ADVANCED WEB TECHNOLOGIES

MULTISCREEN TECHNOLOGIES AND STANDARDS I + II

Louay Bassbouss | Open Distributed Systems | lecture winter term 2016/17

MULTISCREEN TECHNOLOGIES AND STANDARDS

Part I: December 1, 2016
Part II: December 8, 2016

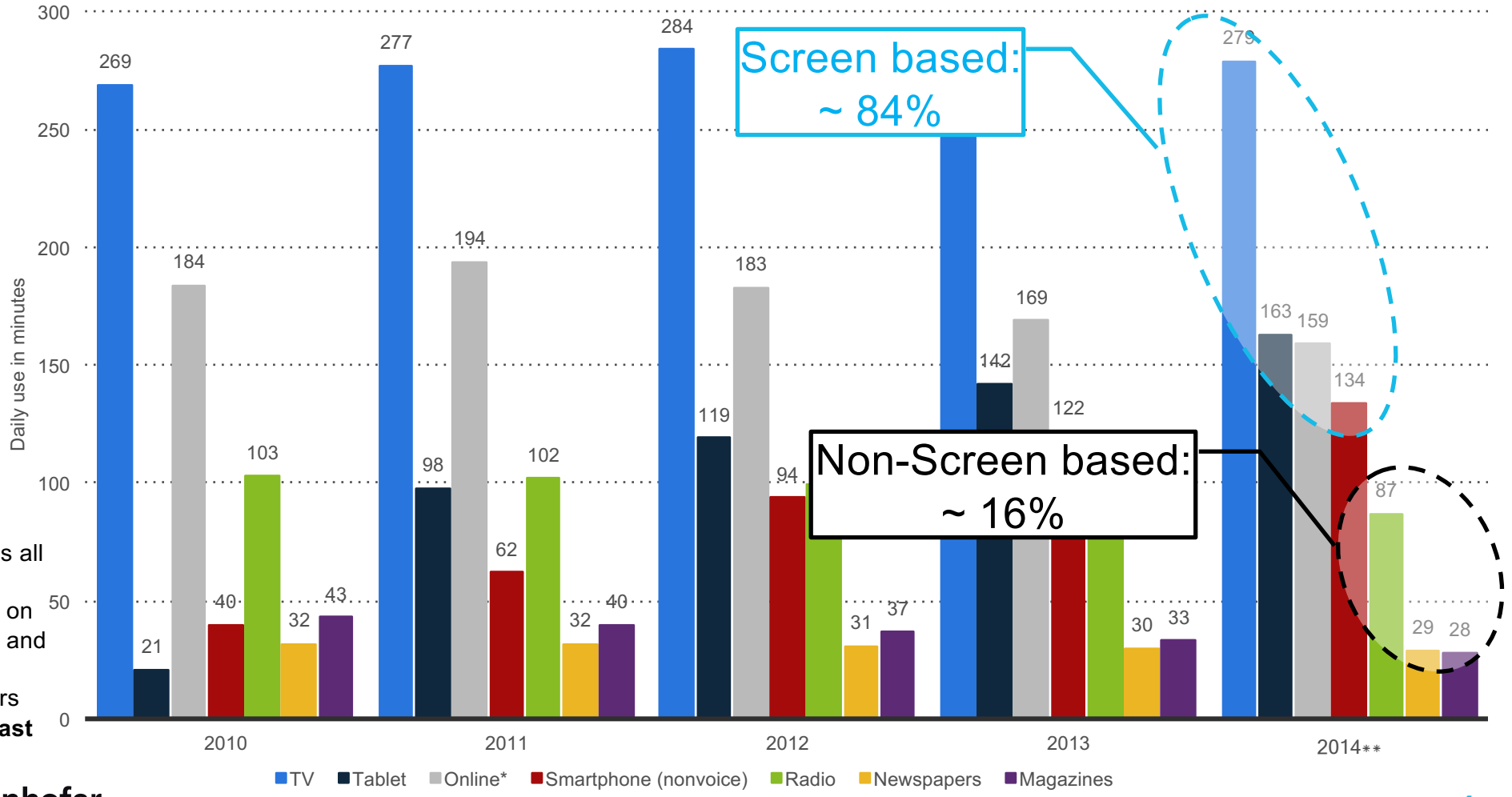
Part I	**Intro & Setting the scene** - Terminology - Multiscreen in Action – practical examples - Use Cases, Features and Challenges
Part II	**State of the Art Technologies and Standards** - W3C APIs, HbbTV, Smart TV SDKs, Android/IOS SDKs - UPnP / DLNA, Bonjour/AirPlay, DIAL, Miracast / WiDi, iBeacon, Physical Web

MULTISCREEN TECHNOLOGIES AND STANDARDS

Part I: November 19, 2015

Part I	**Intro & Setting the scene** - Terminology - Multiscreen in Action – practical examples - Use Cases, Features and Challenges
Part II	**State of the Art Technologies and Standards** - W3C APIs, HbbTV, Smart TV SDKs, Android/IOS SDKs - UPnP / DLNA, Bonjour/AirPlay, DIAL, Miracast / WiDi, iBeacon, Physical Web

AVERAGE DAILY MEDIA USE IN THE UNITED STATES FROM 2010 TO 2014



* Includes all internet activities on **desktop** and **laptop** computers
** **Forecast**

Source: eMarketer; ID 270781 December 2016

TYPICAL TYPES OF SCREENS PEOPLE USE ON A DAILY BASIS

Laptop/PC

Game Console

TV/Set-Top-Boxes

Smartphones

Tablets

BUT THERE ARE MORE …

Public Screens/ Digital Signage

Smart Watches

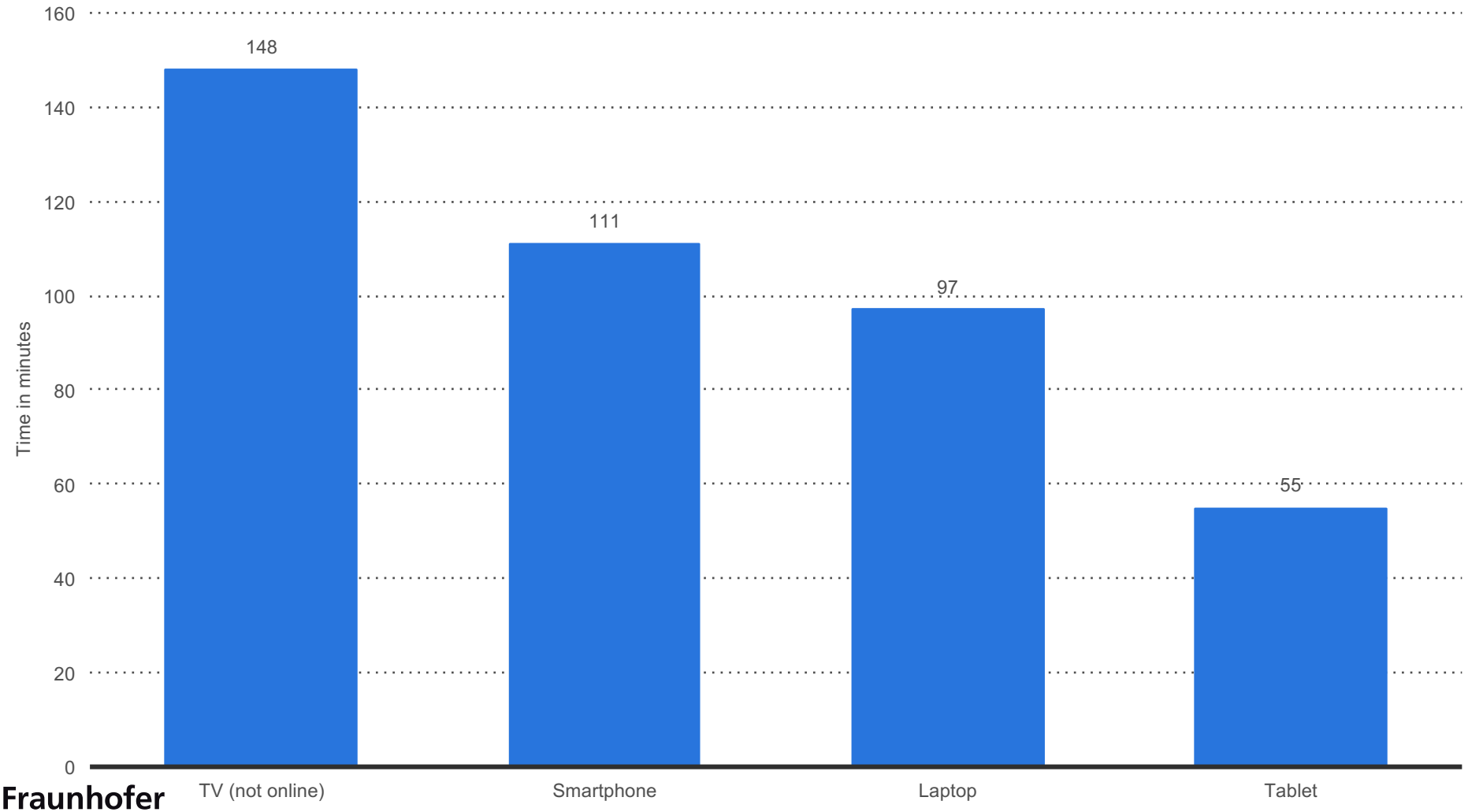
In-car infotainment

Streaming Devices

Digital Photo Frames

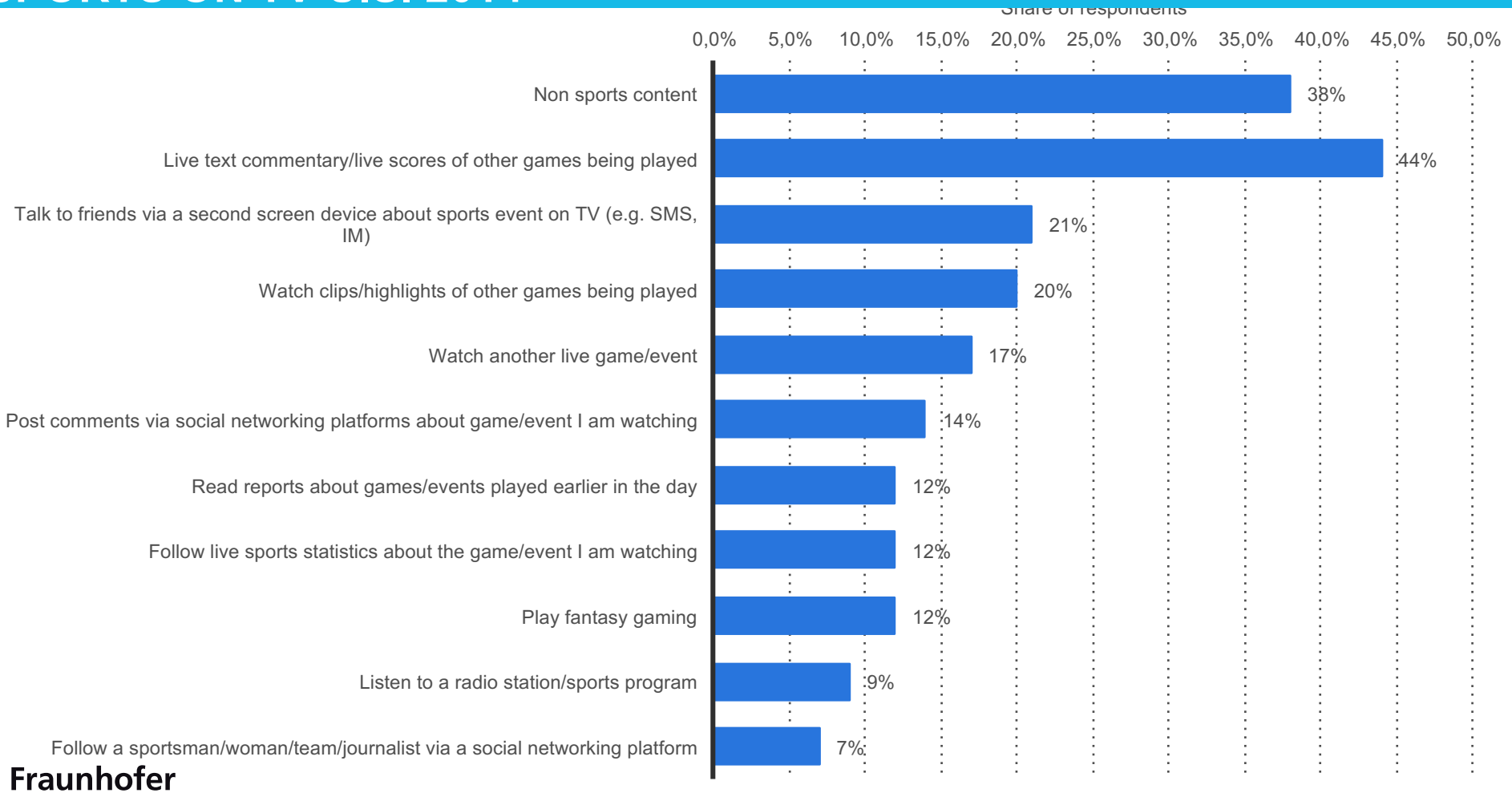
Smart Glasses

DAILY SCREEN TIME PER PERSON IN THE UK 2014, BY DEVICE



Source: Millward Brown; ID 293293 December 2016

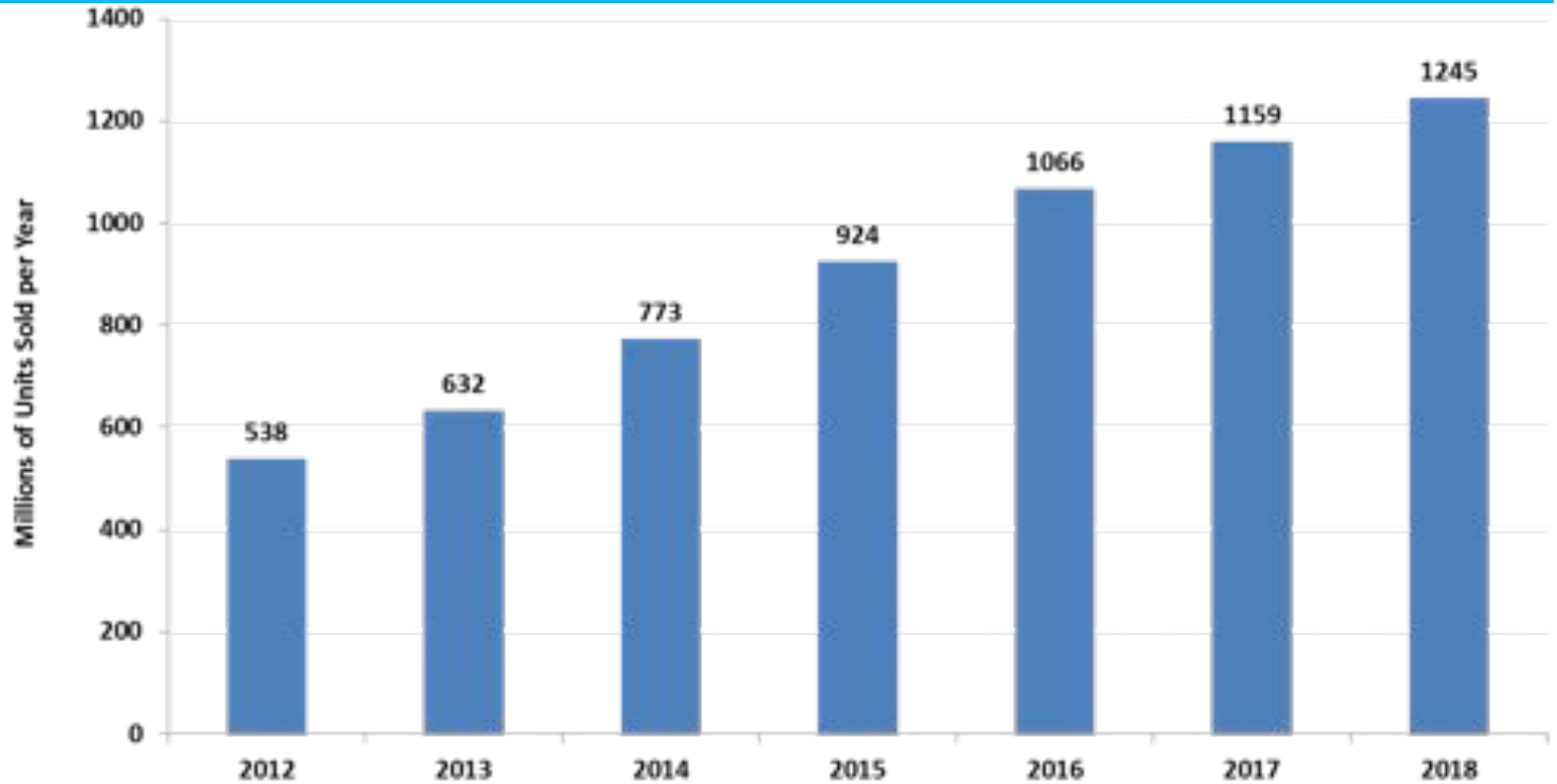
TYPES OF CONTENT ON SECOND SCREEN DEVICES WHILST WATCHING SPORTS ON TV U.S. 2014



Source: Kantar Media; SportBusiness; Perform Group (Sporting News Media); ID 316658

December 2016

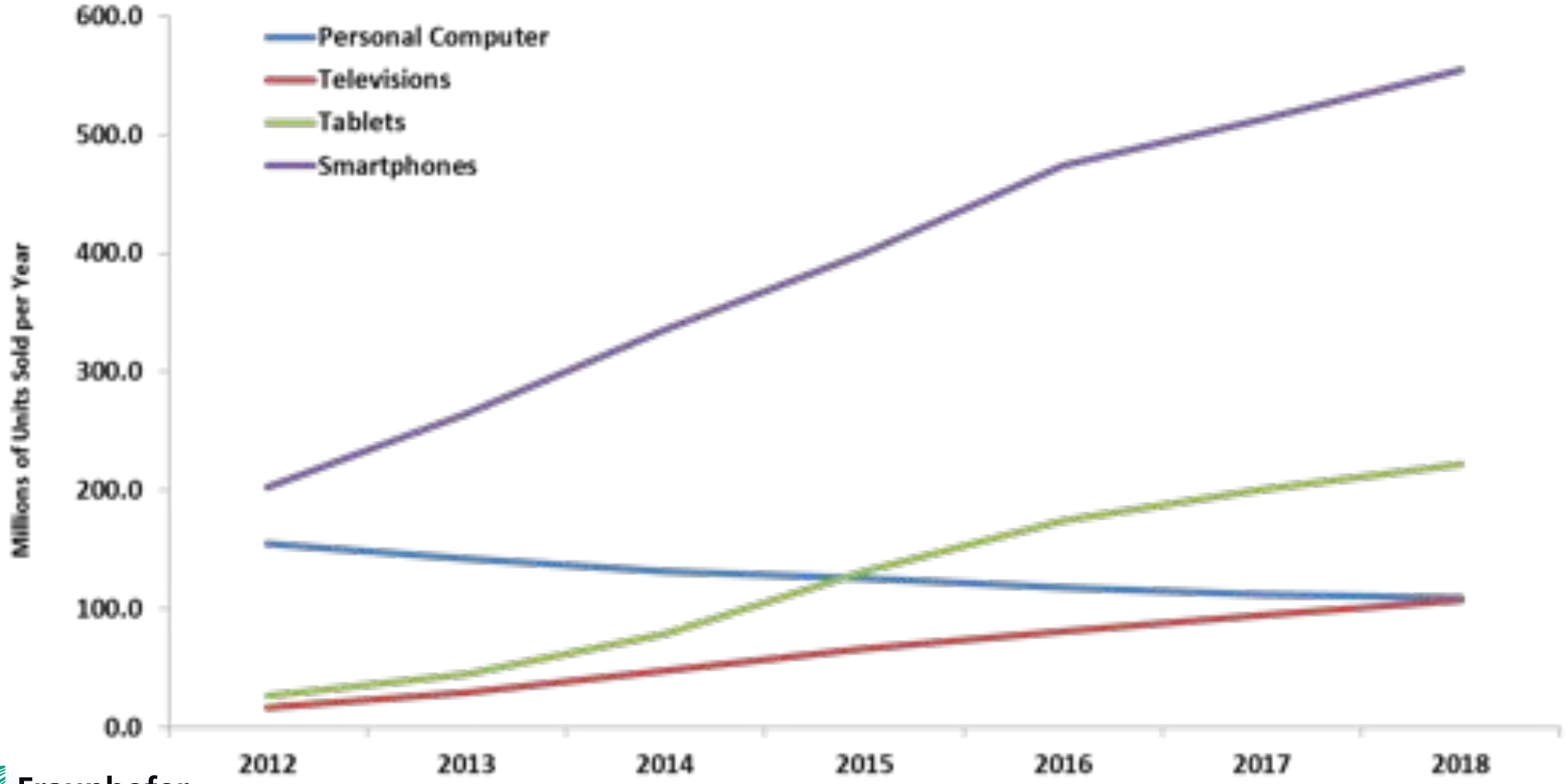
ANNUAL GLOBAL DLNA - CERTIFIED DEVICE SALES: 2012-2018



Source: DLNA Market Overview, Parks associates for DLNA, http://www.dlna.org/dlna-for-industry/newsroom/parks-associates-report-2013

GLOBAL GROWTH OF DLNA-CERTIFIED DEVICES

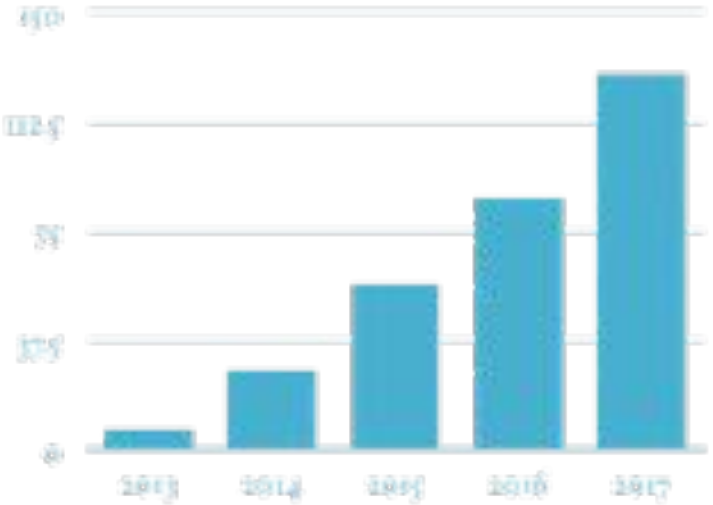
ANNUAL SALES FOR TOP 4 CATEGORIES (2012-2018)



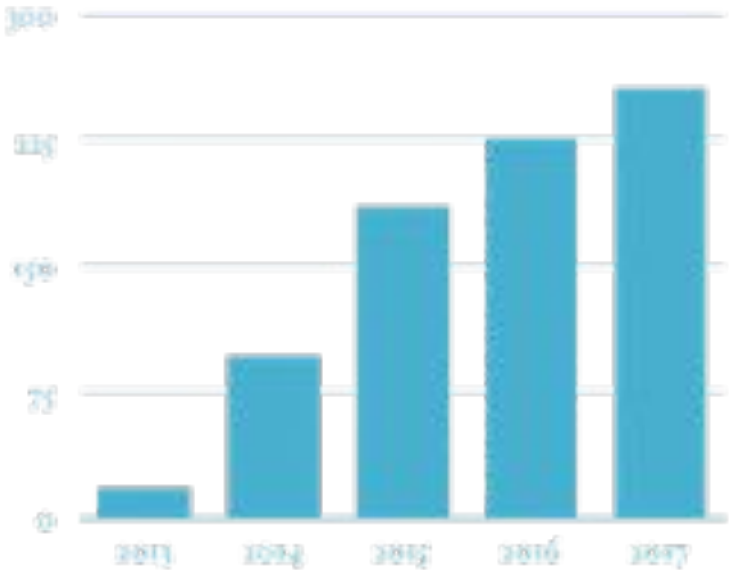
Source: DLNA Market Overview, Parks associates for DLNA, http://www.dlna.org/dlna-for-industry/newsroom/parks-associates-report-2013

MIRACAST, DIAL AND 802.11AC WI-FI FORECAST

Forecast for Miracast and DIAL enabled devices sold per year

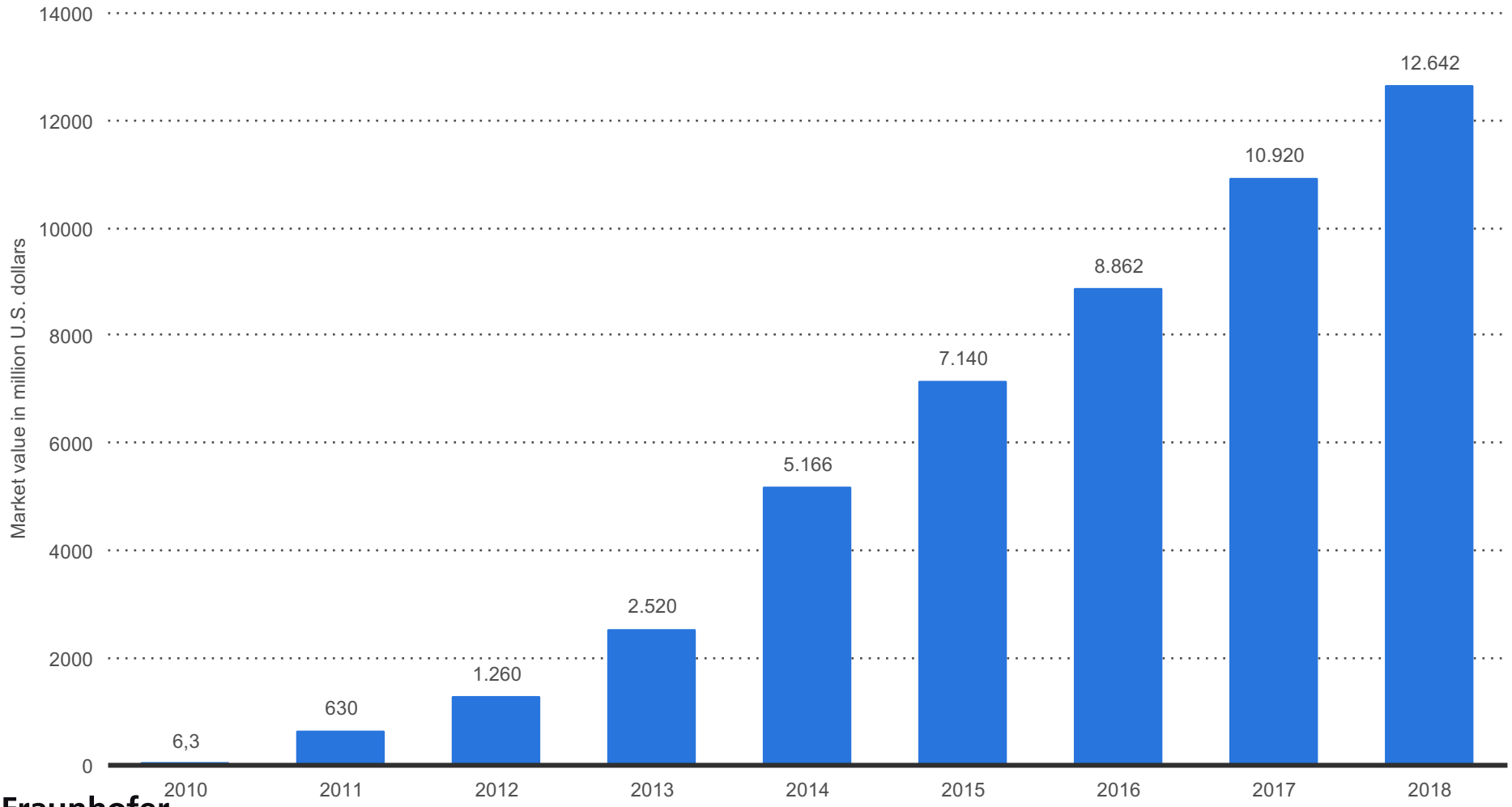


Forecast for high speed 802.11ac Wi-Fi technology sold per year



Source: http://nextmarket.co/blogs/news-1/7329546-miracast-and-dial-enabled-display-devices-to-reach-129-million-by-2017

WEARABLE DEVICE MARKET VALUE FROM 2010 TO 2018 (IN MIL. USD)



Fraunhofer FOKUS

Source: Business Insider; ID 259378 December 2016

TERMINOLOGY: DEFINITIONS VS. BUZZ WORDS

There are different buzz words, but no unique definition

- **Cross-screen**: Access information or services from different screens with display optimizations and with no mutual dependency or common execution context between application instances. Instances substitute each other.
- **Multiscreen**: Participation within a common execution context involving more than one screens with application instances interacting and complementing each other.
- **Second screen** (subset of Multiscreen): For the supplementary access to information and services through an automated or manual contextual link with the primary content distribution channel.
- **Companion Screen**: This term is usually used for devices that run apps linked to a given TV show or TV service.
- **Dual Screen**: another buzz word, mostly used iOS/Apple TV context

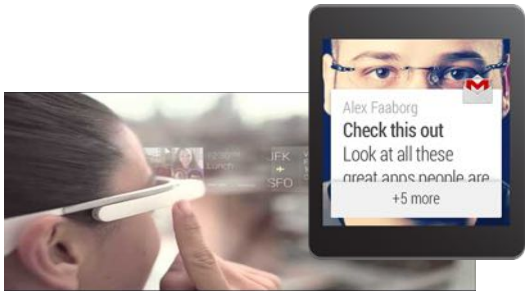
Multiscreen in Action – practical examples

MULTISCREEN IN ACTION

TV and Companion Devices

Streaming Devices

In-Vehicle Infotainment

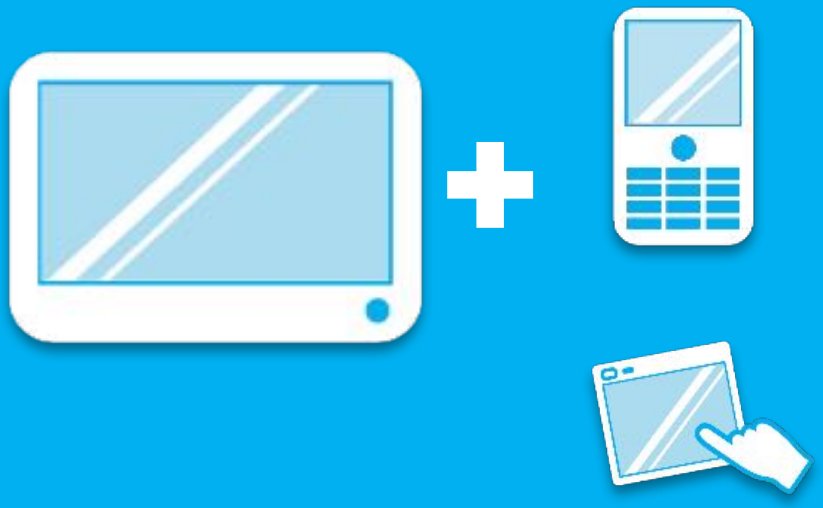


Wearables

Companion devices

PC and companion devices

TV and companion devices

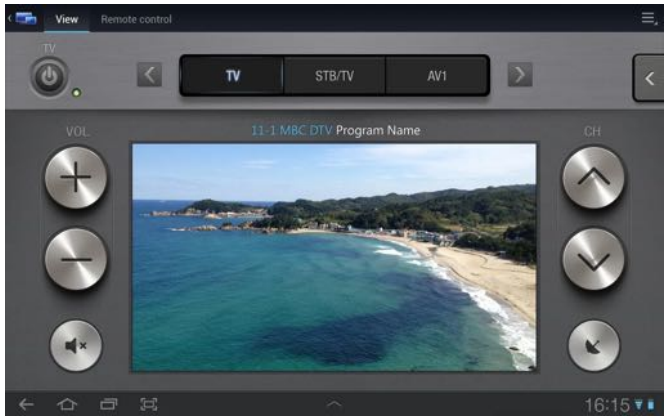


SAMSUNG SMART TV – SMARTVIEW APP

- **Remote Control**: Control TV and registered devices(STB, BDP, HTS)
- **Qwerty Keyboard** to type on TV
- **TV View**: Watch the same content with TV or different content from other input source on your mobile devices
- **Smart View of TV stream**

SmartView (Windows 8)

SmartView (iOS)



SmartView (Android)

LG SMART TV – LG TV REMOTE

- MiniTV : Continuously watching live channels on mobile devices
- Mini Home: Access to premium Content Providers and MyApps service directly
- Touch Pad: Control LG Smart TV on screen pointer(like magic remote) using touch pad
- SmartText: Easy text input via smartphone
- QuickMemo: capture, comment/draw, share TV screen
- Unified search: Searching for smart TV contents
- Smartshare: Media files from mobile on TV

Sources: http://www.lg.com/us/tv-audio-video/discoverlgtvs/smarttv/index.jsp, https://play.google.com/store/apps/details?id=com.lge.tv.remoteapps

PHILIPS SMART TV – PHILIPS MYREMOTE

- Remote Control features
- Easy text entry
- Stream pictures, video and music easily
- Browse EPG on companion device
- view ratings and select the channel without interrupting your current TV watching.
- Compatible with Philips connected products/ Net TV / Smart TV enabled Philips products

SHAZAM (FOR TV)

- **Tag Shows** (Any TV Show in the US), Get Content, Go Social
- **Music in the show**: See all the songs featured in the broadcast
- **Celebrity Buzz**: Find all the latest entertainment news – and gossip! – about the cast, guests and show
- **Trivia**: Get the lowdown on fun facts and "did you knows"
- **Social Sharing**: See what the cast and celebs are tweeting, or tweet what you're thinking

Source: http://www.shazam.com/

INCREASE TV AD RECALL & LIKEABILITY WITH SHAZAMABLE ADS

- According to Nielsen study, TV Ads with a Shazam call-to-action have overall higher recall and likeability than those without
- Brands with Shazamable TV ads are able to extend the conversation with viewers in a way that complements what they are seeing on their TV
- The study showed that Shazamable television ads outperform other ads from the same brands, including:
  - 6 percent higher ad recall,
  - 10 percent higher ad + brand recall,
  - 14 percent higher ad/brand + message recall; and,
  - 13 percent higher recall + likeability.

Source: http://news.shazam.com/pressreleases/absolut-gillette-and-jaguar-increase-television-ad-recall-likeability-with-shazamable-ads-1037055

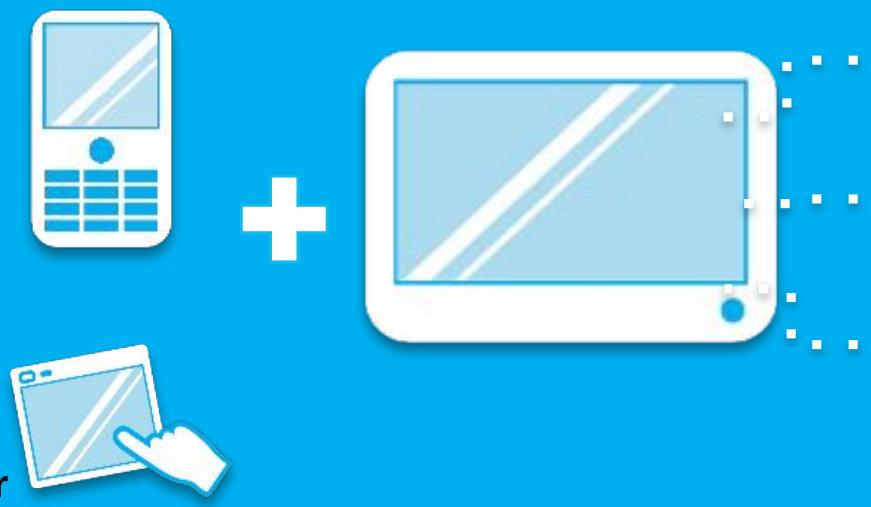
ARD.CONNECT

- Companion App for ARD HbbTV Mediathek
- Use Tablet or Smartphone instead of TV Remote Control
- Pairing of companion device and TV using QR code
- Show additional information and EPG on tablet instead of TV

Source: http://www.ard.de/home/intern/presse/pressearchiv/ARD_connect__Starten_Verbinden_Schauen/1242262/index.html

HbbTV ARD.Connect Demo

Streaming Devices



APPLE TV

- Pre-installed Apps: YouTube, Netflix, etc.
- Build-in iOS features:
  - Streaming (Video, Music, Photos)
  - Screen Mirroring
- Features through API for 3rd party Apps
  - Dual Screen

POPULAR APPLE TV APPS


Apple Remote

Weather


AirWeb


Skype for iPad


Keynote


Google Earth


Tagesschau

CNN


Spotify Music


SoundCloud


Authentic Yoga


FaceTime


YouTube


Flipboard


Snowboard Hero

Source: https://www.apple.com/itunes/

GOOGLE CHROMECAST

Remote free

Chromecast works with devices you already own, including Android tablets and smartphones, iPhones®, iPads®, Chrome for Mac® and Chrome for Windows®

Selected Apps

Source: http://www.google.com/intl/en/chrome/devices/chromecast/index.html

POPULAR CHROMECAST APPS

ROKU

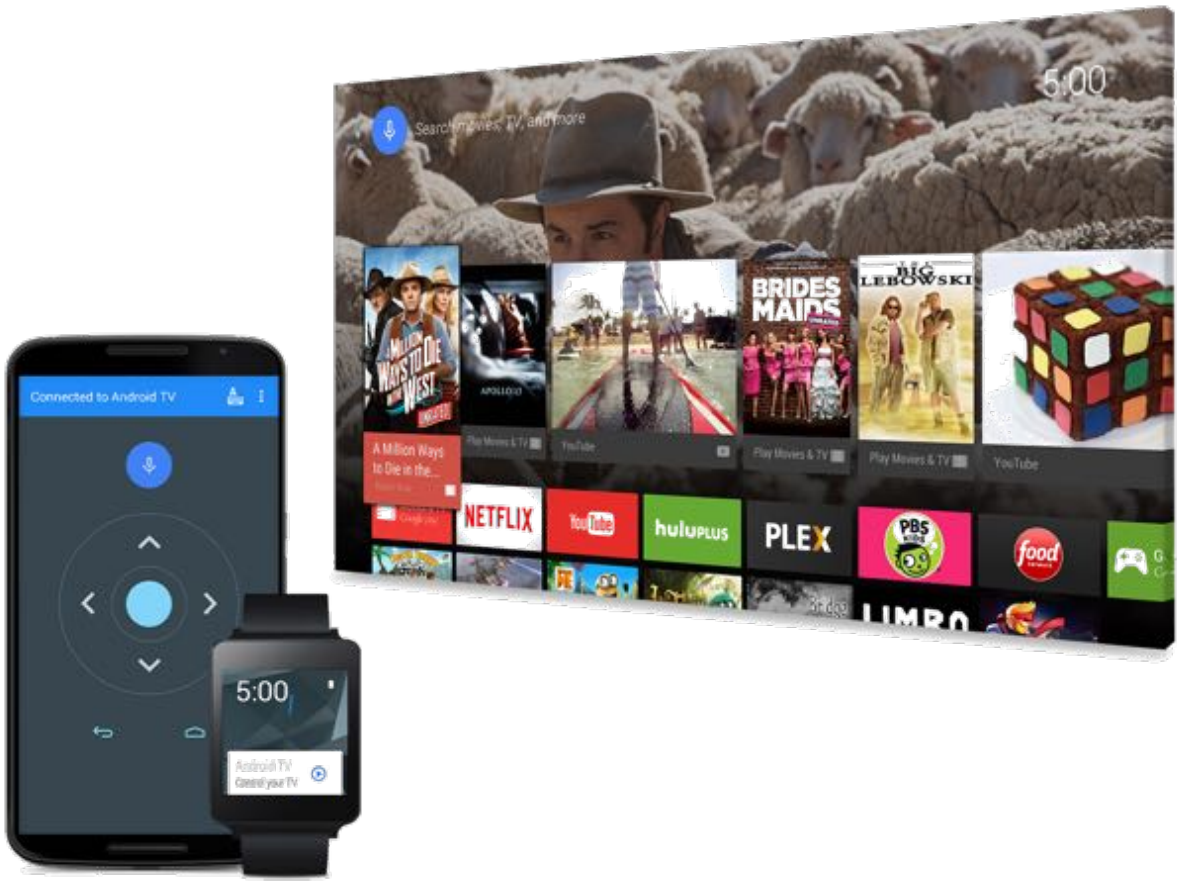
- 2.000+ Channels
- Included remote
- Apps for Smartphone and tablet
- iOS/Android
- Cast directly from YouTube and Netflix Apps

Source: http://www.roku.com/

- Control Roku player from your mobile device
- searching for movies, shows, actors, or directors
- Browse, add, and rate Channels
- Enter text using you device's keyboard
- Stream media from mobile device

ANDROID TV

- Personalized content recommendations from Google Play, YouTube
- Google Cast enabled. All Chromecast compatible Apps work with Android TV
- The remote app offers voice search and an on-screen keyboard
- Install new Applications from Google Play



Source: http://www.android.com/tv/

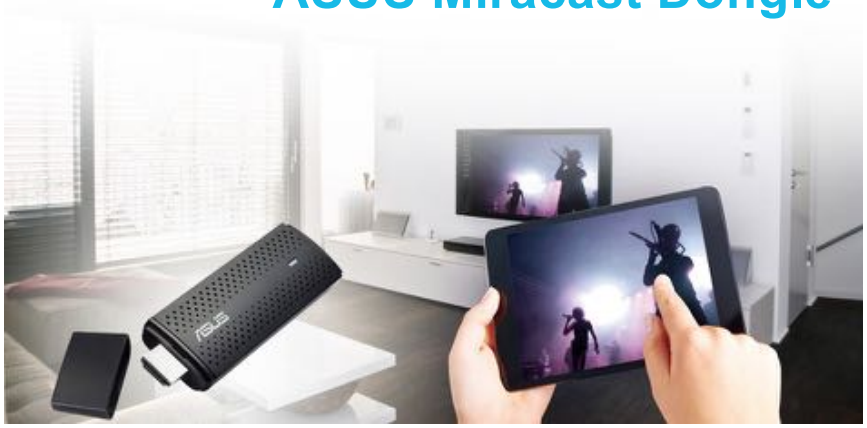
MATCHSTICK

- pocket-sized wifi-enabled streaming Stick built on Firefox OS
- Plugs into an HDMI port on TV or Monitor
- fling" content from any smartphone, tablet, or laptop to stream to a big screen TV
- Use smartphone or tablet as a remote control

Source: http://www.matchstick.tv/

MIRACAST DONGLES

ASUS Miracast Dongle



Source: http://www.asus.com/Tablet_Mobile_Accessories/ASUS_Miracast_Dongle/

- Android 4.2+ devices
- Windows 8.1 devices
- WiDi 3.5+ devices

NETGEAR Push2TV

Source: http://www.netgear.de/products/home/hometheater/media-players/PTV3000.aspx

UNIVERSAL HDMI DONGLES – E.G. EZCAST WIFI DISPLAY

- Support multiple protocols and technologies in one dongle: **Miracast, DLNA, Airplay**
- Cross Platform support: **iOS, Android, Windows, Mac OS**

Source: http://www.ezcastadapter.com/ezcast-dongle/

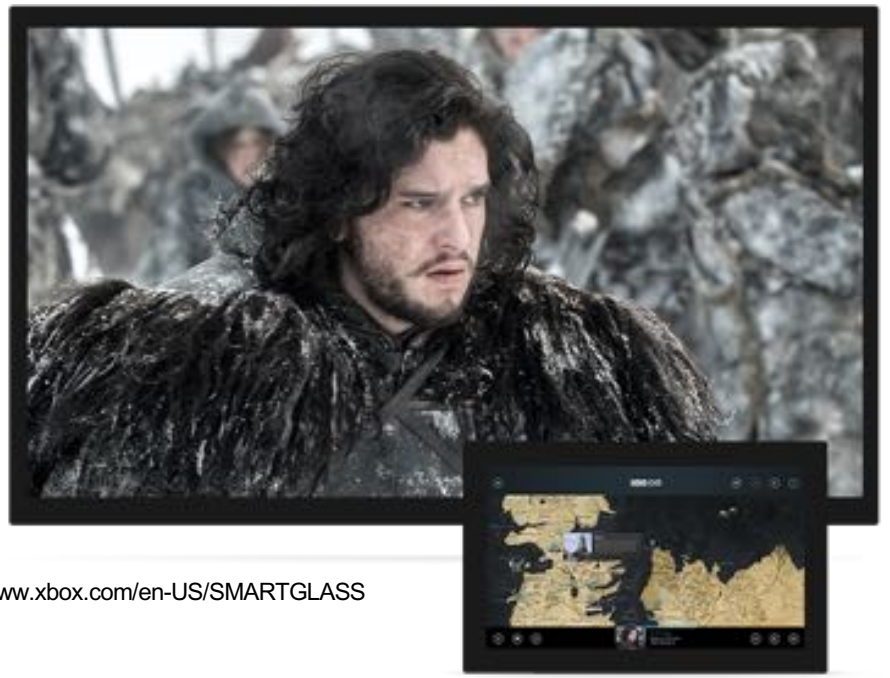
ANDROID HDMI DONGLES - E.G. DELL WYSE

- **Dell Wyse Cloud Connect:** Android OS + Dell Wyse PocketCloud software for remote access to physical or virtual desktops, apps and content
- **Zero-battery technology**: Power for Cloud Connect is supplied through the MHL interface, or separately through the integrated USB po
- **Bluetooth interface** for connecting to a wireless mouse and keyboard
- **Ultrathin client**: works with many existing Dell Wyse thin clients and software products. It can also access Citrix, Microsoft or Vmware environments for web-based apps or a standalone device for local apps and content

Source: http://www.dell.com/us/business/p/cloud-connect/pd

XBOX SMARTGLASS

- “Xbox SmartGlass turns your mobile phone or tablet into a second screen that intelligently interacts with your Xbox to elevate your entertainment experience. Xbox SmartGlass allows for your devices and TV to talk to each other to enhance your favorite TV shows, movies, music, sports and games.”
- Available for Windows, iOS and Android



Source: http://www.xbox.com/en-US/SMARTGLASS

PC and Companion Devices

KODI MEDIA CENTER (FORMALLY XBMC MEDIA CENTER)

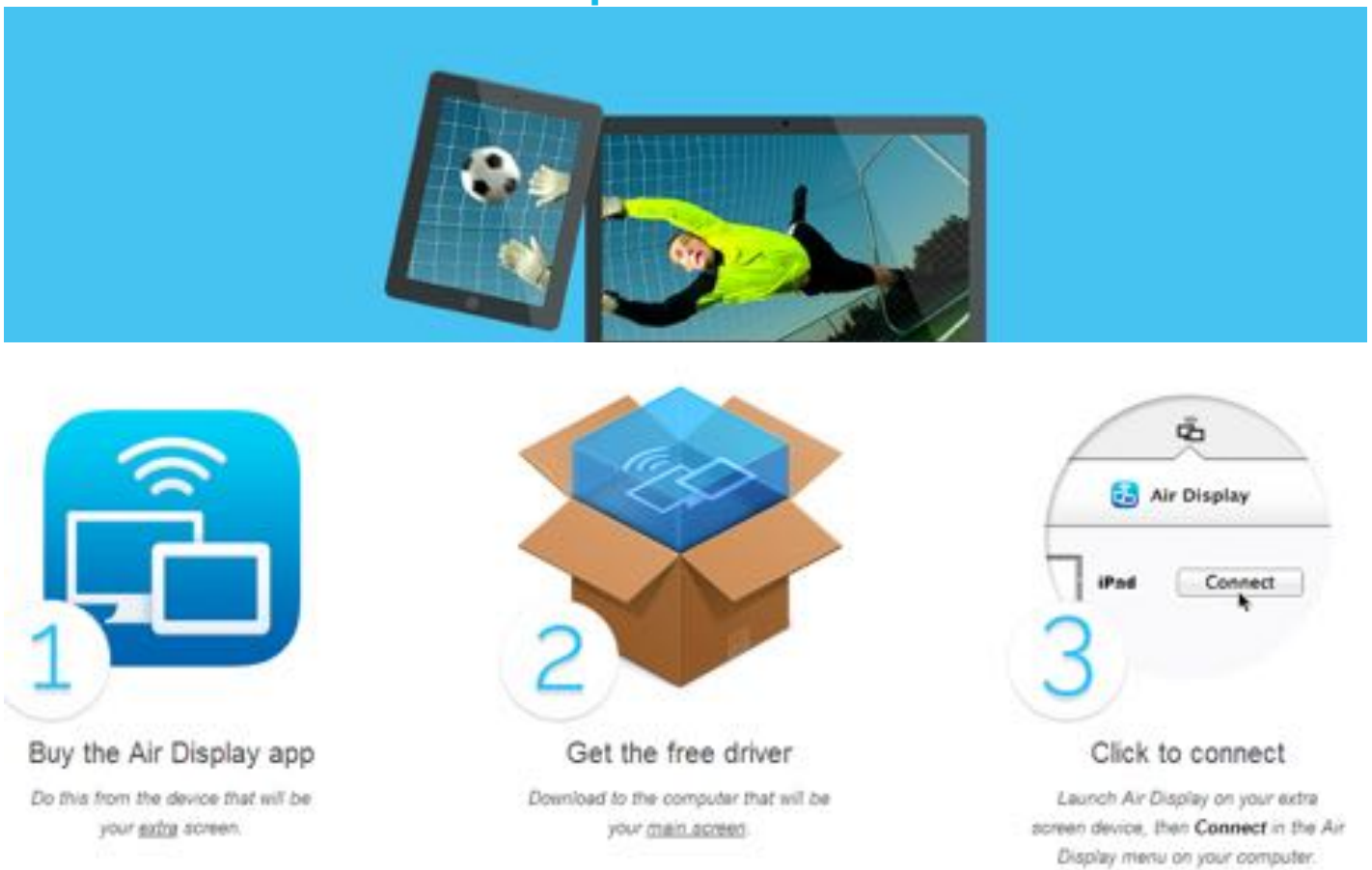
- Kodi™ (formerly known as XBMC™) is an award-winning free and open source (GPL) software media center for playing videos, music, pictures, games, and more. Kodi runs on Linux, OS X, Windows, iOS, and Android, featuring a 10-foot user interface for use with televisions and remote controls
- Support for hundreds of remote controls, CEC-compatible TVs, or one of the new Smartphone and Tablet Apps
- Kodi has **several built-in UPnP A/V features**, including the ability to receive **UPnP and DLNA content** pushed to Kodi, browse UPnP and DLNA media sources, sharing an Kodi library with other UPnP and DLNA devices, and even controlling UPnP and DLNA devices
- Airplay Video Streaming supported (Mirroring not supported)

Source: http://xbmc.org/about/

AIR DISPLAY

Turns iPad into an extra screen for computer

Source: https://www.avatron.com/apps/air-display/



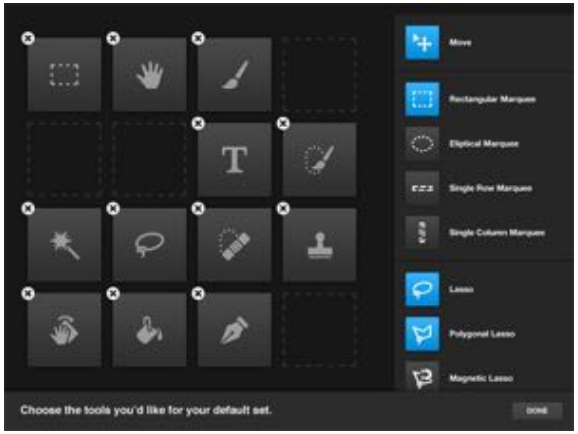
AIRSERVER

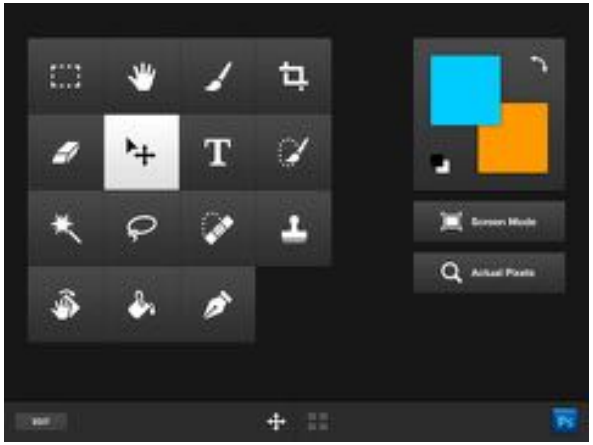
- Universal AirPlay + Miracast receiver for PC
- World's First Miracast Receiver for Windows PC
- Fully compatible with Windows 8.1 based Tablets and PCs, Miracast enabled Android devices, and iOS
- Record whatever is happening on iPad or iPhone

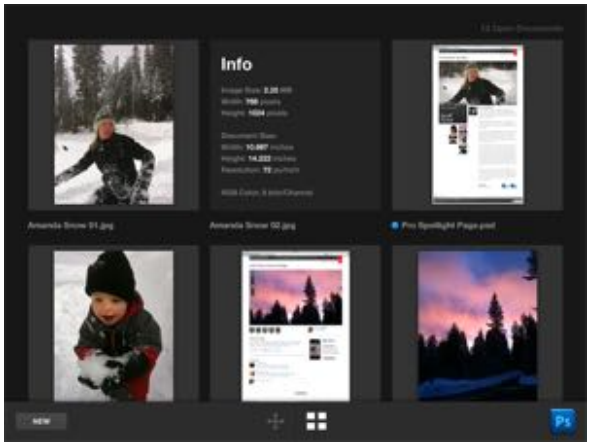
Source: http://www.airserver.com/

ADOBE NAV FOR PHOTOSHOP

- A companion app for Adobe Photoshop software running on PC
- Interact with Photoshop from iPad
- Transfer images instantly to Adobe Photoshop, Intuitive access to your Photoshop files from iPad
- browse open Photoshop documents, and activate tools
- Tool selections from your iPad, Toolbar customization







Source: http://www.adobe.com/africa/products/nav.html

companion devices

PADRACER

- The first "phone to pad" arcade game.
- iPhone becomes a steering wheel, iPad becomes a racetrack.
- If you have two iPads on hand, try the amazing DUAL-iPad mode. A double-size track extends over two iPads.

Source: https://itunes.apple.com/en/app/padracer/id369603739

CHOPPER 2

“If you have both an iPad or Retina display, you can use the fantastic remote control feature to wirelessly control the iPad/Retina display device with an iPhone/iPod touch over Bluetooth or Wi-Fi. Then plug the iPad/Retina display device into a TV or display through the component cable or VGA adapter and play Chopper 2 on your TV from your couch!”

Wearable Technologies

Smart Watch

Smart Glass

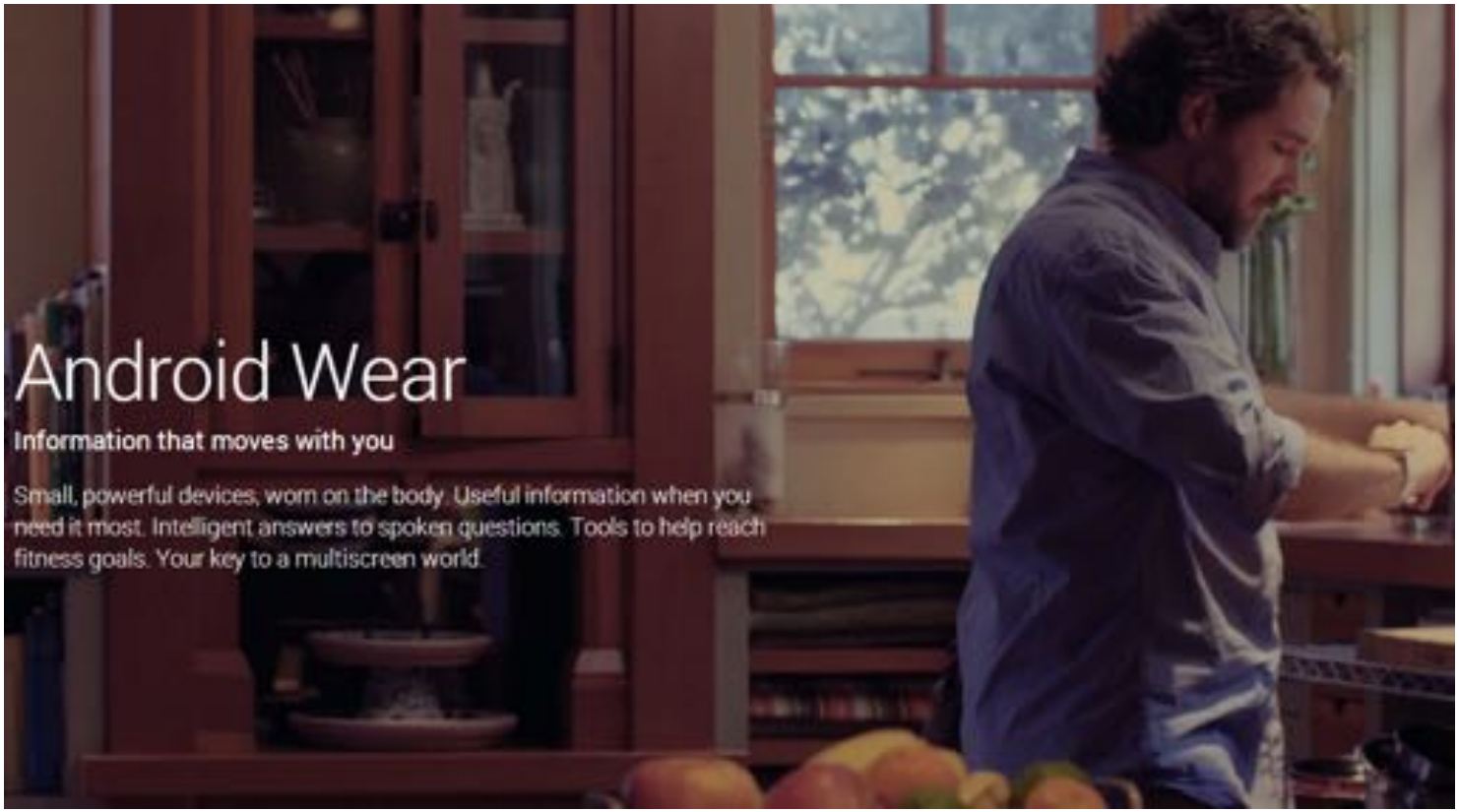
APPLE WATCH

- 3 types of Extensions/Apps
  - **WatchKit Apps**: contain a Full user interface designed for Apple Watch
  - **Glances**: provide users with timely read-only information
  - **Actionable Notifications**: For Notifications and simple actions e.g. Turn Light ON

- WatchKit Apps have two parts: A WatchKit extension that runs on iPhone in the background and set of user interface resources that are installed on Apple Watch.

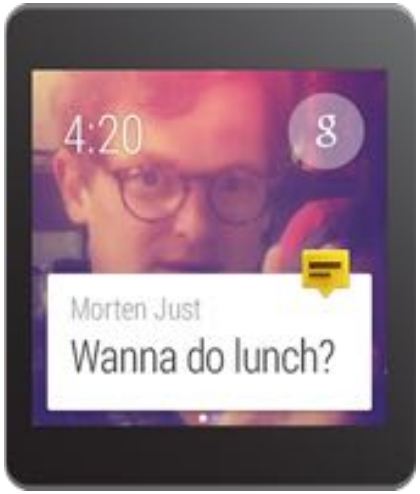
Source: https://developer.apple.com/watchkit/

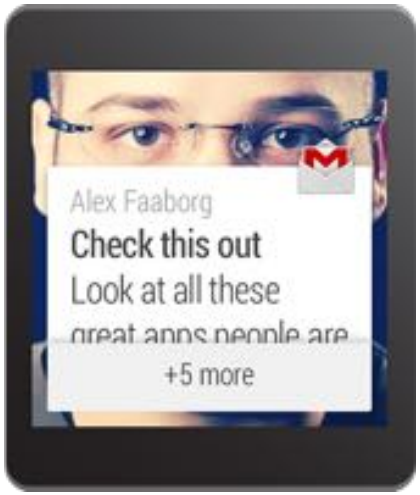
ANDROID WEAR

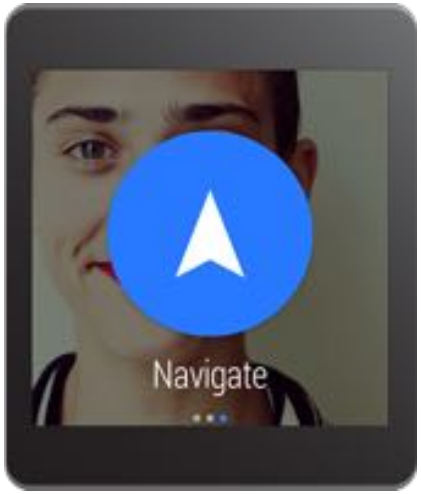


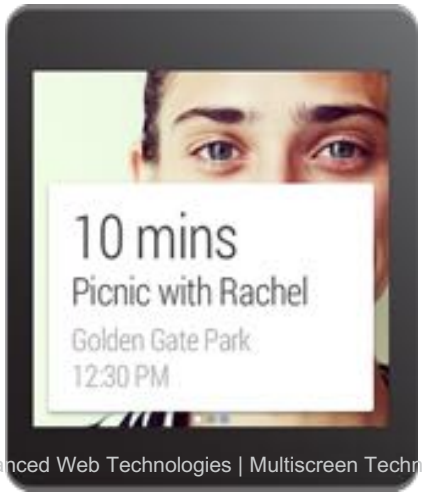
Source: http://developer.android.com/wear/index.html

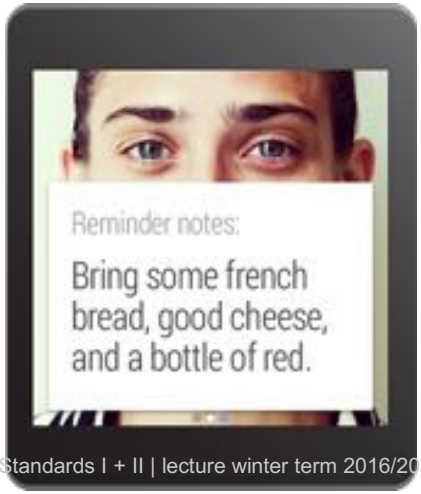
ANDROID WEAR











Source: http://developer.android.com/wear/index.html

LG WEARABLES

- Current LG Smartwatches are running Android Wear
- LG has officially announced the Watch Urbane and the Watch Urbane LTE: The Urbane LTE Smatwatch will run WebOS instead of Android Wear
- The LG Urbane LTE is a fully-featured phone with NFC and a bigger battery
- The Urbane LTE version can support calling, messaging, and richer application
- It can also make calls, albeit via a Bluetooth headset or the watch's tinny speaker

LG Urbane

Sources: http://www.lg.com/us/smartwatch/urbane
http://techcrunch.com/2015/03/01/webos-shambles-back-from-the-grave-in-lgs-latest-smartwatch/

SAMSUNG WEARABLE TECH

Evernote for Galaxy Gear
Add photo and voice notes directly to Evernote notebook

ARGUS Mini Fitness Tracker
Health & Fitness App for count steps, heart rate

CNN for Galaxy Gear
Latest news stories from CNN

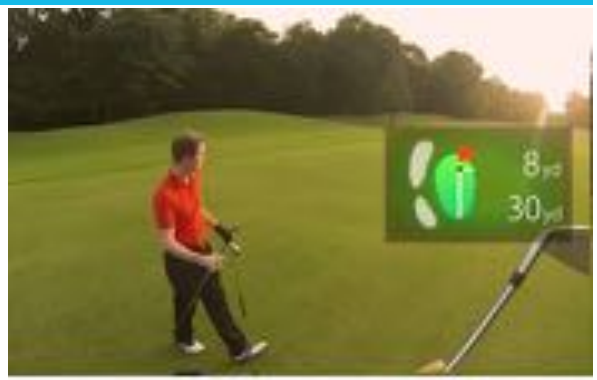
Source: http://developer.samsung.com/gear

WHAT CAN I DO WITH A SMART WATCH?

- ✓ **Notifications**
  - Mails/SMS
  - Breaking news/ Sport Score
  - Weather
- ✓ **Media Control**
  - Photo/Video Snapshots
  - Music playback
- ✓ **Telephony**
  - Accept/Reject Calls
- ✓ **Recommendation**
  - POIs nearby
- ✓ **Simple/Smart Interactions**
  - Remote Camera Shutter
  - Check in a location with one click
  - Voice input
- ✓ **Navigation**
  - Turn-by-turn navigation
- ✓ **Health & Fitness**
  - Heart rate monitor
  - Pedometer

SONY SMARTEYEGLASS / SMARTEYEGLASS ATTACH

- SmartEyeGlass includes an array of features, including a gyroscope, accelerometer, ambient light sensor and built in camera
- Sony has released the essential tools (SmartEyeglass SDK) to allow developers to start coding applications for its SmartEyeGlass
- the SmartEyeGlass Attach accessory was unveiled at CES 2015 and will work with any regular specs
- SmartEyeGlass communicates with a smartphone over Bluetooth and WiFi
- SmartEyeGlass apps run as normal Android apps on the smartphone.



Source: http://developer.sonymobile.com/products/smarteyeglass/

MICROSOFT HOLOLENS

- Microsoft HoloLens is the first fully untethered, see-through holographic computer enabled by Window 10
- It provides the ability to design and shape high-definition, three-dimensional holograms
- Using holograms, users can pin digital content, such as apps, information and multi-dimensional videos, in the physical space around them, then interact with them as they do with physical objects
- Windows 10 supports holographic computing with APIs that enable gaze, gesture, voice …

Source: https://www.microsoft.com/microsoft-hololens/en-us

In-Vehicle Infotainment

APPLE CARPLAY

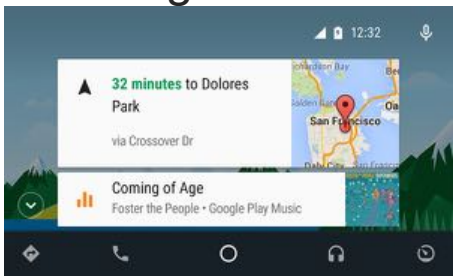
Source: http://www.apple.com/ios/carplay/

APPLE CARPLAY

Source: http://www.apple.com/ios/carplay/

ANDROID AUTO

- Android Auto brings the Android platform into the car with a user interface that's optimized for driving
- Three types of Apps
  - **Maps/Navigation** (Only Google Maps)



  - **Audio Apps**: enable content navigation and playback through a vehicle console

  - **Messaging Apps**: provide text communication services through a vehicle console

Source: https://developer.android.com/auto/index.html

TOYOTA - ENTUNE

Source: http://www.toyota.com/entune/

FORD APPLINK

- Control SYNC compatible mobile app With SYNC AppLink.
- Hands-free Phone
- SYNC Controls
- SYNC Entertainment
- SYNC Services
- SYNC Navigation
- SYNC Media Hub

Source: http://support.ford.com/topics/sync-technology-sync

Classification of Multiscreen Use Cases

CLASSIFICATION OF MULTISCREEN USE CASES

- **Media sharing and control**

  Share Media (Videos, Music, Photos, Documents) available on personal devices or remote sources on large displays and control the playback from mobile device

- **Screen Mirroring**

  Mirror the whole screen of a companion device on a large display (or more displays) to share with others. It is in general a Platform feature that works independent from running apps.

- **Screen Extension**

  Extends the small screen of a companion device on a larger display. This needs in general (on non-window based systems) incorporation with the application to decide what to display on each screen. On window based system like desktop platforms the user can freely move windows between the displays.

CLASSIFICATION OF MULTISCREEN USE CASES

- **Device shifting**

  Start activity on one device and then switch and continue on another device. E.g. start movie on TV, continue watching on tablet. Search for products on companion device, continue purchasing on PC.

- **Remote Control & Smart Interaction**

  Use companion device as controller for TV or TV App using different interaction capabilities like Touch, Gestures, Text Input, …

- **Synchronized Content**

  Keep content on different devices in sync. E.g. get additional information on companion device to current content displayed on TV. Another example is synchronized media playback of multi-view streams (different camera perspectives) on different devices

- **Social TV**

  Share TV watching experience (related to scene in a TV show, actor, etc.) with the public or community over social media networks. Most common features are like, comment, share

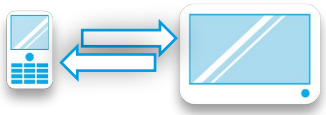
What we need to do “multiscreening”?
Common features and Challenges

TYPICAL FUNCTIONAL BLOCKS

Discovery & screen linkage

Communication & synchronization

Content adaptation & media sharing



COMMON CHALLENGES

1. **How to find screens?**

2. **How to pair screens?**

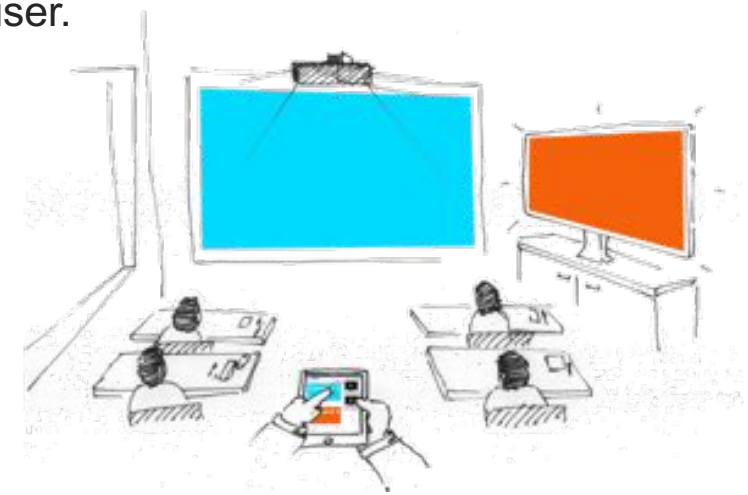
3. **How to use screens?**
   - How to share content?
   - How to launch app?
   - How to communicate?

4. **How to keep content on screens in sync?**

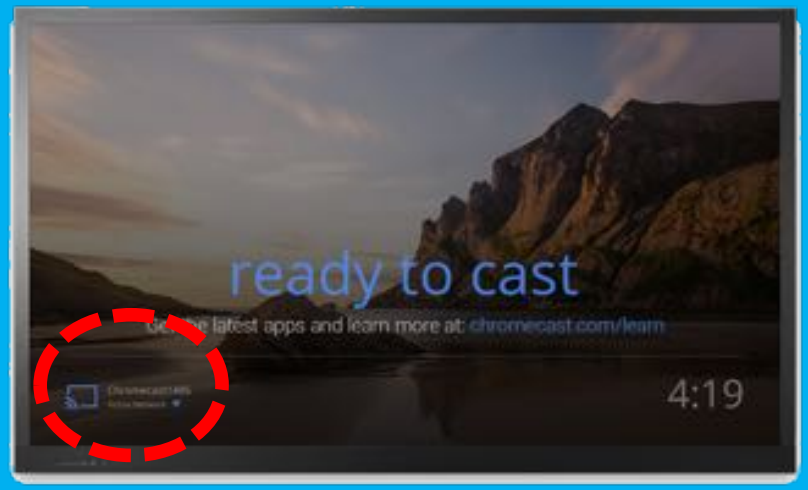
5. **How to adapt to different screens?**

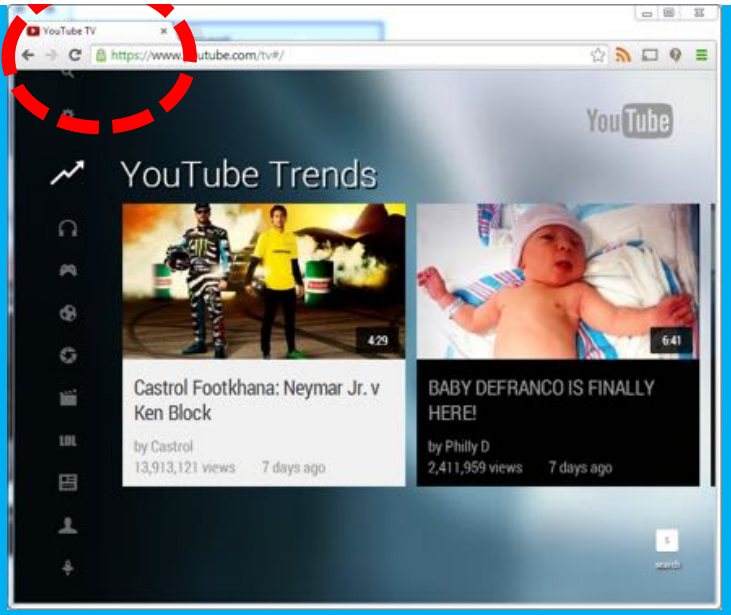
HOW TO FIND SCREENS?

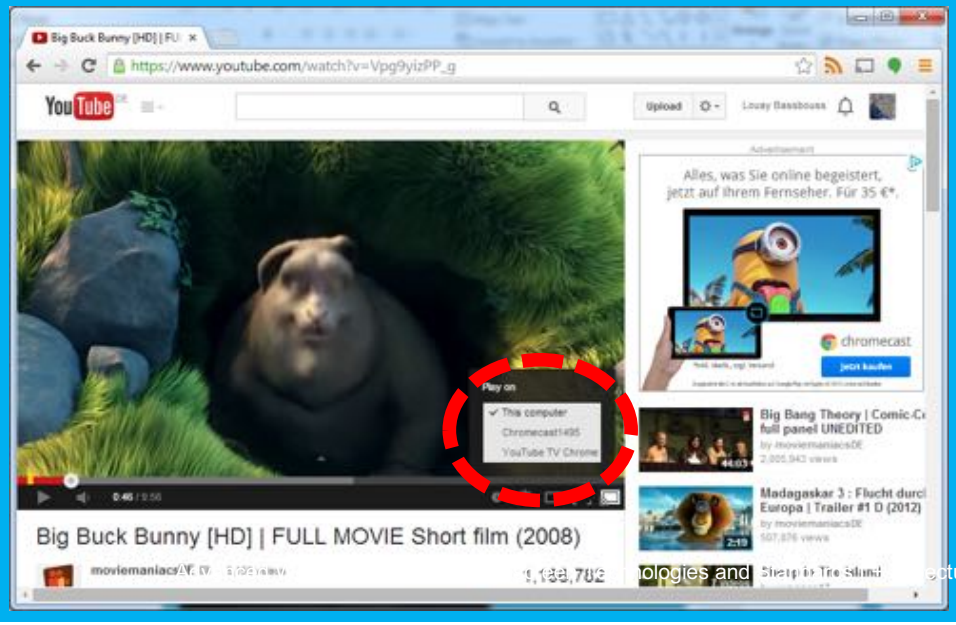
- This is this first and most important step for developing multiscreen applications → **Discovery**
- Objective of this challenge is to find one or more screens that fulfill some criteria or offer a specific service such as Video Player, Photo Viewer, etc.
- There are two basic concepts for discovering screens (or devices in general)
  - **Local Discovery**: discover devices/services nearby or in same local network (Network Service Discovery). Devices need to support human readable names.
  - **Remote Discovery**: Search devices that are registered on a central Server (Registry) and fulfil a specific criteria e.g. get all device belong to the same user.
  - Mix of Local and Remote Discovery is possible
    → YouTube
- In both cases the requesting device gets a list of available screens including names and how to connect to each of them.

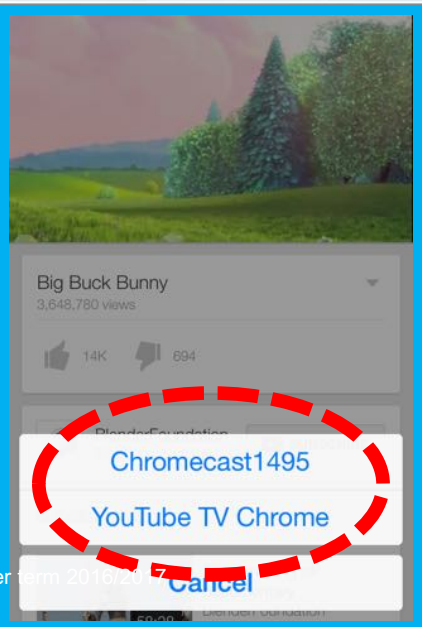


Demo


ready to cast
4:19

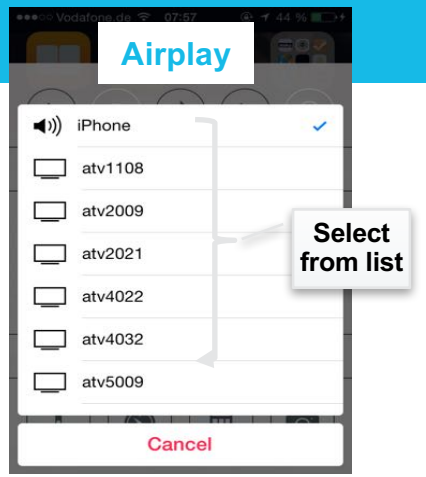

YouTube Trends
Castrol Footkhana: Neymar Jr. v Ken Block
by Castrol
13,913,121 views
7 days ago
BABY DEFRANCO IS FINALLY HERE!
by Philly D
2,411,959 views
7 days ago

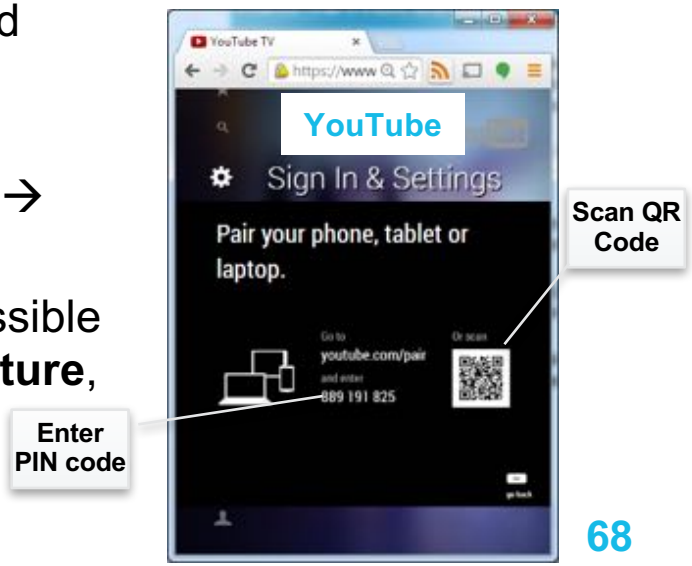

Big Buck Bunny [HD] | FULL MOVIE Short film (2008)


Big Buck Bunny
Chromecast1495
YouTube TV Chrome
Cancel

HOW TO PAIR SCREENS?

- After the discovery process is done, the user gets a list of available screens including human readable names and connection info and other additional (optional) meta data (Location, Capabilities, etc.)
- User selects one screen from the list → User device and selected screen will be coupled with each other → **Pairing**
- There are different forms of pairing:
  - **Pairing on platform Level**: The pairing/connection is shared with all application running on top of the platform → **Airplay**
  - **Pairing on Application Level**: The pairing/connection is available for a specific application (or group of applications) → **YouTube App**
- Discovery is not always required for pairing devices. This is possible using technologies like **QR-code**, **Audio-code**, **PIN-code**, **Gesture**, etc.





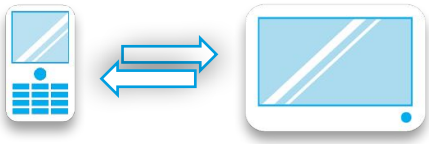
YouTube pairing Demo

HOW TO USE SCREENS?

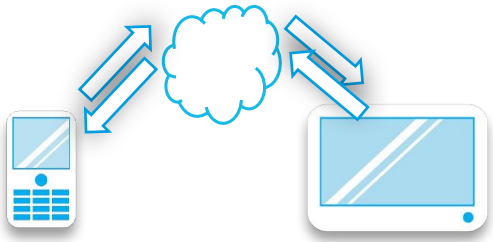
- After pairing process is done, the screens are ready for use → Different usage options:
  - **Use existing/common services/applications**: most common usage for this is **Content or Screen Sharing**. The App provider needs only to pass the content to the paired device or specific service running on that device. Video → Video Player, Music → Audio Player, Photo → Photo Viewer, etc. (technologies: DLNA, Airplay)
  - **Use specific services/applications**: In many situations, App providers need to distribute the logic of the application on the different screens. Instead of starting the application (different components) manually on each screen and then link the application components to each other, the **Remote App Launch** feature will help to do it in one step (technologies: DIAL, Chromecast).
    - A Launcher service need to be available on the paired screen. This service offers a function to Launch a specific native or Web application.
    - It should be possible to pass app specific parameters to the application during Launch. E.g.: Launch YouTube App and Play “big buck bunny” video
    - In some situations a **Wacke-up** of Application and **Notification** of user is needed before Launch. This is relevant for personal devices more than TVs for better user experience (technologies: iBeacon)

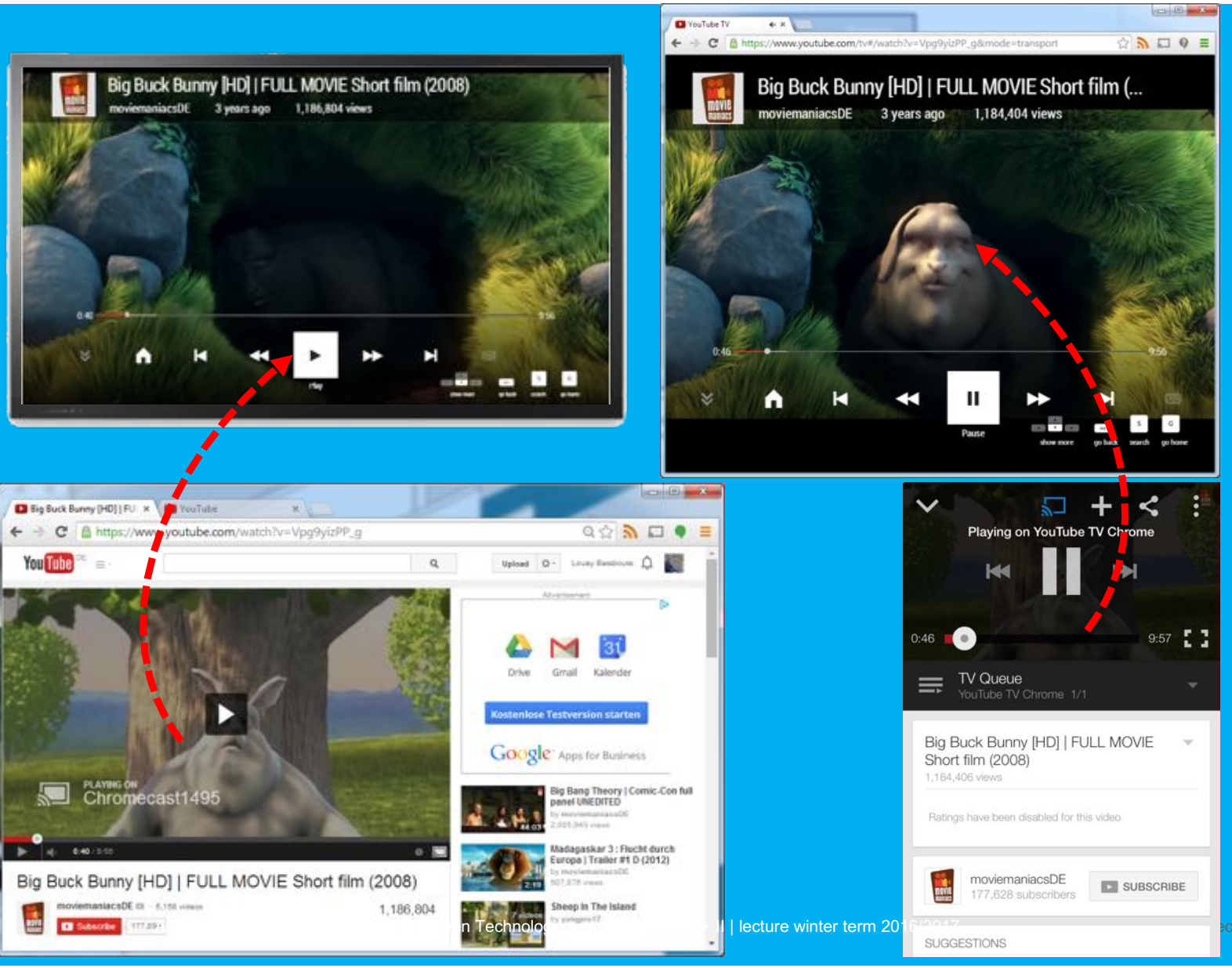
HOW TO USE SCREENS?

- In both usage options described on previous slide, a coordination between the components running on different screens is needed. The **App2App Communication** feature offers the required functionalities.
- For Content Sharing case, the communication is needed to control the media playback on the remote screen e.g. Play, Pause, Seek, etc. or for monitoring the playback state e.g. current time, etc.
- For App Launch case, the communication is needed for exchanging app
- Different Communication mechanisms:
  - **Direct Communication** (Peer2Peer): A direct communication channel between the entities will be established. This is relevant for devices in same network.



  - **Indirect Communication** (Over a Proxy or third entity): This is relevant if a direct communication is not possible. Firewall, different networks, etc.

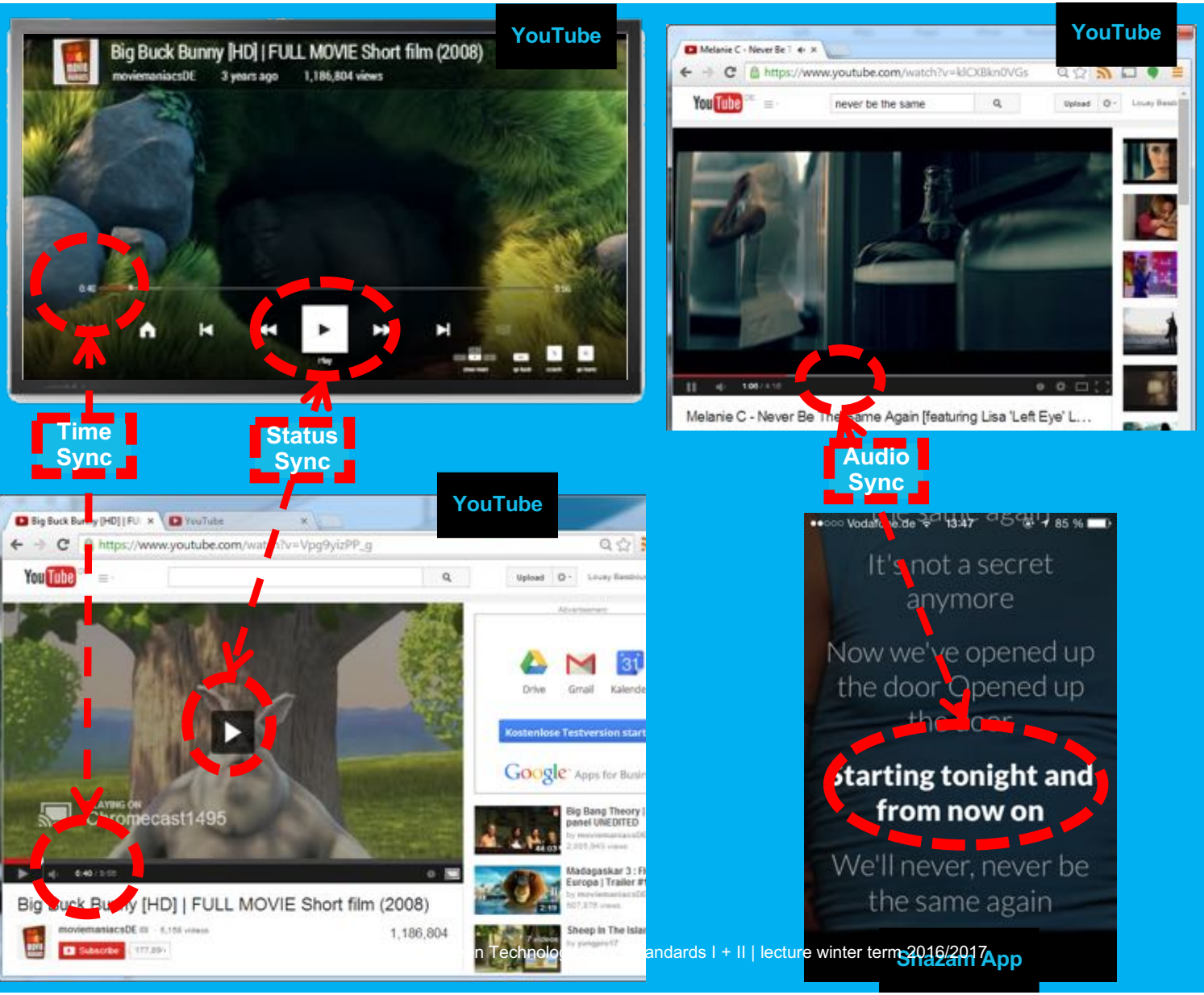



Big Buck Bunny [HD] | FULL MOVIE Short film (2008)
moviemaniacsDE 3 years ago 1,186,804 views
Big Buck Bunny [HD] | FULL MOVIE Short film (...
moviemaniacsDE 3 years ago 1,184,404 views
Pause
Playing on YouTube TV Chrome
0:46
9:57
TV Queue
YouTube TV Chrome 1/1
Big Buck Bunny [HD] | FULL MOVIE Short film (2008)
1,184,406 views
Ratings have been disabled for this video
moviemaniacsDE
177,628 subscribers
SUBSCRIBE
SUGGESTIONS
Kostenlose Testversion starten
Google Apps for Business
Drive
Gmail
Kalender
PLAYING ON
Chromecast1495
Big Buck Bunny [HD] | FULL MOVIE Short film (2008)
1,186,804
Big Bang Theory | Comic-Con full panel UNEDITED
Madagaskar 3 : Flucht durch Europa | Trailer #1 D (2012)
Sheep In The Island

HOW TO KEEP SCREENS IN SYNC?

- There are different ways to do **Synchronization**:
  - **In-App Synchronization**: Uses the channels established for App2App Communication to synchronize content on the different screens
  - **Cross-App Synchronization**: The application uses other channels than for App2App Communication to synchronize content on different screens. E.g. Audio Watermarking/Fingerprinting technologies
- … and different objectives to do synchronization:
  - **Content Synchronization**: Keep Content on different screens always in sync. Updates will be immediately reflected to all connected devices (e.g. Dropbox, Google Drive, etc.)
  - **State Synchronization**: The state of the application distributed on different screens will be kept in sync during the runtime of the application (E.g. state of a multiplayer game)
  - **Timeline Synchronization**: keep the playback of different streams or time-based media in sync. This is relevant for playback of multi-stream media.

Demo

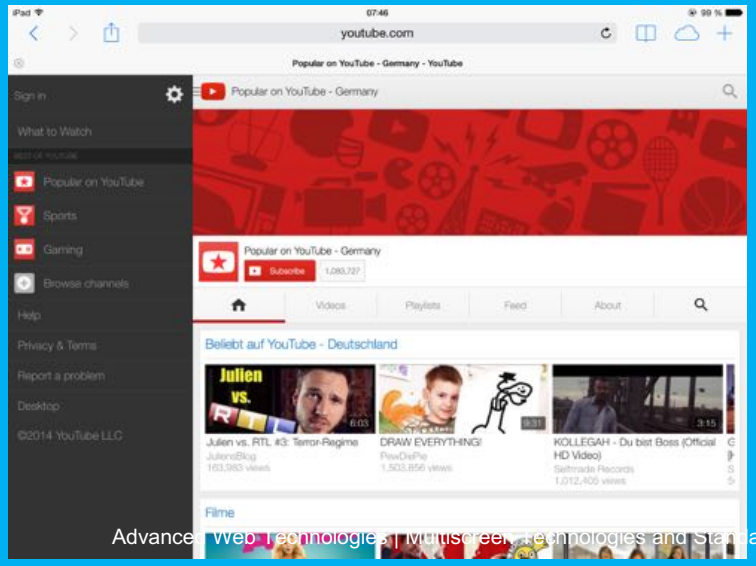

YouTube
Big Buck Bunny [HD] | FULL MOVIE Short film (2008)
moviemaniacsDE
3 years ago
1,186,804 views
Time Sync
Status Sync
YouTube
Big Buck Bunny [HD] | FULL MOVIE Short film (2008)
1,186,804
YouTube
Melanie C - Never Be The Same Again [featuring Lisa 'Left Eye' L...
Audio Sync
It's not a secret anymore
Now we've opened up the door
Opened up the door
Starting tonight and from now on
We'll never, never be the same again
Shazam App

HOW TO ADAPT TO DIFFERENT SCREENS?

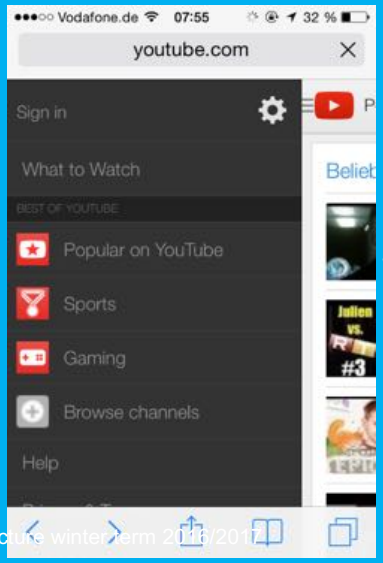
- Multiscreen applications run on devices with **different screen sizes/resolutions and input capabilities**.
  - Small screens on mobile devices, medium screens on PC/Laptop to large TV displays
  - Touch/Voice input on mobile, Mouse/Keyboard on PC/Laptop, Remote Control on TV
- **Adaptation** of applications to target screen is needed even on devices from same category and running same platform (iPhone 4, iPhone5)
- Different options for adaptation:
  - **Implement different apps for the different screens**: time and costs intensive
  - **Implement only UI for each screen type**: This needs to separate the UI from the Logic of the application and to define interfaces between them (MVC)
  - **Implement one App for all screens (responsive design)**: Adapt to target screen at runtime. Adaptation Technologies may help (CSS Media Queries, Twitter Bootstrap, iOS Universal Apps, …)

Chrome
on Desktop

Safari on
iPad mini



Safari on
iPhone 4



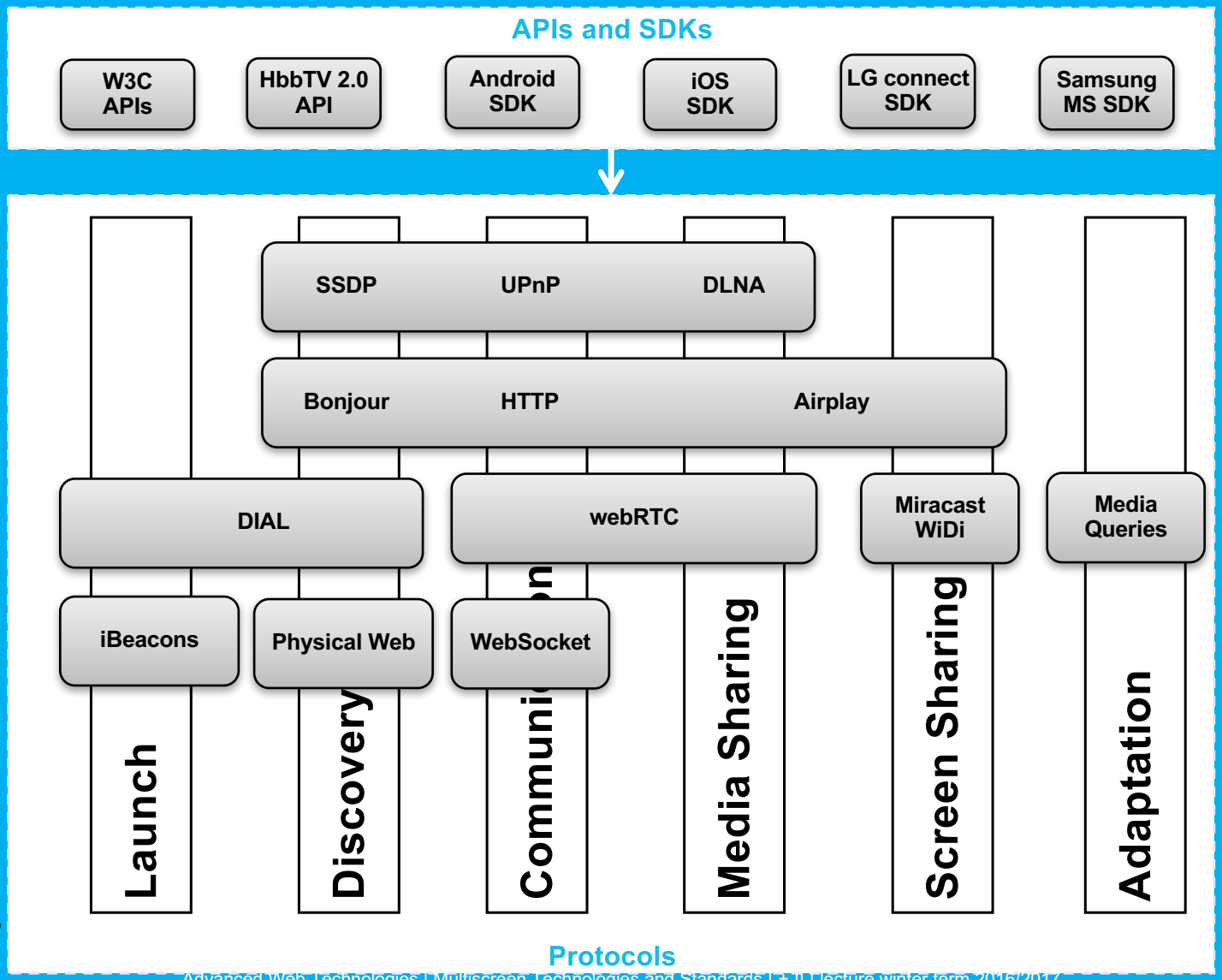
MULTISCREEN TECHNOLOGIES AND STANDARDS

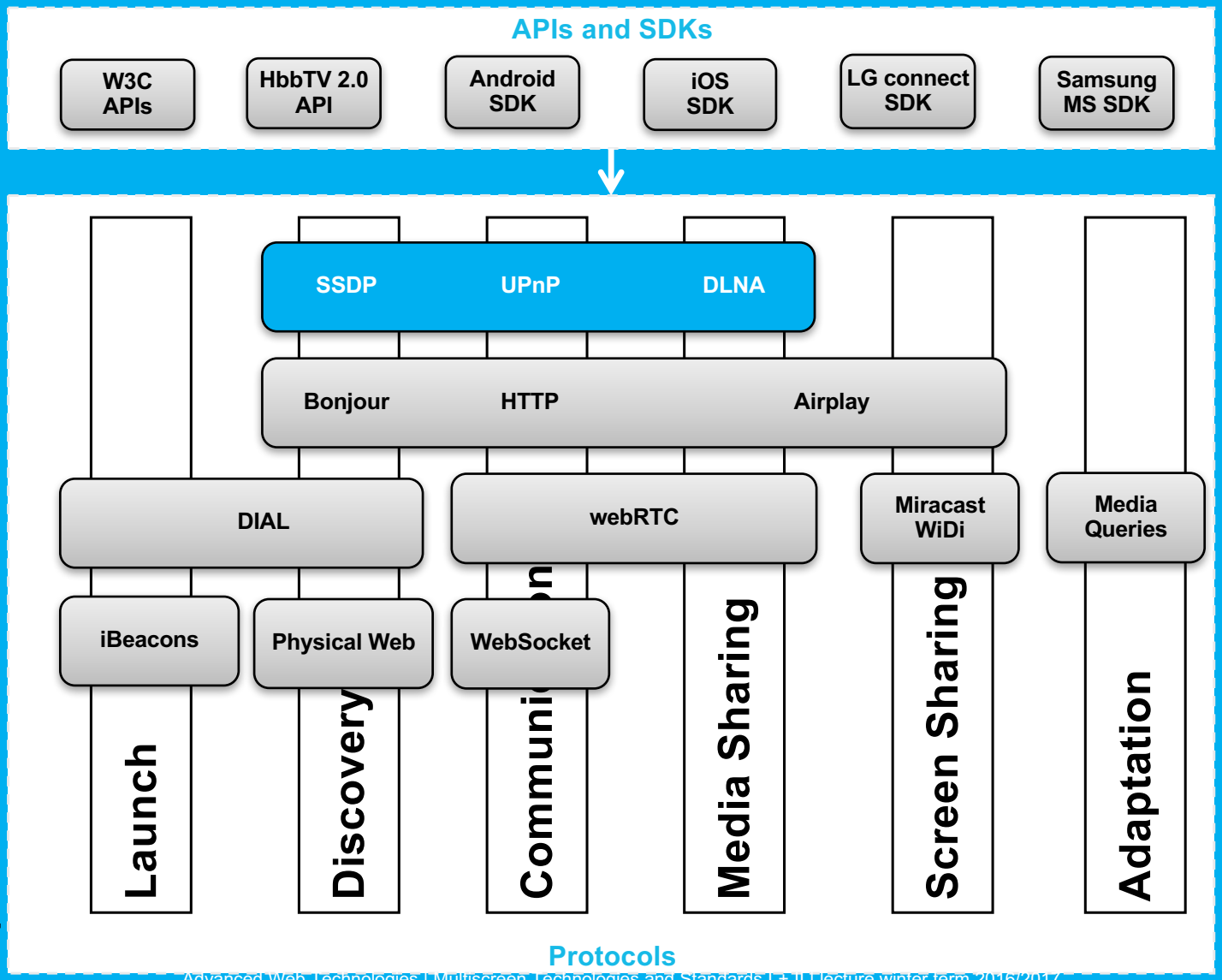
December 8, 2016

Part I	**Intro & Setting the scene** - Terminology - Multiscreen in Action – practical examples - Use Cases, Features and Challenges
Part II	**State of the Art Technologies and Standards** - W3C APIs, HbbTV, Smart TV SDKs, Android/IOS SDKs - UPnP / DLNA, Bonjour/AirPlay, DIAL, Miracast / WiDi, iBeacon, Physical Web

FEATURE-TECHNOLOGY MATRIX

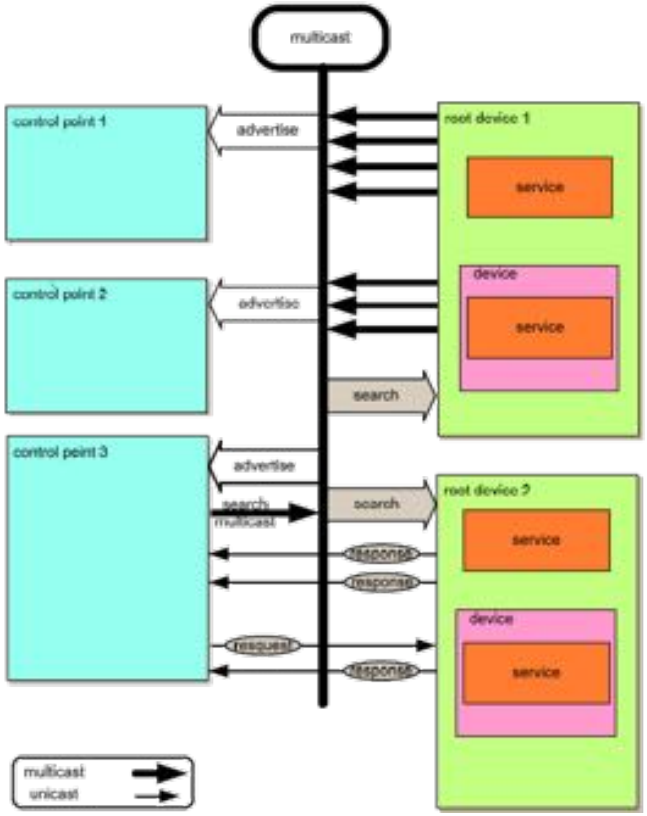
	SSDP/ UPnP/ DLNA	mDNS/ Bonjour	Miracast/ WiDi	Airplay/ AirDrop	DIAL	Audio Watermarks	QR, Pin, Audio code	WebSockets / WebRTC	W3C Presentation API	W3C Media Queries	BLE/ iBeacon	BLE/ Physical Web	HbbTV 2.0 CS
Discovery	+	+			+						+	+	+
Pairing				+			+						
Wake-up & Notification											+	+	
Launch					+								+
Communication	+							+			(+)	(+)	+
Content Sharing	+			+									
Screen Sharing			+	+					+				
Synchronization	+					+					+	+	
Adaptation										+			


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

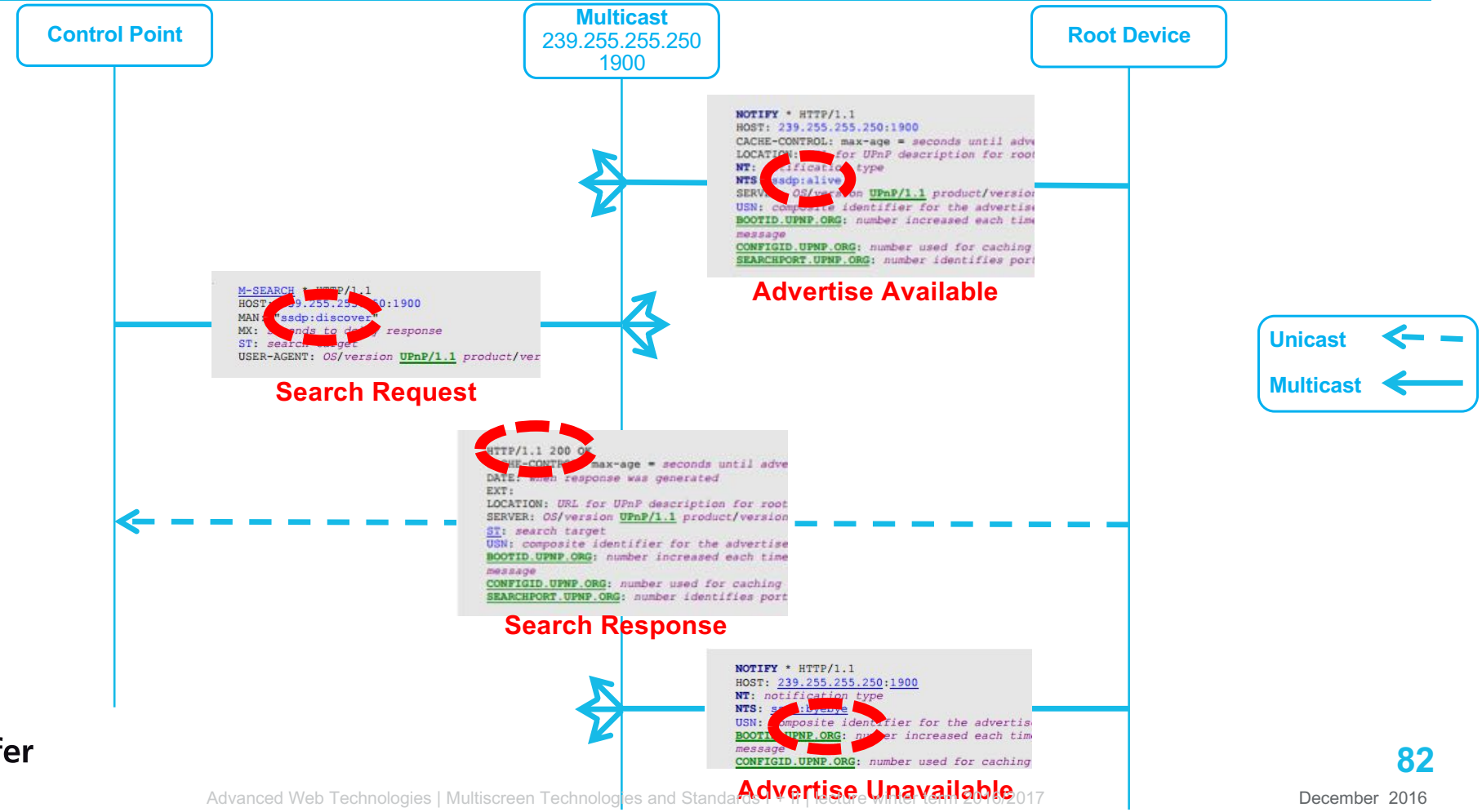
SIMPLE SERVICE DISCOVERY PROTOCOL SSDP

- **SSDP** is the **discovery** protocol for **UPnP**
- Allows devices to **advertise** their services to control points or control points to **search** for devices of interest on the network
- The fundamental exchange in both cases is a **discovery message** containing a few, essential specifics about the device or one of its services
- SSDP Message formats (**SSDP Start-line**)
  - **NOTIFY * HTTP/1.1\r\n** → **Advertisement**
  - **M-SEARCH * HTTP/1.1\r\n** → **Search**
  - **HTTP/1.1 200 OK\r\n** → **Response**
- Multicast Address and Port
  - **239.255.255.250:1900**



Source: http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

SSDP MESSAGE FLOW EXAMPLE



Source: http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

UNIVERSAL PLUG AND PLAY PROTOCOL UPNP

– The UPnP Device Architecture specifies **6 Steps** for UPnP Networking

Each involved device should obtain an IP Address → **(0) Addressing**

UPnP Discovery is SSDP → **(1) Discovery**

XML Description of Root devices and Services → **(2) Description**

UPnP Device Control is based on SOAP → **(3) Control**

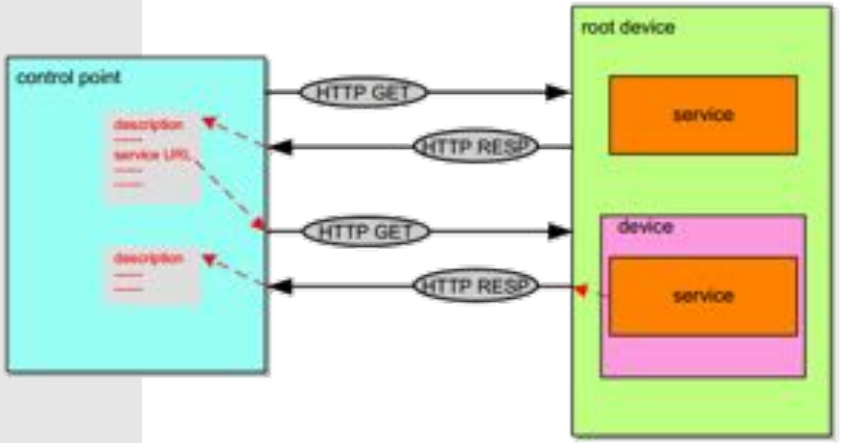
Publish/Subscribe based, Unicast and Multicast Eventing → **(4) Eventing**

Allow Root devices to provide URL for presentation in Browser → **(5) Presentation**

Source: http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

UPNP DEVICE DESCRIPTION EXAMPLE

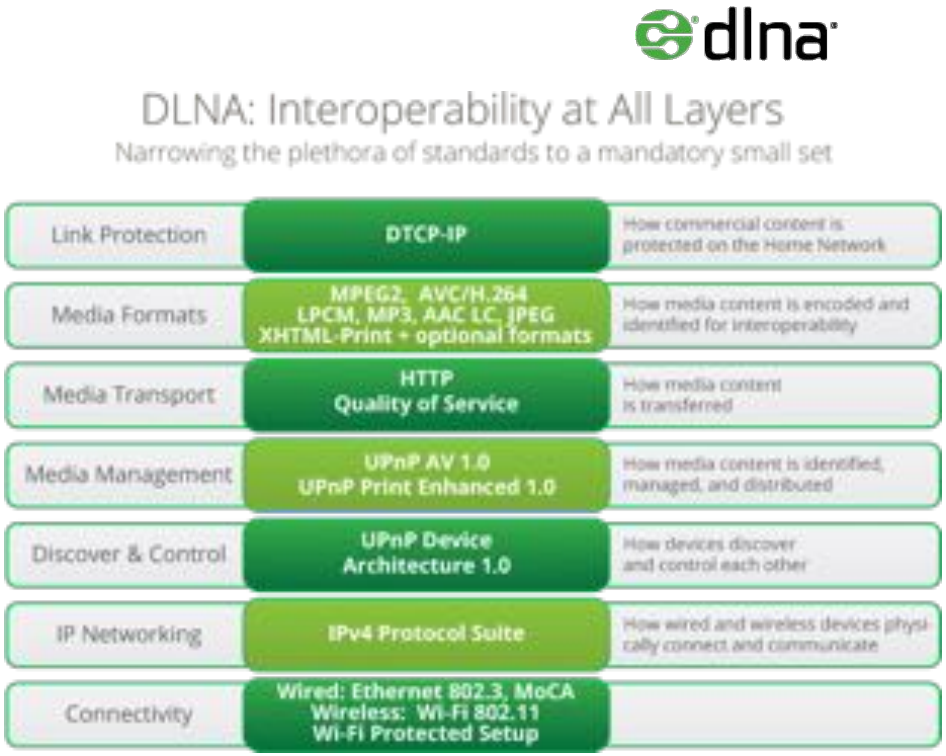
```
<?xml version="1.0"?>
<root xmlns="urn:schemas-upnp-org:device-1-0"
 configId="configuration number">
    <specVersion>
        <major>1</major>
        <minor>1</minor>
    </specVersion>
    <device>
        <deviceType>urn:schemas-upnp-org:device:deviceType:v</deviceType>
        <friendlyName>short user-friendly title</friendlyName>
        <manufacturer>manufacturer name</manufacturer>
        <manufacturerURL>URL to manufacturer site</manufacturerURL>
        <modelDescription>long user-friendly title</modelDescription>
        <modelName>model name</modelName>
        <modelNumber>model number</modelNumber>
        <modelURL>URL to model site</modelURL>
        <serialNumber>manufacturer's serial number</serialNumber>
        <UDN>uuid:UUID</UDN>
        <UPC>Universal Product Code</UPC>
        <iconList>
            <icon>
                <mimetype>image/format</mimetype>
                <width>horizontal pixels</width>
                <height>vertical pixels</height>
                <depth>color depth</depth>
                <url>URL to icon</url>
            </icon>
            <!-- XML to declare other icons, if any, go here -->
        </iconList>
        <serviceList>
            <service>
                <serviceType>urn:schemas-upnp-org:service:serviceType:v</serviceType>
                <serviceId>urn:upnp-org:serviceId:serviceID</serviceId>
                <SCPDURL>URL to service description</SCPDURL>
                <controlURL>URL for control</controlURL>
                <eventSubURL>URL for eventing</eventSubURL>
            </service>
            <!-- Declarations for other services defined by a UPnP Forum working committee
                 (if any) go here -->
            <!-- Declarations for other services added by UPnP vendor (if any) go here -->
        </serviceList>
        <deviceList>
        <!-- Description of embedded devices defined by a UPnP Forum working committee
             (if any) go here -->
        <!-- Description of embedded devices added by UPnP vendor (if any) go here -->
        </deviceList>
        <presentationURL>URL for presentation</presentationURL>
    </device>
</root>
```



Source: http://upnp.org/specs/arch/UPnP-arch-DeviceArchitecture-v1.1.pdf

DIGITAL LIVING NETWORK ALLIANCE (DLNA)

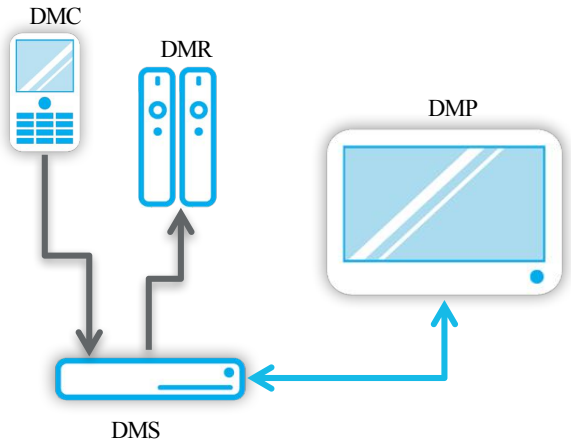
- Interoperability guidelines for sharing of digital content between devices
- Appliances, media players, consoles, computers, NAS, mobile devices, photo frames, printers, projectors, TV, STB, ...
- Over **20,000 device models** have been DLNA Certified by over **100 member companies** since the launch of the program in 2005
- **4 Billion DLNA** Certified devices



Source:http://www.dlna.org/

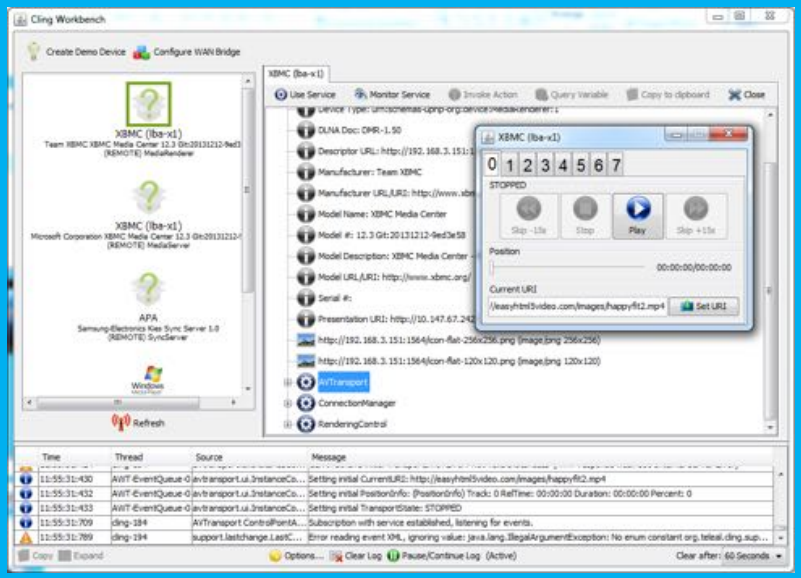
DLNA DEVICE TYPES FOR CONTENT SHARING

- **Digital Media Server (DMS)**
  - Provides digital content
- **Digital Media Player (DMP)**
  - Finds and plays digital content
- **Digital Media Renderer (DMR)**
  - Plays digital content
- **Digital Media Controller (DMC)**
  - Find digital content, controls playback
- **Digital Media Printer (DMPr)**
  - Network printing

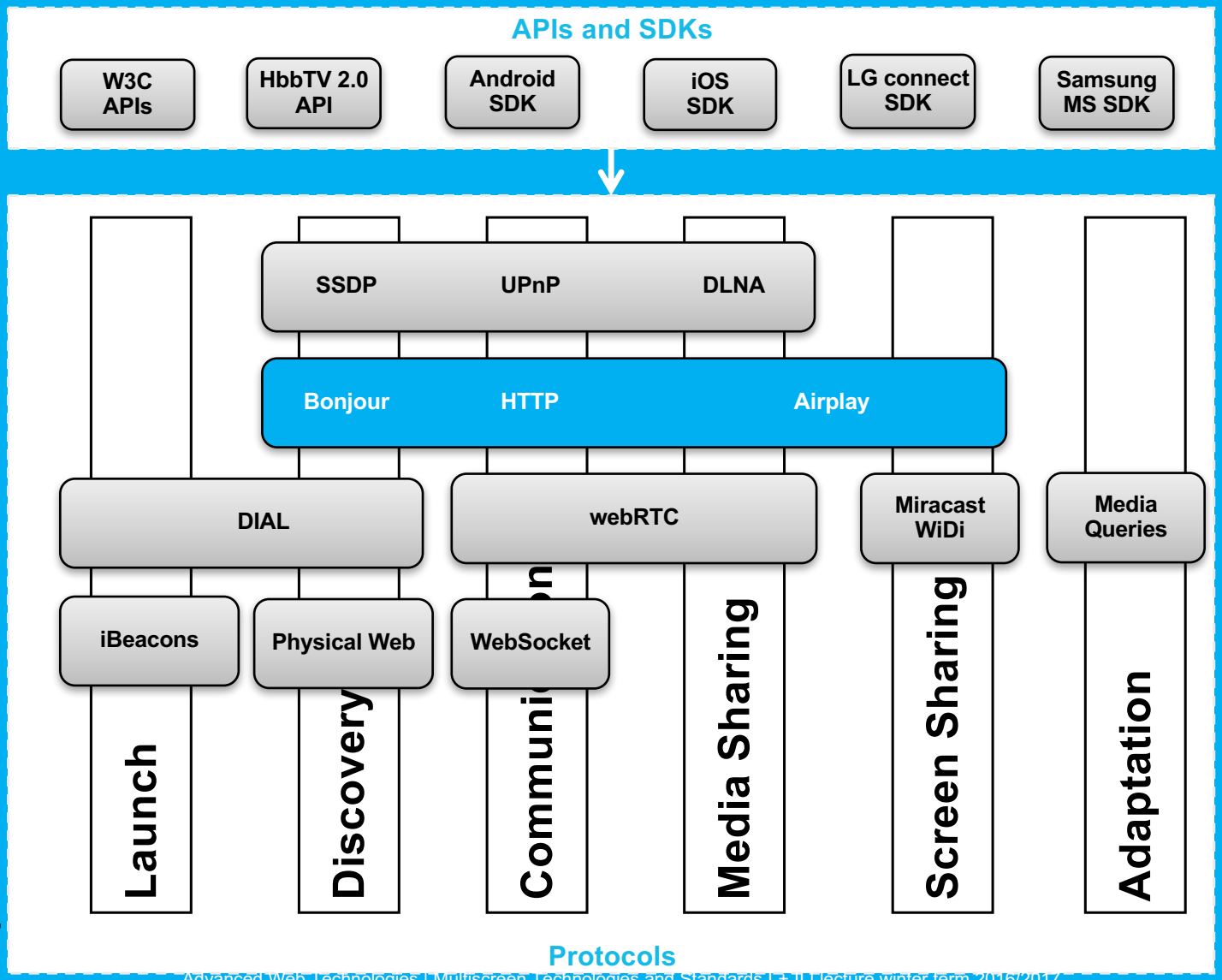


UPnP Demo

KODI UPnP Media Renderer



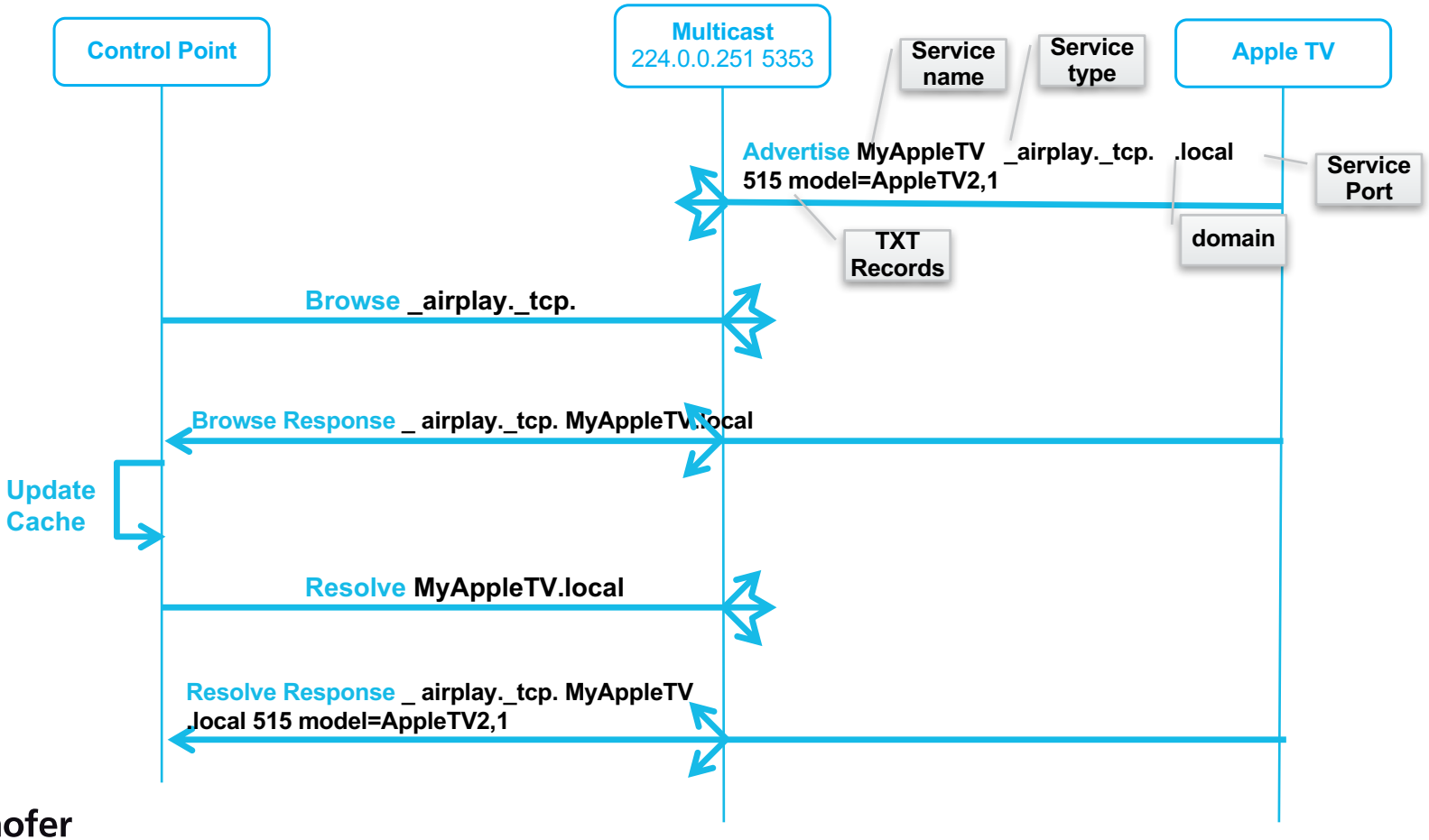
Cling UPnP Workbench


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

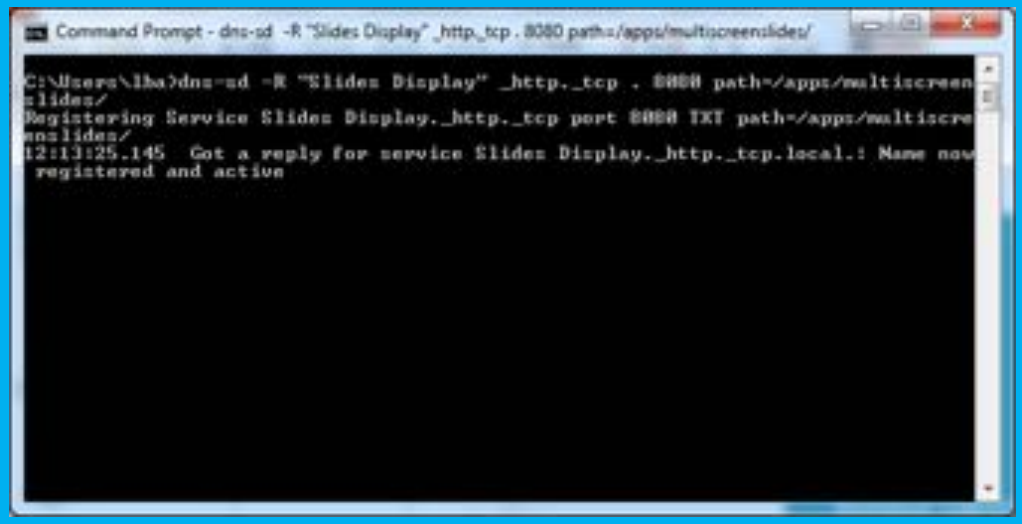
MULTICAST DNS (MDNS) - DNS SERVICE DISCOVERY (DNS-SD)

- A way of using DNS in a small local network where no conventional DNS server has been installed.
- mDNS queries are sent via **multicast** UDP to 224.0.0.251 (or FF02::FB IPv6) and port 5353
- mDNS responses are sent to all hosts in the network and update their caches
- Defines special top level domain (TLD) ".local." which is always link to local
- Multicast requests and responses are cached to reduce network traffic
- The Service Discovery is a two step process: (1) find the name of hosts providing a certain service and (2) resolve the IP Address, Port and other info for a specific host name.
- DNS-SD uses, PTR, SRV and DNS TXT records
- DNS Pointer records PTR define mapping between service type and list of service names e.g. *_airplay._tcp.local. 28800 PTR MyAppleTV._airplay._tcp.local.*
- DNS SRV records specifies protocol of service types e.g. *_airplay._tcp*
- DNS TXT records defines optional additional information as key-value pairs e.g. *model=AppleTV2,1 deviceid=58:55:….*

MDNS/DNS-SD MESSAGE FLOW EXAMPLE



Bonjour Demo
Browse Airplay and Chromecast Devices



dns-sd command line tool

AIRPLAY

- Uses Bonjour, Apple implementation of mDNS and DNS-SD, for service discovery
- AirPlay mDNS Service type is ***_airplay._tcp***

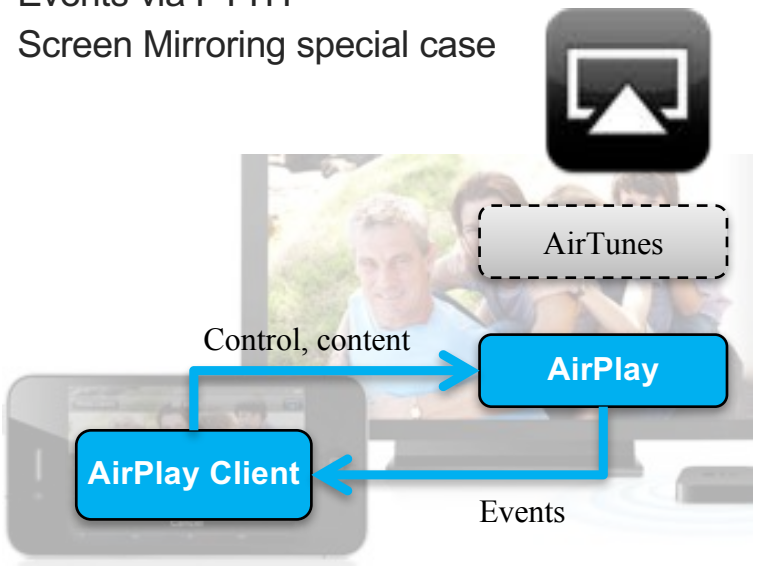
AIRPLAY SERVICE

```
name: Apple TV
type: _airplay._tcp
port: 7000
txt:
 deviceid=58:55:CA:1A:E2:88
 features=0x39f7
 model=AppleTV2,1
 srcvers=130.14
```

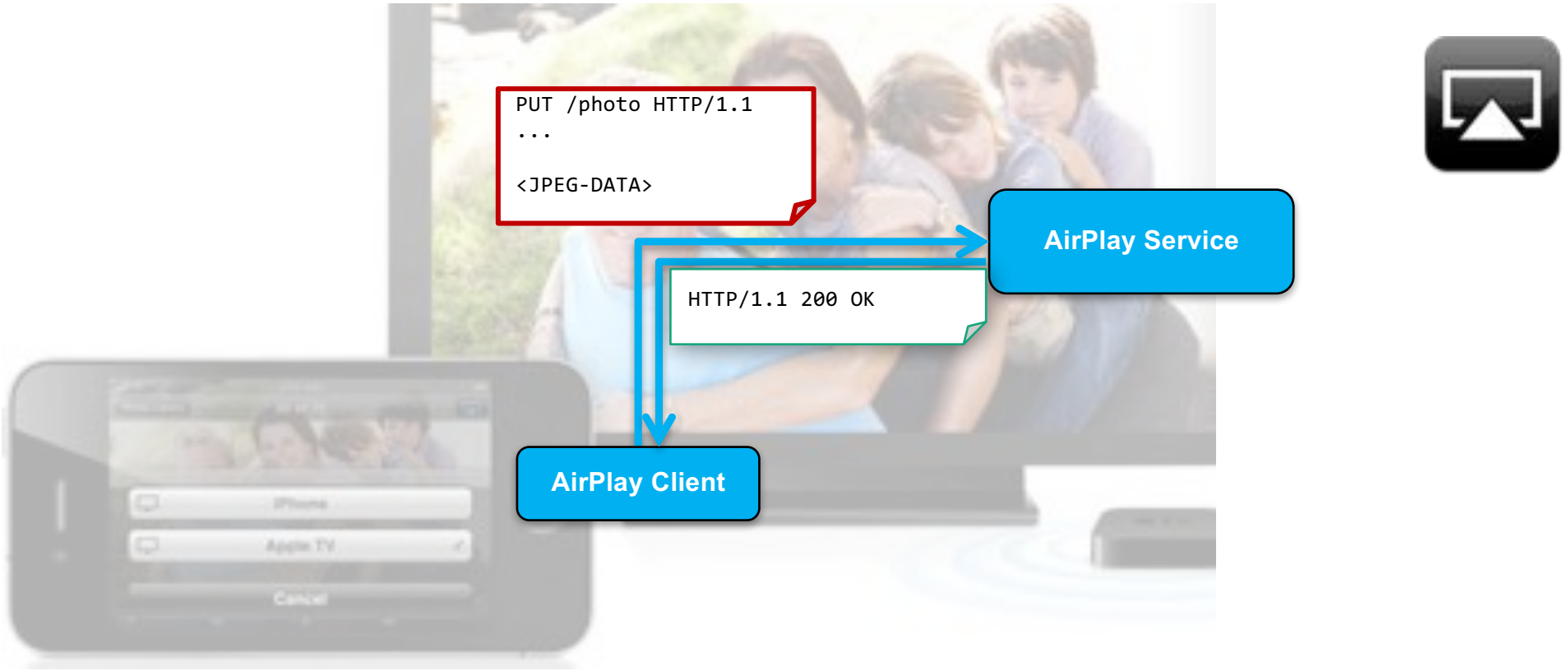
AIRPLAY

- Connect one {iPhone, iPad, MacBook, ...} to an Apple TV
  - Stream audio, photos, videos
  - Mirror screen
- Apple TV advertises two services
  - AirTunes
    - Remote Audio Output Protocol (RAOP)
    - RTSP Server
    - Audio Streaming
  - AirPlay
    - HTTP Server
    - Photo slideshows and video streaming
  - Service discovery over Bonjour (mDNS, DNS-SD)
  - Screen Mirroring special case

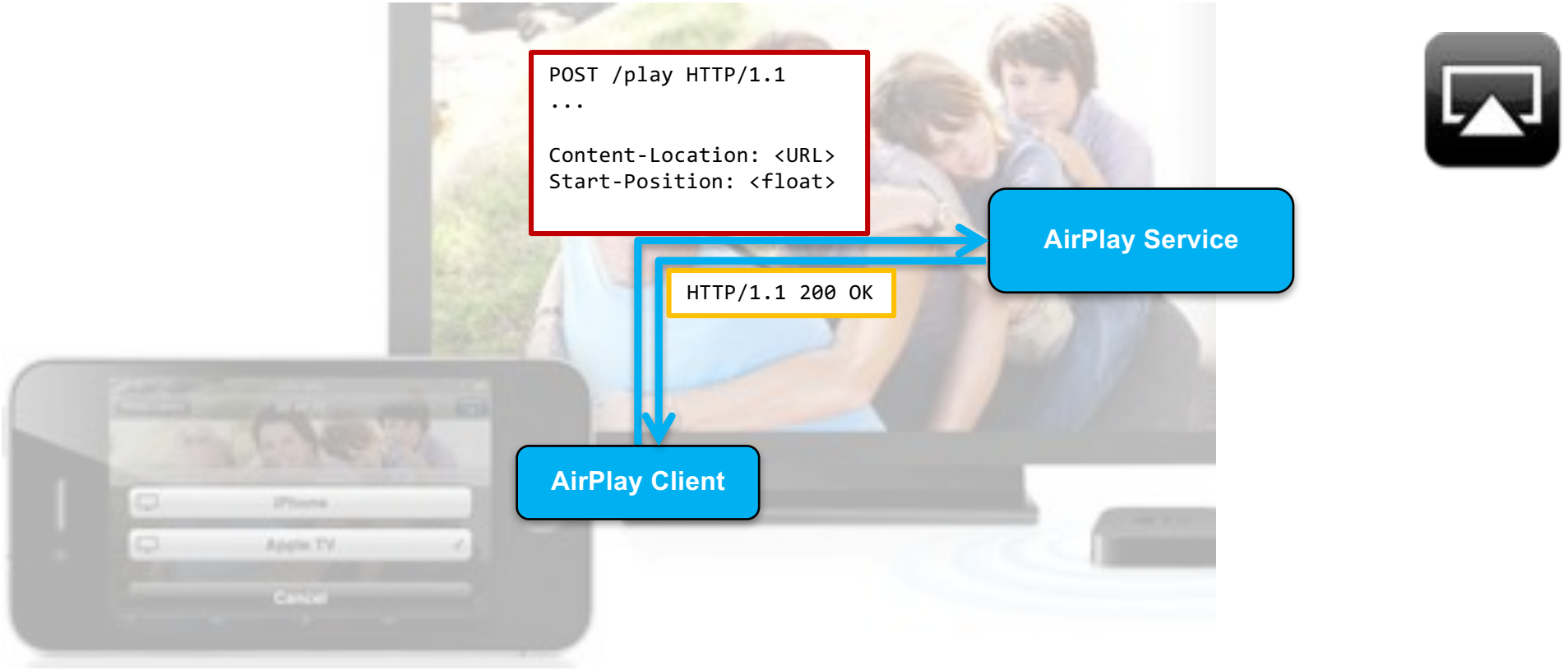
- Two connections between AirPlay service and client
  - Control and content via HTTP
  - Events via PTTH
  - Screen Mirroring special case



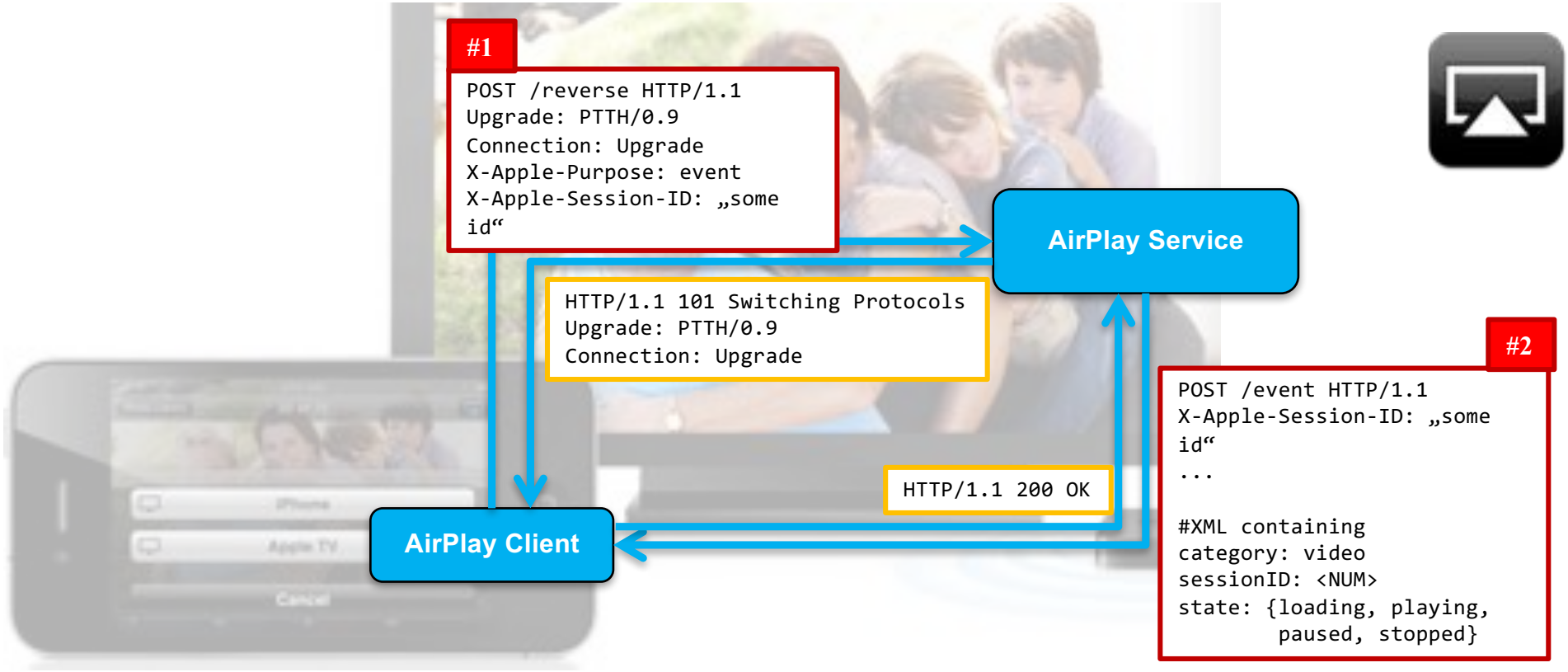
AIRPLAY: PHOTO SHARING



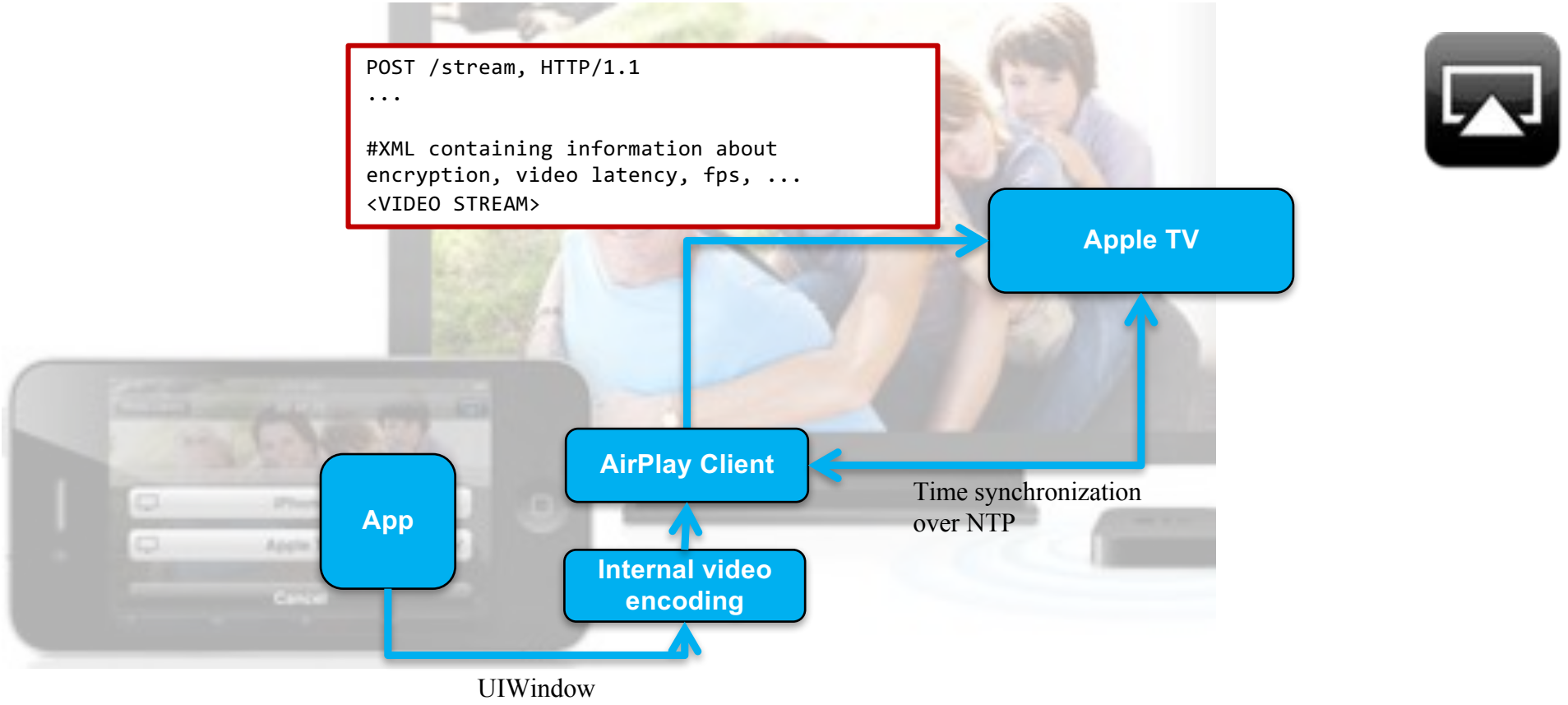
AIRPLAY: VIDEO SHARING

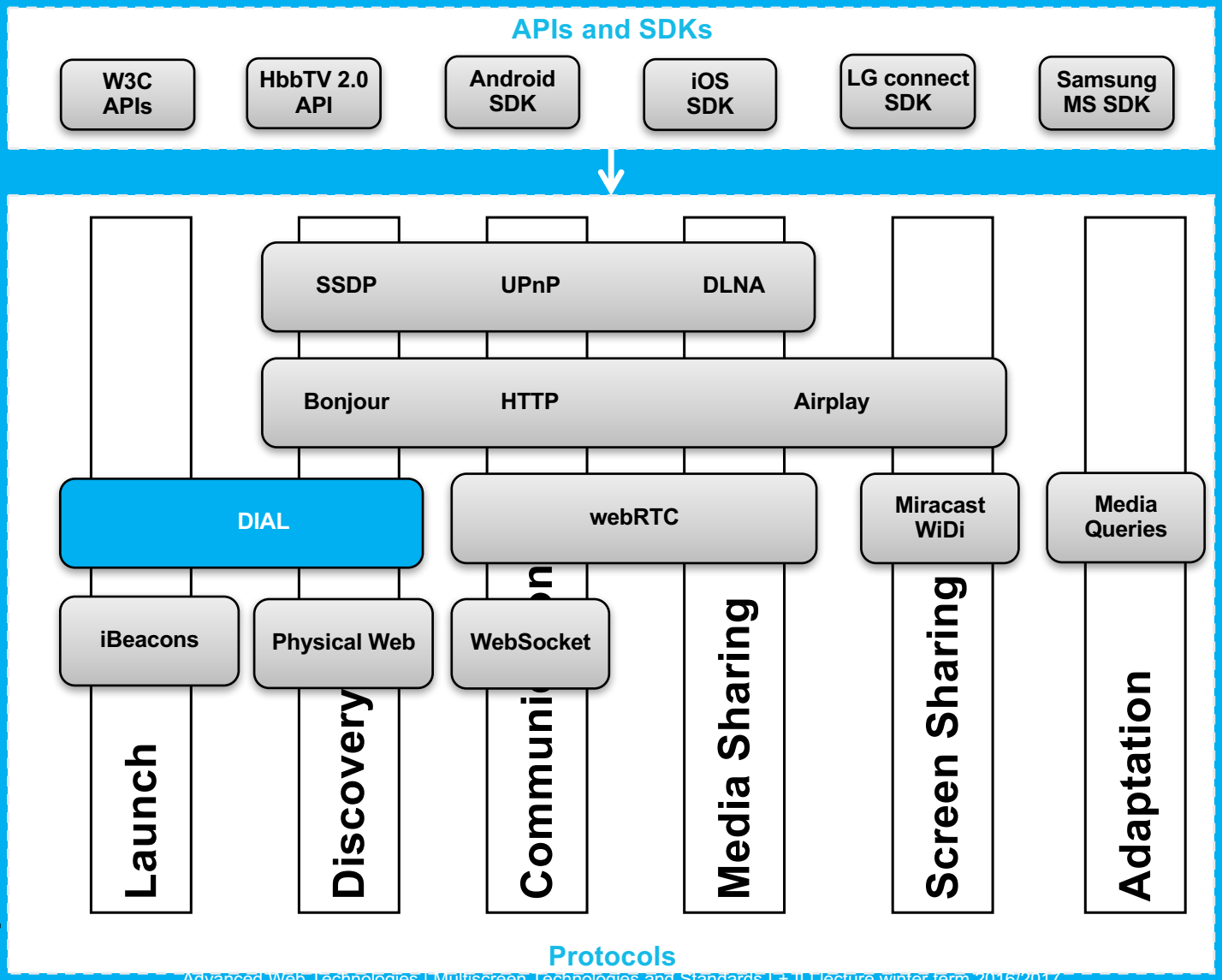


AIRPLAY: EVENTS OVER PTTH



AIRPLAY: SCREEN MIRRORING




APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Media Sharing
Screen Sharing
Adaptation
Protocols

DISCOVERY AND LAUNCH PROTOCOL DIAL

- Is a simple protocol that 2nd screen devices (Mobile devices) can use to discover and launch apps on 1st screen devices (TV and Large displays)
- Uses Discovery (SSDP) and Device/Service descriptions from UPnP Stack
- DIAL UPnP Service type is: *urn:dial-multiscreen-org:service:dial:1*
- New Layer for Launch in form of a REST API
- The end-point of the REST API is returned in the HTTP Header "*Application-URL*" of the device description response
- Use HTTP GET to check if application exists
  - GET <Application-URL>/YouTube → 200 OK response means YouTube exists
- Use HTTP POST to Launch an application
  - POST <Application-URL>/YouTube → 201 CREATED response means YouTube App is launched successfully
- Use HTTP DELETE to stop an application
  - DELETE <Application-URL>/<id> stops an Application with running instance <id>

(1) Discovery

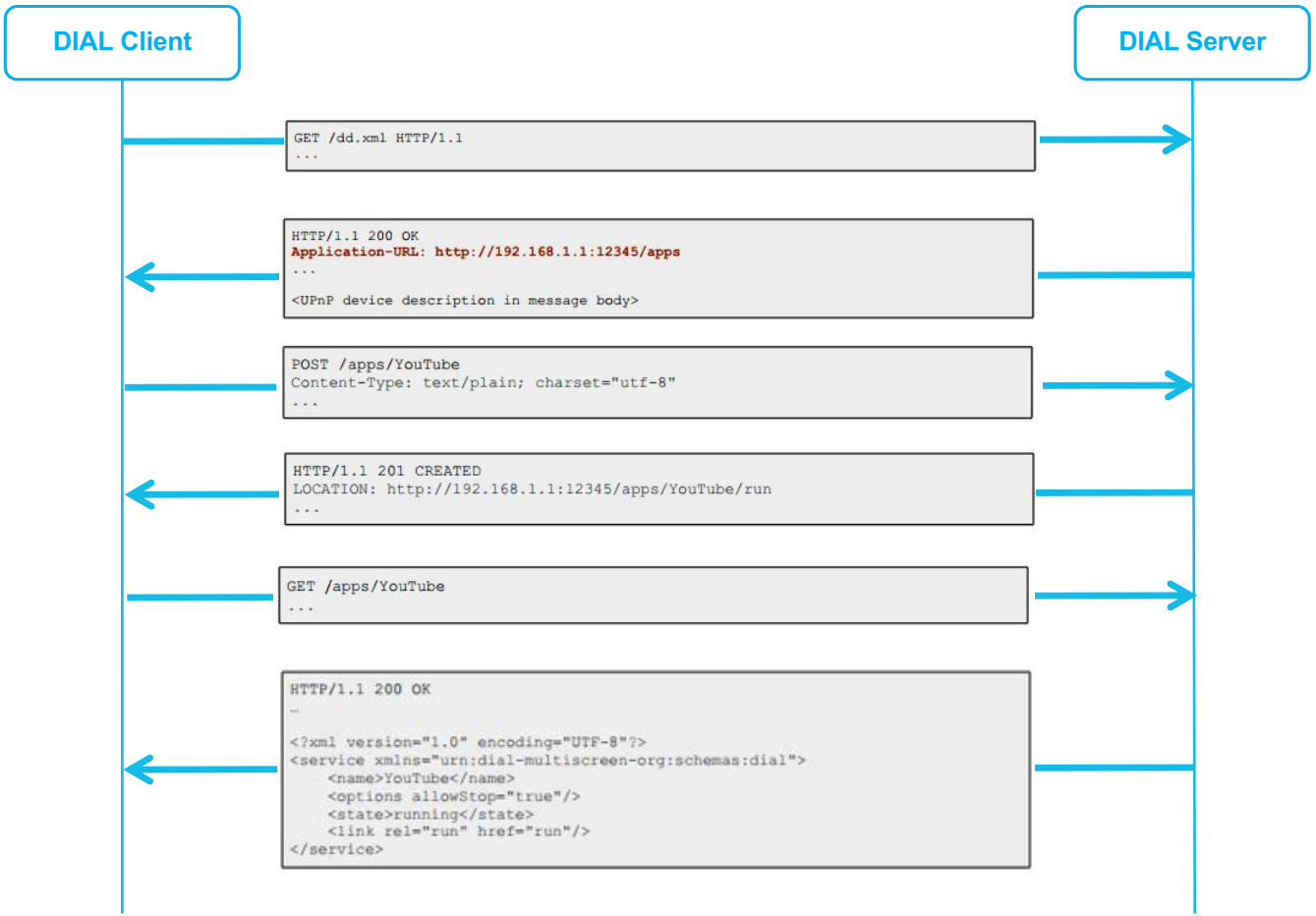
(2) Description

UPnP

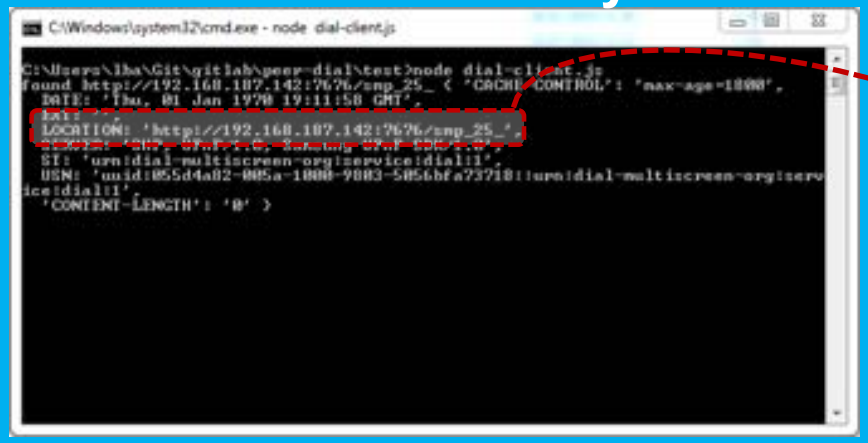
(3) Launch

Source: http://www.dial-multiscreen.org/

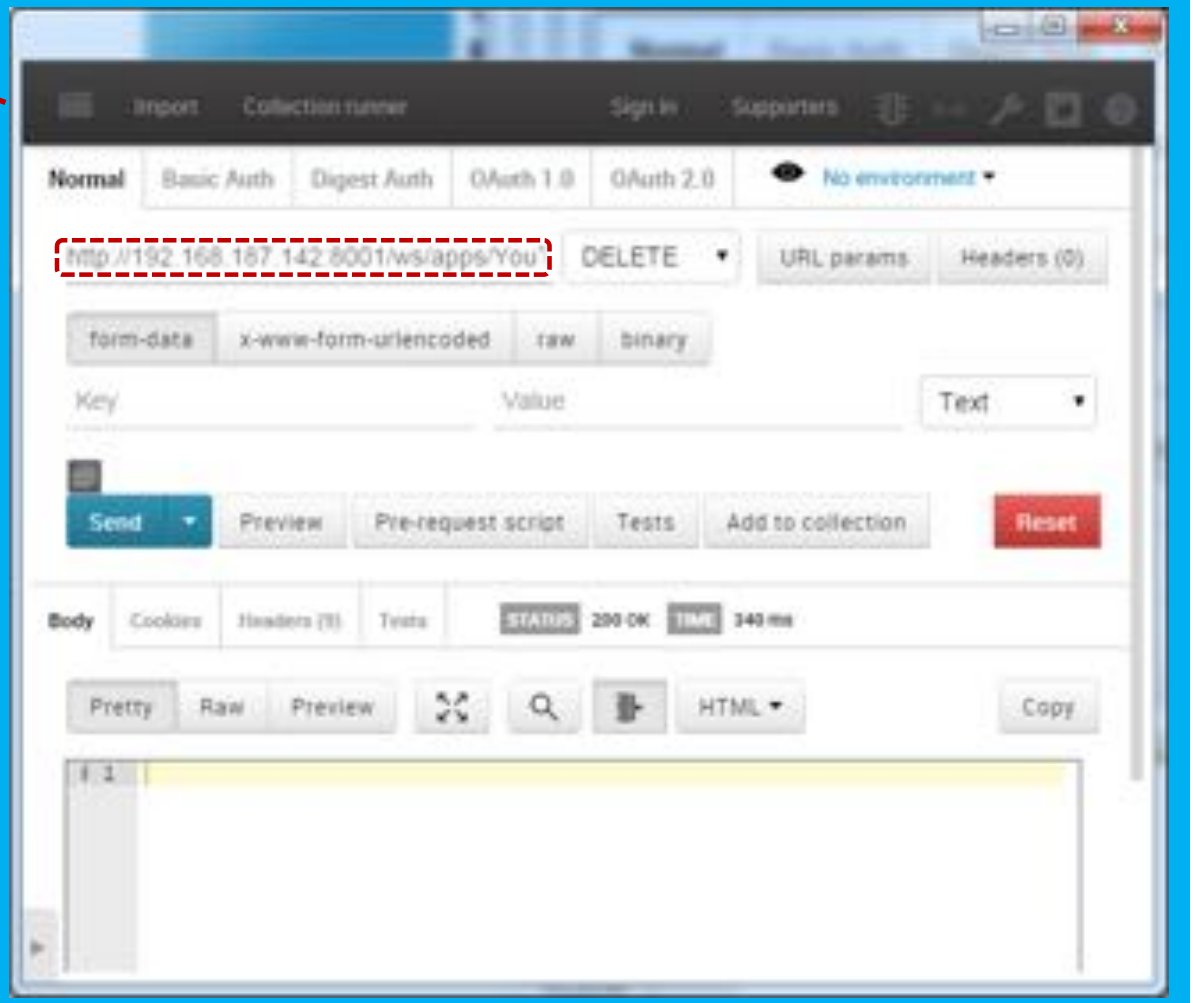
DIAL MESSAGE FLOW EXAMPLE

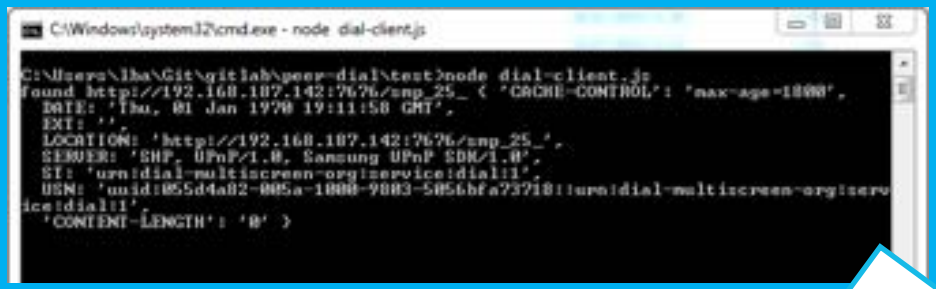


SSDP Discovery



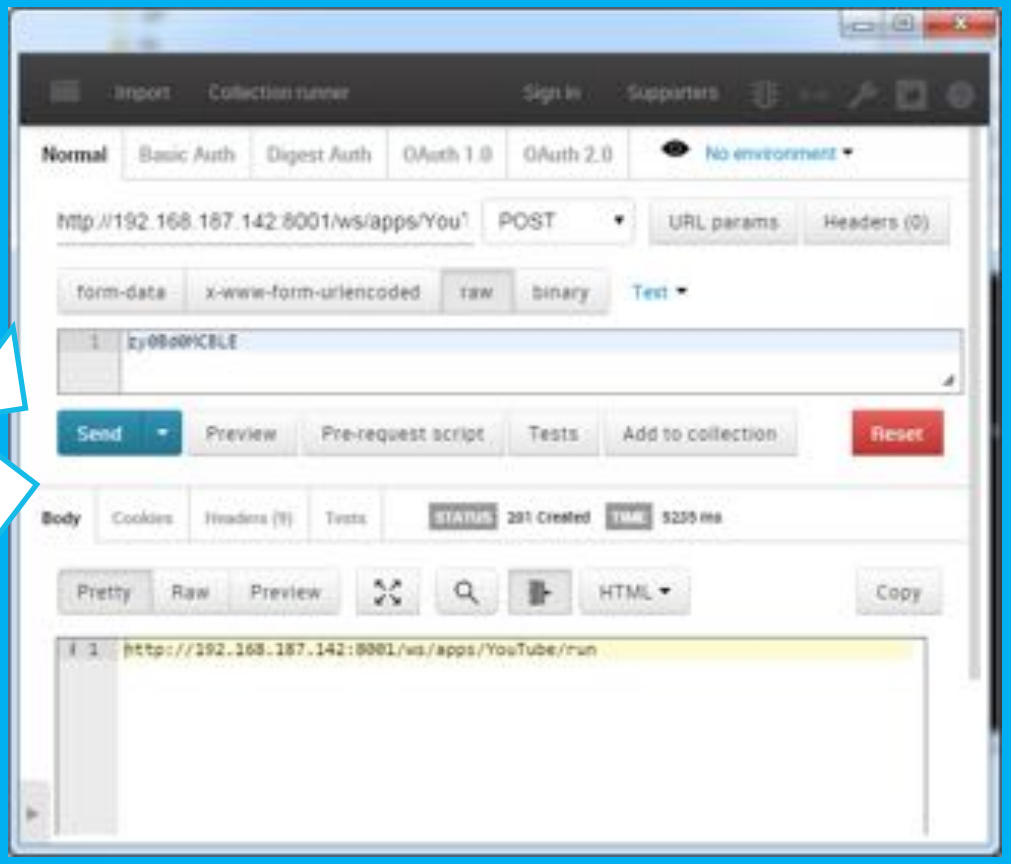
HTTP Client

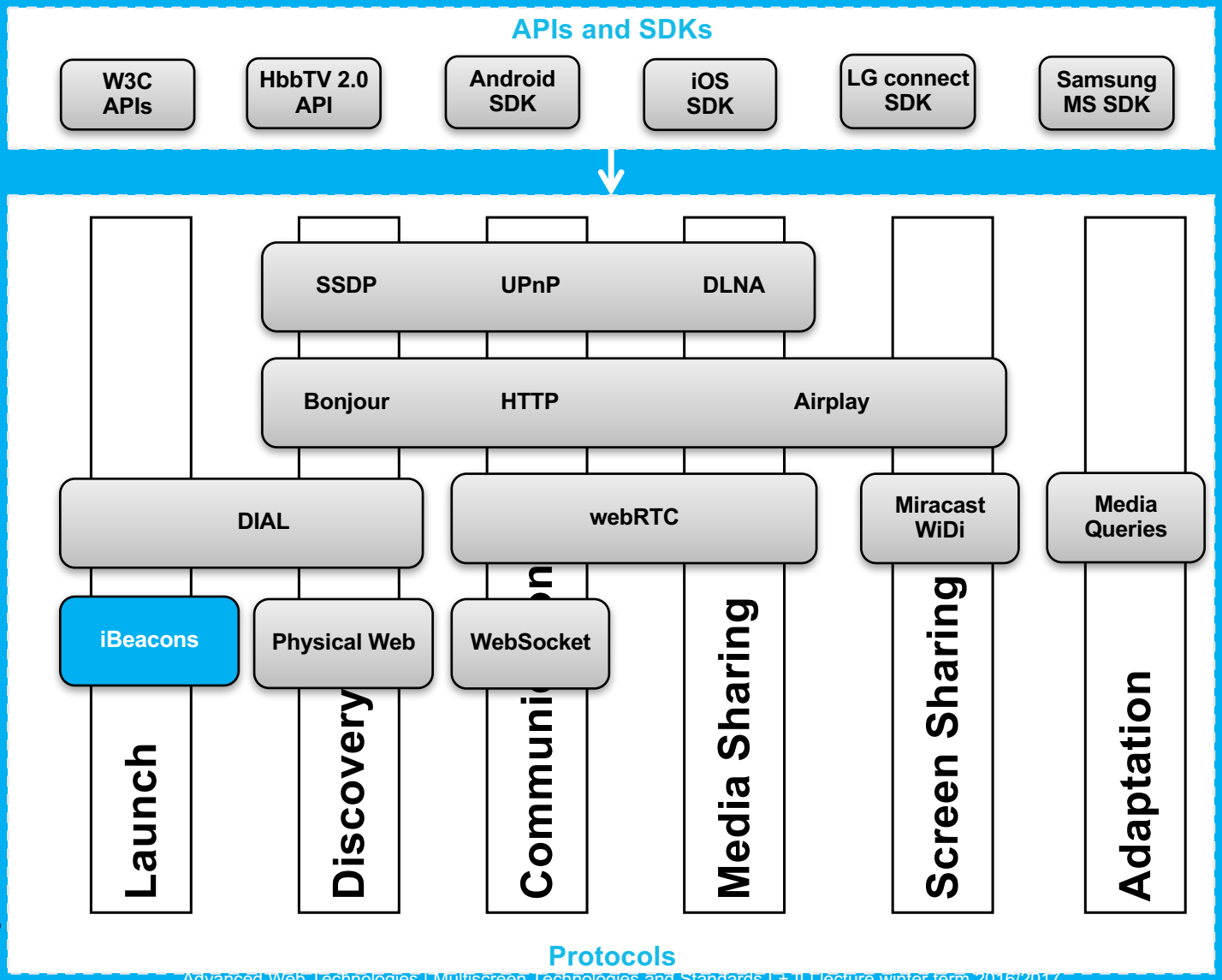



C:\Windows\system32\cmd.exe - node dial-client.js
C:\Users\lba\Git\gitlab\peer-dial\test>node dial-client.js
found http://192.168.187.142:7676/smp_25_ { 'CACHE-CONTROL': 'max-age=1800',
DATE: 'Thu, 01 Jan 1970 19:11:58 GMT',
EXT: '',
LOCATION: 'http://192.168.187.142:7676/smp_25_',
SERVER: 'SHP, UPnP/1.0, Samsung UPnP SDK/1.0',
ST: 'urn:dial-multiscreen-org:service:dial:1',
USN: 'uuid:855d4a82-885a-1000-9883-5856bfa73718::urn:dial-multiscreen-org:service:dial:1',
'CONTENT-LENGTH': '0' }

DIAL Demo

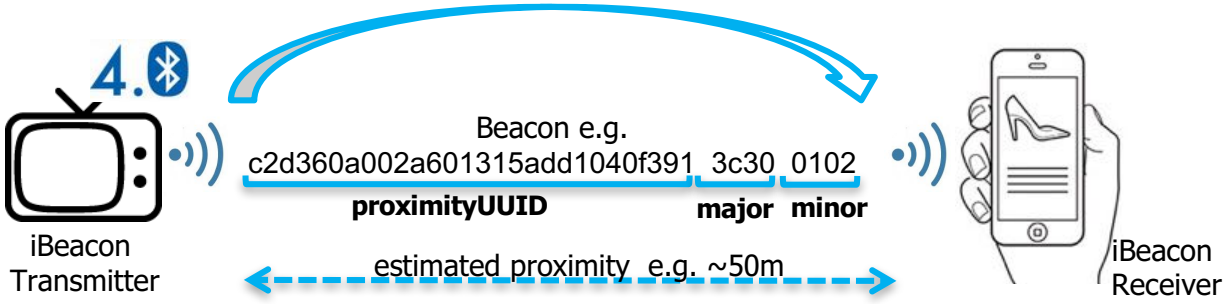

YouTube


Import
Collection runner
Sign in
Normal
Basic Auth
Digest Auth
OAuth 1.0
OAuth 2.0
No environment
http://192.168.187.142:8001/ws/apps/You
POST
URL params
Headers (0)
form-data
x-www-form-urlencoded
raw
binary
Text
Send
Preview
Pre-request script
Tests
Add to collection
Reset
Body
Cookies
Headers (9)
Tests
STATUS
201 Created
TIME
5235 ms
Pretty
Raw
Preview
HTML
Copy
1 http://192.168.187.142:8001/ws/apps/YouTube/run


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Media Sharing
Screen Sharing
Adaptation
Protocols

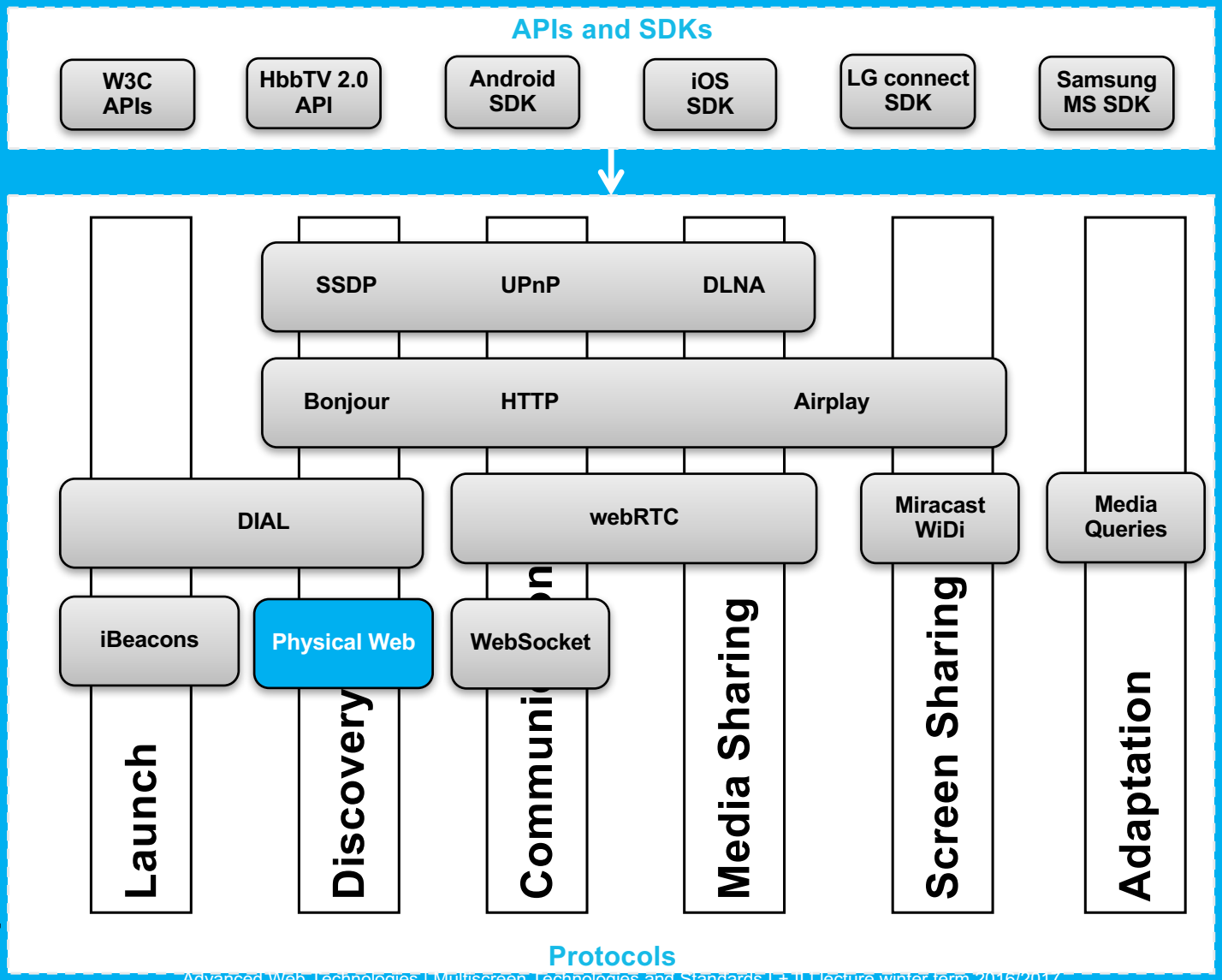
IBEACONS

- iBeacon is a new technology that extends Location Services on iOS (iOS7)
- **Monitoring**: alert App when it enters or leaves a region of an iBeacon
- **Ranging**: App can also estimate proximity to an iBeacon
- Uses Bluetooth low energy signal (BLE) → Bluetooth needs to be turned on
- compatible iOS device: iPhone 4s or later, iPad 3 or later, iPad mini or later, iPod touch 5 or late.



- For multiscreen usage, any device supporting BLE can be used as iBeacon transmitter to wake-up nearby devices and notify users

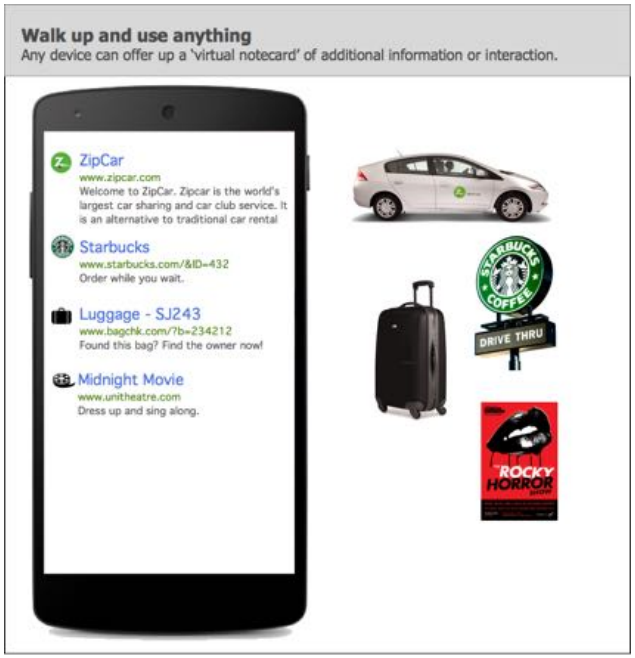
Source: http://estimote.com/api/index.html
Source: http://support.apple.com/kb/HT6048


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

PHYSICAL WEB

Introduction

- The Physical Web is an effort to extend the core superpower of the web - the URL - to everyday physical objects.
- Users are able to walk up to any “smart” physical object (e.g. a vending machine, a poster, a toy, a bus stop, a rental car) and interact with it without first downloading an app.
- The user experience of smart objects should be much like links in a web browser: i.e., just tap and use.
- The Physical Web is an open standard that everyone can use.



→ https://github.com/google/physical-web

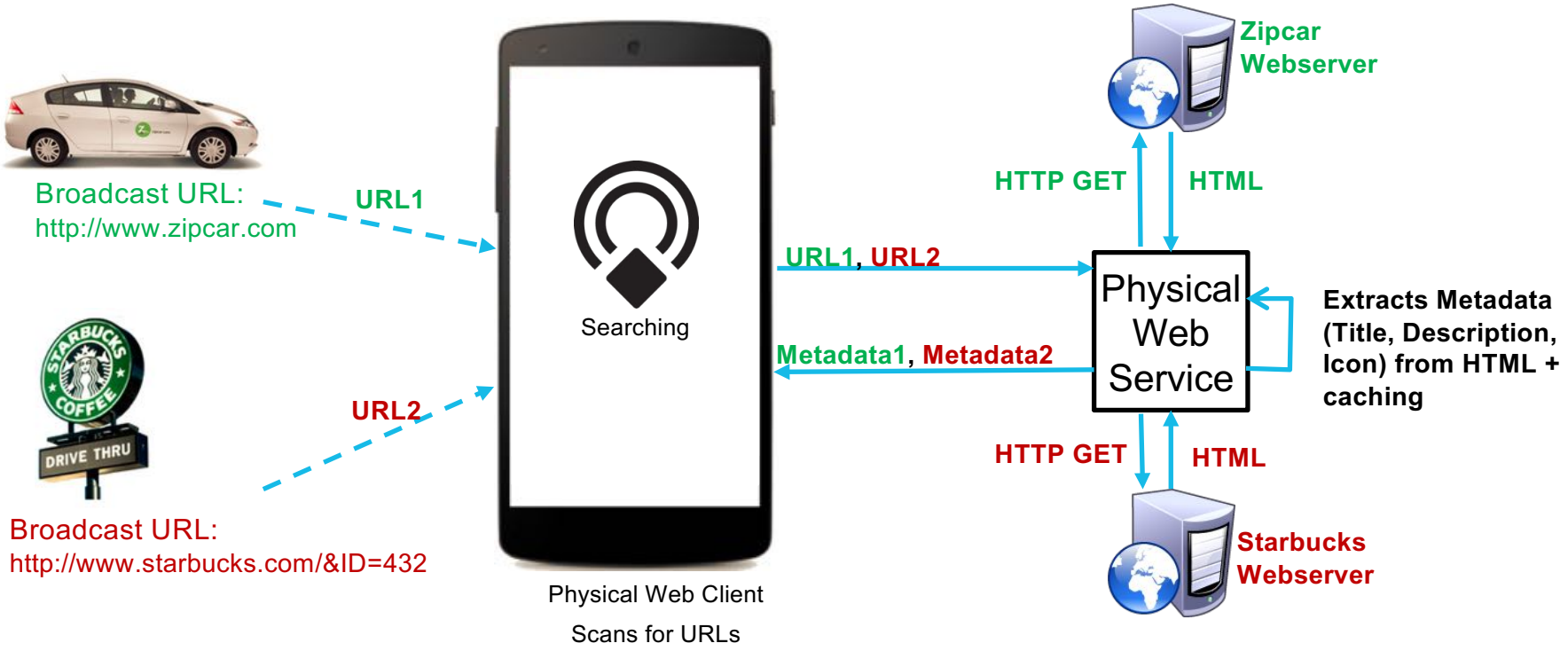
PHYSICAL WEB

How can be used?

- The Physical Web is a discovery service: a smart object broadcasts relevant URLs that any nearby device can receive
- Any nearby display such as a phone or tablet can then see these URLs and offer them up to the user
- It mirrors the basic behavior we have today with a search engine:
  - The user requests a list of what's nearby.
  - A ranked list of URLs is shown.
  - The user picks one.
  - The URL is opened in a full screen browser window.

PHYSICAL WEB

How it works?



PHYSICAL WEB

How to broadcast URLs?

- There are many possible ways to broadcast a URL. The following methods are currently supported:
  - **Bluetooth Low Energy (BLE)** : The initial version of Physical Web uses BLE as it is so ubiquitous on mobile phones and tablets today. It is very energy efficient. There are tiny BLE devices that can broadcast for nearly 2 years on a single coin cell → Protocol Specification: UriBeacon
  - **mDNS**: Broadcast URLs in local network using multicast DNS. URLs are only visible to devices in same network. There is no limitation on URL length
  - **SSDP**: similar concept to mDNS but uses the Simple Service Discovery Protocol used in UPnP to broadcast the URLs.

SSDP Concept and Implementation developed and contributed by Fraunhofer FOKUS/FAME

HOW TO BROADCAST URLS?

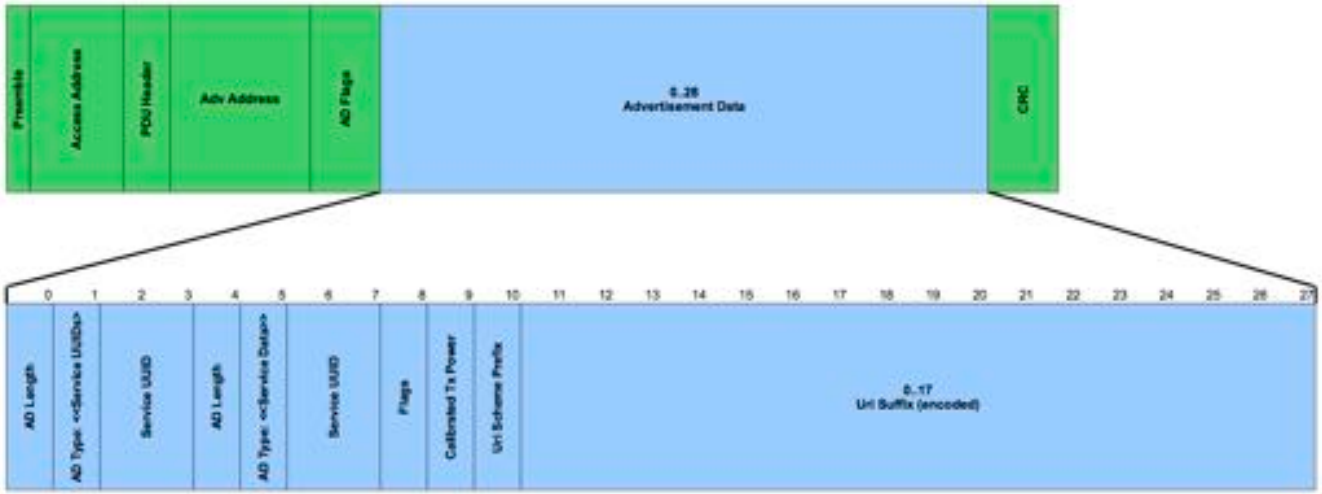
UriBeacon: Concept

- UriBeacon is a wireless advertisement format for broadcasting Uris to nearby smart device:
  - Compliant with Bluetooth 4.0 Advertising messages.
  - Compatible with iOS and Android low power scanning modes.
  - Independent of any particular hardware, user-agent, service or mobile platform.
  - Openly licensed under Apache 2.0 and as such free of restrictive and proprietary terms.
- UriBeacon connects low power beacons to the family of Open Web technologies and is one step towards making the Internet of Things as easy to use as the World Wide Web.

HOW TO BROADCAST URLS?

UriBeacon: Specification (https://github.com/google/uribeacon)

- The UriBeacon consists of two basic data types in an Advertising Data (AD) block: <<*Service UUID*>> and <<*Service Data*>
- The <<Service UUID>> provides the mechanism for efficient, cross platform background scanning. Both Android and iOS allow background scanning for Services
- The <<Service Data>> provides the mechanism for transporting the payload of Uri, Tx Power Level (for ranging), and flags.



HOW TO BROADCAST URLS?

mDNS

- https://github.com/google/physical-web/blob/master/documentation/mDNS_Support.md
- Physical Web is about getting URLs into the physical world. However, this isn't limited to just BLE beacons (UriBeacon)
- mDNS is a service broadcast technique used in Wi-Fi networks. It has two advantages over BLE:
  - Only people logged into same Network can see the mDNS URLs
  - It doesn't have the length restrictions of BLE
- “dns-sd” Tool from Apple can be used to broadcast URLs over mDNS. Example:

```
dns-sd -R "My Test" _http._tcp . 80 path=/path-to-page.html
```

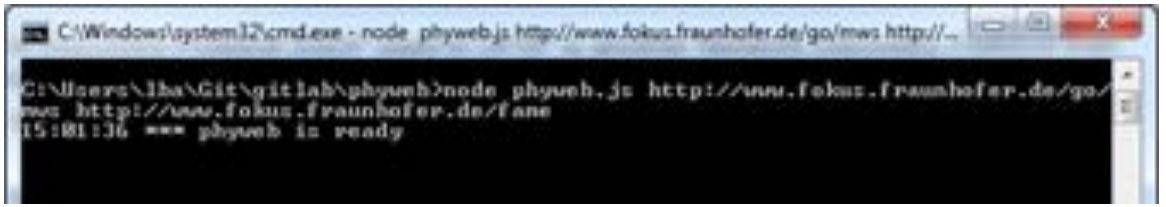
HOW TO BROADCAST URLS?

SSDP

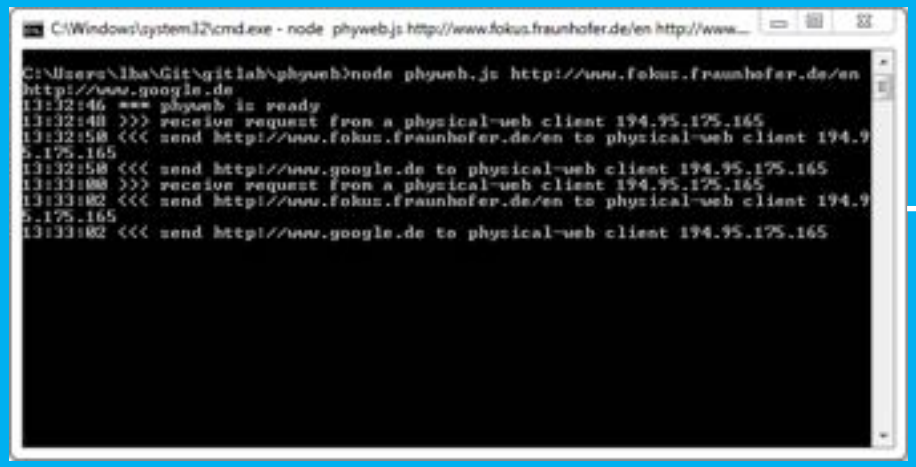
SSDP Concept and Implementation developed and contributed by FAME

- https://github.com/google/physical-web/blob/master/documentation/ssdp_support.md)
- Simple Service Discovery Protocol SSDP used in UPnP is another Network Discovery Protocol than can be used broadcast the URLs
- Concept and implementation of integrating SSDP in Physical Web is developed and contributed by FAME
- SSDP is now supported in the Physical Web Android App. iOS integration is work in progress.
- The SSDP Physical Web Advertiser Node.js Tool (for Win, Linux and Mac) provided by FAME can be used to broadcast one or more URLs to devices in local network. Tools is available here: https://github.com/fraunhoferfokus/phyweb. Following Example broadcasts two URLs:

```
node phyweb http://www.fokus.fraunhofer.de/go/mws http://www.fokus.fraunhofer.de/fame
```

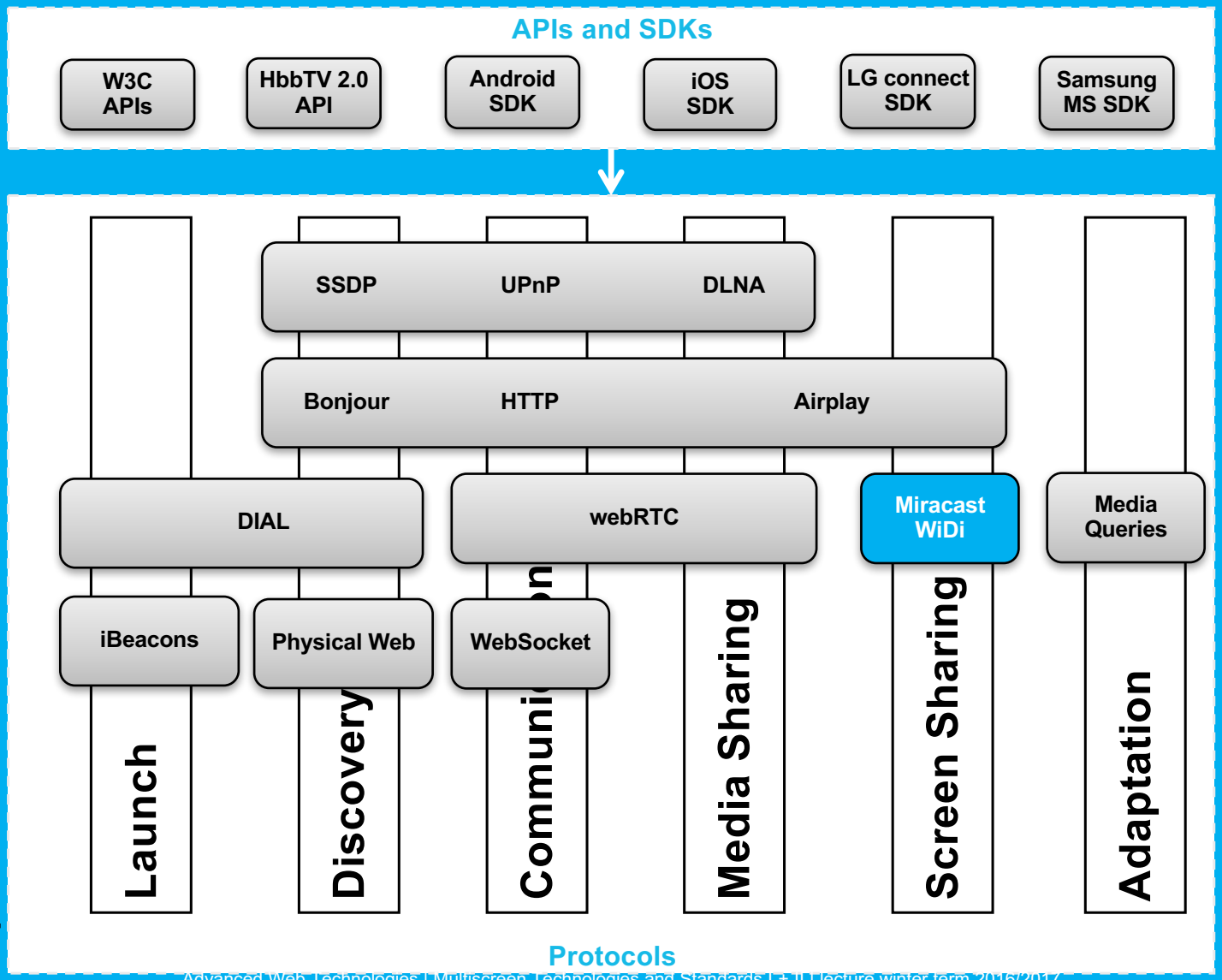


Physical Web Demo



SSDP

BLE/UriBeacon


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Media Sharing
Screen Sharing
Adaptation
Protocols

Physical Web Article on heise.de

MIRACAST/INTEL WIDI

- Peer-to-peer wireless screencast standards
- Wi-Fi Direct for wireless ad-hoc peer-to-peer connections

- **Miracast**
- Developed by Wi-Fi Alliance as open, platform-independent standard
- First devices certified in 09/2012
- Today 540 source devices, 1771 target devices are (2014-05-05)
- Smartphones and Wi-Fi adapters
- STB, Wi-Fi adapters, TVs, blue-ray players, ...
- LG, Samsung, Toshiba, Sony, Philips, Sharp, Intel, Qualcomm, TI, Realtek , Broadcom, NVIDIA, ...

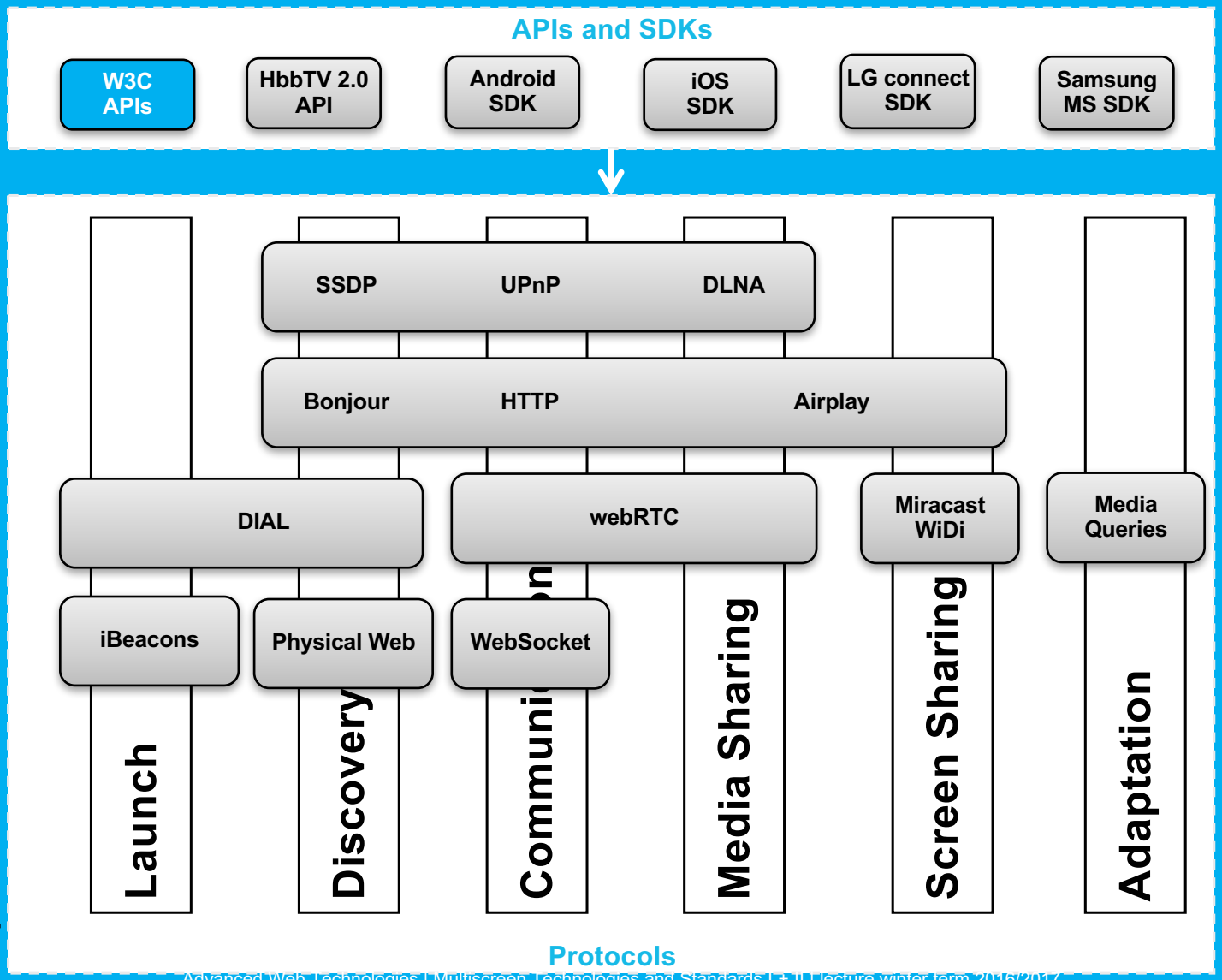
- Support 1080p video, 5.1 audio
- **Official support in Android 4.2+**

- **Intel WiDi**
- Introduced 2010 by Intel, proprietary
- Processor renders second virtual display, broadcast via Wi-Fi
- Intel My Wi-Fi implements Wi-Fi Direct
- Supported by
- LG, Toshiba, Samsung, Sony, Belkin, Netgear
- **Since v3.5 fully compatible to Miracast**

Source: http://www.wi-fi.org/discover-wi-fi/wi-fi-certified-miracast

Miracast demo on Android and NETGEAR Push2TV Display Adapter

Source: http://www.netgear.com/


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

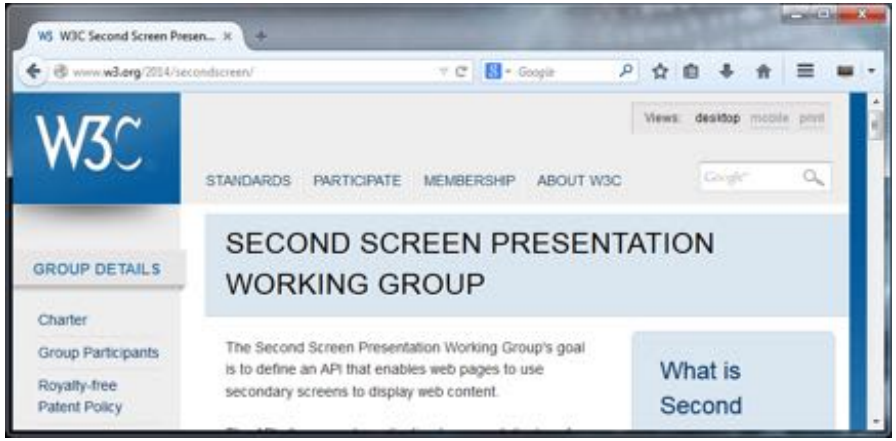
W3C Second Screen Presentation API

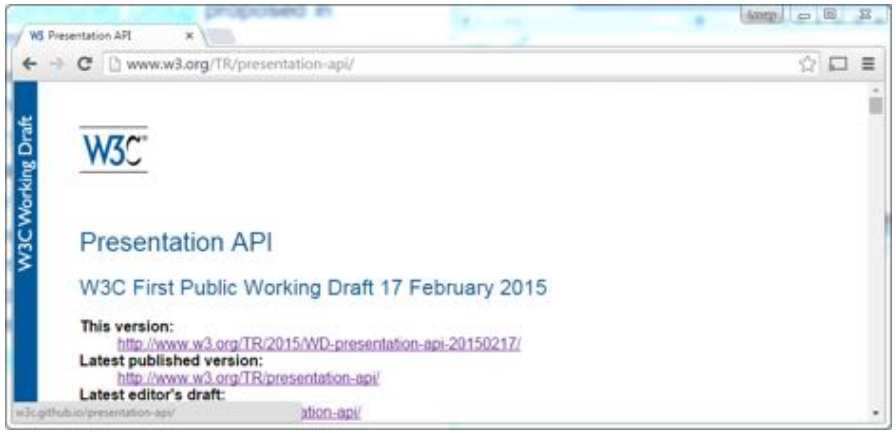
W3C SECOND SCREEN PRESENTATION CG

- W3C Community Group proposed in September 2013 by Intel
- Key partners: Intel, Google, Mozilla, Fraunhofer FOKUS, Netflix, LGE, etc.
- Goal: “Is to define an API that allows web applications to use secondary screens to display Web content”
- Final Report of the CG published in July 2014.

W3C SECOND SCREEN PRESENTATION WG

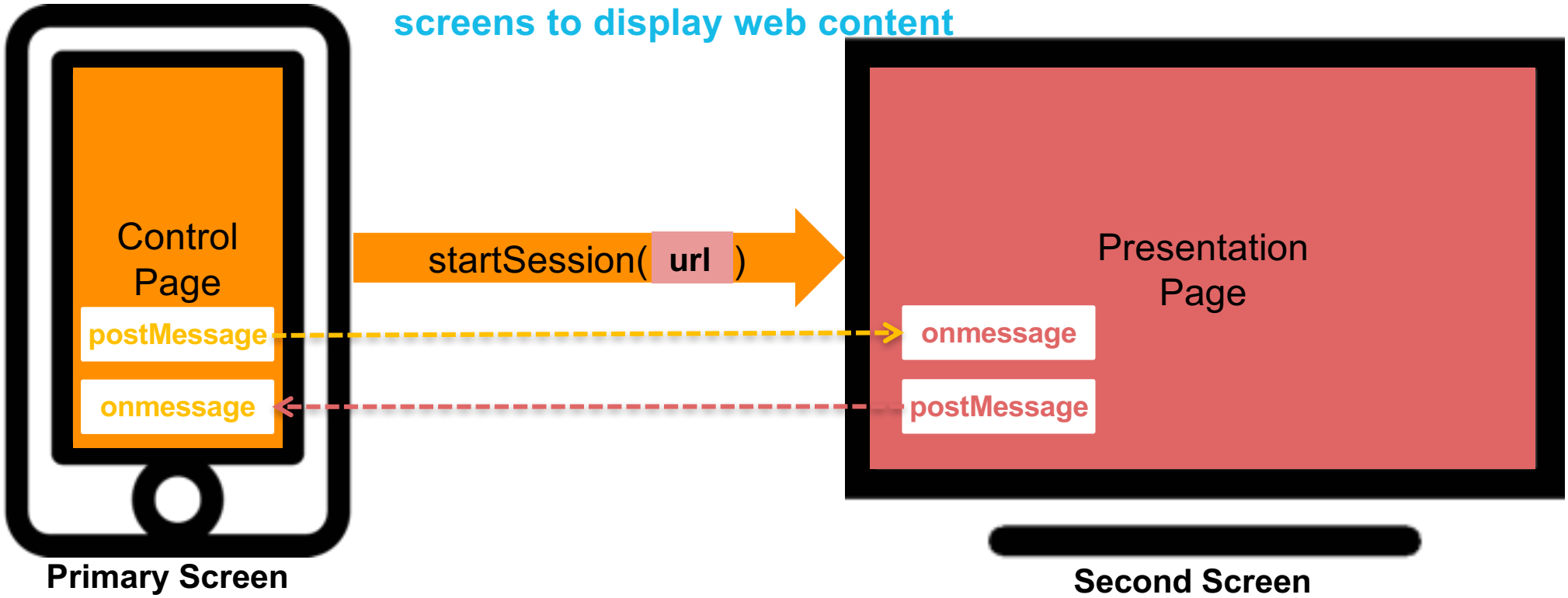
- The work of the Second Screen Presentation API is continued in a Working Group
- The Working Group was created in October 2014 → End date: 31 October 2016
- The WG took the final report of the CG as initial working draft for the Presentation API





W3C SECOND SCREEN PRESENTATION API

Goal is to define an API that enables web pages to use secondary screens to display web content

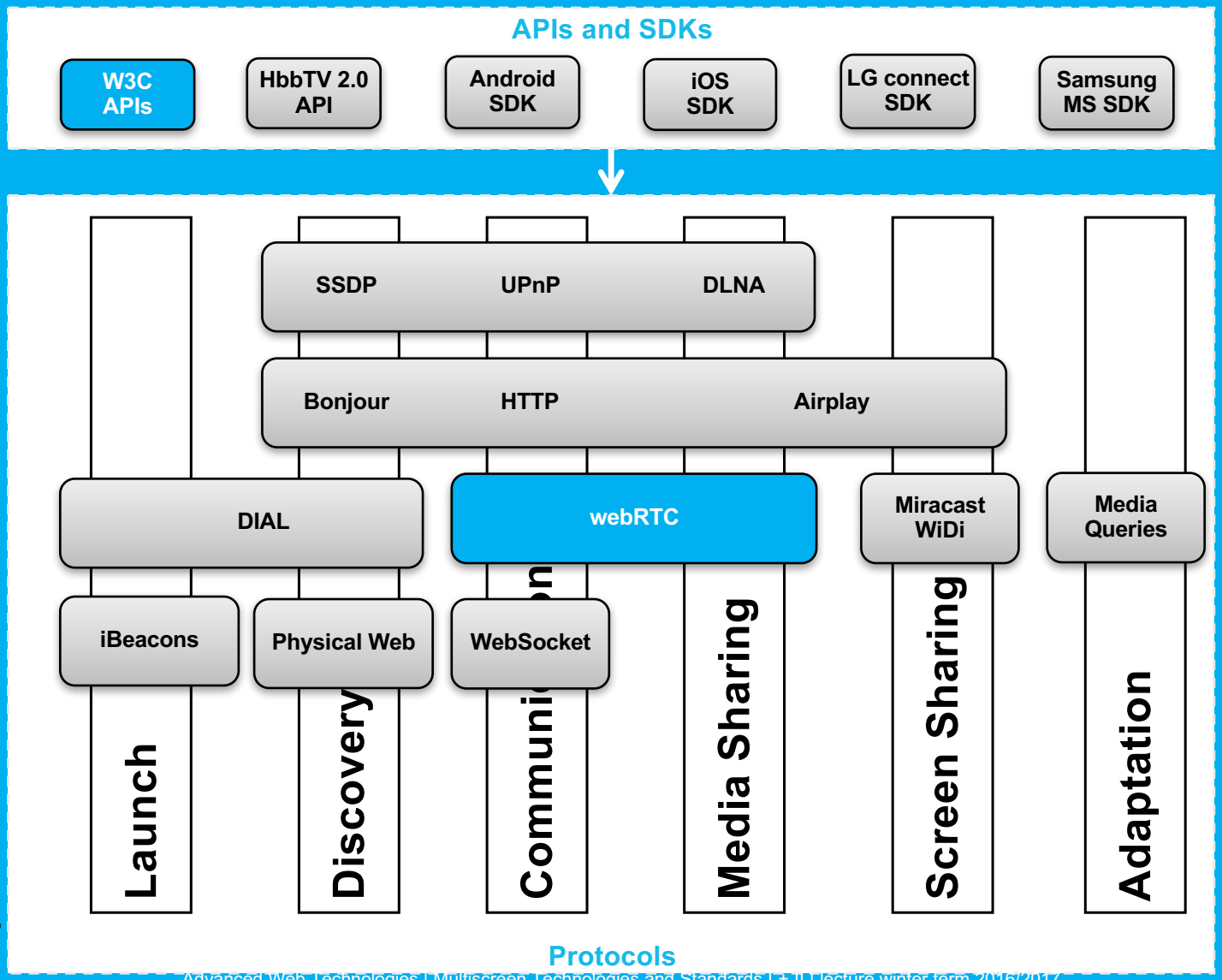


W3C SECOND SCREEN PRESENTATION API

Scope

- Define an API that allows a web application to:
  - ... request display of web content on a connected display
  - ... communicate with and control the web content
  - ... identify whether at least one secondary screen is available for display
- The web content may comprise HTML documents, web media types such as images, audio, video, or application-specific media
- The specification includes security and privacy considerations

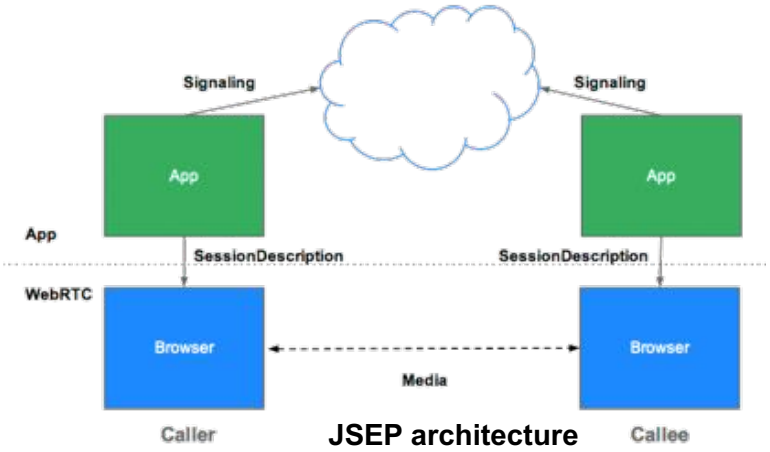
FAMIUM Presentation API Demo


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Media Sharing
Screen Sharing
Adaptation
Protocols

WEBRTC

- W3C specification for real-time audio/video communication in the browser
  - IETF specifies **RTCWeb**, including **JSEP** (JavaScript Session Establishment Protocol) for session establishment
- Key features
  - **MediaStream** – audio and video capturing using **mic**, **camera** and **screen** → Rendering of media streams in **<audio />**, **<video />**, all CSS3 goodness
  - **PeerConnection** – Peer-to-peer audio/video connection
  - **DataChannel** – Peer-to-peer application data transfer, e.g. JSON send(), onmessage(). Same methods can be used to send/receive Blobs, etc.

Source: http://www.webrtc.org/

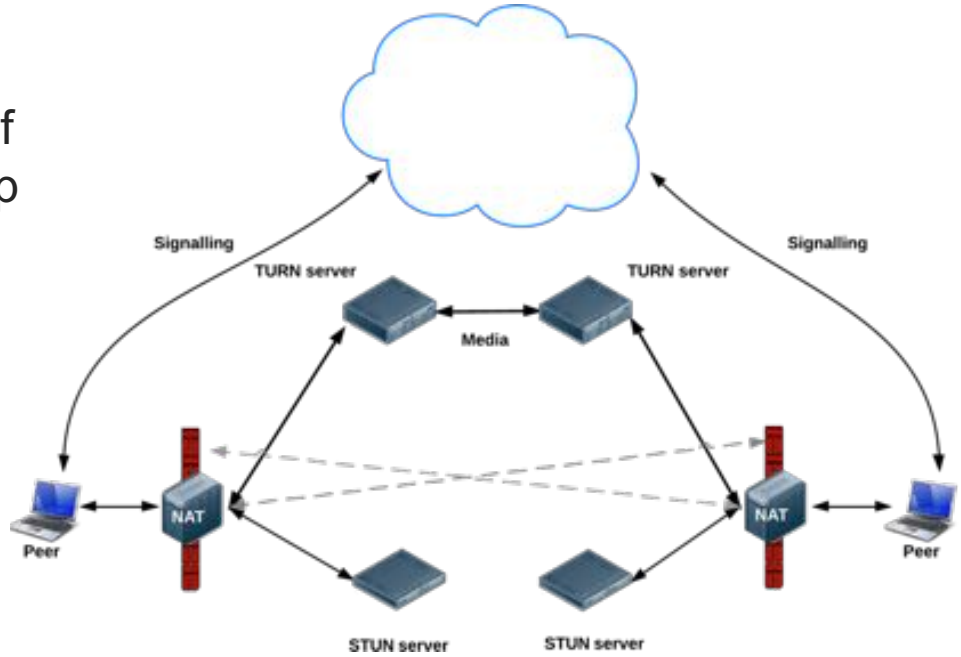


JSEP architecture

Source: http://www.html5rocks.com/en/tutorials/webrtc/infrastructure

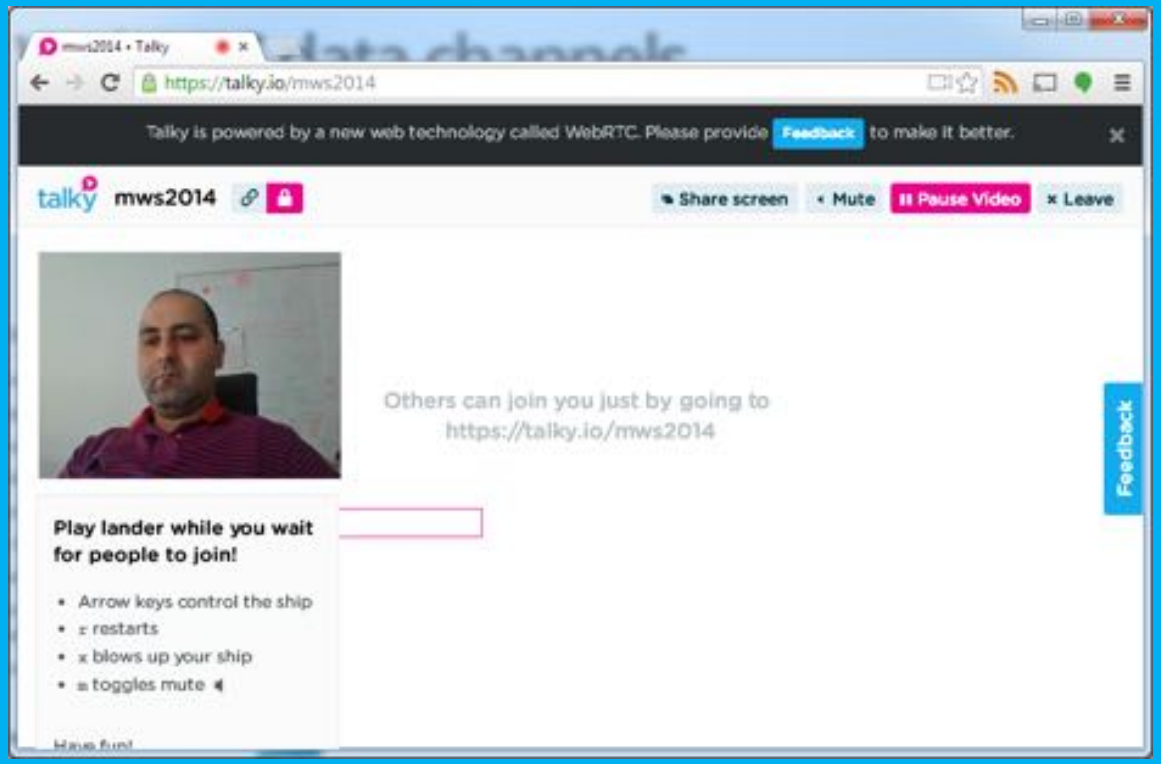
WEBRTC

- Although WebRTC enables peer-to-peer communication, it still needs servers:
  - **For signaling**: to enable the exchange of media and network metadata to bootstrap a peer connection
  - **To cope with NATs and firewalls**: by using the ICE framework to establish the best possible network path between peers, by working with STUN and TURN servers

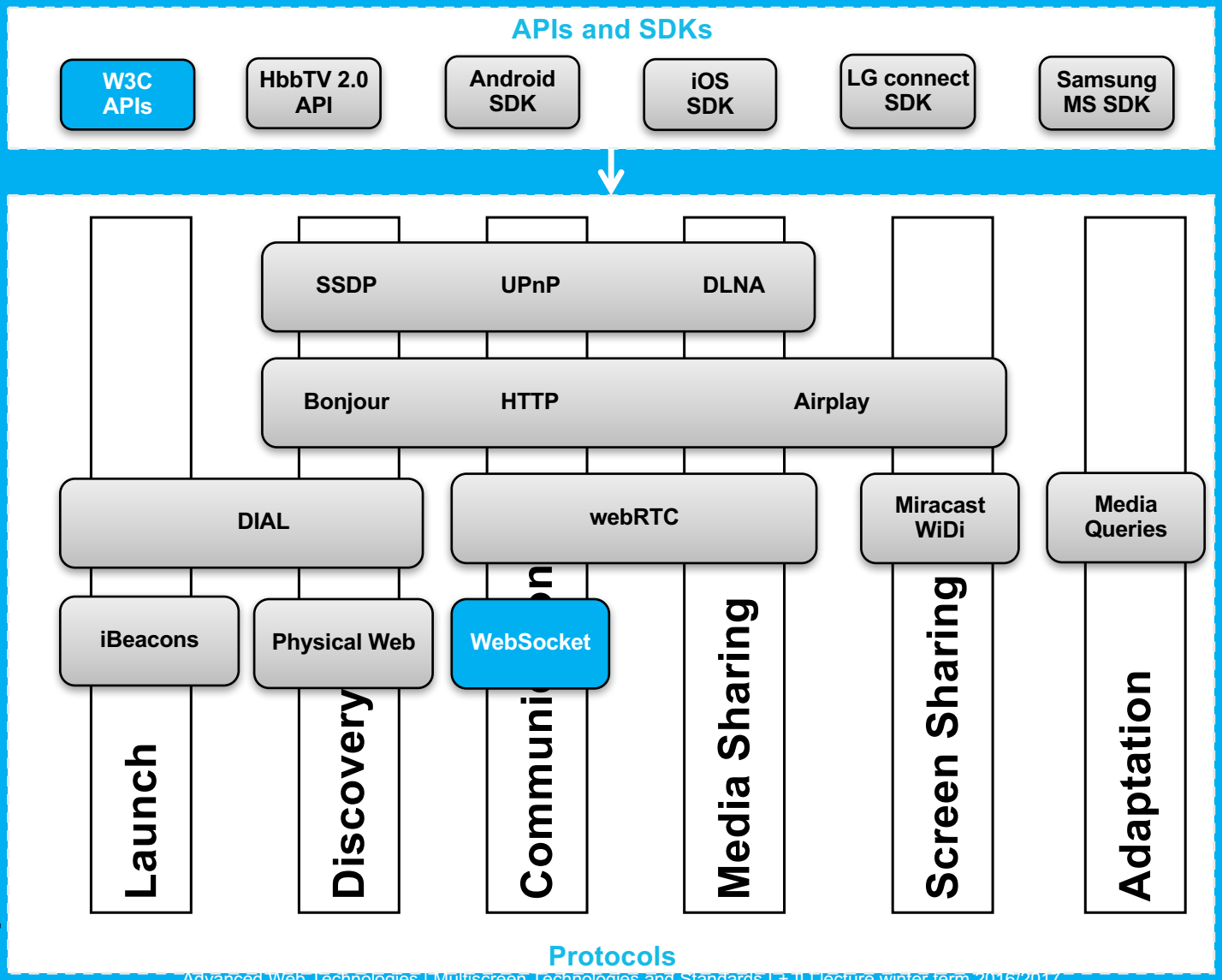


Source: http://www.html5rocks.com/en/tutorials/webrtc/infrastructure

WebRTC Demo



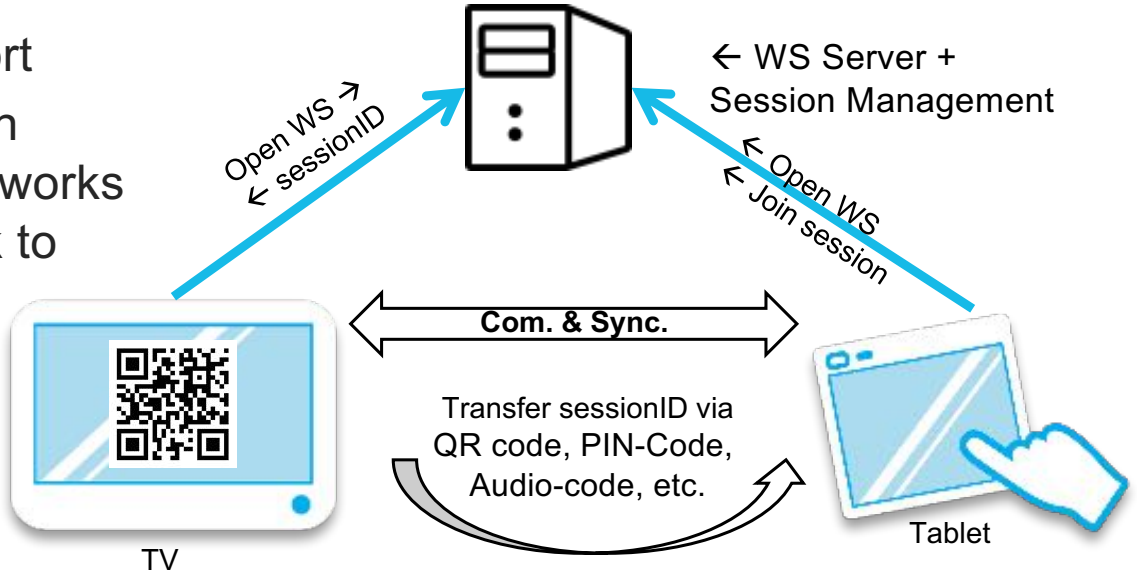
https://talky.io/mws2014

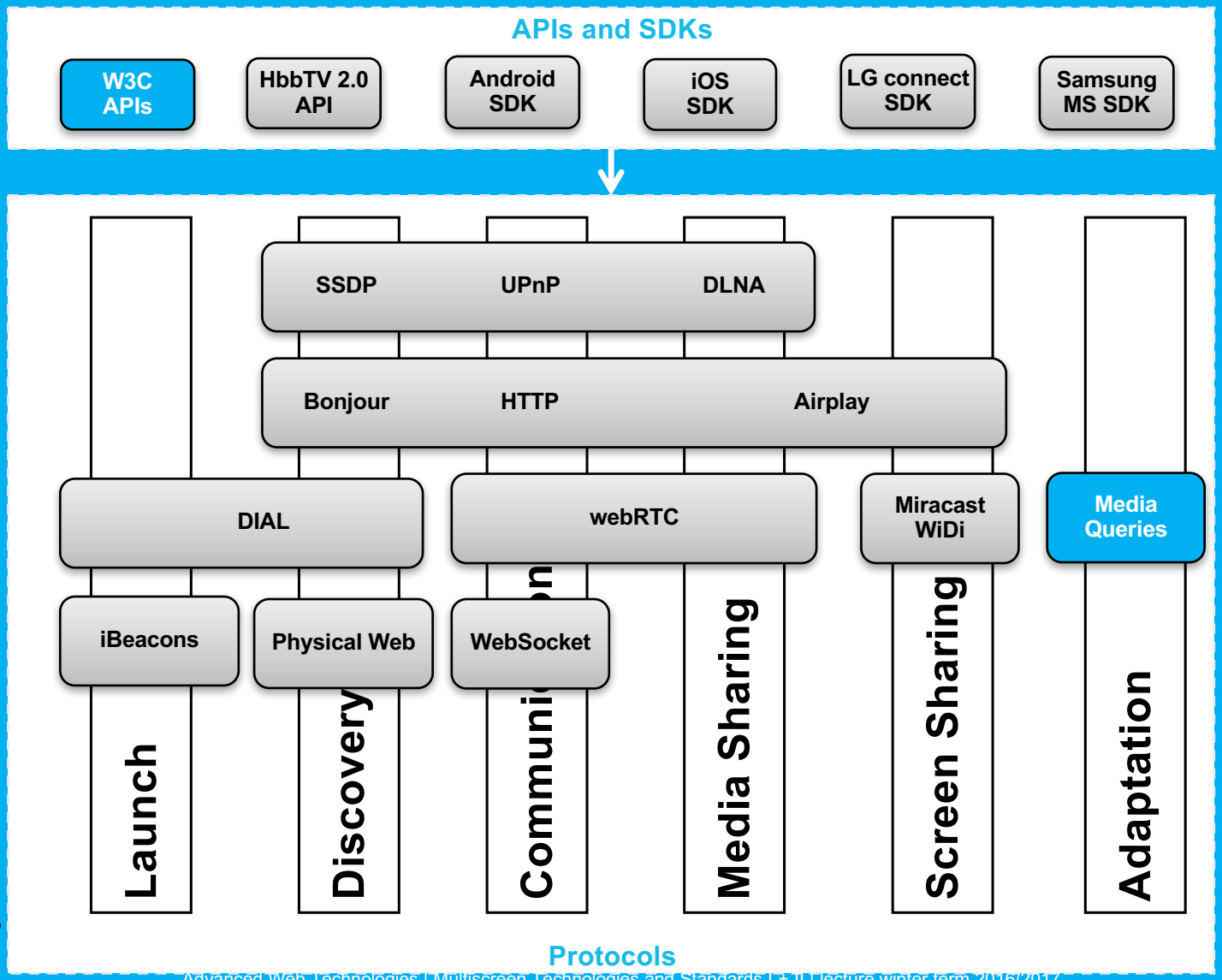

APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

WEBSOCKET

- **WebSocket Protocol** (defined by IETF) enables **two-way communication** between a **client and server** → https://tools.ietf.org/html/rfc6455
- W3C **WebSocket API** enables Web pages to use the WebSocket protocol → http://www.w3.org/TR/websockets/
- **DOMString**, **Blob**, **ArrayBuffer** Support
- Implemented in all modern Browsers on Desktop and Mobile → 3rd party Frameworks (e.g. socket.io) can be used as fallback to support older browsers

Typical Multiscreen usage



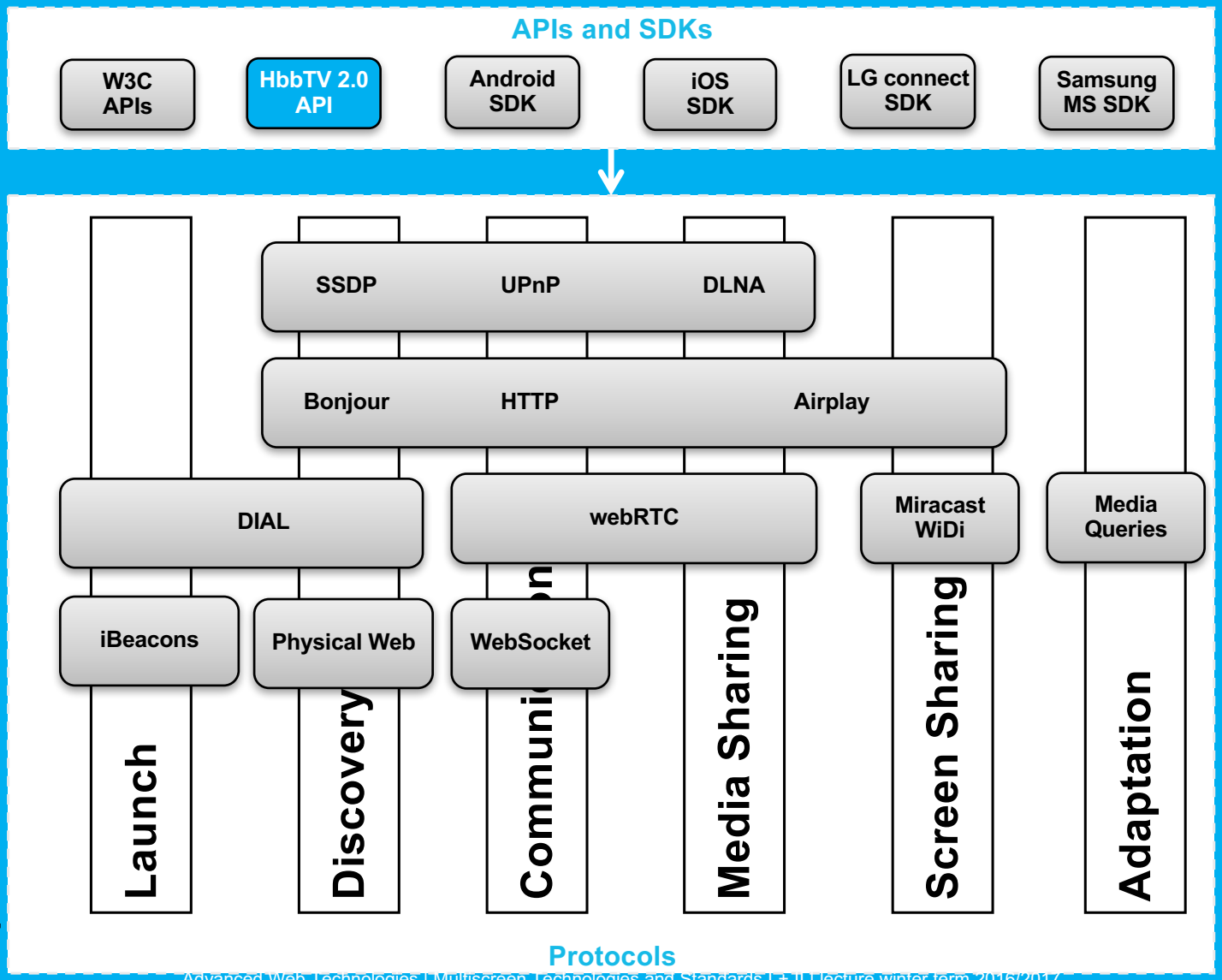

APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

W3C CSS3 MEDIA QUERIES

- W3C Recommendation 19 June 2012 → http://www.w3.org/TR/css3-mediaqueries
- A media query consists of a media type (e.g. screen, print, speech, etc.) and zero or more expressions that check for the conditions of particular media features.
- Examples:
  - ***<link rel="stylesheet" media="screen and (color), projection and (color)" href="example.css">***
  - ***@import url(color.css) screen and (color);***
  - ***@media screen and (min-width: 400px) and (max-width: 700px) { /*CSS*/}***
  - ***@media handheld and (min-width: 20em), screen and (min-width: 20em) {/*CSS*/}***
  - ***@media screen and (device-aspect-ratio: 16/9) {/*CSS*/}***
  - ***@media print and (min-resolution: 118dpcm) {/*CSS*/}***

Simple CSS3 Media Queries Example

http://www.maxdesign.com.au/articles/css3-media-queries/media-sample/


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

HBBTV

- **Hybrid Broadcast Broadband TV** → "European initiative aimed at harmonising the broadcast and broadband delivery of entertainment to the end consumer through connected TVs and set-top boxes" (https://hbbtv.org)
- **HbbTV 2.0** is the most recent version of the HbbTV specification. It has been published by HbbTV in early February 2015.(older versions is 1.0, 1.5)

Source: http://hbbtv.org

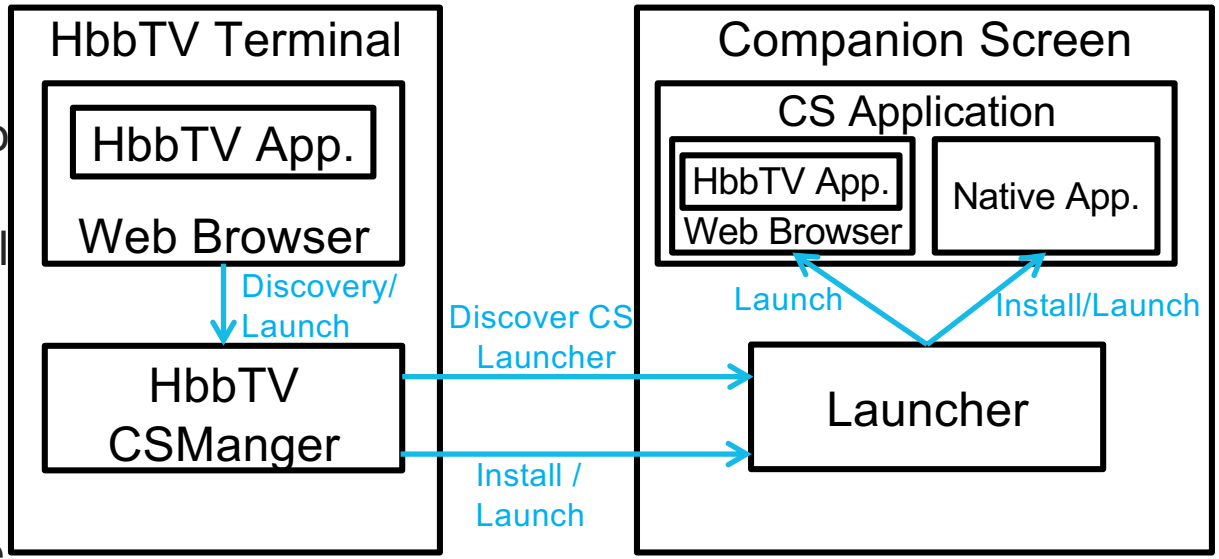
NEW FEATURES SUPPORTED IN HBBTV 2.0

- HTML5 and associated technologies: many modules from CSS3, DOM3, Web Sockets, Web Messaging, Canvas 2D, Web Workers, Web Storage, …
- HEVC video
- Subtitles for broadband delivered content
- Advert insertion into VoD content (Support of multiple Video Elements)
- **Companion Screen Features:**
  - Launching a Companion Screen Application from the HbbTV 2.0 App on the TV
  - Application to Application (App2App) Communication
  - Remotely Launching an HbbTV 2.0 Application from an App running on a companion device e.g. Tablet or Smartphone
  - Synchronizing Applications and Content across devices. This allows an app on the smartphone or tablet to synchronize with the video being presented by the TV or STB

Source: http://hbbtv.org

HBBTV2.0 CS: LAUNCHING A COMPANION SCREEN APPLICATION

- The HbbTV 2.0 App uses the HbbTVCSManager interface to discover CS devices with a running Launcher and to install or Launch native or web CS Apps
- Communication protocol between HbbTVCSManager and Launcher is not part of the HbbTV 2.0 Spec



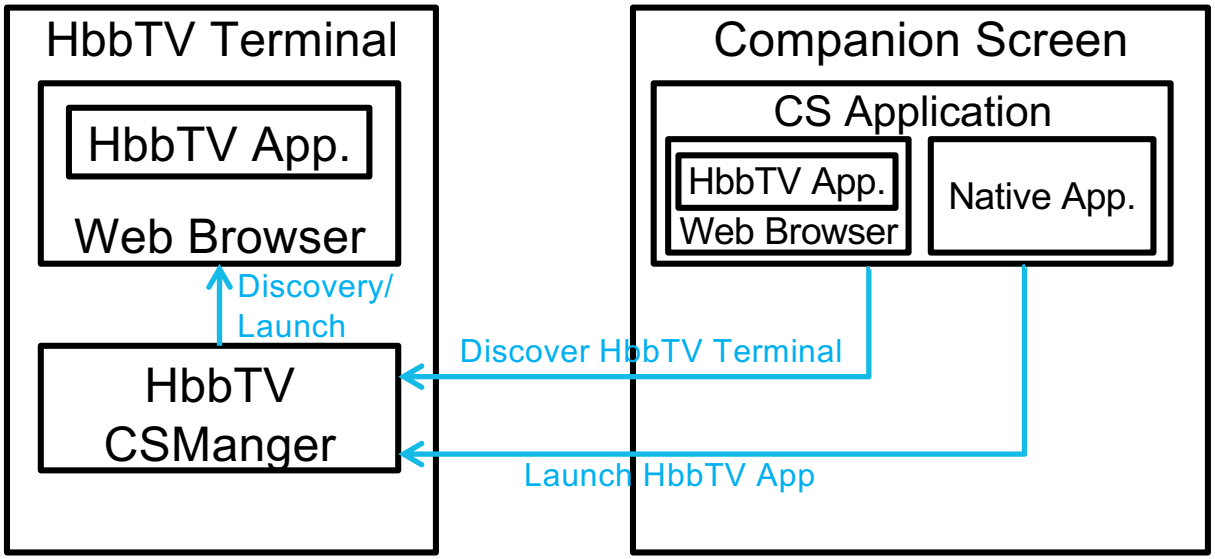
- The HbbTV 2.0 App obtains platform information, including the OS that the Companion Screen is running.

http://hbbtv.org/pages/about_hbbtv/HbbTV_specification_2_0.pdf

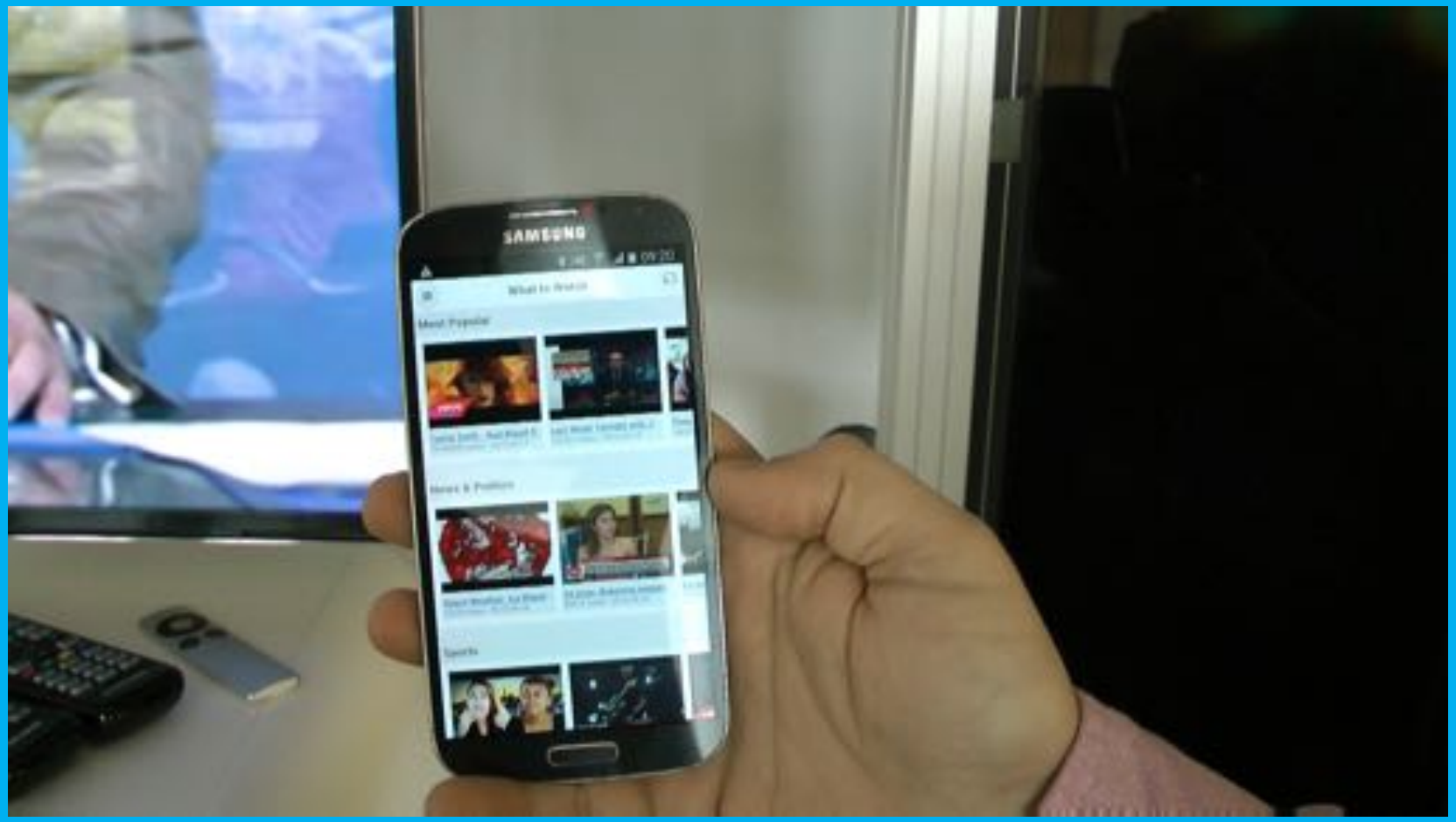

Fraunhofer
FOKU
Intro
The
Fut

HBBTV2.0 CS: LAUNCHING AN HBBTV 2.0 APPLICATION

- Launch Broadcast independent HbbTV App
- HbbTV Terminal exposes itself as DIAL App “HbbTV”
- Companion Screen App discovers (using SSDP) available DIAL servers and checks if “HbbTV” is available
- Companion Screen App launches the HbbTV App by sending HTTP POST request to the DIAL Endpoint
- The body of the HTTP POST request is an XML that contains information about the HbbTV App to launch

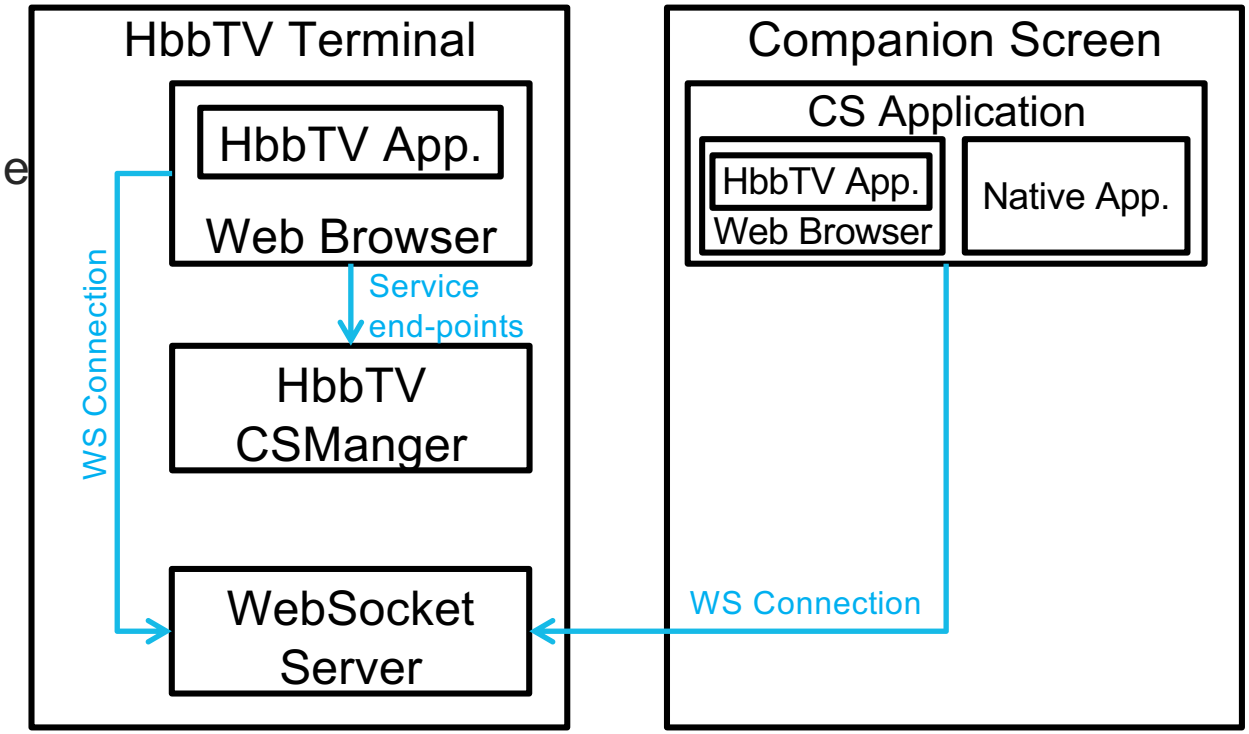


http://hbbtv.org/pages/about_hbbtv/HbbTV_specification_2_0.pdf


SAMSUNG

HBBTV2.0 CS: APPLICATION TO APPLICATION COMMUNICATION

- The HbbTV Terminal runs a Websocket Server
- App2App end-point contains the address of the WS Server
- The HbbTV App requests the App2App end-point from the HbbTVCSManager
- The CS App gets the App2App end-point after DIAL App Launch
- The HbbTV and CS Apps establish WS connections

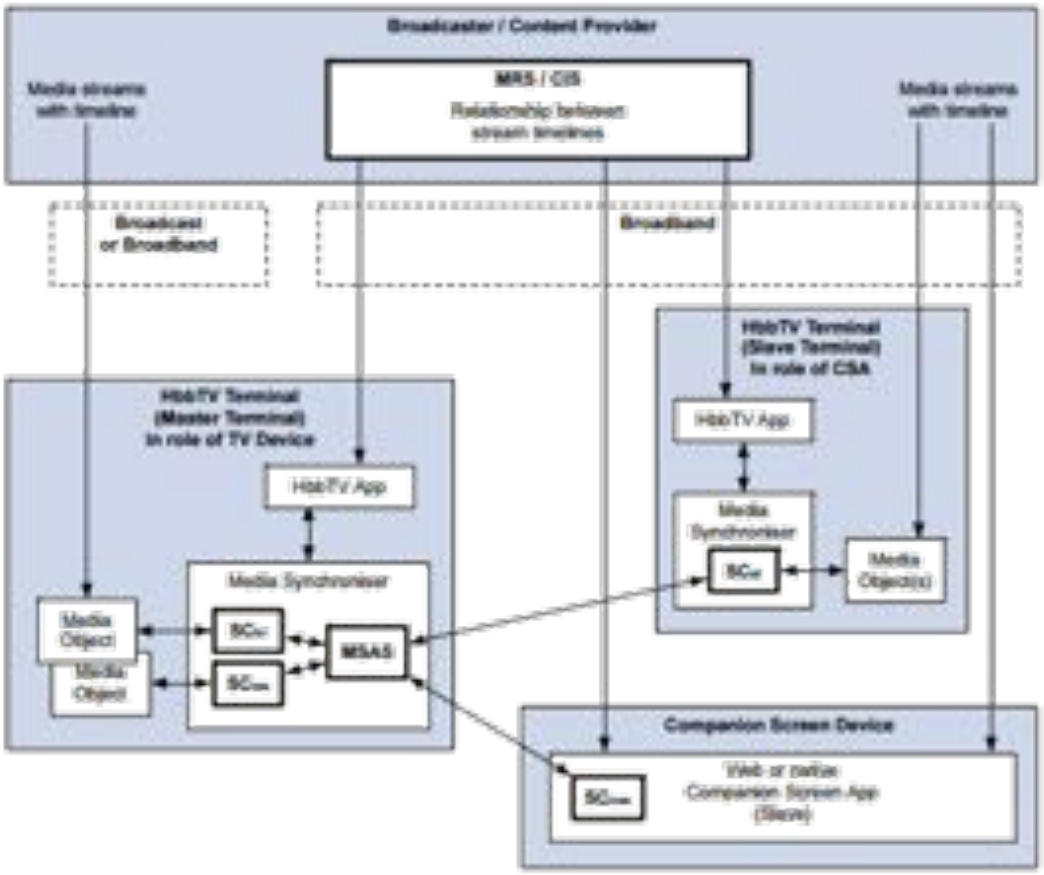


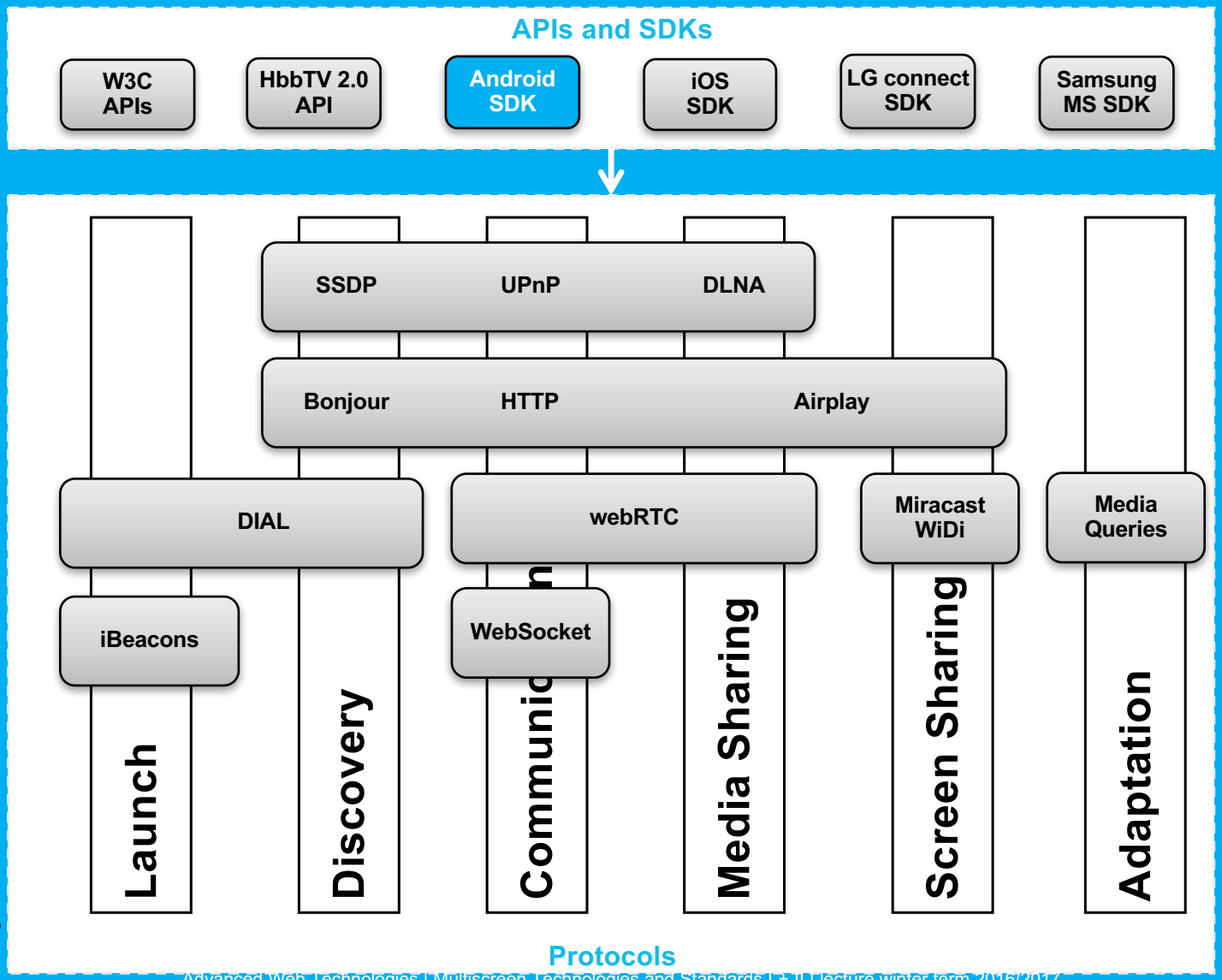
http://hbbtv.org/pages/about_hbbtv/HbbTV_specification_2_0.pdf

- The Websocket Server acts as relay and pairs the WS connections on the same channel

HBBTV2.0 CS: SYNCHRONIZING APPLICATIONS AND CONTENT

- **Multi-stream synchronization**: defines how streams delivered via broadcast and broadband can be synchronized
- **Synchronizing applications and content across devices**: This allows an app on the smartphone or tablet to synchronize with the video being presented by the TV or STB based on the DVB Companion Screen ("CSS") specification.
- Application and content synchronization builds on the "TEMI" timeline recently standardized in MPEG




APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
WebSocket
Launch
Discovery
Communic
Media Sharing
Screen Sharing
Adaptation
Protocols

Source: http://developer.android.com/reference/android/app/Presentation.html

ANDROID SDK

- Android 4.2+ allows apps to display unique content on additional screens that are connected to the user's device over either a wired connection or Wi-Fi.
- To create unique content for a secondary display, the app developer need to extend the **Presentation** class and override the **onCreate()** method.
- The **onCreate()** method of the Presentation class is very similar to the **onCreate()** method of the **Activity** class.
- Use the **DisplayManager** or **MediaRouter** APIs to detect Secondary Disp
- **DisplayManager** provides fun be connected at once.
- **MediaRouter** can be used ins default display for presentatio

```
MediaRouter mediaRouter = (MediaRouter) context.getSystemService(Context.MEDIA_ROUTER_SERVICE);
MediaRouter.RouteInfo route = mediaRouter.getSelectedRoute();
if (route != null) {
    Display presentationDisplay = route.getPresentationDisplay();
    if (presentationDisplay != null) {
        Presentation presentation = new MyPresentation(context, presentationDisplay);
        presentation.show();
    }
}
```

ANDROID SDK

- Android NSD Manager → http://developer.android.com/training/connect-devices-wirelessly/nsd.html
- Native mDNS support → No need for 3rd party frameworks

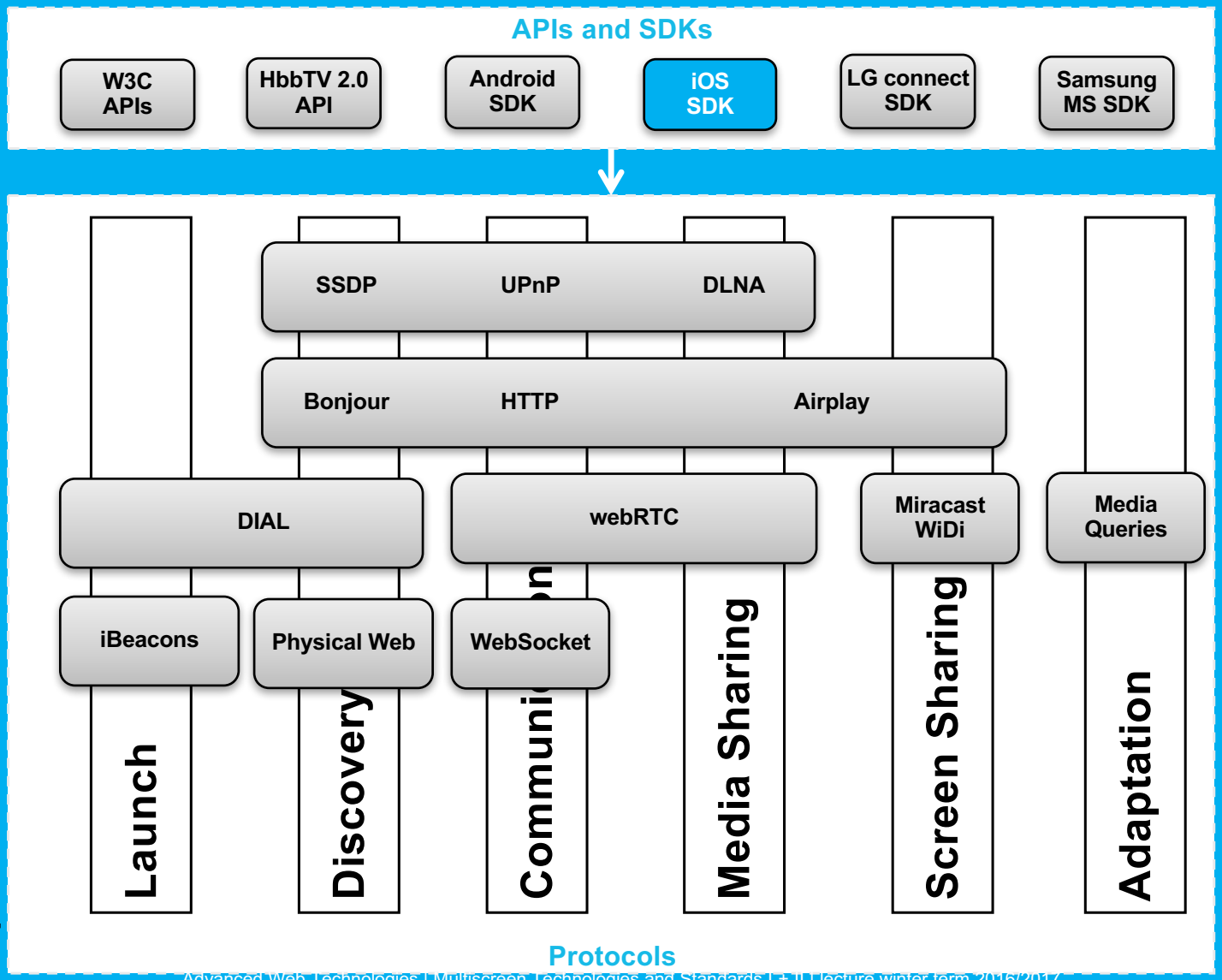
```
public void registerService(int port) {
    NsdServiceInfo serviceInfo  = new NsdServiceInfo();
    serviceInfo.setServiceName("NsdChat");
    serviceInfo.setServiceType("_http._tcp.");
    serviceInfo.setPort(port);

    mNsdManager = Context.getSystemService(Context.NSD_SERVICE);

    mNsdManager.registerService(
            serviceInfo, NsdManager.PROTOCOL_DNS_SD, mRegistrationListen
}
```

Advertisement

Discovery

```
mNsdManager.discoverServices(
    SERVICE_TYPE, NsdManager.PROTOCOL_DNS_SD, mDiscoveryListener);
```

```
@Override
public void onServiceFound(NsdServiceInfo service) {
    // A service was found!  Do something with it.
    Log.d(TAG, "Service discovery success" + service);
    if (!service.getServiceType().equals(SERVICE_TYPE)) {
        // Service type is the string containing the protocol and
        // transport layer for this service.
        Log.d(TAG, "Unknown Service Type: " + service.getServiceType());
    } else if (service.getServiceName().equals(mServiceName)) {
        // The name of the service tells the user what they'd be
        // connecting to. It could be "Bob's Chat App".
        Log.d(TAG, "Same machine: " + mServiceName);
    } else if (service.getServiceName().contains("NsdChat")){
        mNsdManager.resolveService(service, mResolveListener);
    }
}
```


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

AIRPLAY & MULTISCREEN

- Screens can be connected to an iOS Device:
  - via Airplay over WiFi
  - via Lightning to HDMI Adapter

Detect added Screens on App launch

```
if ([[UIScreen screens] count] > 0) {
        for (UIScreen *screen in [UIScreen screens]) {
            if (![screen isEqual:[UIScreen mainScreen]]) {
                [self addScreen:screen]; // new screen
                break;
            }
        }
}
```

AIRPLAY & MULTISCREEN

Detect added Screens on the fly
Add Observer to OS Notifications

Adding Screen:

```
[[NSNotificationCenter defaultCenter] addObserver:self
                                 selector:@selector(screenDidConnect:)
                                 name:UIScreenDidConnectNotification
                                 object:nil];

- (void)screenDidConnect:(NSNotification *)notification
{
   UIScreen *tvScreen = (UIScreen *)[notification object];
   [self addScreen:tvScreen];
}
```

Removing Screen:

```
[[NSNotificationCenter defaultCenter] addObserver:self
                                 selector:@selector(screenDidDisconnect:)
                                 name:UIScreenDidDisconnectNotification
                                 object:nil];

- (void)screenDidDisconnect:(NSNotification *)notification
{
   UIScreen *tvScreen = (UIScreen *)[notification object];
   [self removeScreen:tvScreen];
}
```

COMMUNICATION BETWEEN IOS DEVICES VIA GAME CENTER

Setup Gamecenter and listen to Peers and their states

```
-(void)setupGamecenter
{
        self.gkSession = [[GKSession alloc] initWithSessionID:@“DomainName“
                                displayName:@”DeviceName”  sessionMode:GKSessionModePeer];
        self.gkSession.delegate = self;   // set delegate (listener)
}

-(void)session:(GKSession *)session peer:(NSString *)peerID didChangeState:(GKPeerConnectionState)state
{
   switch (state)
   {
      case GKPeerStateAvailable:                        // Peer available, save
                  break;
      case GKPeerStateUnavailable:          // Peer unavailable, remove
                  break;
      case GKPeerStateConnected:            // Peer connected, save
                  break;
      case GKPeerStateDisconnected:         // Peer disconnected, remove
                  break;
      case GKPeerStateConnecting:           // Peer connecting
                  break;
   }
}
```

Other Delegate Methods

```
-(void)session:(GKSession *)session didReceiveConnectionRequestFromPeer:(NSString *)peerID;
-(void)session:(GKSession *)session connectionWithPeerFailed:(NSString *)peerID withError:(NSError *)error
-(void)session:(GKSession *)session didFailWithError:(NSError *)error
```

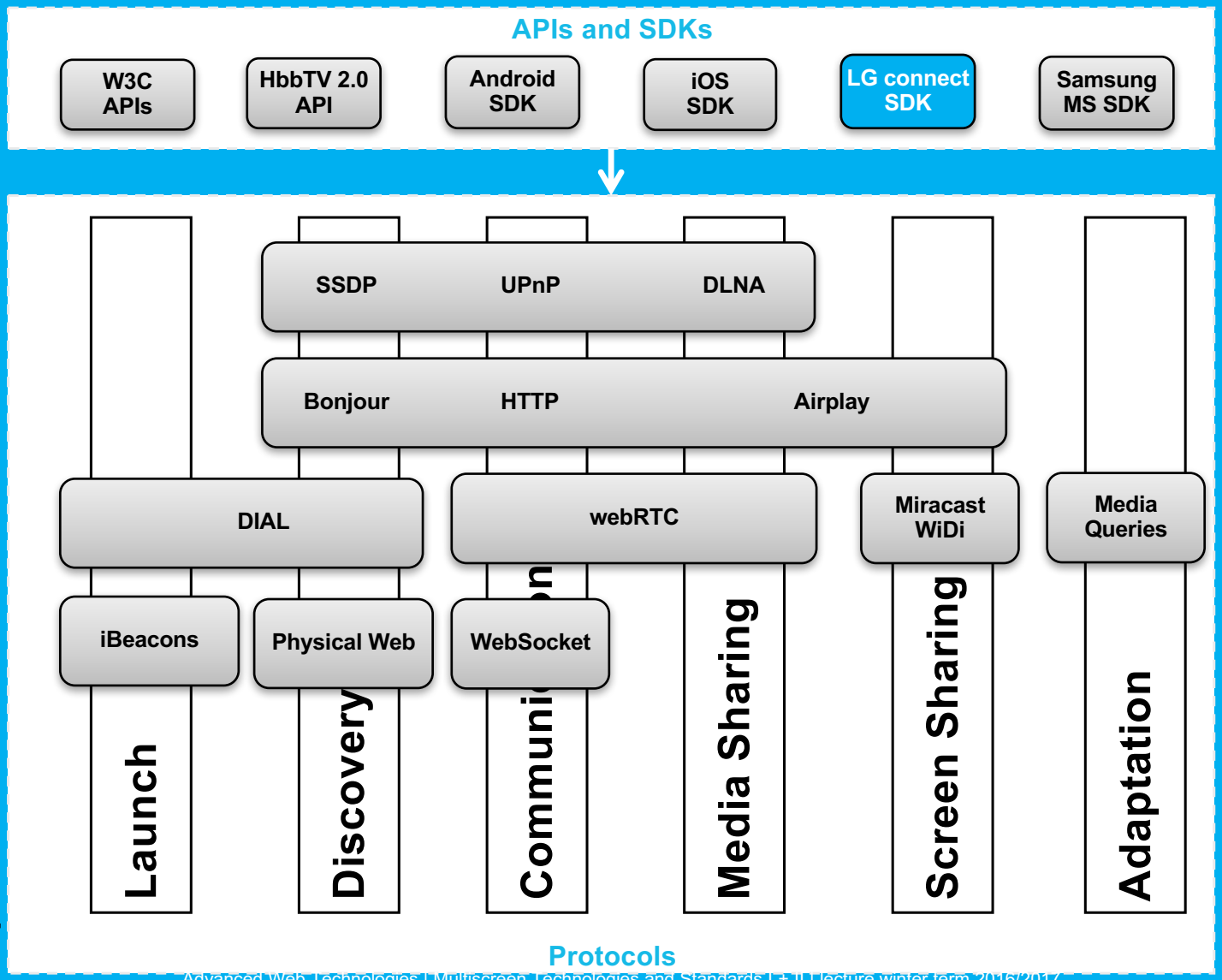
COMMUNICATION BETWEEN IOS DEVICES VIA GAME CENTER

Send data:

```
-(void)sendString:(NSString *)text toPeers:(NSArray *)peers
{
    NSData *data = [text dataUsingEncoding:NSUTF8StringEncoding];  // String to Bytes
    [self.gkSession sendData:data toPeers:peers withDataMode:GKSendDataReliable error:nil];
}
```

Receive data:

```
-(void)receiveData:(NSData *)data fromPeer:(NSString *)peer inSession: (GKSession *)session context:(void *)context
{
    // Bytes to String
    NSString *text = [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding];
}
```


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

LG CONNECT SDK

- Multiscreen SDK supports multiple TV Platforms
- It is an open source framework that connects mobile apps with multiple TV platforms.
- APIs for 3 client Platforms

Android Cordova iOS

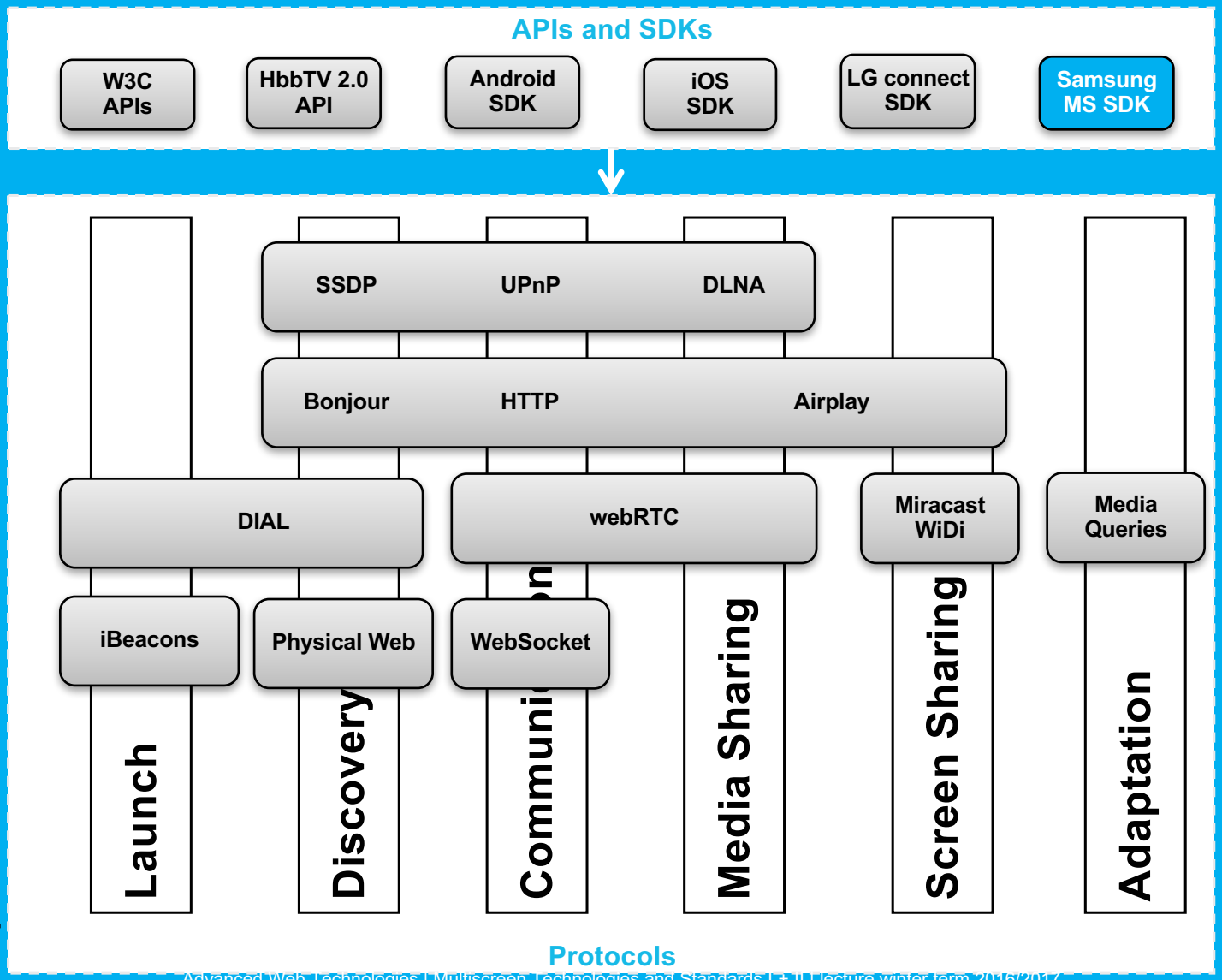
- Well documented
- Sample Apps for all client Platforms available
- Main features:
  - Discovery, App Launch, Web App Beaming, Media Sharing, TV Control, Media Playback Control, System Controls

Source: http://connectsdk.com/

LG CONNECT SDK FEATURES (V. 1.4.4)

Category	Feature	LG Smart webOS '14	Chromecast	Apple TV	Roku	Fire TV	LG Smart TV '13	LG Smart TV '12	DIAL	Sonos Speakers	Xbox One	LG Music Flow speaker
Apps	Beam Web App	✓	✓	✓								
	Launch My app	✓	✓		✓	✓	✓	✓	✓			
	Get list of installed apps	✓			✓		✓	✓				
	Mobile app to TV app messaging	✓	✓	✓								
	Deeplink into app store	✓			✓		✓					
	Beam Youtube	✓	✓		✓	✓	✓	✓	✓		✓	
Media	Beam video	✓	✓	✓	✓		✓	✓			✓	
	Beam audio	✓	✓	✓	✓		✓	✓		✓	✓	✓
	Beam photo	✓	✓	✓	✓		✓	✓			✓	
	Media pause	✓	✓	✓	✓		✓	✓		✓	✓	✓
	Media stop	✓	✓	✓	✓		✓	✓		✓	✓	✓
	Get media duration	✓	✓	✓			✓	✓		✓	✓	✓
	Seek media	✓	✓	✓			✓	✓		✓	✓	✓
	Play State Subscription	✓	✓				✓	✓		✓	✓	✓
	Media Info Subscription		✓				✓	✓		✓	✓	✓

Category	Feature	LG Smart webOS '14	Chromecast	Apple TV	Roku	Fire TV	LG Smart TV '13	LG Smart TV '12	DIAL	Sonos Speakers	Xbox One	LG Music Flow speaker
System Controls	Show toast alert	✓										
	Keyboard input	✓			✓		✓	✓				
	5-way controls	✓			✓		✓	✓				
	Mouse controls	✓					✓	✓				
	Input selector	✓					✓	✓				
	Power off device	✓					✓	✓				
TV Controls	Volume up/down	✓	✓				✓	✓		✓	✓	✓
	Set volume	✓	✓					✓		✓	✓	✓
	Tuner channel control	✓					✓	✓				
	Volume Subscription	✓	✓				✓	✓		✓	✓	✓
Playlist	Beam Playlist									✓		
	Play Next									✓		
	Play Previous									✓		
	Jump To Track									✓		


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Media Sharing
Screen Sharing
Adaptation
Protocols

SAMSUNG MULTISCREEN SDK

How Does Samsung Multiscreen Work?

Discover

A mobile application can use the API to find a Samsung SmartTV Device that it can connect to.

Launch

Once a device has been discovered, the mobile device can launch an Application on your SmartTV.

Communicate

Once launched, the SmartTV and mobile devices can communicate through a Channel.

Source: http://multiscreen.samsung.com/

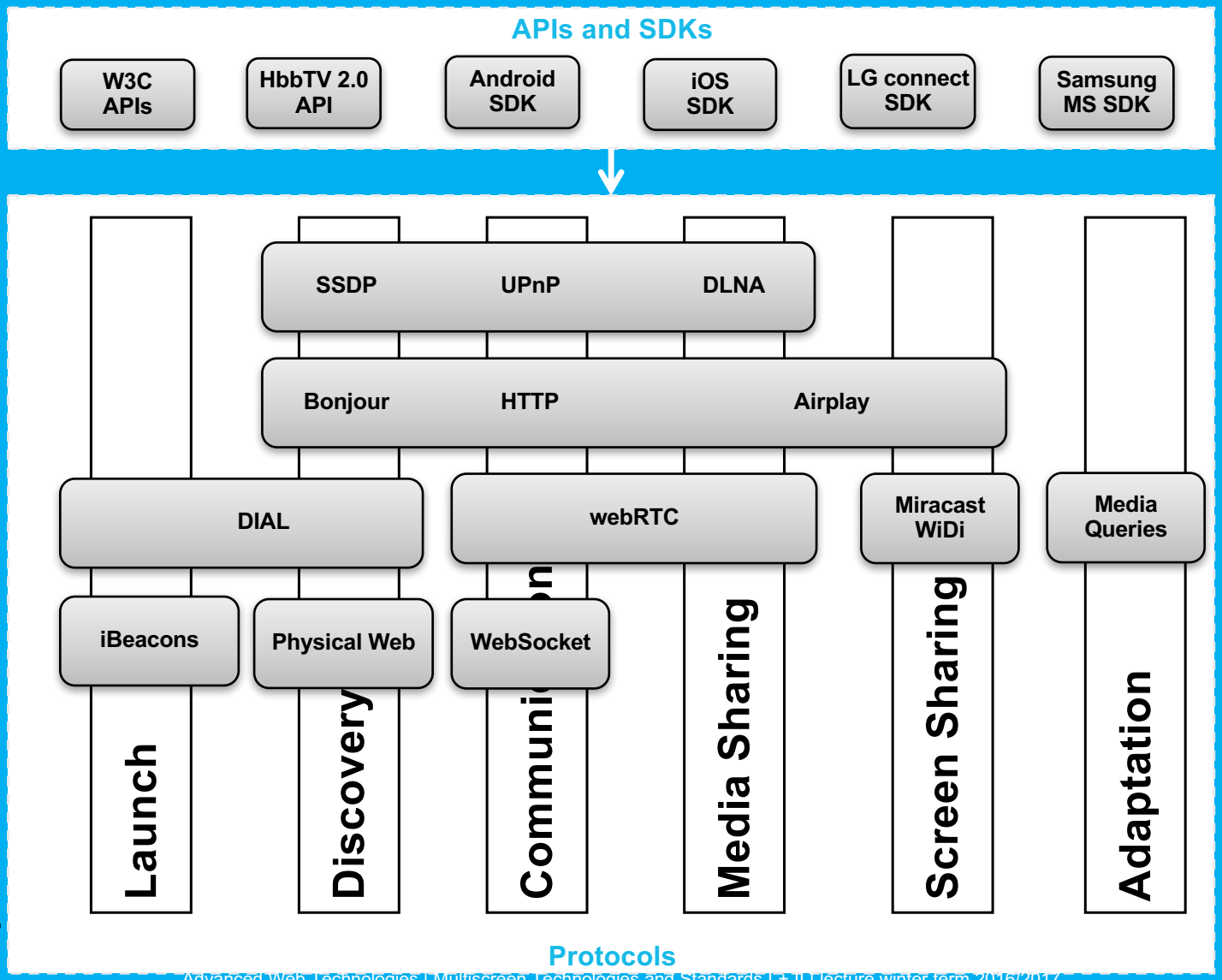
SAMSUNG MULTISCREEN SDK

iOS, Android and JavaScript libraries

Source: http://multiscreen.samsung.com/

SAMSUNG MULTISCREEN SDK 2.0

- Samsung announced to publish the upcoming version 2.0 of the Multiscreen SDK as open source including Android, iOS and JavaScript libraries.
- Samsung is also planning to publish the raw communication protocols so that developers can create additional libraries and components for community
- New features:
  - Binary channels
  - Launch hosted web applications
  - Launch any native application by id
  - …


APIs and SDKs
W3C APIs
HbbTV 2.0 API
Android SDK
iOS SDK
LG connect SDK
Samsung MS SDK
SSDP
UPnP
DLNA
Bonjour
HTTP
Airplay
DIAL
webRTC
Miracast WiDi
Media Queries
iBeacons
Physical Web
WebSocket
Launch
Discovery
Communi
Media Sharing
Screen Sharing
Adaptation
Protocols

Q&A

THANK YOU FOR YOUR ATTENTION