Engineering Automotive Systems

a visionary outlook

AUDIOWIDE - REGULAR 36

AGENDA SLIDE

- Introduction
- Goals and non goals
- Engineering northstar and areas
- Engineering systems
- Engineering design areas

Goals and non-goals of this talk

- Goal is
  - To showcase software engineering practices and designs to match the vision of Eclipse SDV
  - Understand what can be leveraged for automotive cloud and vehicle solutions
  - Understand the benefits of a cloud native approach and provide food for thought to dive deeper into topics
- Non goal is
  - Discuss solution architectures
  - Discuss all aspects of engineering processes and principles (we only have 30 minutes)
  - Provide a final solution/answer to everything (we already know it's 42)

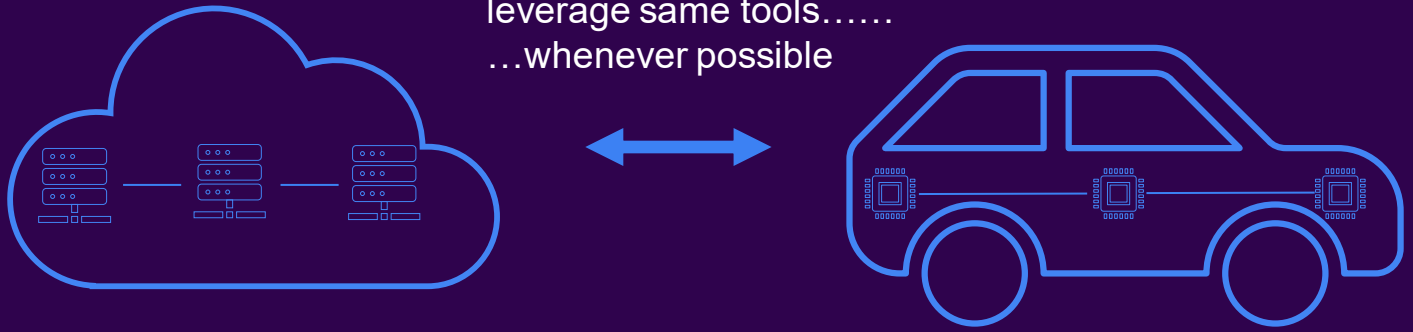
Automotive guy and software guy

“Great, we are planning to connect about 100 million devices and need to serve multiple tenants while honoring regulations and by the way we need to transform onboard architectures to mimic cell phones”

“Does everything in automotive need to be complicated!”

The engineering northstar

leverage same process
leverage same technology
leverage same tools……
…whenever possible



while ensuring safety

Two different worlds

Category	Traditional	Cloud Native
Quality of code check-ins	Depends on scenario	Validated through unit tests
Environment Creation/Configuration	Manual	Automated
Deployment Frequency	1-2x a month (or less frequent)	Deploy whenever needed, including several per day
App Deployment Process	Requires meetings and planning	Push-button deployment
Deployment validation	Manual	Automated
Observability	Minimal to none, mainly monitoring	Observability of the entire stack
Dev and Ops relationship	Blame culture	Culture of trust

Engineering principles that should be considered (excerpt)

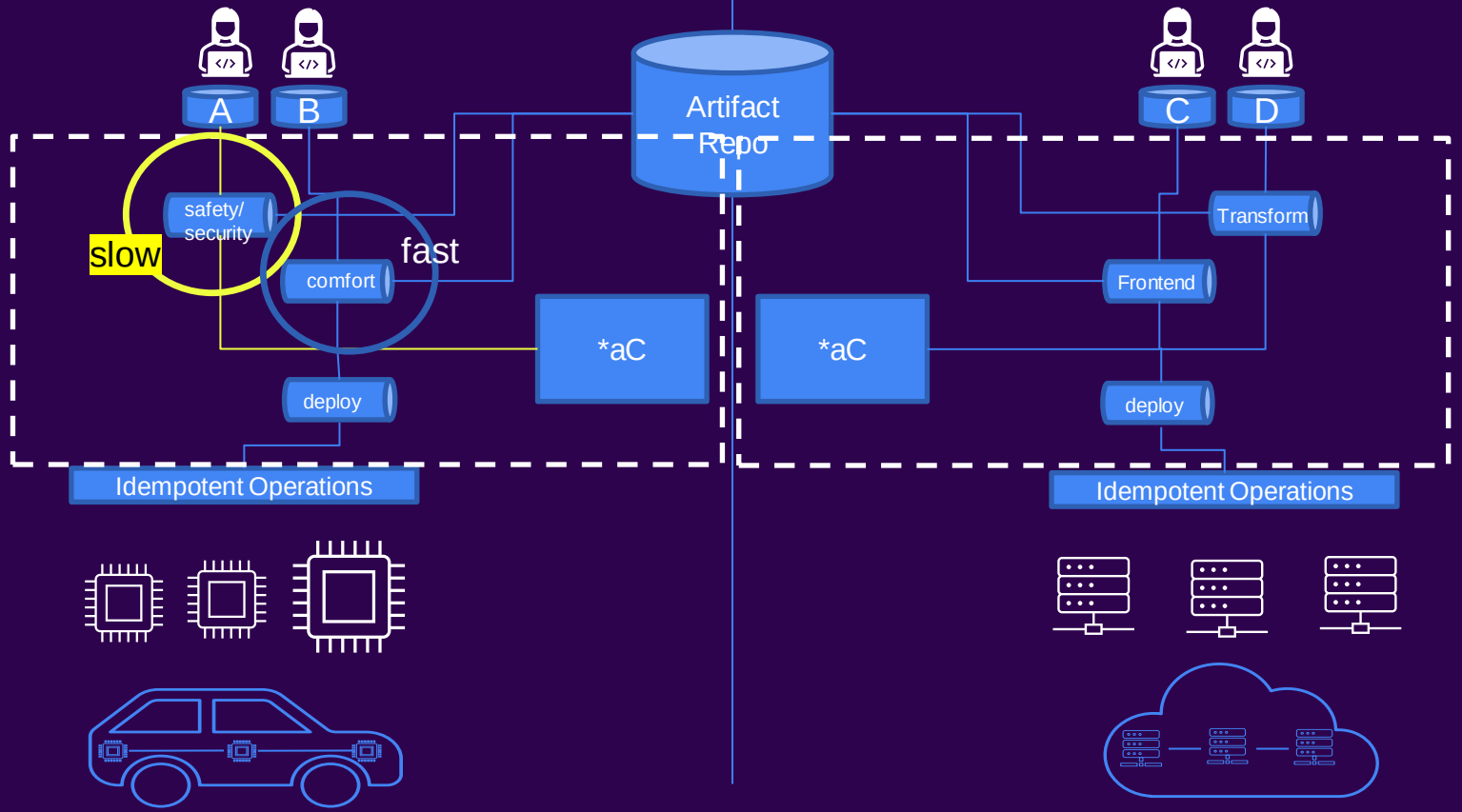
- Organizational challenges
  - Small team with bounded context
  - You build it, you run it
- Automation
  - IaC
  - Gitops
  - Gated Rollouts
- Observability
  - Monitor the entire stack
  - Correlate events
  - Use the same standard, e.q. OpenTelemetry
- Testing
  - Shift to a testing pyramid (shift left)
  - Testing services in isolation does not offer much value
  - Decide between safe and fast
- Service Design
  - Use tools like OpenAPI
  - Consider serialization costs

Deployment velocity: Safe vs Fast

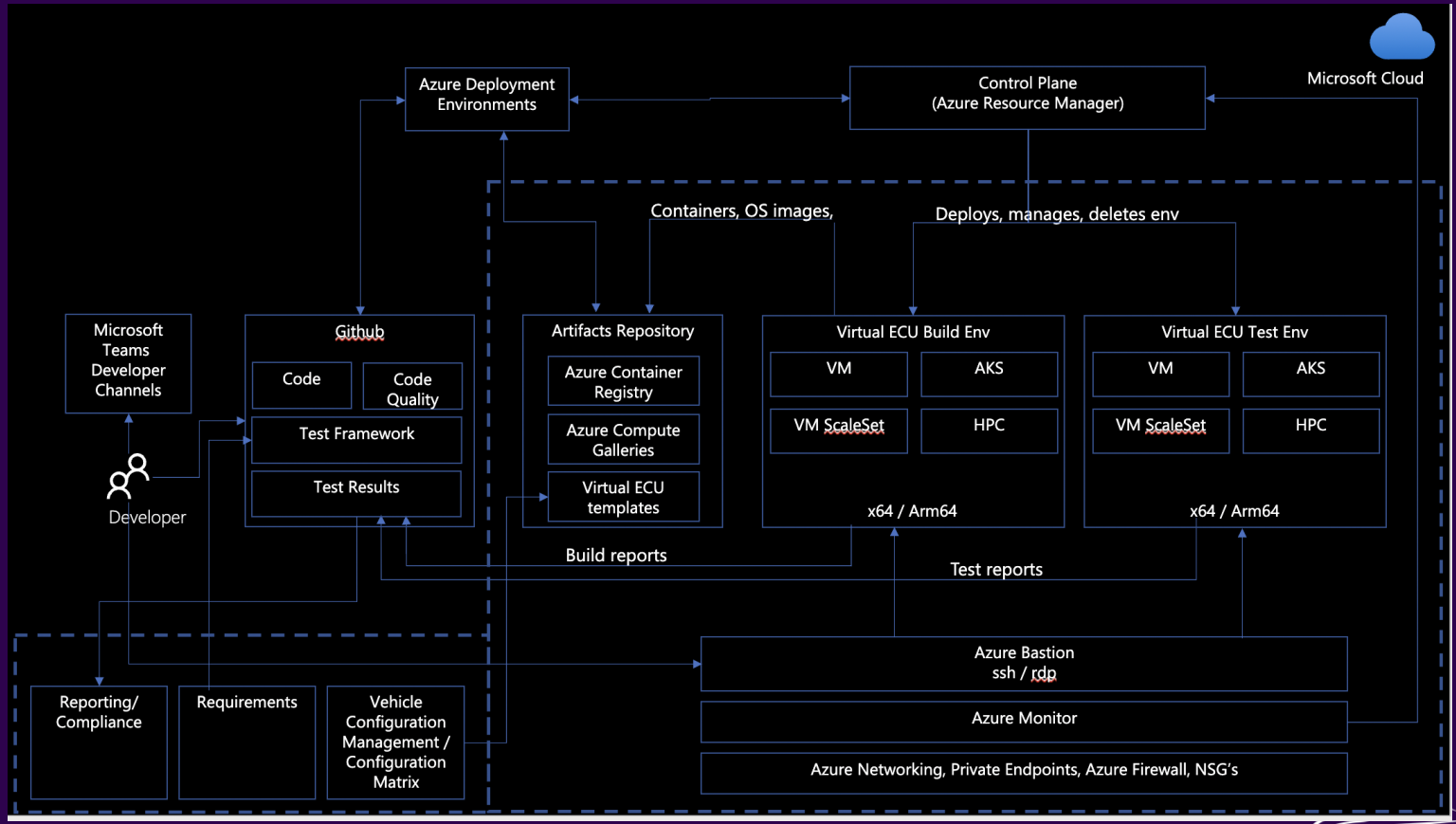
- SAFE
  - Follows traditional automotive development
  - ASPICE/ISO
  - Less frequent component update
  - Safety/security critical components
  - Artifacts binaries
  - In car only

- FAST
  - Follows cloud native processes (DevOps, GitOps)
  - Queued and staged deployments
  - Frequent component update
  - Non safety/security relevant components
  - Cloud and car

High level engineering system to support vehicle and cloud

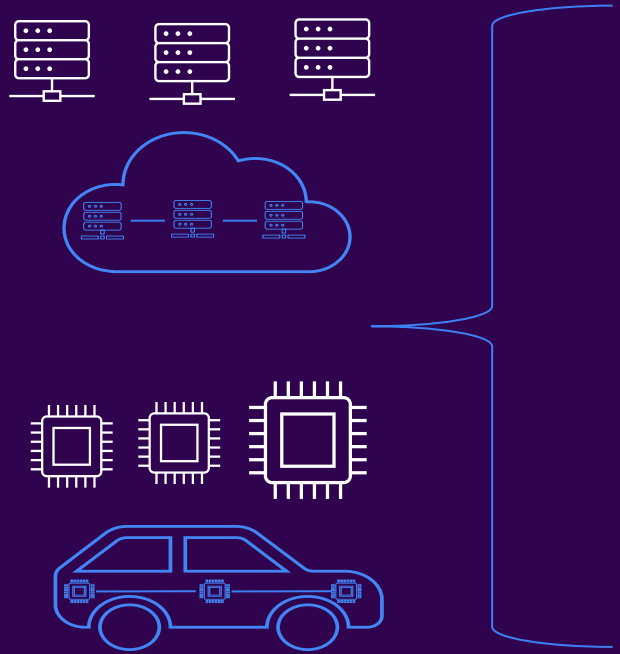


On board system zoom in



Important engineering design principles that can be translated to automotive

Moving into “SOA” means moving into distributed systems



Engineering design principle	Cloud	Vehicle
Loose Coupling	Y	Y
Fault tolerance	Y	Y
Scalability	Y	N*
Consistency	Y	In car**
Security	Y	Y
Performance	Y	Y
Interoperability	Y	Y

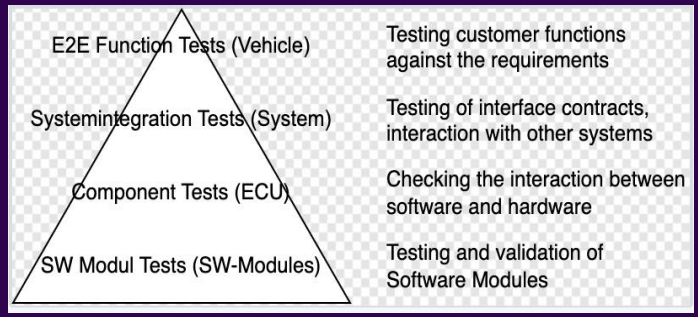
*Currently only process/node
** Strong consistency not required

THANK YOU!

Engineering building blocks – cloud native perspective

- Project/Program Management
  - Agile/Scrum
  - Work Item Tracking
- Development
  - Developer Experience (DevEx)
  - Design Reviews
  - Code Reviews
- Testing
  - Executed during CI/CD as pyramid
- Operations
  - Observability
  - Source Control

Security



SDV Eclipse Software Defined Vehicle