JLR's Experiences with Open Source in IVI

Matt Jones
Jaguar Land Rover

Feb 2015

What’s happened previously

- A collection of stand-along generations of infotainment system.
- Proprietary hardware and software from individual suppliers.
- Designed to deliver the initial feature set and stop.

- This has lead to the support of multiple, concurrent architectures

	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017
Gen 1.0	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪		
Gen 2.0				🟨	🟨	🟨	🟨	🟨	🟨	🟨				
Gen 2.1							🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦

- Similar feature sets, different specifications, different Tier 1 vendors.

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

- This is a fully featured home entertainment network…

OEMs want to increase profits…

- Not locked into a single supplier:

- We hate moving suppliers, but commercially we sometimes have to…

Splitting the Sourcing Model

- JLR have a Hardware Tier 1 and a Software Integration Partner

- The Software Integration Partner will (using the JLR toolset) check for suitability and take all of the subsections of software that are specified, identify and fill gaps in the stack to deliver the feature, integrate the software stack, test and validate the software, deploy the software package to the hardware manufacturer, and warrant the software.

Learning

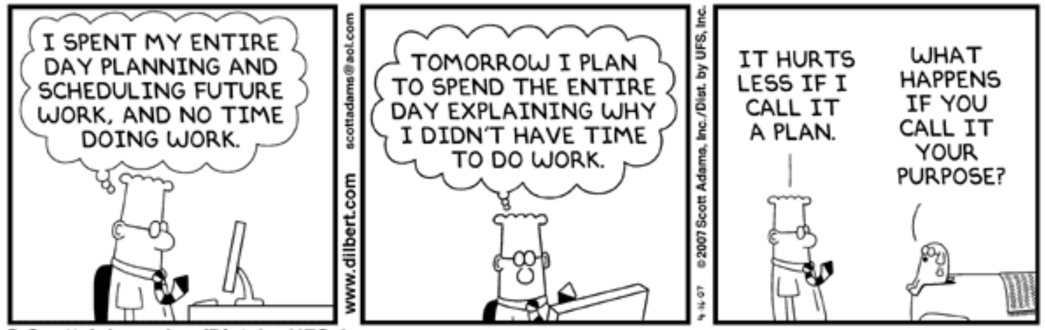
Learning

- Use case studies on failed software projects to learn from, not to replicate.

- Do not underestimate the time to change an organisation.

1. Project Management

- Software is not the same as hardware
- Software timings are estimates
- Never let OEM managers estimate!
- Plan for the end goal at a high level, focus on the next month…


I SPENT MY ENTIRE DAY PLANNING AND SCHEDULING FUTURE WORK, AND NO TIME DOING WORK.
TOMORROW I PLAN TO SPEND THE ENTIRE DAY EXPLAINING WHY I DIDN'T HAVE TIME TO DO WORK.
IT HURTS LESS IF I CALL IT A PLAN.
WHAT HAPPENS IF YOU CALL IT YOUR PURPOSE?
www.dilbert.com
scottadams@aol.com
© 2007 Scott Adams, Inc./Dist. by UFS, Inc.

© Scott Adams, Inc./Dist. by UFS, Inc.

2. Requirements

- Know what you are building
  - Specs may seem complete, but they should be signed off before sourcing.
  - Ensure all parts of the system have specifications – even HMI

- Version control is critical:
  - Use cases
  - Non functional requirements
  - APIs – especially these!!!

ENGINEERING LIAISON
TELL ME YOUR PROJECT STATUS AND I'LL TRANSLATE FOR OUR CLIENTS.
THE PROJECT WILL NEVER BE COMPLETED BECAUSE OUR IDIOT CLIENTS CHANGE THE REQUIREMENTS EVERY OTHER DAY.
I'LL JUST SAY YOU'RE DRUNK.
www.dilbert.com scottadams@aol.com
© 2000 United Feature Syndicate, Inc.
4/26/00

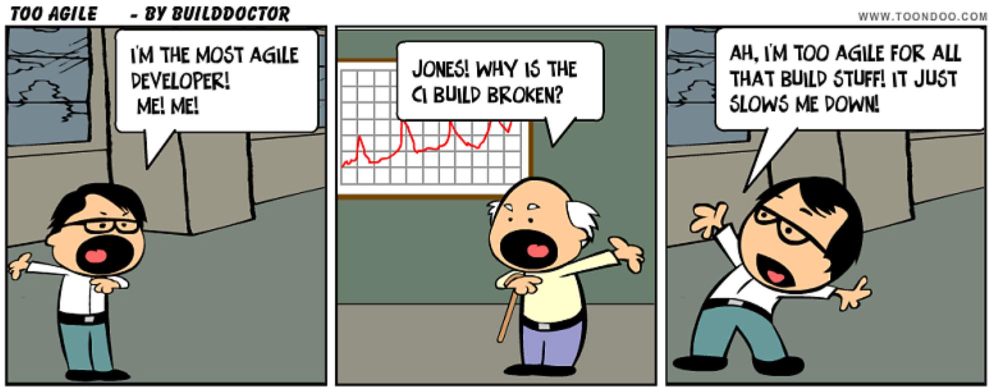
3. Continuous Integration

- Ensure all of the developers have access to a build service, whenever they need it

- Don't allow check outs for more than a few days

- Do not end up with a process that gives three weekly builds

- Nightly and weekly builds are a must

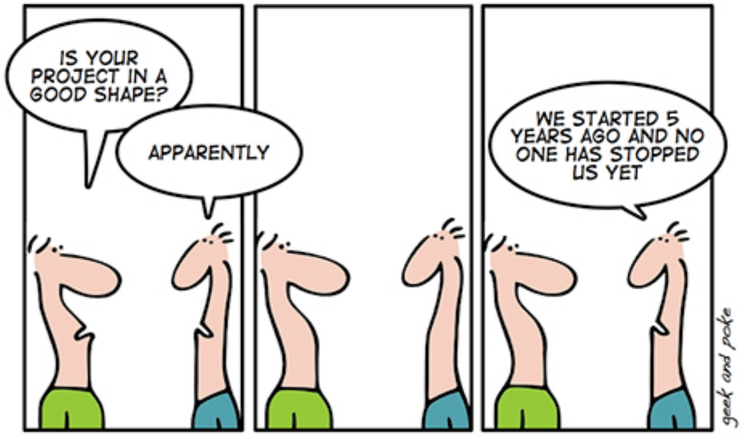

TOO AGILE - BY BUILDDOCTOR
WWW.TOONDOO.COM
I'M THE MOST AGILE DEVELOPER! ME! ME!
JONES! WHY IS THE CI BUILD BROKEN?
AH, I'M TOO AGILE FOR ALL THAT BUILD STUFF! IT JUST SLOWS ME DOWN!

4. Task Management

- Ensure management focus on the roadblocks, not the overall project

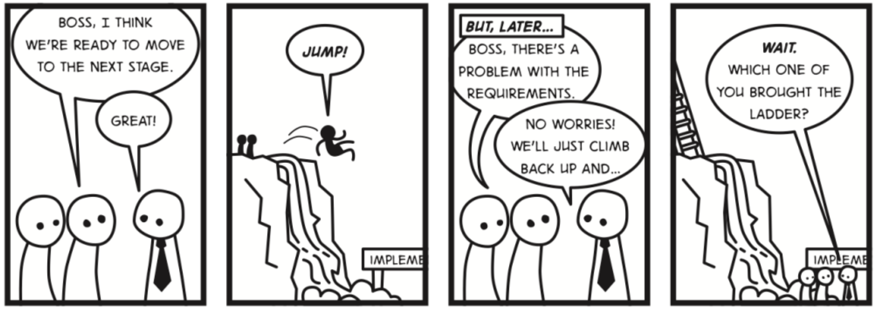
- Enable engineers to report out their status regularly

- Track task completion
  and use it to align resources


IS YOUR PROJECT IN A GOOD SHAPE?
APPARENTLY
WE STARTED 5 YEARS AGO AND NO ONE HAS STOPPED US YET
geek and poke

5. Testing

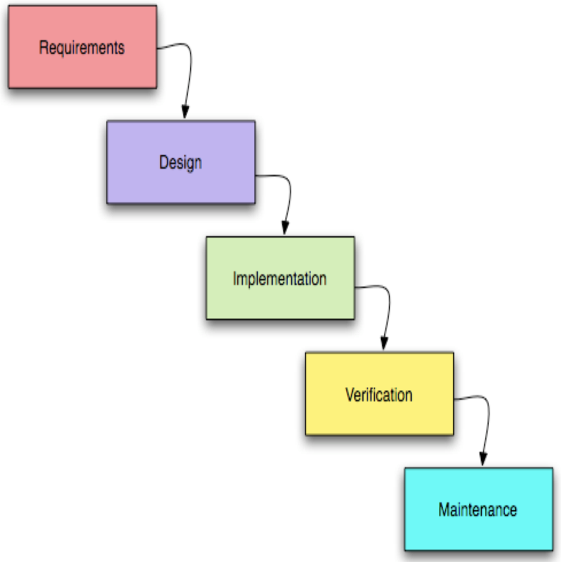
- Write the test with the requirement
- Test early and often
- Test every building block
- Expect to change requirements
- Do not have engineers write DVPs based on what they know of the system…


BOSS, I THINK WE'RE READY TO MOVE TO THE NEXT STAGE.
GREAT!
JUMP!
IMPLEME
BUT, LATER...
BOSS, THERE'S A PROBLEM WITH THE REQUIREMENTS.
NO WORRIES! WE'LL JUST CLIMB BACK UP AND...
WAIT. WHICH ONE OF YOU BROUGHT THE LADDER?
IMPLEME

Open Source

Development Model

Typical Waterfall Method



OSS Development Model

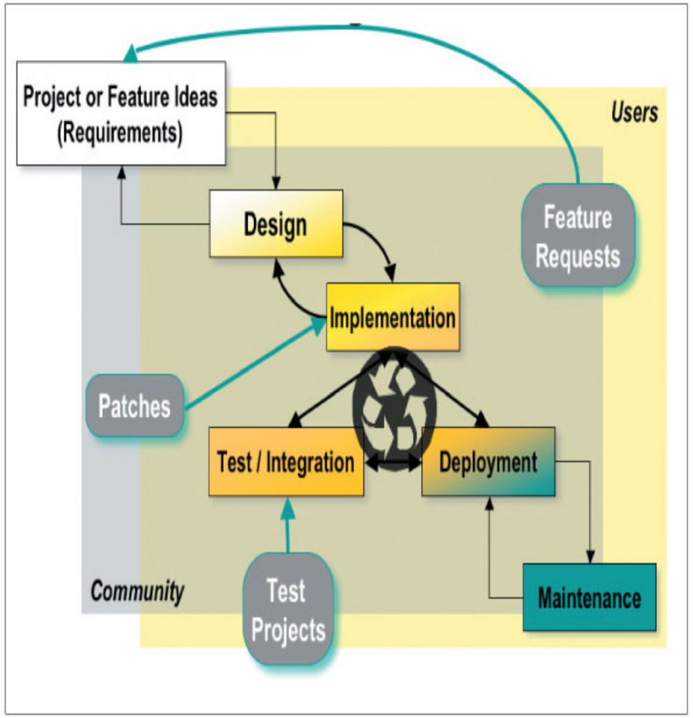


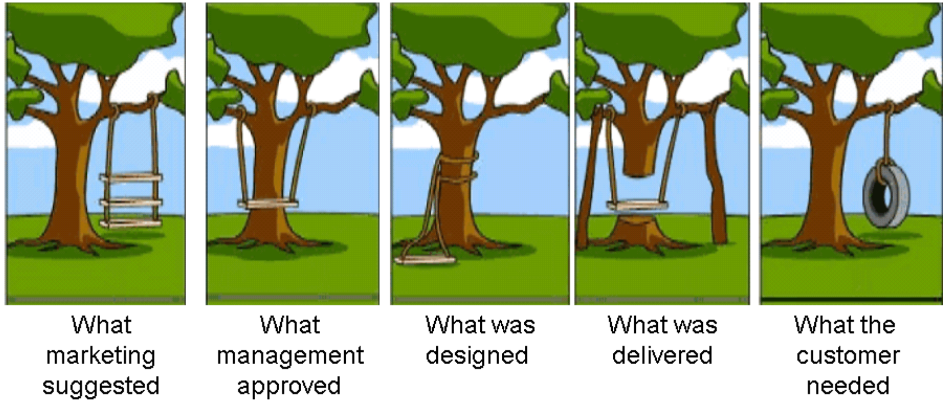
Figure 2: Open source development model

Entering an OSS Project

•Subscribe mailing lists relevant to the project
•Read documentation and mailing lists
•Assess community work environment
  •Community governance structure
  •Community resources
  •Communication etiquette
•Explore available resources
•Understand project structure
•Identify key stakeholders
•Participate in community activities
•Start to leverage Open Source resources (e.g. code, documentation, test) for the project
  •Start small to develop skills and understanding

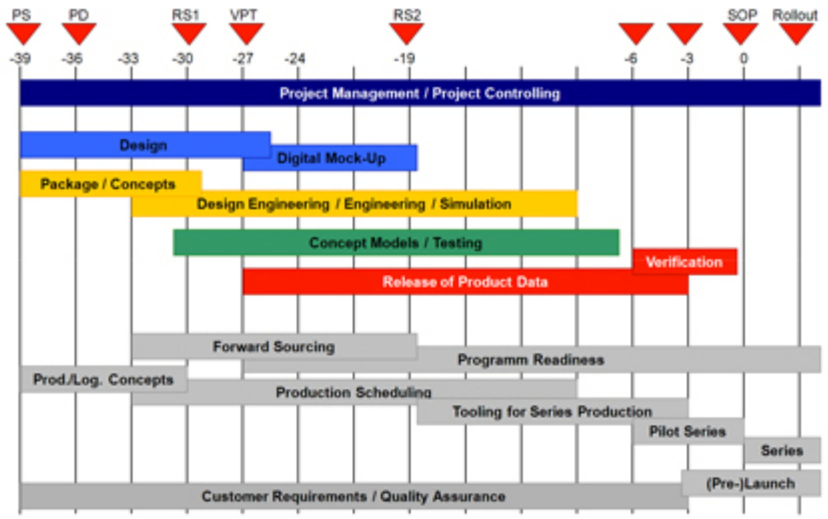
New Features / Projects

- Discuss the new feature or project with the community and solicit feedback before beginning your work
  - Even before code is available, discussion of requirements and plans can lead to very useful feedback AND buy-in by the community
  - After initial feedback is obtained, make contributions in small, manageable pieces.
  - Expect many iterations before your code is accepted



Time and Money

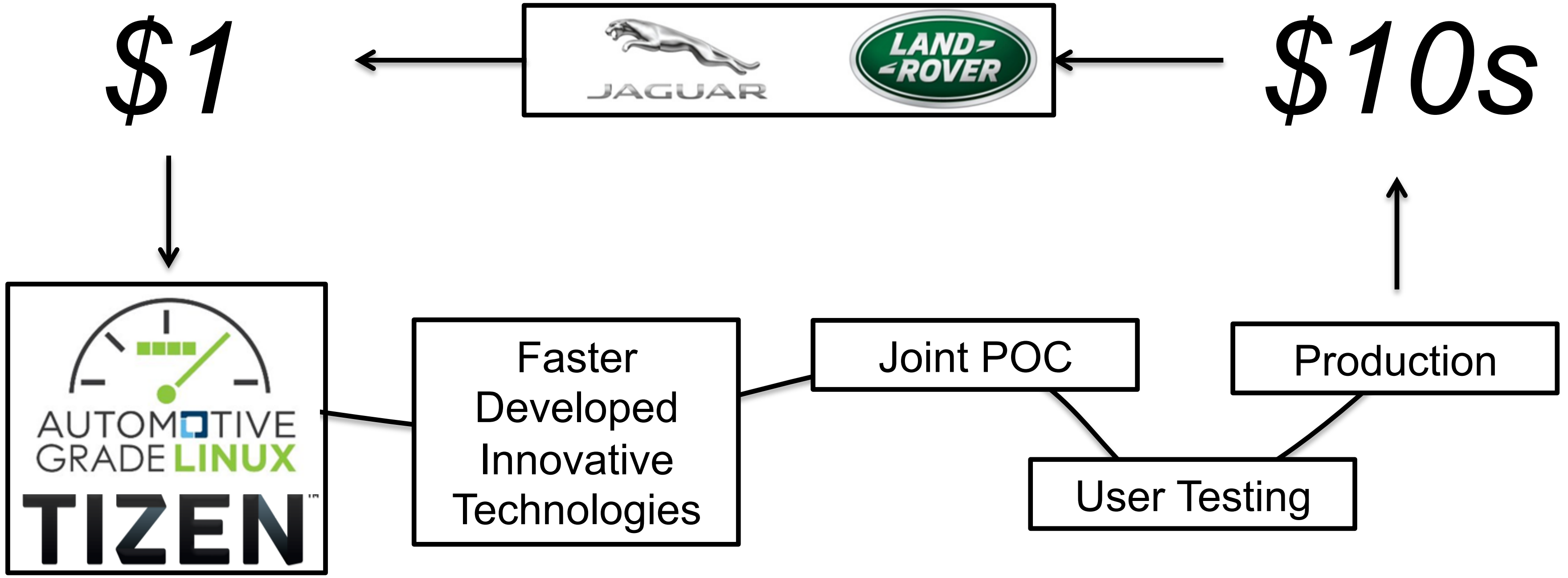
Timescales…



Average concept to deployment in the industry is 39 months…

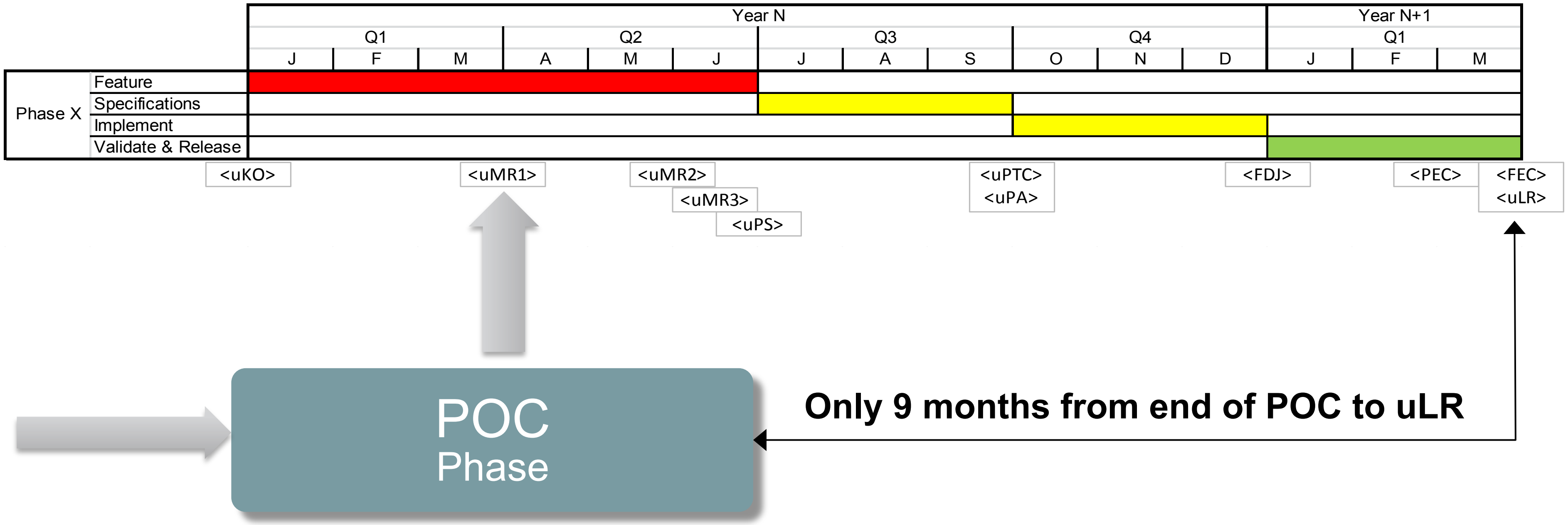
Average lifespan of a startup is 18 months…

In To Production



From POC To Production

How can we reduce the timescales?



Roadmaps

- We’re buying-in to technology roadmaps:
  - Toolset
  - Silicon suppliers
  - Network technologies
  - Linux distribution
  - Software providers

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… Me*
  - 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…