Web and North American Cable TV

February 9, 2011

Clarke Stevens c.stevens@cablelabs,com

CableLabs®

…Revolutionizing Cable Technology®

North America Cable Goal

- Support one web delivery platform for commercial video services across all devices including television, set-top-box, personal computer and mobile devices.

CableLabs®
...Revolutionizing Cable Technology®

North American Cable TV and the Web: Gap Analysis

Requirement	W3C Capability
Remote user interface (RUI)	HTML5, 2D Context, WebGL, Web open fonts, Web storage, Web sockets, remote control key codes in DOM3
Content advisories, client ad insertion, subtitles, media associated applications	Timed Tracks
Content protection	No direct W3C impact. Defined as part of DLNA media format guidelines for HTML RUI platforms
Tailor RUI server behavior to client capabilities	HTML5 appears to provide features to cover these requirements
Alternate audio program	JavaScript API for a Multitrack Media Resource (Issue 152)
Home network content access	No existing W3C spec. Need discovery and media playback APIs

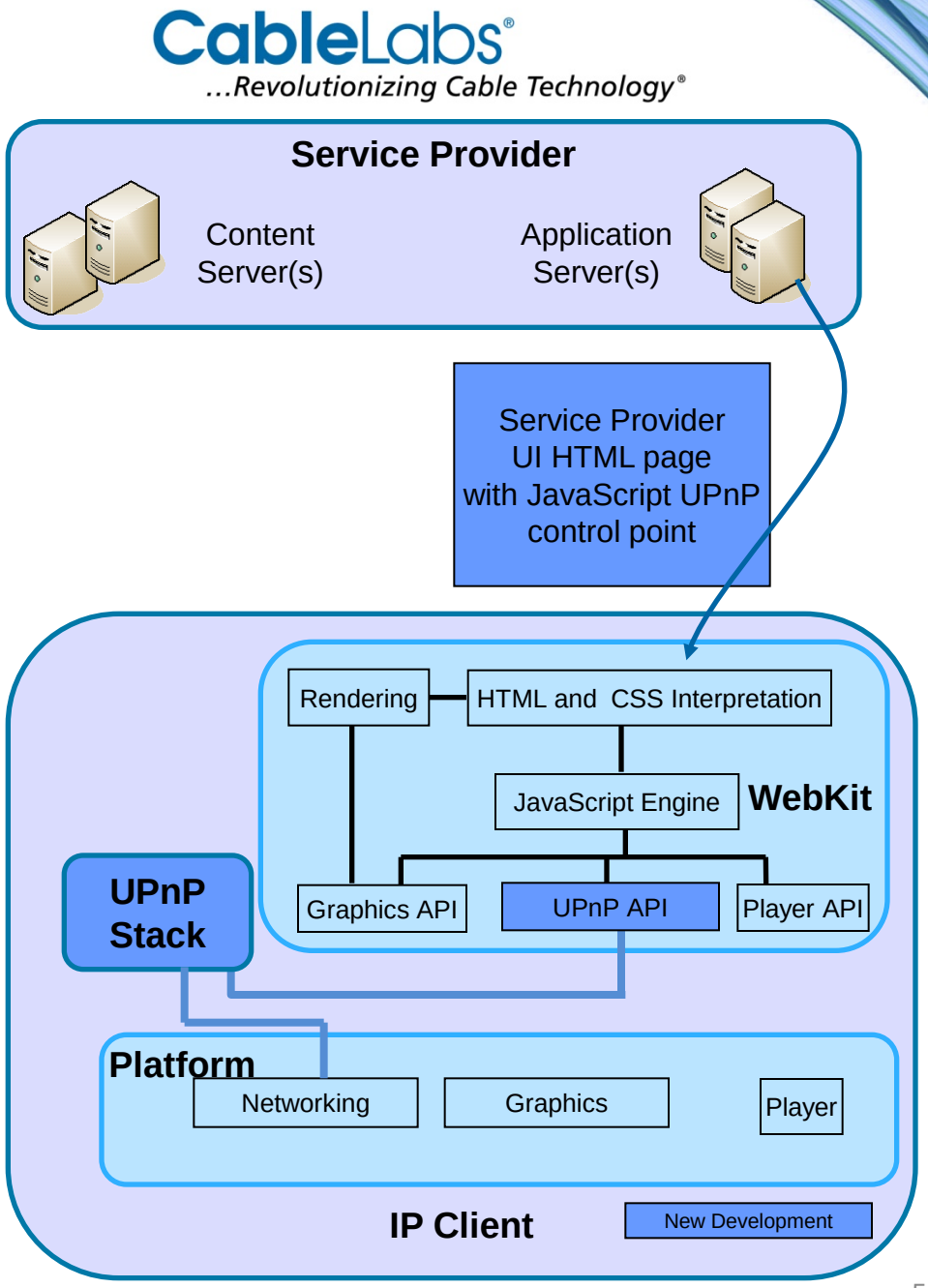
HTML5 WG Bug (11326) Identified HN Access Req

1. JavaScript API for discovering home network servers that host applications providing content serving, playback, user interface and other services.
2. JavaScript API for control of services found in the home network.
3. User agent support to determine platform support for content transport protocols, codecs and content protection capabilities.
4. Security against unauthorized access to home network services.

HTML5-WG recommendation was to work with browser implementers. To that end, CableLabs has developed an implementation on WebKit and is interested in working with browser vendors on implementation.

CableLabs' Prototype Implementation

- Port existing UPnP stack (Cyber Garage)
- UPnP WebIDL API to expose UPnP stack to JavaScript applications
- JavaScript UPnP control point
  - Local content discovery and UI presentation
  - UI for control of over UPnP devices in the home



How It Works

Example HTML Application

- When the browser loads this HTML page, a JavaScript application on the client uses the developed API to search the home network for player devices and video sources with content.
- **Select Player** and **Select Video** causes a connection to be created from the source device containing the video to the player
- **Play, Pause** and **Stop** enable the RUI client to control playback from a remote device to another remote device.

Next Steps

- Work with browser vendors on implementation
  - Plan to open-source source working code under BSD-type license.
  - CableLabs is providing design information and source code it has created to interested browser vendors.
- Generalize for other home networking protocols
  - Currently working on generalizing for mDNS/DNS-SD (e.g. Bonjour)
- Need to address security
  - Be consistent with other Web services, e.g. geolocation and device API
  - User should “opt in” to allowing web page access to devices and content
    - Can an app discover devices
    - Can an app discover content (and which content)
    - Can an app play content

CableLabs®
...Revolutionizing Cable Technology®

Informal Web and TV Workshop Poll

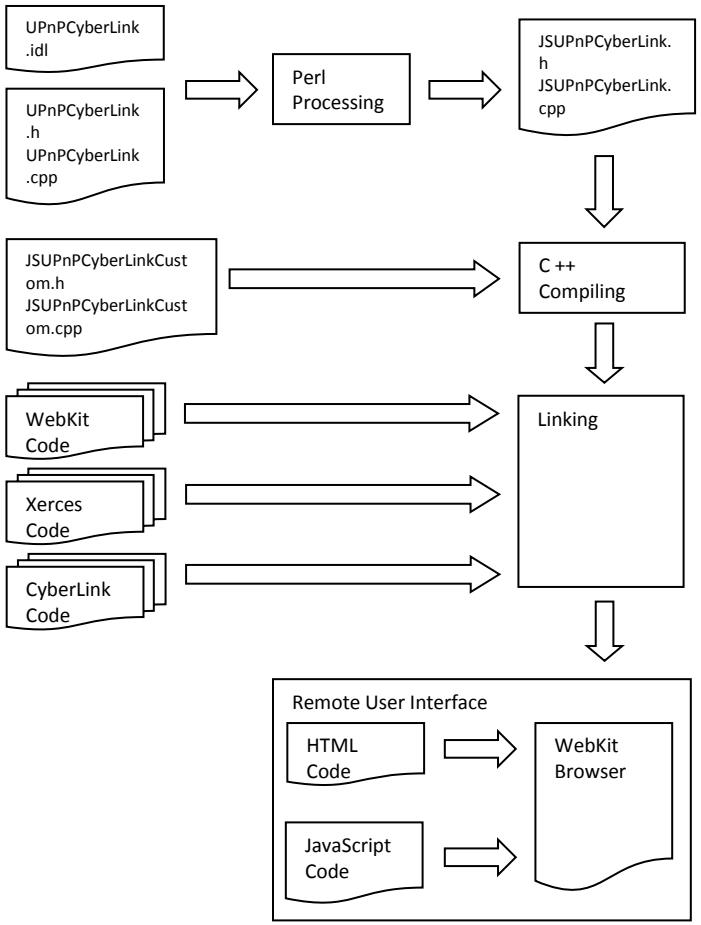
- Support HTML5 JavaScript API for selecting multiple audio/video tracks? (e.g. multi-track audio, etc.)

- Support W3C JavaScript API for local network device/service discovery?

Backup

Existing Implementation

- Clinkc UPnP stack (BSD like license)
- API implemented in WebKit
- API is defined using .idl file
- API is implemented with CyberGarage and Xerces libraries
- New JavaScript APIs are used to implement UPnP/DLNA
- Discovery control point implemented in HTML and JavaScript



API Definition

.idl	UPnP	Zeroconf
Discovery		
Discover Devices	AutoIP	AutoIP
	Browse()/Search() to get device name, etc	mDNS name resolution
	DeviceAdded(); DeviceRemoved(); DeviceChanged()	mDNS; DNS Dynamic Update
Discover Services	Browse()/Search() to get services info for each device	DNS-SD to discover basic service info (device name, IP, domain, interface, service name & type)
	Service interfaces defined in UPnP and DLNA guidelines	Service interface discovery application specific
Service Control (Assign Methods & Attributes)	Device and service control defined by UPnP. New interfaces can be learned from XML (SOAP messages)	Device and service control vendor defined. A priori information required.

.idl Interface in WebKit

- Web interface definition language (IDL) used to define abstract interface
- Common abstracted interface can accommodate UPnP, Bonjour and other protocols
- IDL provides abstraction layer from source code language
- Development of common APIs will be interactive process

```
module core {
interface [
        DelegatingGetOwnPropertySlot,
        DelegatingPutFunction,
        CustomDeleteProperty,
        CustomGetPropertyNames,
        CustomDefineGetter,
        DelegatingPrototypePutFunction,
        CustomPrototypeDefineGetter,
    ] UPnPCyberLink {
        [Custom, V8OnInstance] void assign(in DOMString url);
        [Custom, V8OnInstance] void replace(in DOMString url);
        [Custom, V8OnInstance] void reload();

        // added UPnP functions
        [Custom] void myPlay(in int contentNumber, in int playerNumber);
        [Custom] void myPause(in int playerNumber);
        [Custom] void myStop(in int playerNumber);
        [Custom] void discoverDevices();
        [Custom] void deviceName(in int playerNumber);
        [Custom] void discoverContent();
        [Custom] void contentName(in int contentNumber);
        [Custom] void contentResource(in int contentNumber);

        // UPnP implementation attributes
        attribute [CustomSetter] DOMString numDevices;
        attribute [CustomSetter] DOMString deviceNumber;
        attribute [CustomSetter] DOMString isDevicesChanged;
        attribute [CustomSetter] DOMString numContentItems;
        attribute [CustomSetter] DOMString contentItem;
        attribute [CustomSetter] DOMString isContentChanged;

#if defined(LANGUAGE_JAVASCRIPT) && LANGUAGE_JAVASCRIPT
        [DontEnum, Custom, V8OnInstance, V8ReadOnly] DOMString toString();
#endif
    };
}
```