A proposal for a Web Standards based AR RM
A Position Paper for the ARStandards.org Meeting, Basel October 2011

Rob Manson, Managing Director, MOB
roBman@mob-labs.com

Abstract
This paper proposes a Web Standards based Reference Model for Augmented Reality applications. This work is a logical extension of the initial "Patterns of Interest" model proposed at the Seoul 2010 AR Standards workshop[1] and integrates the state of the art in terms of Web Standards developments.

All of this functionality is now possible in specialised builds of the mainstream web browsers and are likely to be common in the marketplace within the next 12 months.

It is hoped that by providing a common model that shows how these disparate standards can be integrated, the overall diffusion of web based AR applications will develop more quickly and efficiently.

Introduction
In 2010, the Patterns of Interest proposal outlined a goal that is just now starting to be realised:

It aims to extend the existing web to be sensor aware and automatically event driven while encouraging the presentation layer to adapt to support dynamic spatialised information more fluidly.[1]

A wide number of developments across the W3C, WHATWG and Khronos group have brought us to the point that this vision can now be realised in a working system.


The proposed Reference Model
This model is divided into 5 key layers:

- User
- Interfaces
- APIs
- Processor/Application
- User Agent

Here is a brief description of each of these layers and how they relate to each other.

User
This is the person who is situated in the Reality that is being Augmented. They can impact the Sensors through their Interfaces and can receive output via Interfaces like the Screen and the Speakers.

Interfaces
These are digital components that link the Augmented Reality application to the Real world. They can be connected as both Input and Output. Examples of Input Interfaces are Touch Sensors, GPS, Accelerometer, Camera or Microphone. Examples of Output Interfaces are Displays and Speakers.

APIs
These Application Programming Interfaces allow the Interfaces to be connected to the Processor in a standardised way. The sensor based APIs include GeoLocation, Orientation and Sensor. The HTMLMediaElements[14] provide the <audio>[15] and <video>[16] elements that can be interacted with via Stream based frameworks like the MediaStream Processing API[9]. This includes capturing and analysing inbound video streams[15] as well as capturing, analysing and generating[16] audio streams. The getUserMedia()[7] method now provides access to both local streams from cameras and microphones as well as remote streams for peer-to-peer communication. Output video streams can also be generated via the <canvas>[17] element using 2d[18] and 3d[19] contexts as well as <audio> and <video> playthrough.

Processor/Application
This is a javascript[20] based application that would implement a range of standard components including the ability to host sessions as well as retrieve, store and manage filter patterns, points of interest and digital assets. This would also need to be able to dynamically load modules for specialised pattern processing which may also include WebCL integration. It would also manage detection of local device capabilities and manage web workers[21] and network connections through xhr2[22] and web sockets[23].

User Agent
This is the operating environment the Augmented Reality application is running within. It consists of the underlying device, it's host operating system, the navigator and the DOM this application is running within.
Major breakthroughs in the APIs

Here is a more detailed description of the biggest changes that are enabling this new reference model. The first is the introduction of the getUserMedia() method and the underlying MediaStream framework. The getUserMedia() method is an outcome of work in the W3C’s DAP and WebRTC Working Groups and it allows a JavaScript application to access audio and video streams from remote and local cameras, microphones, etc.

A proposed extension on top of this is Robert O'Callahan's MediaStream Processing API that opens the streams up to allow us to access the data within each of the MediaStreamTracks[24]. At the moment the pixel data of video frames can also be accessed using drawImage(video)/getImageData()[25] via a <canvas> element's 2d graphic context.

The GeoLocation API has now stabilised and accessing or monitoring a device's lat/lon/alt is now widely available on mobile web browsers. The Orientation API that also provides access to Orientation and Acceleration data is now also fairly widely available.

And an interesting generic solution called Sensor API has been proposed that spans the Orientation sensors as well as Temperature, Ambient Light, Magnetic Field, Proximity and Atmospheric Pressure.

Video output

The <canvas> based 2d graphics context is now widely available making this an effective and relatively efficient way to render images and moving video.

The <canvas> based 3d graphics context (e.g. WebGL) is now quite widely accessible and often now doesn't need any user configuration at all.

WebCL is now working on both the webkit and gecko platforms and can relatively easily be injected to improve rendering performance with some examples showing 500% to 1000% improvement.

Processor/Application

To validate this reference model we have created a working prototype of this JavaScript based application that integrates a range of these standard components.

In order to manage security and to allow customisation this model is designed around a session based perspective.

At the center we have also placed the collection of Points of Interest. This is designed to work with the W3C POI WG's[26] proposed data model and looks to extend that with an AR Profile and support for non-geographic anchors as an alternative to the POI location. The processor looks after retrieving these POIs from remote services and managing and caching them in the local system.

The rules for which patterns of Sensor Data from the Interfaces will drive the conversion of certain POI into AR Projections is another collection that needs to be retrieved, managed and stored. These AR Projections are the 2d and 3d images that are rendered over the top of the video playback to augment this reality.

Then we have also designed in support for specialised pattern processing modules which may just utilise standard JavaScript or even more advanced WebCL integration.

JSARToolkit[27], one example module is a JavaScript port of the ARToolKit fiducial marker AR libraries and it's performance is quite impressive. Many new algorithms will become available over time and once this Web Standards based AR Reference Model is commonly available then upgrading to a new form of Visual Processing or Pattern Recognition will simply involve clicking on a link or visiting a new web address.

And in order to make sure this application can operate effectively on the local device the Processor would also manage the detection of local device capabilities. To ensure that the User Experience is fluid and non-blocking it would also manage the use of web workers and background network connections through xhr2 and web sockets.

The User Agent

This is generally thought of as the Web Browser and these APIs and frameworks are now working in leading edge versions of the major browser engines. They are likely to be widely available for Opera, Mozilla, Safari and Chrome over the next 6-12 months.

With these browsers gaining wider access to the native device Interfaces we expect to see more vertically integrated device strategies pop up such as that proposed in the "Boot to Gecko"[28] initiative from Mozilla.

In Summary

What was only a speculative possibility a year ago is now a technical reality and well on track to quickly becoming a market revolution. In 2009, when Saqoosha released FLARToolkit[29] (a Flash based port of ARToolKit) we saw an explosion in the awareness of and access to Augmented Reality because it could now be delivered through the web.

With the realisation of this Web Standards based AR Reference Model we will see a much bigger and faster explosion of adoption and awareness.

Today the web is mainstream and it is also AR's path to becoming mainstream too.[30]
Figure 1. Web Standards based AR Reference Model
References

[1] Patterns of Interest proposal
AR Standards Workshop, Seoul, 2010

http://www.w3.org/2008/geolocation/

[3] GeoLocation API
http://www.w3.org/TR/2010/CR-geolocation-API-20100907/

[4] Orientation API
http://dev.w3.org/geo/api/spec-source-orientation.html

http://www.w3.org/2009/dap/

http://www.w3.org/2011/04/webrtc/

[7] getUserMedia() method

[8] Stream API

[9] MediaStream Processing API

[10] Sensor API
http://dev.w3.org/2009/dap/system-info/Sensors.html

http://www.khronos.org/webgl/

[12] Khronos WebCL
http://www.khronos.org/webcl/

http://www.w3.org/community/declarative3d/

[14] HTMLMediaElements
http://www.w3.org/TR/html5/video.html#htmlmediaelement

[15] <audio> element
http://www.w3.org/TR/html5/video.html#audio

[16] <video> element
http://www.w3.org/TR/html5/video.html#video

[17] <canvas> element

[18] <canvas> 2d graphics context
http://www.whatwg.org/specs/web-apps/current-work/multipage/the-canvas-element.html#2dcontext

[19] <canvas> 3d graphics context
https://www.khronos.org/registry/webgl/specs/1.0/#2

[20] Javascript / ECMAScript
http://en.wikipedia.org/wiki/ECMAScript

[21] Web Workers
http://dev.w3.org/html5/workers/

[22] xhr2 / XMLHttpRequest2
http://www.w3.org/TR/XMLHttpRequest2/

[23] Web Sockets
http://dev.w3.org/html5/websockets/

http://www.whatwg.org/specs/web-apps/current-work/multipage/video-conferencing-and-peer-to-peer-communication.html#mediastreamtrack

[25] drawImage(video)/getImageData() pipeline
https://developer.mozilla.org/En/Manipulating_video_using_canvas

[26] W3C Points of Interest Working Group
http://www.w3.org/2010/POI/

[27] JSARToolkit libraries
http://fhtr.org/JSARToolkit/

[28] Mozilla’s Boot to Gecko/Web
https://wiki.mozilla.org/B2G

[29] FLARToolkit libraries
http://saqoo.sh/a/en/flartoolkit/start-up-guide

[30] A test framework for this new web
http://isweb3here.com