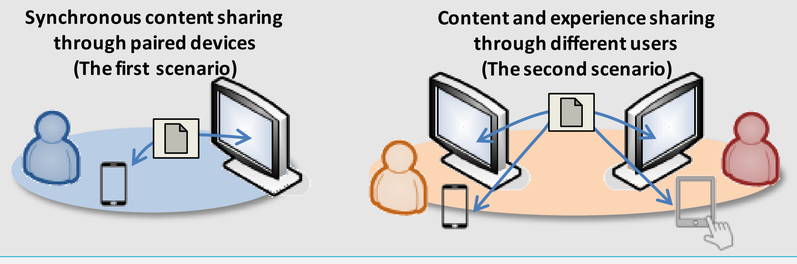
Cloud session maintenance to synchronise HbbTV applications and home network devices

Mikel Zorrilla, Iñigo Tamayo, Angel Martin and Igor G. Olaizola {mzorilla@vicomtech.org}

Digital TV & Multimedia Services, Vicomtech-ik4
Donostia-San Sebastián, Spain

Introduction

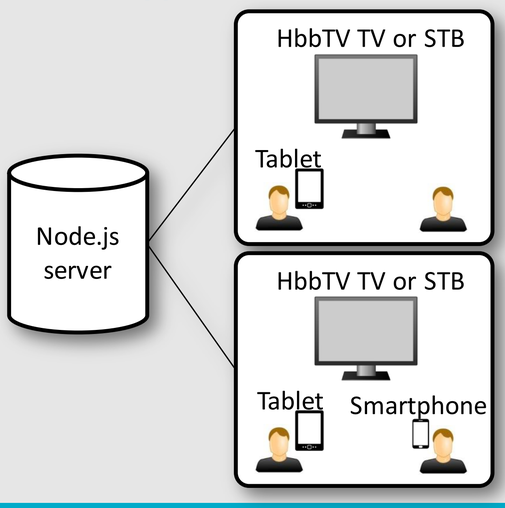
- User experience around TV has changed:
  - 2nd screen, multi-screen, multi-device experiences
- Smart TVs are a commercial success worldwide
- Heterogeneous TV platforms:
  - CE manufacturer created vertical platforms (Samsung Smart TV, LG Smart TV)
  - Proprietary horizontal approaches (Google TV, Apple TV)
  - Standard approaches involving broadcasters: **HbbTV** in Europe
- HbbTV v.1.1 and v.1.5 no specifications for multi-screen
- **Two main scenarios have been addressed in this work:**



End to end solution to synchronise HbbTV apps (v.1.1 & v.1.5) with 2nd screen devices through standard HTML5 apps
- By using cloud session maintenance in the server side
- Session variables to identify & pair different users and their devices
- Dynamic context adaptation

Cloud session maintenance architecture

- **Event-driven server architecture** with 3 main modules:
  - **Node.js server:** Manages the *cloud session maintenance* and apps' behaviour
  - **HbbTV app**: for the Smart TV
  - **HTML5 app**: for the 2nd screen devices (tablets, smartphones, laptops, etc.)



Node.js is a framework for developing high-performance concurrent programs using an **event-driven** programming model.

Enables to create **bi-directional communication** channels.

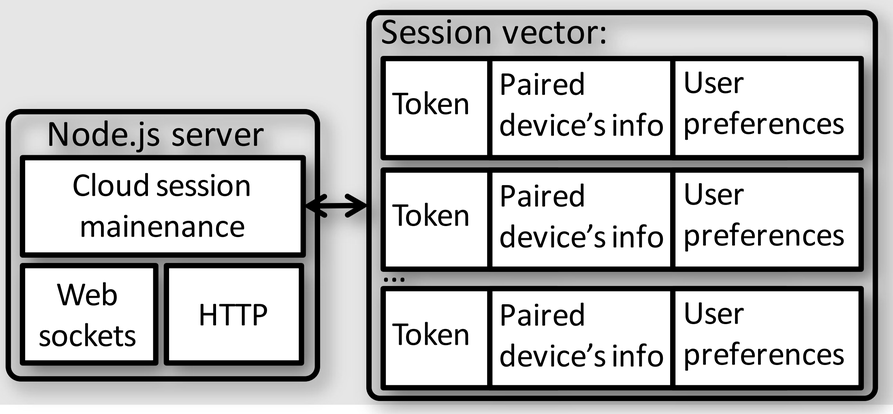
It receives **user interaction** events and generates reaction commands to **push other apps**.

System Workflow

- Service starts with an HbbTV app linked to a broadcast live content
- Each TV app is linked to a unique *session vector* in the server:
  - A unique session token
  - Information about already synchronised mobile devices
  - User preferences
- A QR code for pairing (REST-based information)
- In a multi-device experience the service adapts to the new context: TV and mobile app change their behaviour

Server components

- **HTTP Server**: Delivery of the HTML-based apps (HbbTV & HTML5)
- **Websocket module**: Establish bi-directional communication between the server and the end-devices
- ***Cloud session maintenance* module**: maintaining and updating a *session vector* for each HbbTV app; creating the commands and events depending on the multi-device behaviour of the service; and managing the Websocket module and communications.



Advantages of the architecture

- Local network protocols such as UPnP or DLNA are not required
  - TV & 2nd screen devices do not require to be connected to the same network
  - HbbTV device does not require local communication capabilities
- Allows multi-user experience sharing
  - Users can be geographically distributed

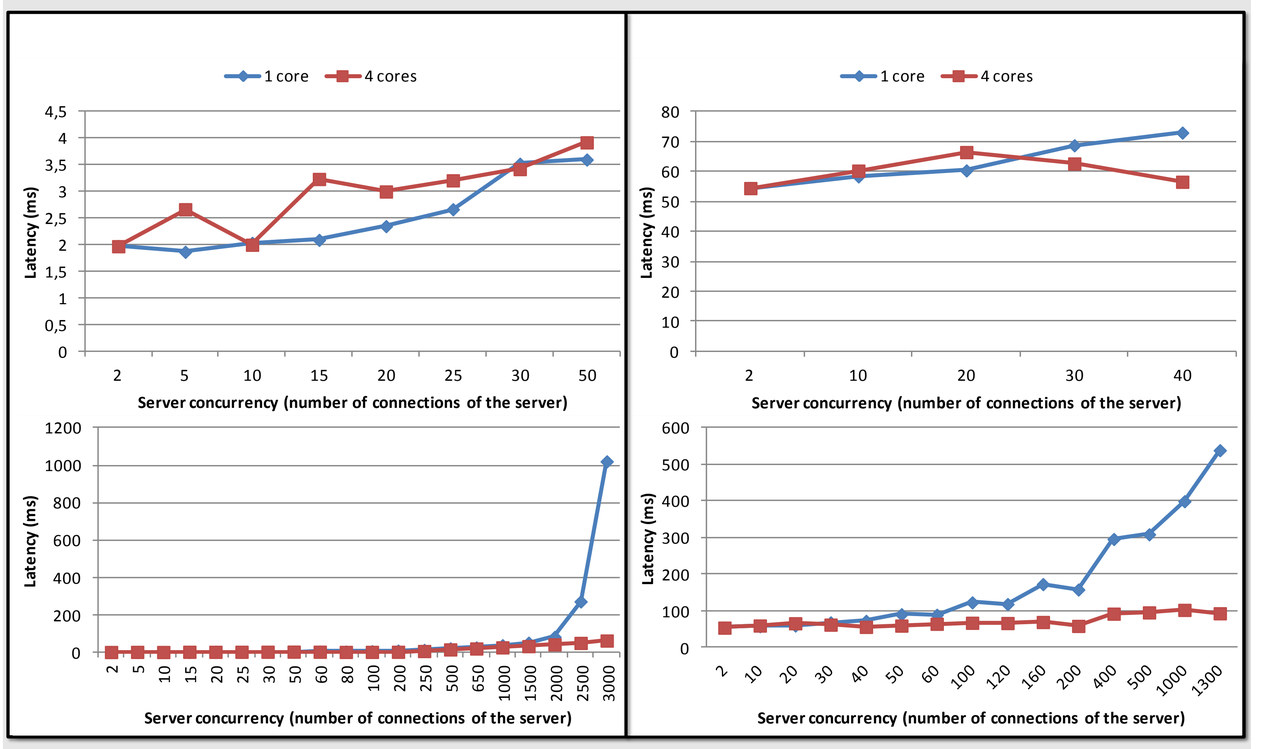
Risks of the architecture

- Latency and concurrency
  - Cloud-based solutions must keep latency performance even under stress conditions with massive users performing concurrent requests
- Security
  - Not afforded because it can be addressed on top of this architecture

Validation experiments and results

- Detailed latency and concurrency analysis of the Node.js-based server
- Experiments carried out with Websockets and AJAX (an alternative for those devices which are not compatible Websockets, e.g. HbbTV v1.1)
- Variable number of CPU cores in the server to evaluate the performance.
- **Results**
- Best latency score 1.97 ms

Websocket | **AJAX**



Conclusions

- The **latency is suitable** for most of the 2nd screen services
- Websockets provide a much better latency
  - AJAX-based solution could be also adequate
- Incrementing the number of simultaneous connections there is a ***stable zone* where the latency value keeps over quality threshold**.
  - Due to the fact that all sessions can be concurrently managed, **the scalability of the system straight forward** and can be achieved ensuring the bandwidth requirements and computing power,.
- The server architecture **ready for a multi-device ecosystem of many users interacting concurrently**

This paper provides a suitable **end to end solution** based on a **cloud session maintenance** architecture that allows **synchronised 2nd screen services** with Connected TVs and mobile devices.

This architecture performs a solution over already commercialised devices such as **HbbTV compatible** Connected TVs and OS-based smartphones and tablets with a standard mobile **web browser**.