Embedding Openness in the Connected Car

Matt Jones
Jaguar Land Rover

25 March 2015

Customer Expectations

Customer Expectations

- Multimodal HMI
  - HD Displays
  - Improved Voice Control
- Connected World
  - Telematics
  - Connected Navigation (augmented offboard)
  - App Store
- Media Management – Online & Local

- This is a fully featured home entertainment network…

Customer Expectations

- Multimodal HMI
  - HD Displays
  - Improved Voice Control
- Connected World
  - Telematics
  - Connected Navigation (augmented offboard)
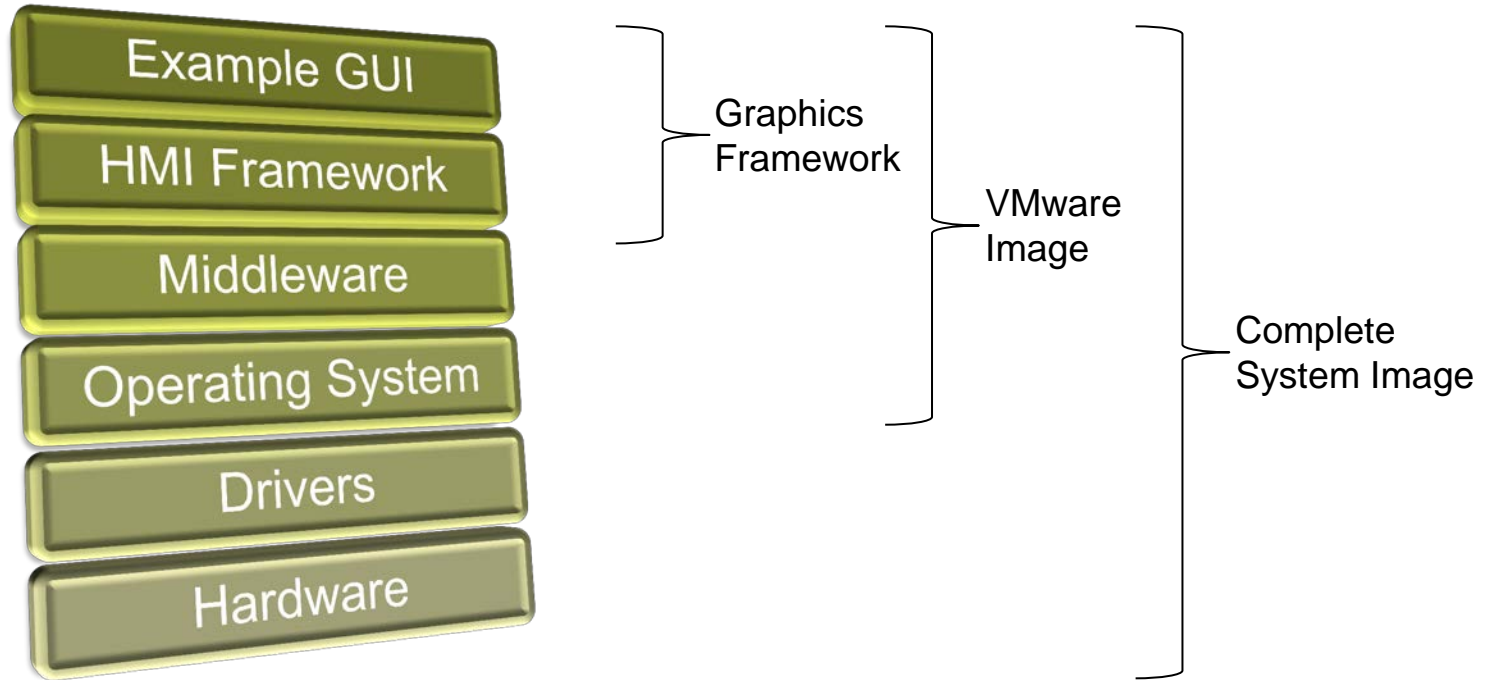 - App Store
- Media Management – Online & Local

Regular Updates and Additions

- This is a fully featured home entertainment network…

Developer Expectations

Easy to Use Platform

| Stack	Grouping
Example GUI	Graphics Framework
HMI Framework	
Middleware	VMware Image
Operating System	
Drivers	Complete System Image
Hardware	

The POC Platform

- JLR worked with the Automotive Grade Linux project to create a base platform that others can build on:

  - Tizen 3.0 platform
  - GENIVI compliant
  - HTML5 based application environment using wgts
  - Server backend to push / pull apps

  - Complete documentation
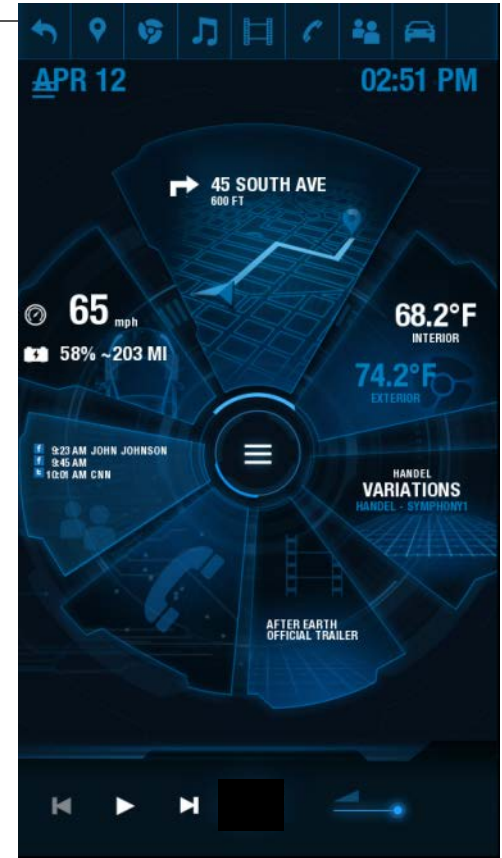
Progress

- See we address link:
  - *http://automotive.linuxfoundation.org/agl-demonstrator*

- Today you can download:
  - HTML5 framework – complete with sample user interface
  - Download the Virtual Machine image
  - Pre-built image – with detailed instructions to get up and running

- Over 4000 downloads in just over four weeks!
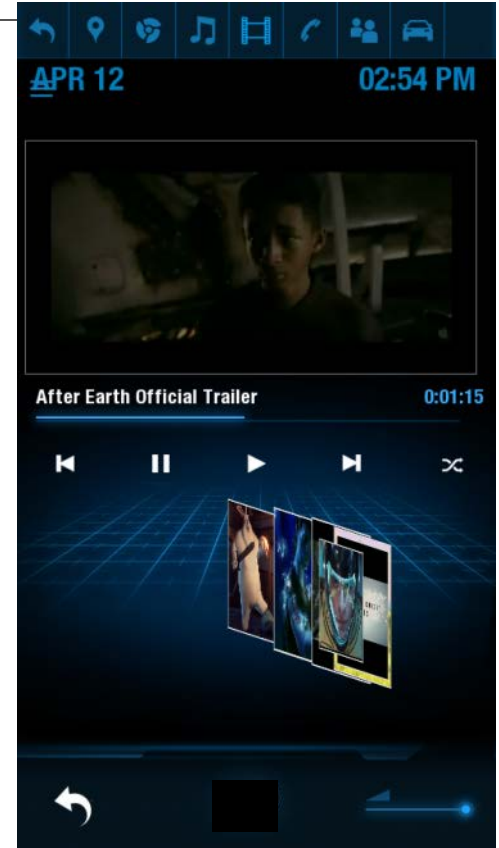
AMB –Automotive Message Broker

AMB stands for the Automotive Message Broker. AMB is a framework for accessing vehicle information from an application without the application having to know the specific details of the vehicle's network. The work for this project is to extend the AMB project so that it can be used to control HVAC settings on JLR vehicles using specialised plugins.

Use cases include:
- Climate fully auto scenario
- Max defrost scenario
- Fan speed scenario
- Air distribution scenario
- Temperature control scenario
- Air condition scenario
- Front screen heating scenario
- Rear window heating scenario

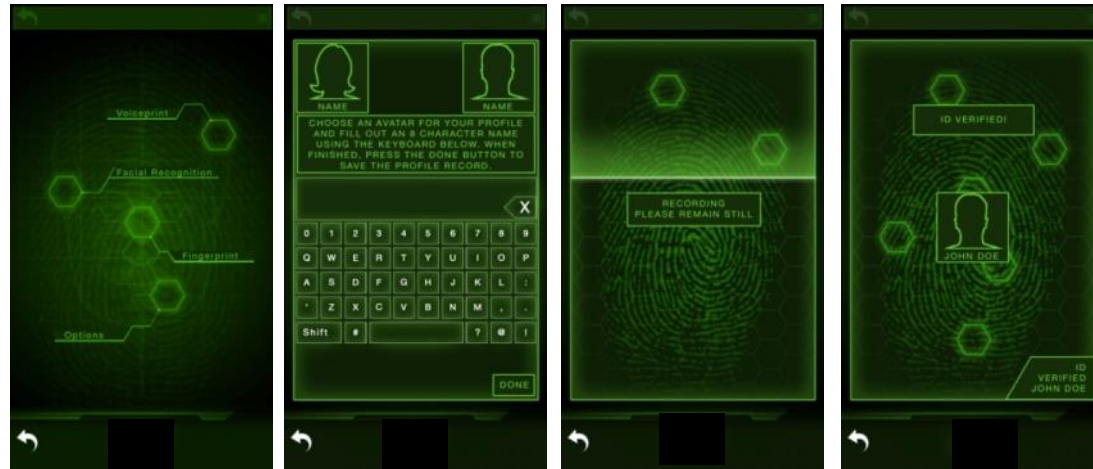https://wiki.tizen.org/wiki/Automotive_Message_Broker

Finger Print Scanning

The project objective is to implement Finger Print recognition as an integrated and working feature on the Tizen IVI platform.

Use cases include:

- Learning a finger print using an external scanning device.
- Recalling a previously recorded finger print.
- Deleting finger print records.

Face Recognition

The project objective is to implement Face recognition as an integrated and working feature on the Tizen IVI platform.

Use cases include:

- Learning a new face using an external cameras.

- Recalling a previously recorded face.

- Deleting face records.

Seeding the tech

- Adding functionality with the aid of the industry:

  - Fingerprint recognition

  - Voiceprint recognition

  - Face recognition

  - Camera integration

  - NavIt integration

  - Near Field Comms

  - Many more coming…

Remote Vehicle Interactions

RVI – SOTA	• Updating features and functions on the vehicle
RVI - Control	• Controlling the vehicle functions remotely
RVI - Big Data	• Securely getting the data from the vehicle to the cloud

Remote Vehicle Interactions

- **SOTA**
  Handle uploads, dependency checks, campaigns, cost control, transmission, installation, and reporting in a single open source system.

- **Big Data**
  Subscribe to vehicle data and analyse driving patterns, battery cycles, feature usage, etc, providing data for product refinement and value added services.

- **Car Sharing**
  Provide a single portal for owners, fleet managers and TaaS providers to remotely unlock, monitor, and disable vehicles.

Remote Vehicle Interactions
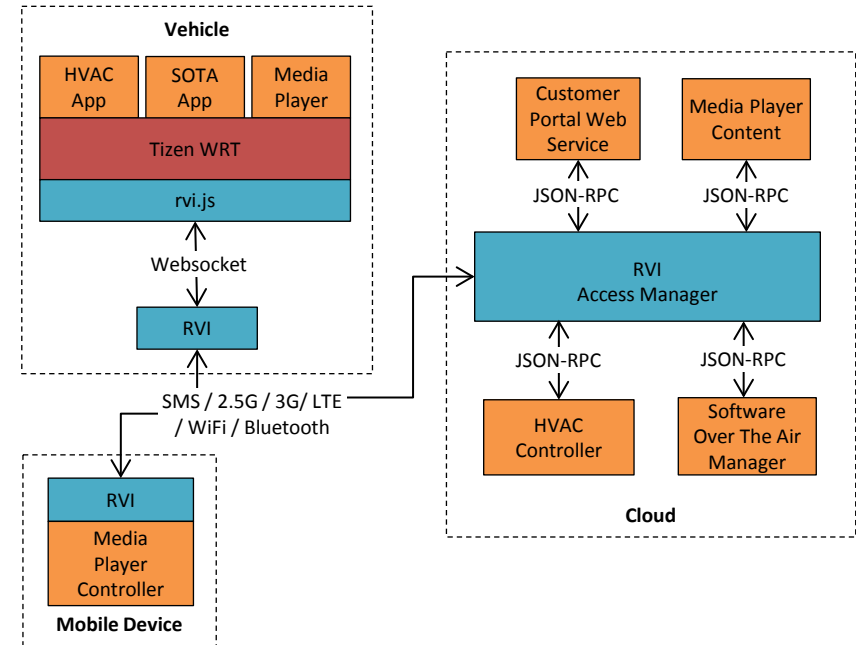
- **API based**
  All Tizen apps interface RVI through a standard JS API.

- **Simple backend service integration**
  Backend services can invoke a Tizen app through its service name. No network awareness necessary.

- **P2P**
  A Mobile device can interface Tizen apps through Bluetooth or local WiFi even when no Internet coverage is available
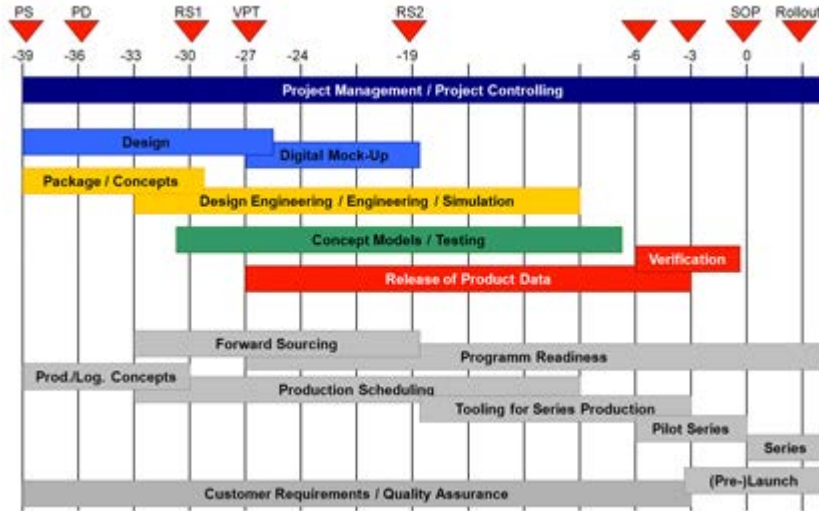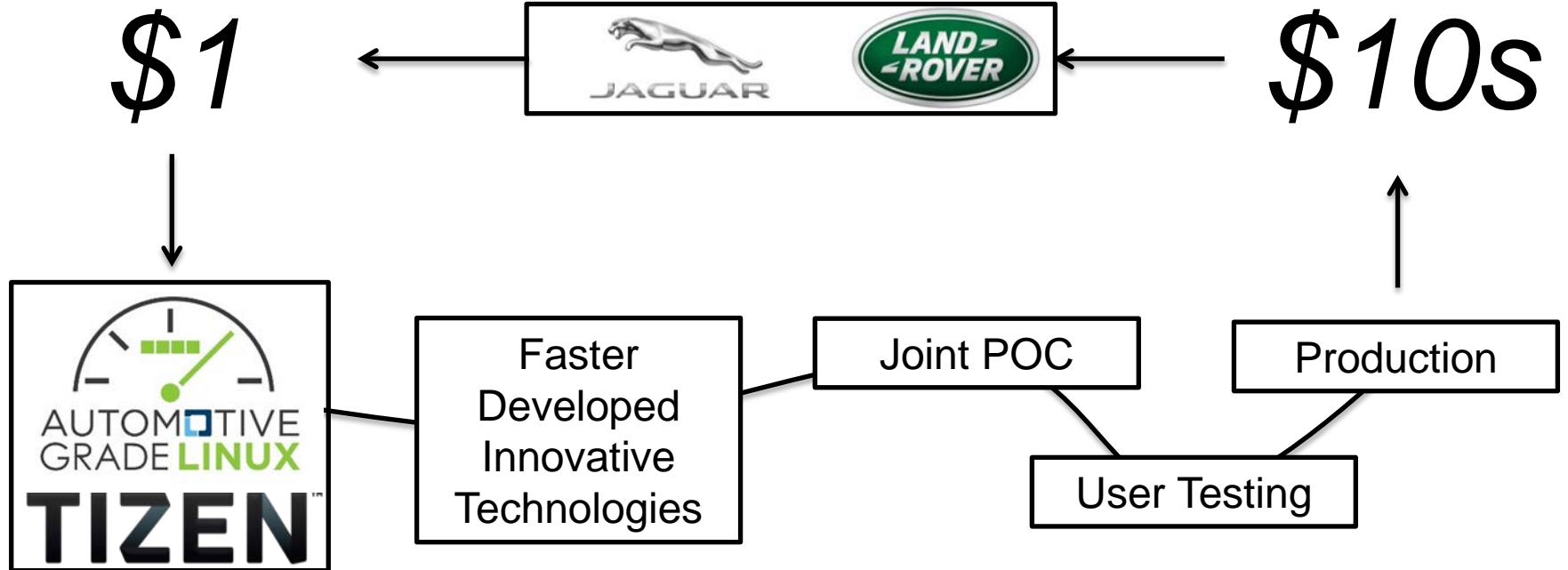
Time and Money

Timescales…

Average concept to deployment in the industry is 39 months…

Average lifespan of a startup is 18 months…

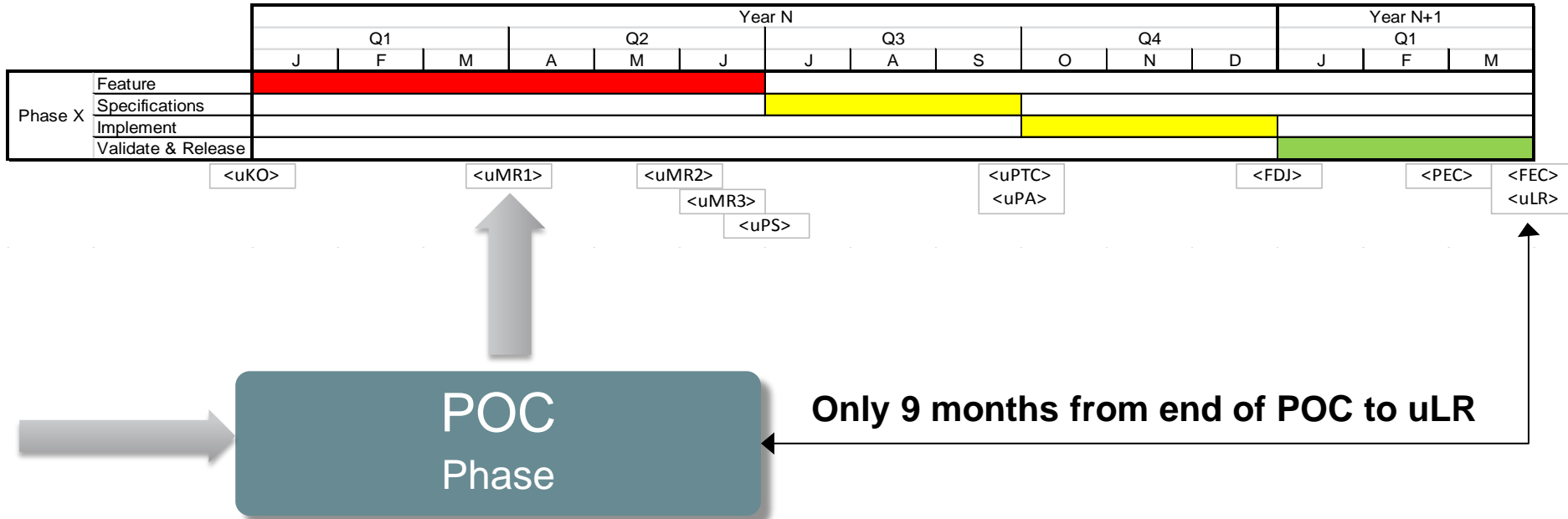# In To Production

From POC To Production

How can <u>we</u> reduce the timescales?

Phase X		Year N											Year N+1		
		Q1			Q2			Q3			Q4			Q1	
		J	F	M	A	M	J	J	A	S	O	N	D	J	F	M
	Feature															
	Specifications															
	Implement															
	Validate & Release															

<uKO> <uMR1> <uMR2> <uPTC> <FDJ> <PEC> <FEC>
<uMR3> <uPA> <uLR>
<uPS>

POC
Phase

Only 9 months from end of POC to uLR

Partnerships

- We're creating partnerships, rather than pure sourcing relationships.

- These are starting during the specification phase with engineering development partners.

- Not just at an OS level, but with individual applications

Open Source Software

- JLR are committed to Open Source Software:
  - It is our intention to push out any software that does not give JLR a competitive advantage.
  - We have set up code scanning tools to check all licenses
  - We have driven down the ability to release code to the lowest possible level… <u>Me</u>*
  - Our code locker will automatically push Open Source licensed software to an open website.

*This really scares our lawyers!

The Rest of the Journey

- Phase 2, 3, 4, 5, 6, 7…

Thank You!