SONY

startSession("WoT Devices")

Tatsuya Igarashi

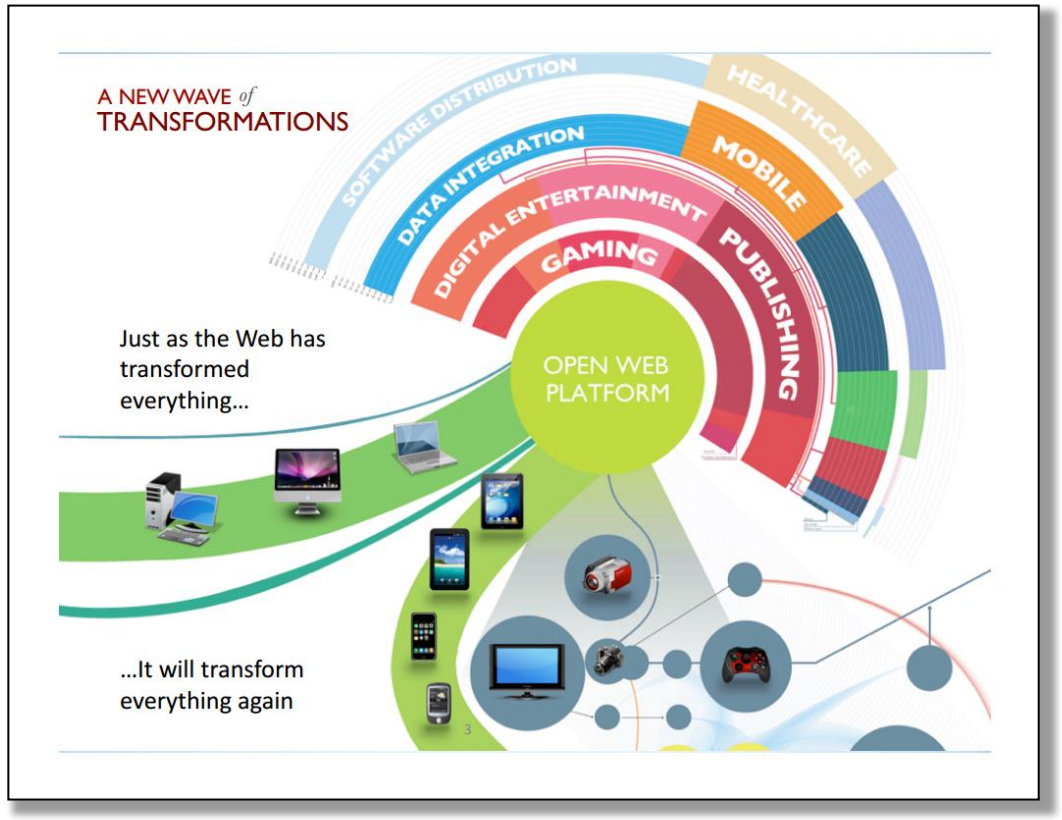
Sony Corporation

W3C TPAC 2014
Break-out Session

Oct. 29, 2014

Expanding Open Web Platform

Mobile, TV, Publishing, Automotive, etc.



Jeff's slide at TPAC 2011
http://www.w3.org/2011/Talks/jj-tpac-20111102.pdf

Jeff's slide at TPAC 2012
http://www.w3.org/2012/Talks/jj-tpac2012-plenary.pdf

Now, Web of Things

SONY

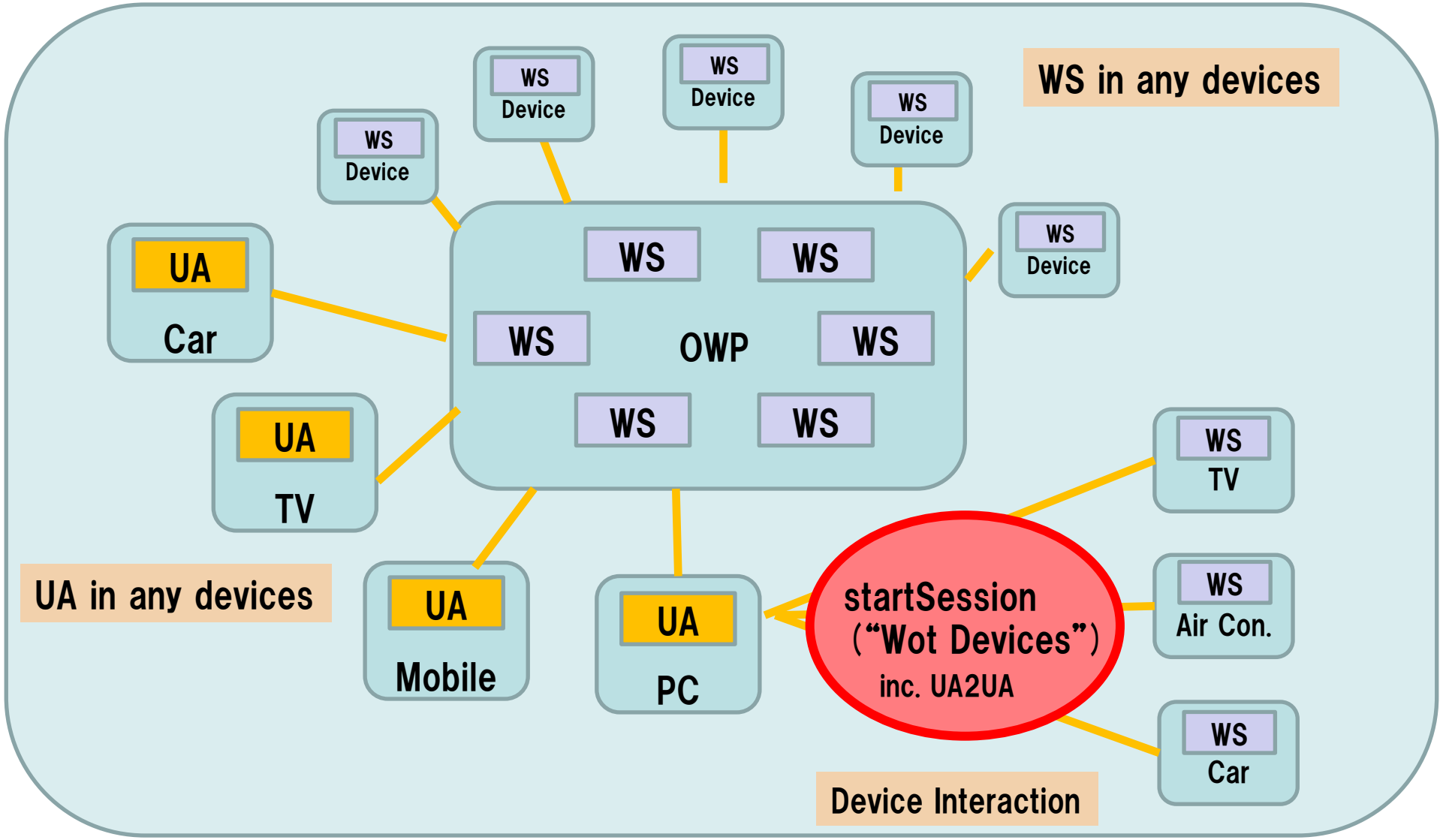
Open Web Platform reaches to devices functioning as tags, sensors and actuators for the physical environment, i.e. the Internet of Things (IoT).

Web of Things Workshop : http://www.w3.org/2014/02/wot/

startSession(“WoT Devices”)

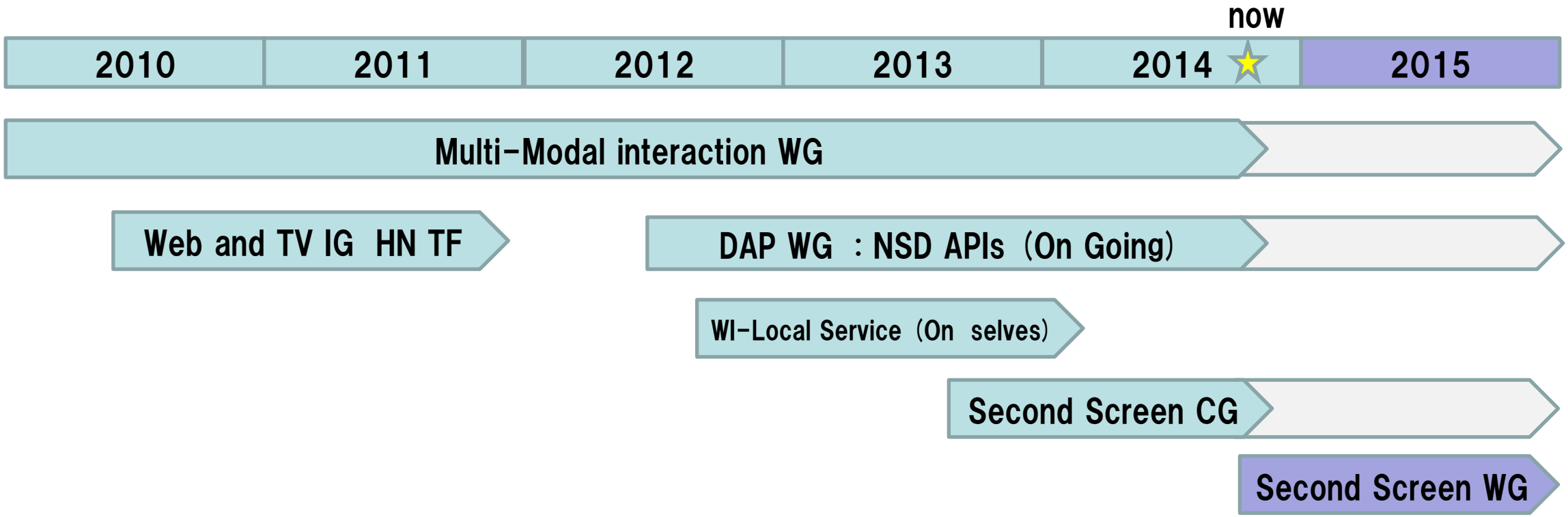
SONY

Some aspects of expanding OWP, but narrow the discussion at Today’s session.



Challenges for Device Interaction

SONY



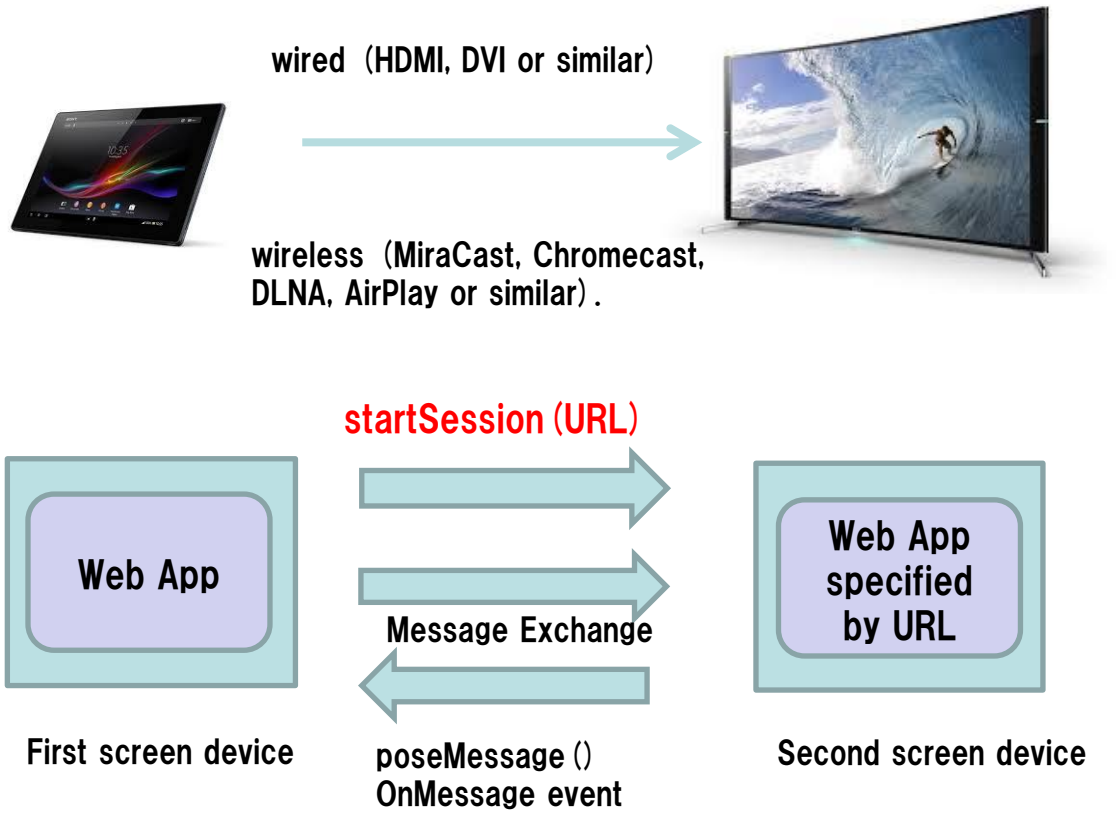
Dec 2011 Web and TV IG : Requirements for Home Networking Scenarios (Note)
July 2012 MMI WG: Registration & Discovery of Multimodal Modality Components (Note)
Aug 2012 Device APIs WG: Network Service Discovery and Messaging (FPWD)
Oct 2012 Device APIs WG: Web Intents Addendum-Local Services (FPWD)
Nov 2013 Second Screen CG: Presentation API (Draft Report)
Nov 2014 Second Screen WG is approved. (Charter)

Presentation API

SONY

The API aims to request to present a HTML document at external screen and establish a session to communicate with the web application. Note that underlying protocols or interfaces are out of scope.

```
/* controller.html */
<button disable>Show</button>
<script>
var presentation = navigator.presentation,
                   showButton = document.querySelector('button');
presentation.onavailableChange = function€ {
    showButton.disabled = !e.available;
    showButton.onclick = show;
}
function show() {
                   presentation.startSession("http://example.org/").then(
                   function(session) {;
                    // Communicate with opener page.
                                  session.postMessage(/*...*/);
                                  session.onmessage = function() {/*...*/};
                                  session.onstatechange = function() {
                                        switch (this.state) {
                                              case "disconnected":
                                              // Handle disconnection from opener page.
                                  }
                   },
                   function() {
                                  // User cancelled, etc.
                   });
};
</script>
```



Note that depending on the underlying protocols UA may be not implemented in second Screen device. For example, in case of MiraCast, screen images rendered by UA in the first screen device is transferred to second screen device via the interface.

What Presentation API is different ?

SONY

		Presentation API	Network Service Discovery API
Use Cases		Limited. presenting a web content at remote screen	Broad. Control any device/service which can communicate via UA supported protocols, e.g. HTTP.
API	Discovery	No. UA supports a selection among discovered devices. The API does not expose discovery information. (No Privacy Concern)	Yes. API provides list of discovered services.
	Messaging	Yes. Cross-Origins Access is at UA's discretion. i.e. Out of Scope.	Yes with CORS. XHR, WebSocket, WebRTC, etc.
Protocols	Discovery	Out of Scope	Yes UA supports SSDP/mDNS/DIAL
	Messaging		Yes UA supports HTTP/WebSocket/WebRTC, etc.

Open Questions

SONY

Presentation API is a reasonable approach just to standardize a common messaging API with other application. But, some open questions.

1) Why the common messaging API should not be used to interact with non-screen devices, i.e. WoT devices ?

2) How to ensure interoperability if WoT device are provided by different vendors ?

3) What is the security model for cross-origin access to WoT devices ?

SONY

1) Common Messaging API for WoT Devices

- startSession(URL or URI?)
  - URL is used to specify a document to be presented in 2nd screen
    - E.g. startSession(“http://example.com/wot.html”)
  - URI is also be used to identify an (native) application on WoT Devices ?
    - E.g. startSession(“urn:schema-example-com:wot”)
- Discovery is UA implementation dependent
    - Note that UA should filter the device list because every device does not support the specified application.
- Messaging API is very generic enough to support some use cases that a web page interacts with a WoT device.

SONY

2) Interoperability with WoT Devices

- Some good stories about role-sharing, e.g. WebSocket, WebRTC
  - API: W3C
  - Protocol: IETF

- How about WoT ?
  - IETF defines mDNS, CoAP as IP layer
  - PHY/MAC layers vary in WoT devices, e.g. WiFi, BT, Zigbee, ANT+

SONY

3) Security model to interact with WoT Devices

- Does Content Security Policy for Web also work for WoT ?
  - Can CORS of XHR and WebSocket be used along with Presentation API ?
  - WoT devices would not be able to support TLS/SSL practically.

- Can we standardize a security model ?
  - Presentation API leaves security policy to UA implementation dependent.
  - This seems to be pragmatic, but do we have any other way ?