and playback
  - embedded camera
  - ability to initiate a phone call
  - persistent storage
  - calendar, address book, personal preferences, ...
- Speech synthesis and recognition
  - using embedded or remote speech engine
- Geographic location

“service” is used loosely for anything that Web applications might want to make use of

Binding to a Service

- Binding as a scripting interface
  - Input a service name or description
  - Output an object that proxies for the service
- May be restricted and based upon proving membership of appropriate access control list
  - Issues of trust, identity, privacy and security
  - Usability issues, e.g. asking user for decision
    - Is it okay to send location to web app?
    - Is it okay to grant access to camera?
    - What information to provide as context?
    - What if user isn't present?

Service Discovery

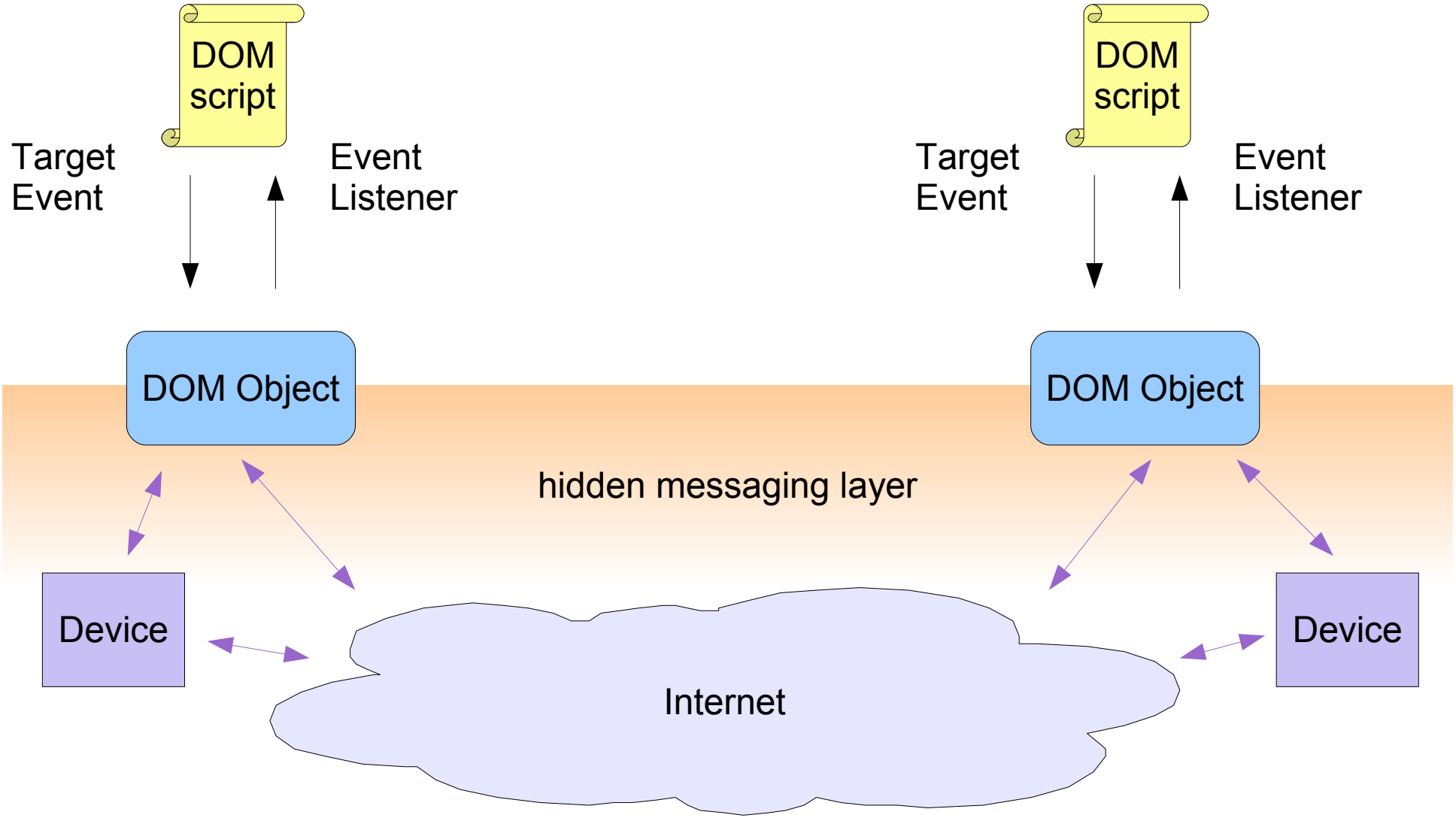
- Name service or describe its characteristics
  - URI for service or service description
  - Description as content for XML element that will act as DOM proxy for the service
- Discovery mechanism may be implicit
  - Provided by run-time environment, e.g. UPnP
- Discovery mechanism may be explicit
  - Provided by a Web server
  - Based upon external description of service

Delivery Context Client Interfaces

- Enable applications to dynamically respond to changes in user preferences, device capabilities and environmental conditions
- Exposed as tree of XML DOM Nodes
  - For example, display characteristics, playback volume level, memory size, geographical location, battery level, network availability, etc.
  - Nodes may support additional interfaces for accessing services, e.g. dimming display, or muting microphone
  - Nodes act as proxies for accessing capabilities

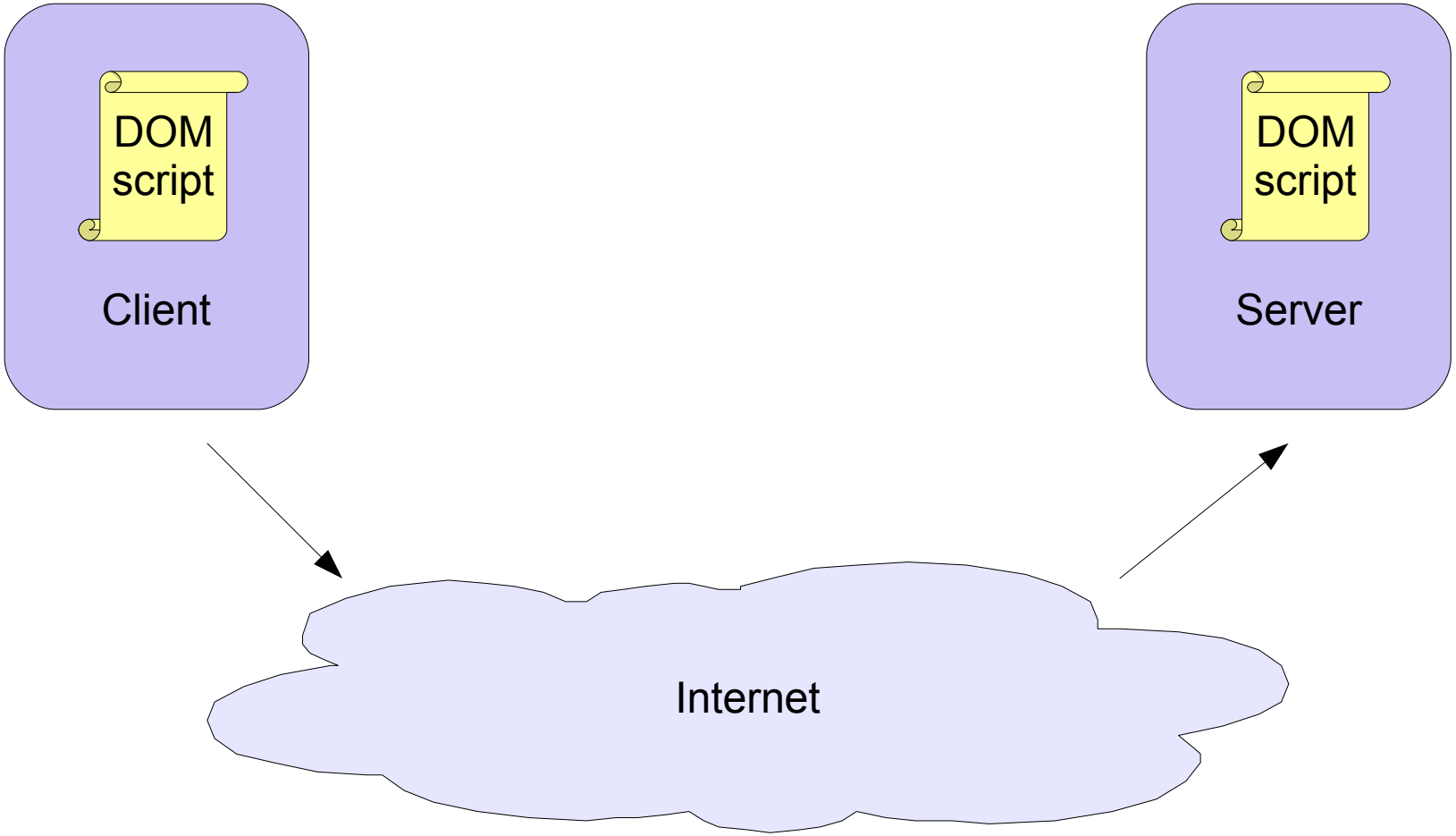
DOM = Document Object Model

Proxies for accessing services



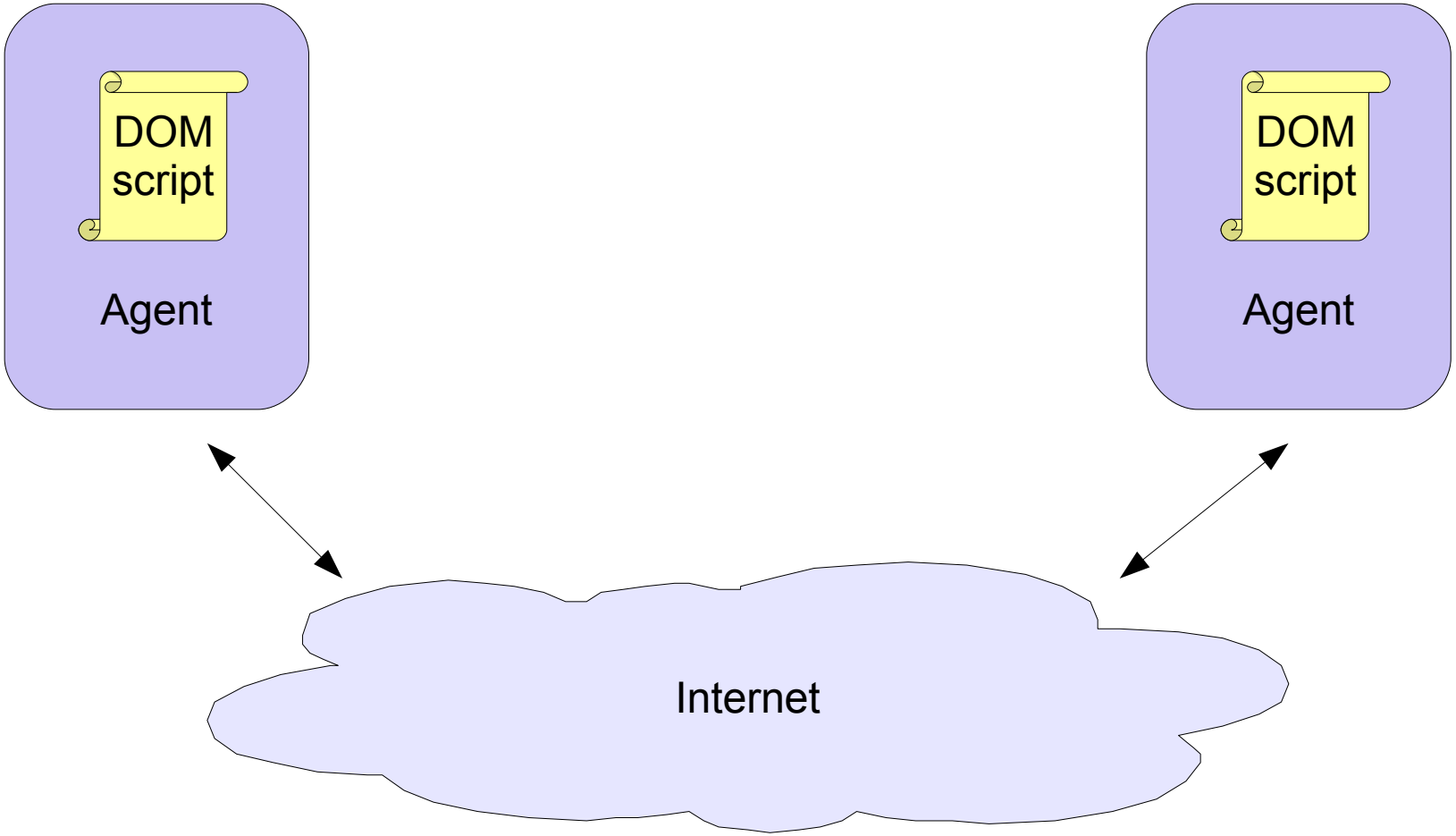
DOM – XML Document Object Model

Client or Server?



Client or Server?

Agent combines client and server

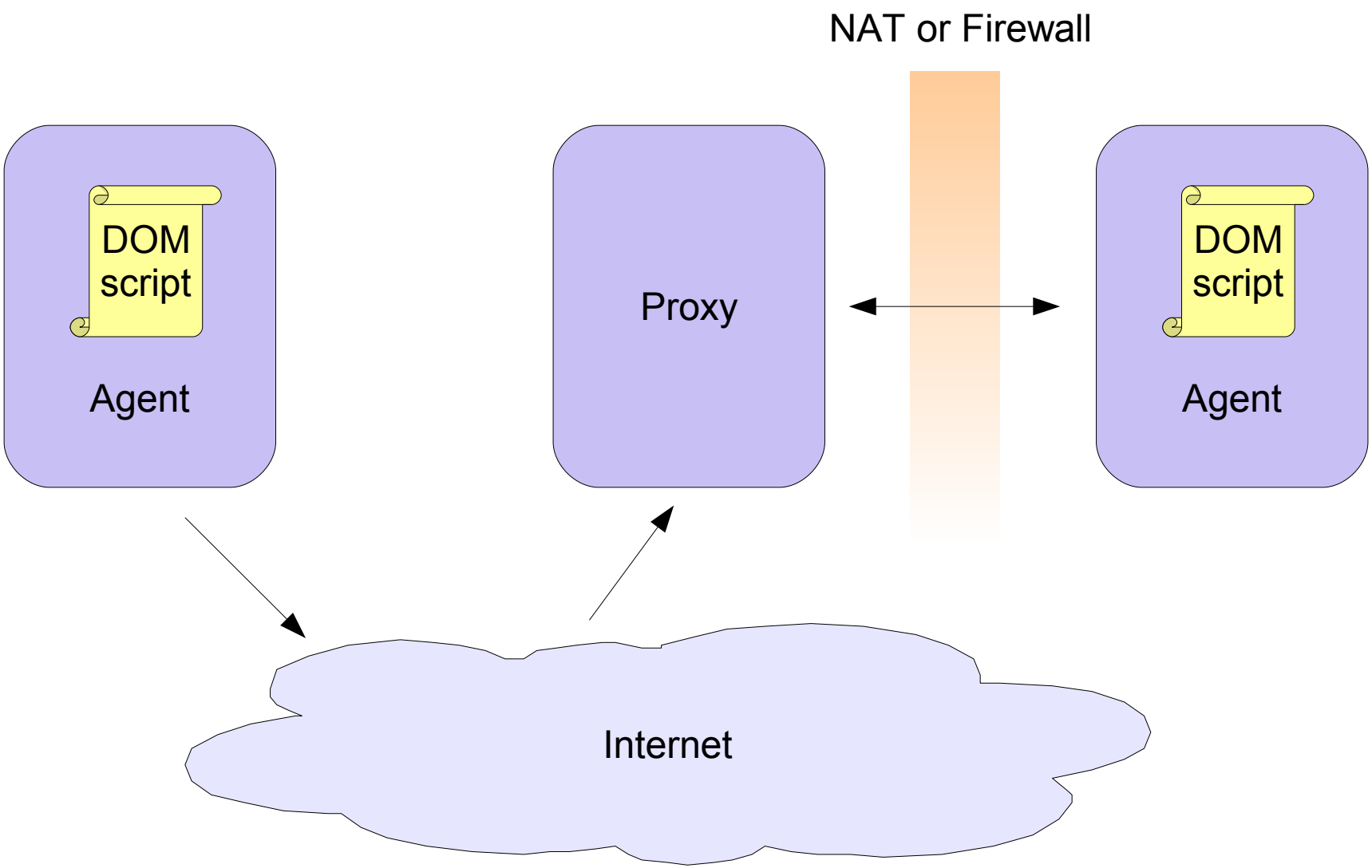


Event Transport

How to deliver events to devices?

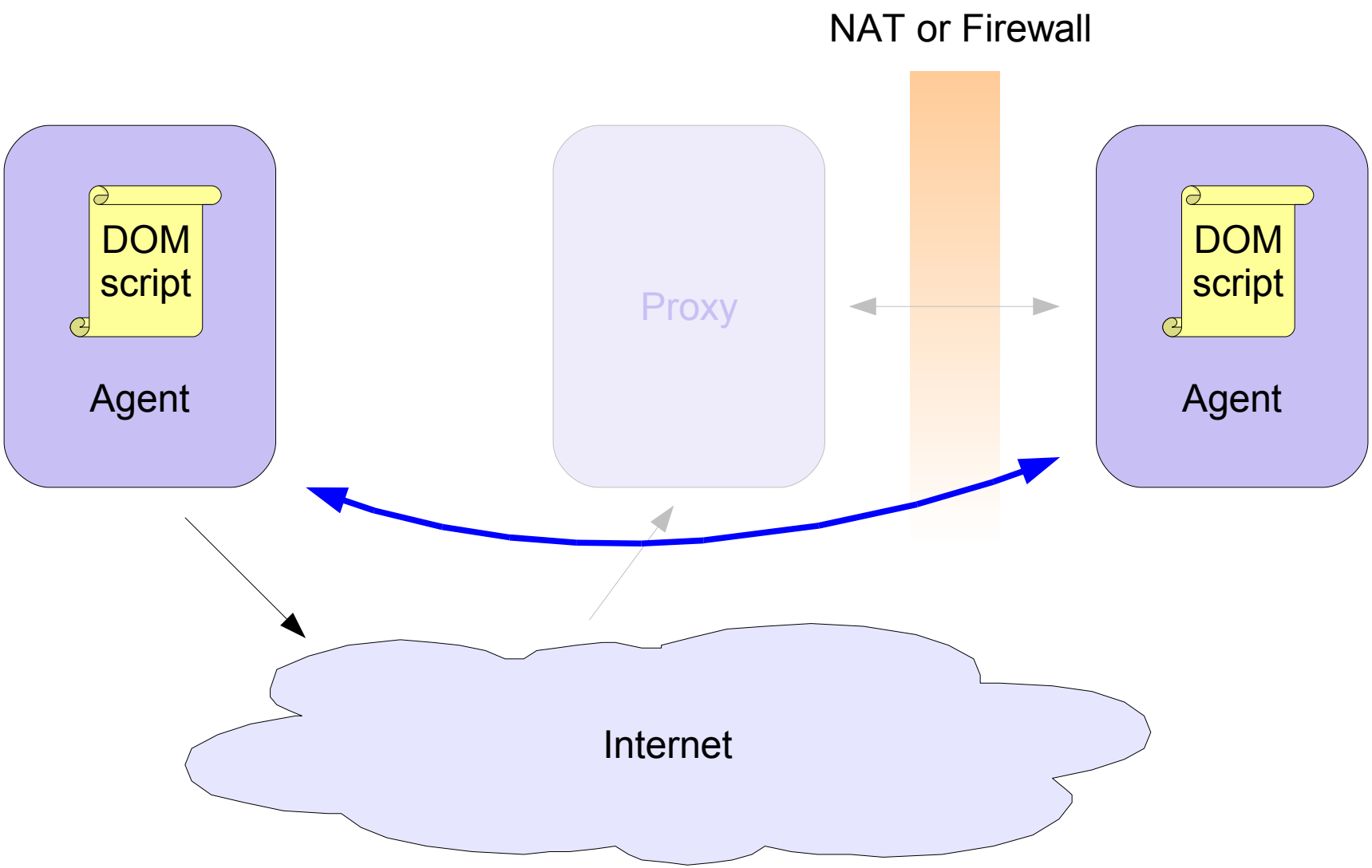
- HTTP
  - a) Add HTTP server to each device
    - But problems with firewalls/NAT
  - b) Emulate via polling/long lived connection
    - Hacks with Ajax
- Overloading SMS on GSM networks
- SIP and IMS
  - Each device acts as client and server
    - IETF/3GPP standards
    - XML representation of event as SIP message payload

Tunnelling through NAT

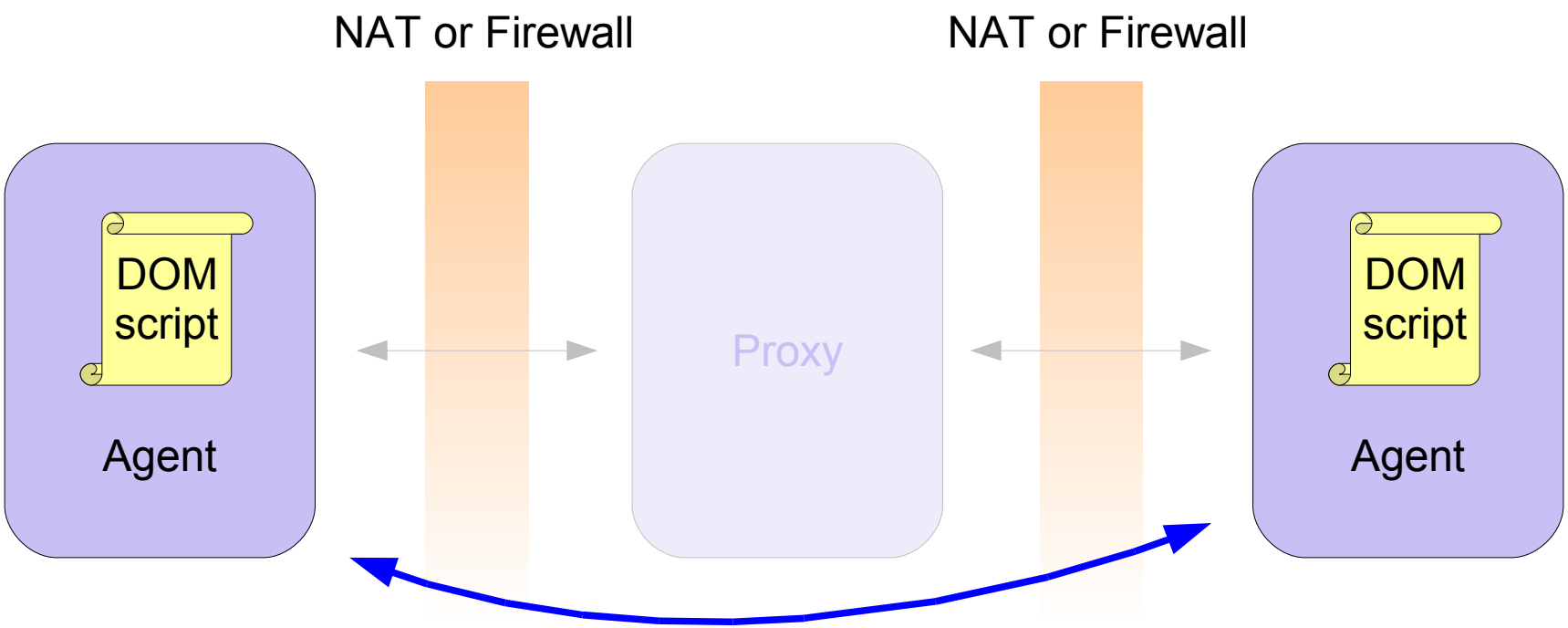


Tunnelling through NAT

Proxy may arrange for direct link through NAT

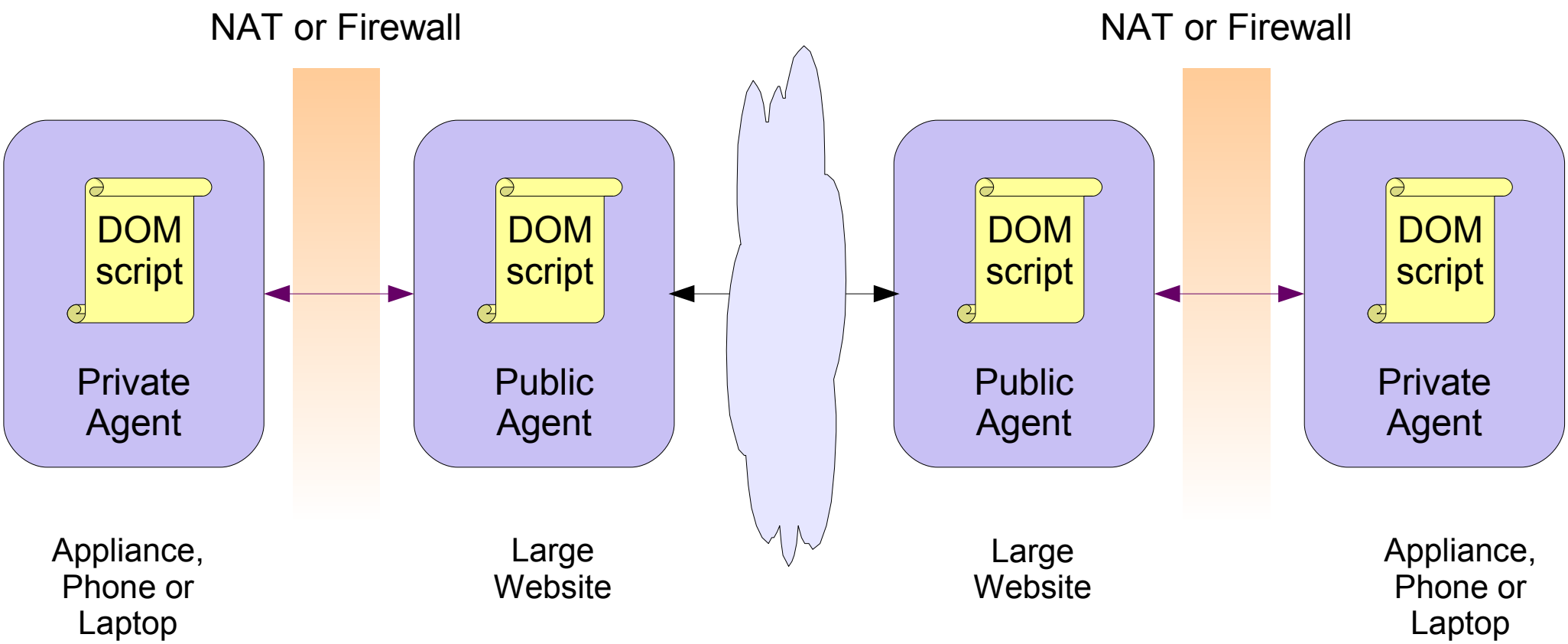


Tunnelling through NAT



Connecting devices behind different NATs

Public and Private Agents



What's needed?

- Interfaces for accessing services from web scripts
  - Need standards for common services
  - Need standards for discovery and binding
- Descriptions that can be used for discovery and adaptation purposes
  - Semantic Web technologies like Ontologies
- Policies for discovery and binding
  - Need standards for describing them
  - Cover security and privacy considerations

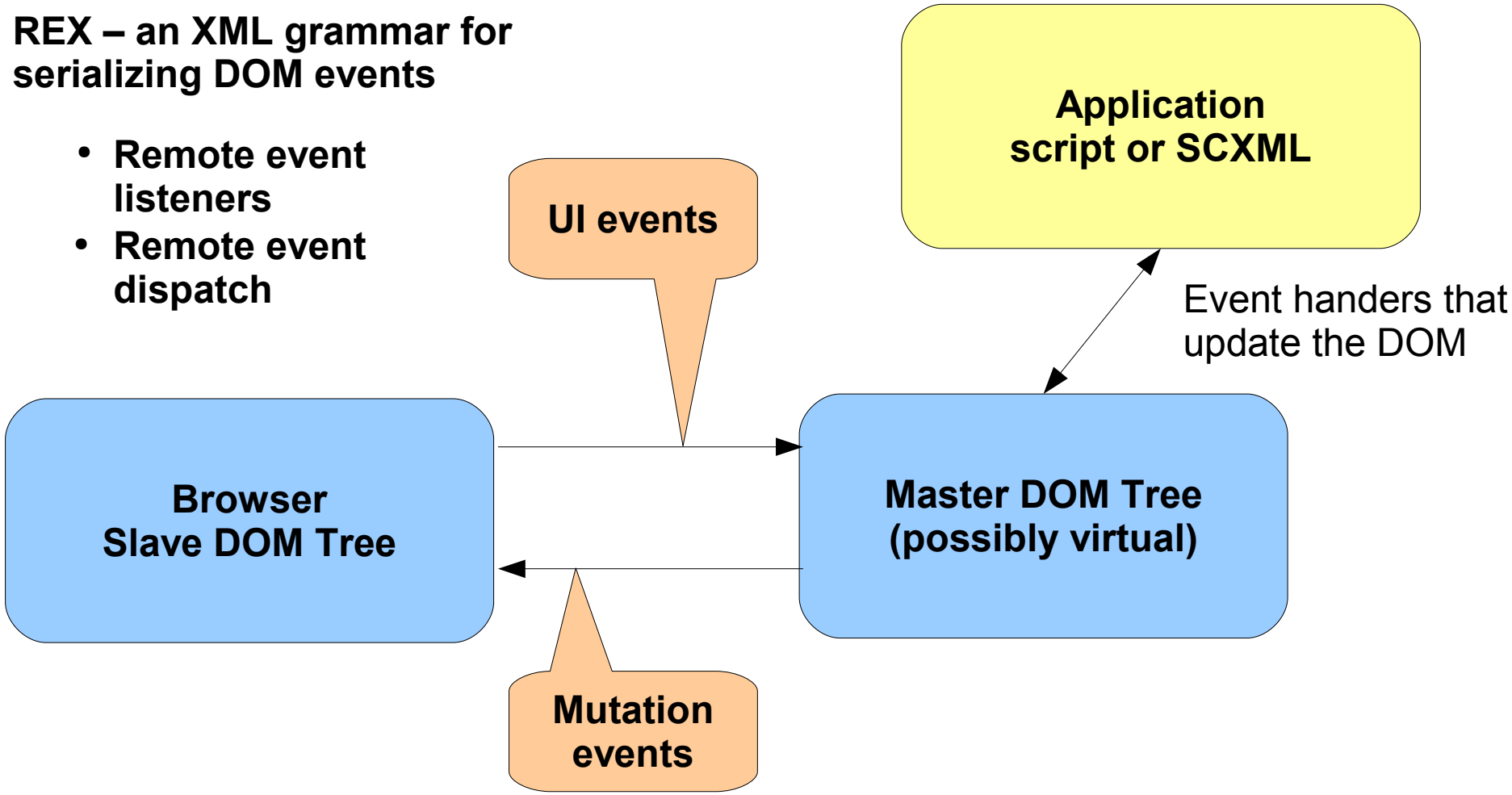
Remote User Interfaces

- Model behaviour as script or state machine
  - Interaction Manager (IM)
- Model UI as XML (XHTML, SVG, ...)
- Run UI and behaviour on separate devices
- IM sends events to update remote UI's DOM
- IM receives events from UI as result of user input
- UI can be distributed on multiple devices and controlled via single interaction manager
  - rich UI: mobile phone or remote + flat screen display
  - simple UI with buttons and indicator

REX for Distributed Components

REX – an XML grammar for serializing DOM events

- **Remote event listeners**
- **Remote event dispatch**



REX = Remote Events for XML
DOM = Document Object Model

REX

Serialize sequence of events as XML Grammar

```
<rex>

  <event target="..."  name="..." timeStamp="...">
    XML serialization of event's data
  </event>

  <event target="..."  name="..." timeStamp="...">
    XML serialization of event's data
  </event>

  ...
</rex>
```

event target identified using XPath,
timeStamp defines when to dispatch event

REX

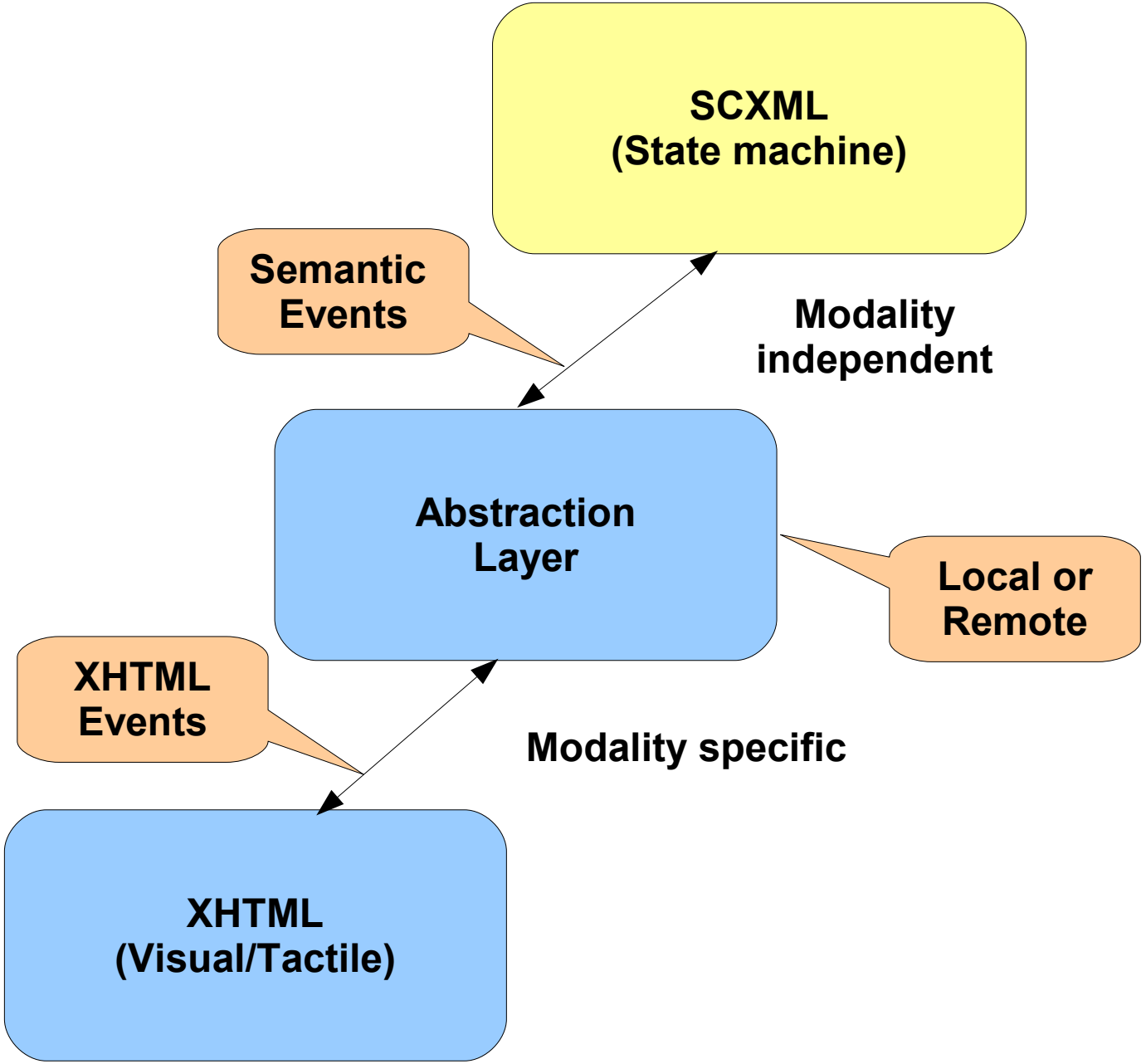
Example that updates SVG DOM tree

```
<rex xmlns="http://www.w3.org/2006/rex"
  xmlns:svg="http://www.w3.org/2000/svg">

 <event target="/svg/g/circle[1]"
   name="DOMNodeRemoved" timeStamp="1000"/>

 <event target="/svg/g"
   name="DOMNodeInserted" position="0" timeStamp="3000">

     <svg:rect x="0px" y="0px" width="200px" height="200px"
       fill="#888" stroke="#000"/>

 </event>

</rex>
```

Abstracting control

- Describe behaviour as event-driven state machine
  - Runs as agent
- Application level semantic events
- Couple UI to state machine via event transport
- XHTML + DOM operates at lower level of abstraction
- Introduce abstraction layer to mediate between XHTML events and application level events
- Abstraction layer can be located anywhere in network

Abstraction layer for Events



Adaptation

Describing applications in a way that makes them easier to run on a range of devices

Challenge of device diversity

- An ever increasing diversity of devices
- It is expensive to test on lots of devices
- My employer Volantis Systems has a database of over 4000 mobile devices with several hundred properties for each
  - browsers vary in details of scripting support, CSS bugs, etc.
  - variations in display size, fonts, kinds of buttons, memory, etc.
- Much tougher challenge than for desktop browsers

Policy-based Adaptation

- Author markup in device independent representation
  - authoring format is freed from browser restrictions
  - high level events in place of low level scripts
- Describe policies for adaptation to classes of devices
  - what layout, images, style sheets, scripts, etc.
  - skinning apps as combo of markup, CSS, script
- Adaptation process executes policies for specific delivery context
  - e.g. generate HTML4 if appropriate
  - split content for low memory devices
  - exploit client APIs for rich web apps (e.g. Ajax)

External Groups

with potential relevance to W3C work on Ubiquitous Web Apps

- 3GPP – protocols for mobile devices (GSM, W-CDMA)
- DLNA – device coordination for home entertainment
- FIPA – IEEE CS standards for agent-based technology
- HGI – devices acting as home gateways
- IETF – protocols including HTTP and SIP
- OMA – mobile application environment
- PUCC – device and service metadata for devices

Ubiquitous Web Applications WG

- Home page http://www.w3.org/2007/uwa
- Follow on to former Device Independence WG
- Plus broadened focus on Ubiquitous Web Applications
- Looking for companies interested in working on
  - enabling applications across multiple devices
  - content adaptation for multi-channel delivery
- UWA WG Charter
  - http://www.w3.org/2006/10/uwa-charter.html
  - chair: Dave Raggett <dsr@w3.org>
  - team contact: Stéphane Boyera <boyera@w3.org>

Ubiquitous Web Applications

Questions?

End Users

- Want attractive easy to use applications that justify the cost of purchase and installation
- Mix of devices from different vendors and different product generations
- Care about security, but it has got to be really simple to deal with
- Aren't Geeks and expect technology to just work!

Application Developers

- Seek to add value through innovative ways to combine devices and services
- Need standards to reduce development and testing costs
- Need to ensure applications will work across vendors and product generations
- Interested in device descriptions and flexibility when it comes to adapting to each device, and exploiting capabilities unique to that device
- Concerned with security and usability

Device Vendors

- Digital homes/offices offer opportunities for new kinds of devices
- Applications add value to the device, and standards are needed to realize that value
- Standards allow devices to be used in ways that device vendors may not have conceived e.g. use of mobile phone to control television
- Concerned about security and usability
- Devices should be able to act as agents in distributed applications

Network Operators

- Looking for new ways to drive revenues from data services
- Interested in attracting developers to create applications involving remote services that add value to the network
- Role as intermediaries e.g. for location, billing and security services

Browser Vendors

- Browser as application platform
- Interested in ways to increase value of the browser for both device vendors and application developers
- One way is through richer access to device capabilities
- Browser as agent rather than client
- Understand need for careful attention to security and usability considerations

Website and Content Owners

- Looking for ways to increase value to end users and application developers, whilst retaining control over content and services
- One approach is to add remote APIs
- Another is to support autonomous agents (wbots) that act on behalf of users and which are controlled by them
  - agents as event driven state machines
- Want to provide effective user experience over a wide range of devices/browsers
  - concern over costs arising from device diversity