CONTINUOUS INTEGRATION/CONTINUOUS DEPLOYMENT (CI/CD) IN LARGE ENTERPRISE ENVIRONMENTS

Michael Orosz, Research Director and Research Associate Professor

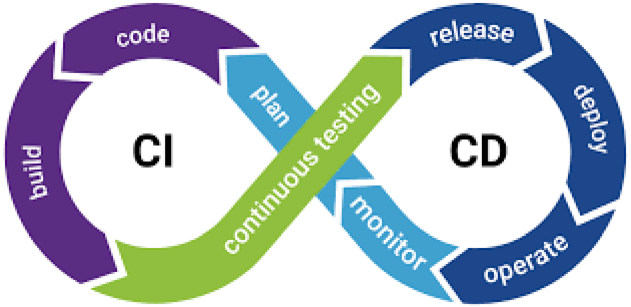
mdorosz@isi.edu

University of Southern California
Information Sciences Institute

September 2022

SYSTEMS ENGINEERING RESEARCH CENTER

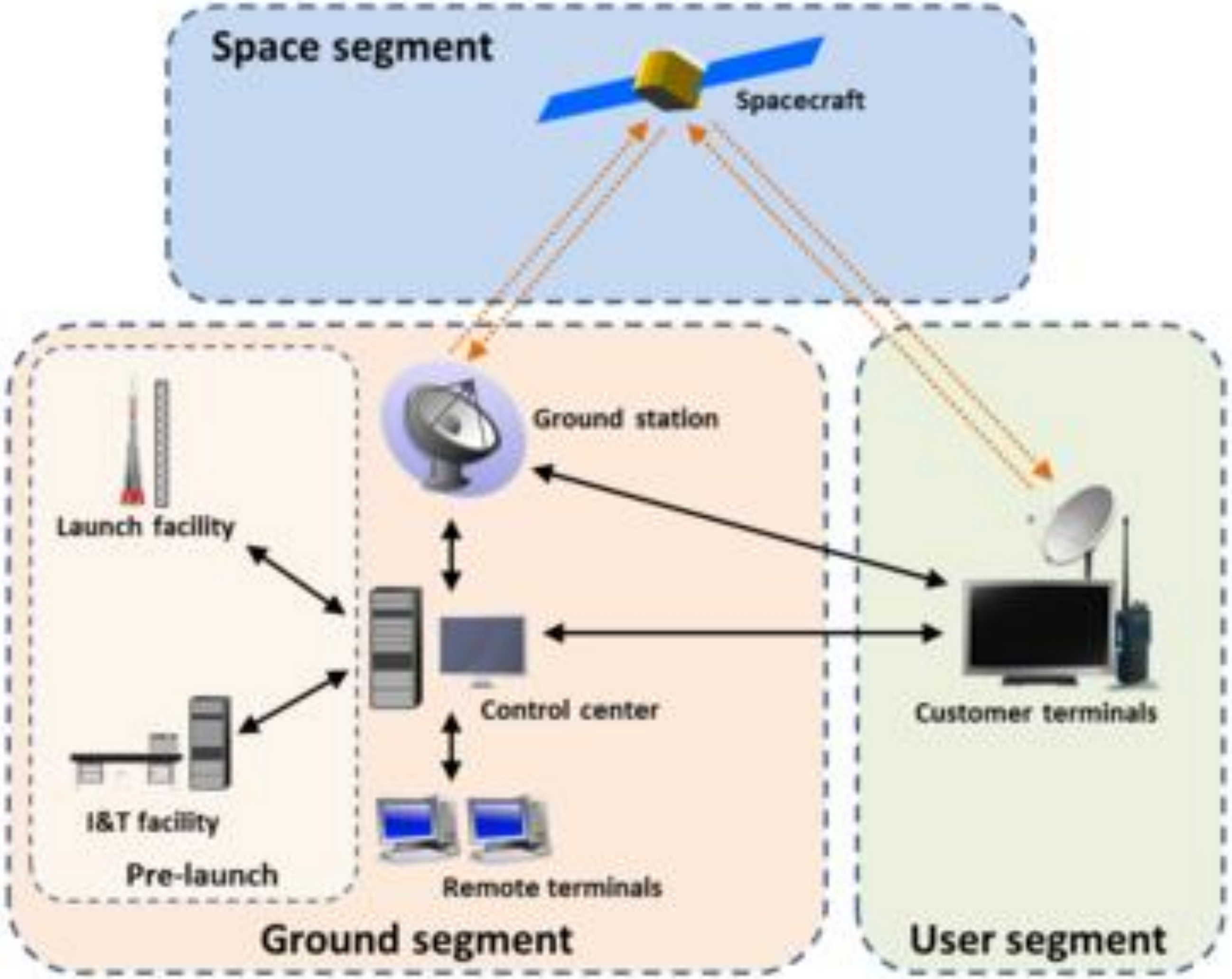
Why CI/CD?



- **Customer Perspective**: Get product into the hands of the customer as soon as possible (i.e., provide value) and an opportunity to collect feedback (i.e., capture evolving customer needs)
- **Developer Perspective**:
  - — Quickly respond to customers' evolving needs (*if Agile principles are part of the effort*)
  - —**Early discovery and mitigation of discrepancies**

The Enterprise System

1. Systems of systems
2. Comprised of both hardware and software elements
3. Evolving requirements
4. Multiple and often separate development pipelines
5. Long development timelines
6. Multiple and often changing vendors
7. Often multiple and highly vulnerable supply chains

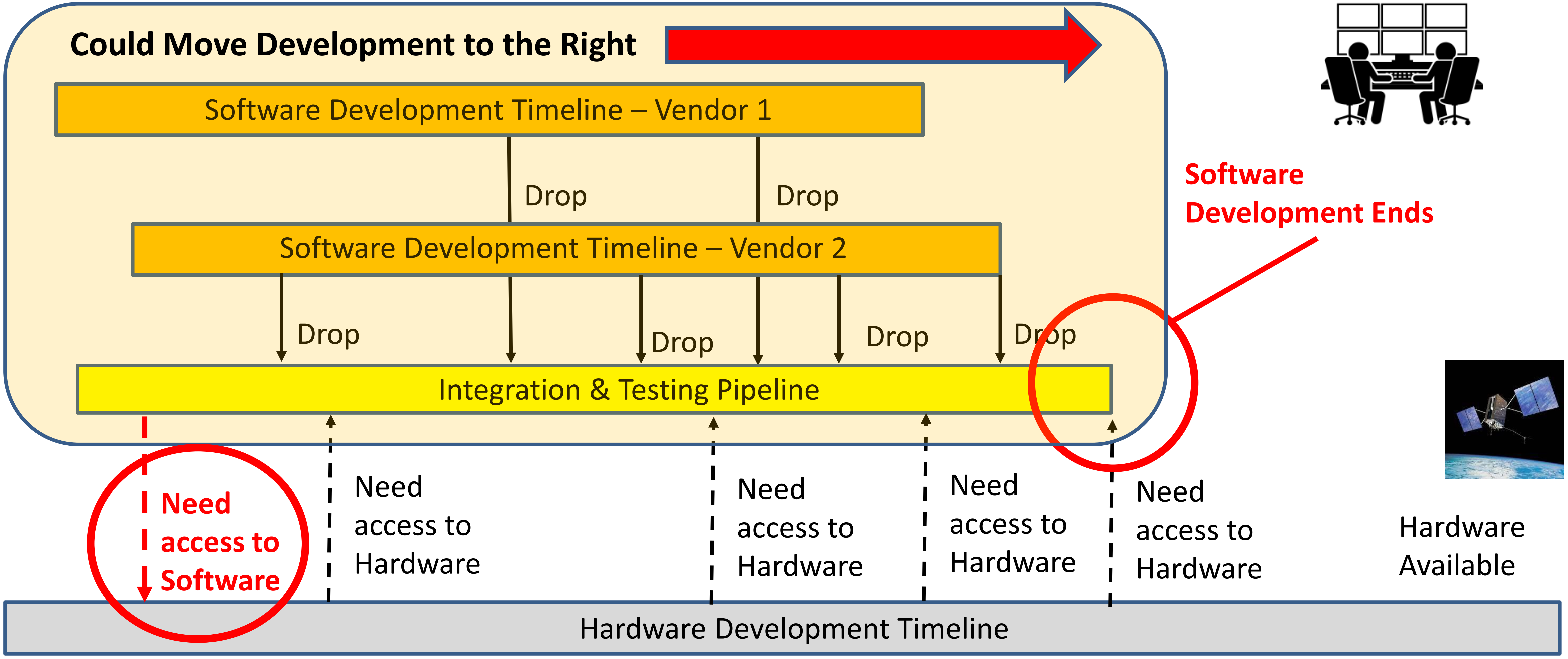


Challenges to CI/CD

1. Access to the complete system for integration and testing may not be possible (e.g., mission critical nature, components of the enterprise may not be available, etc.)

2. It may not be possible to delivery a product to the end-user on a continuous or near-continuous basis

Enterprise System Timelines – The Challenge



Could Move Development to the Right

Software Development Timeline – Vendor 1

Drop

Drop

Software Development Timeline – Vendor 2

Drop

Drop

Drop

Drop

Integration & Testing Pipeline

Software Development Ends

Need access to Software

Need access to Hardware

Need access to Hardware

Need access to Hardware

Need access to Hardware

Hardware Available

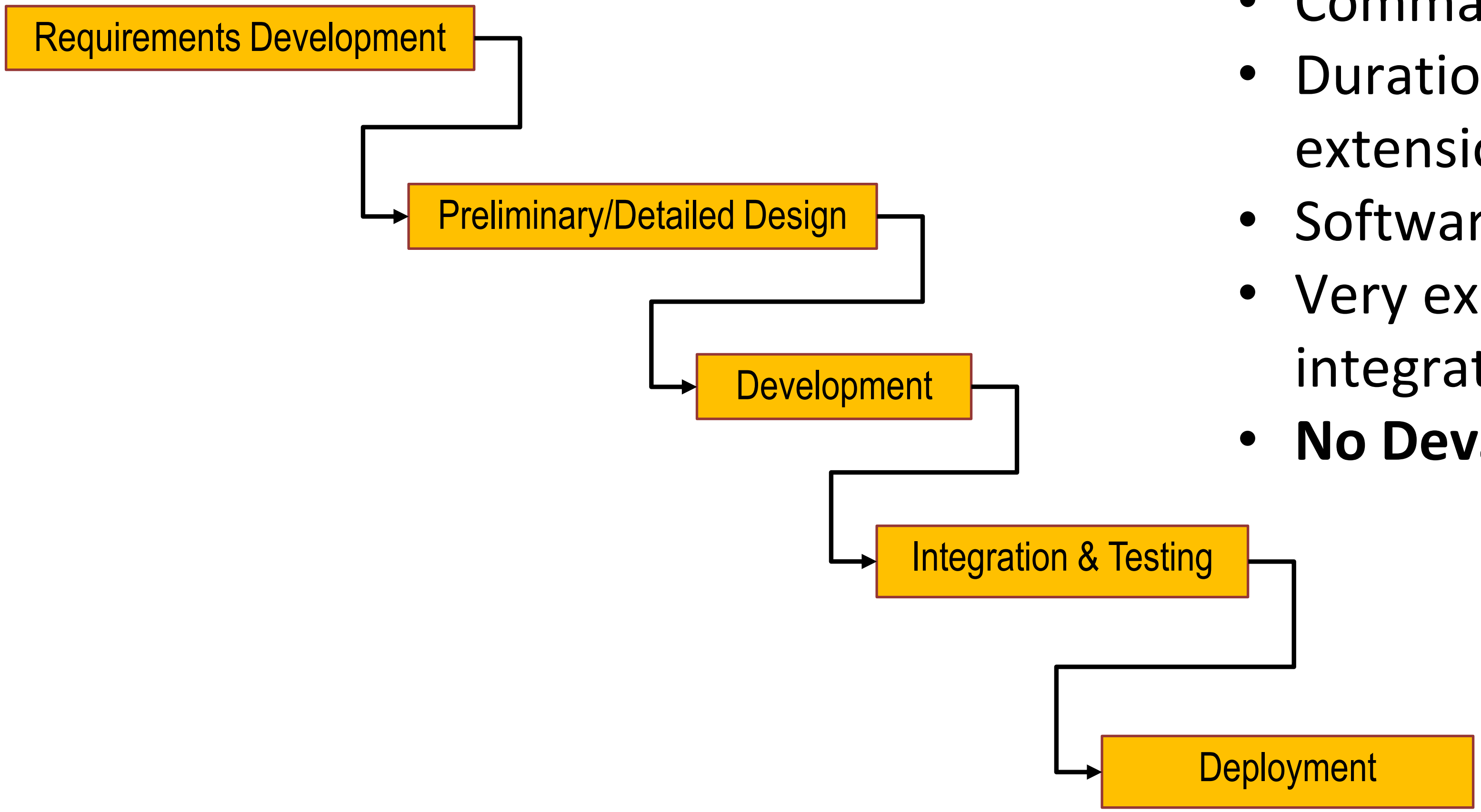
Hardware Development Timeline

What about vendor availability? Access to supply chains?

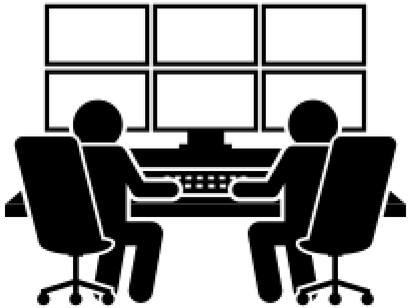
Moving timelines can be difficult

Case Study

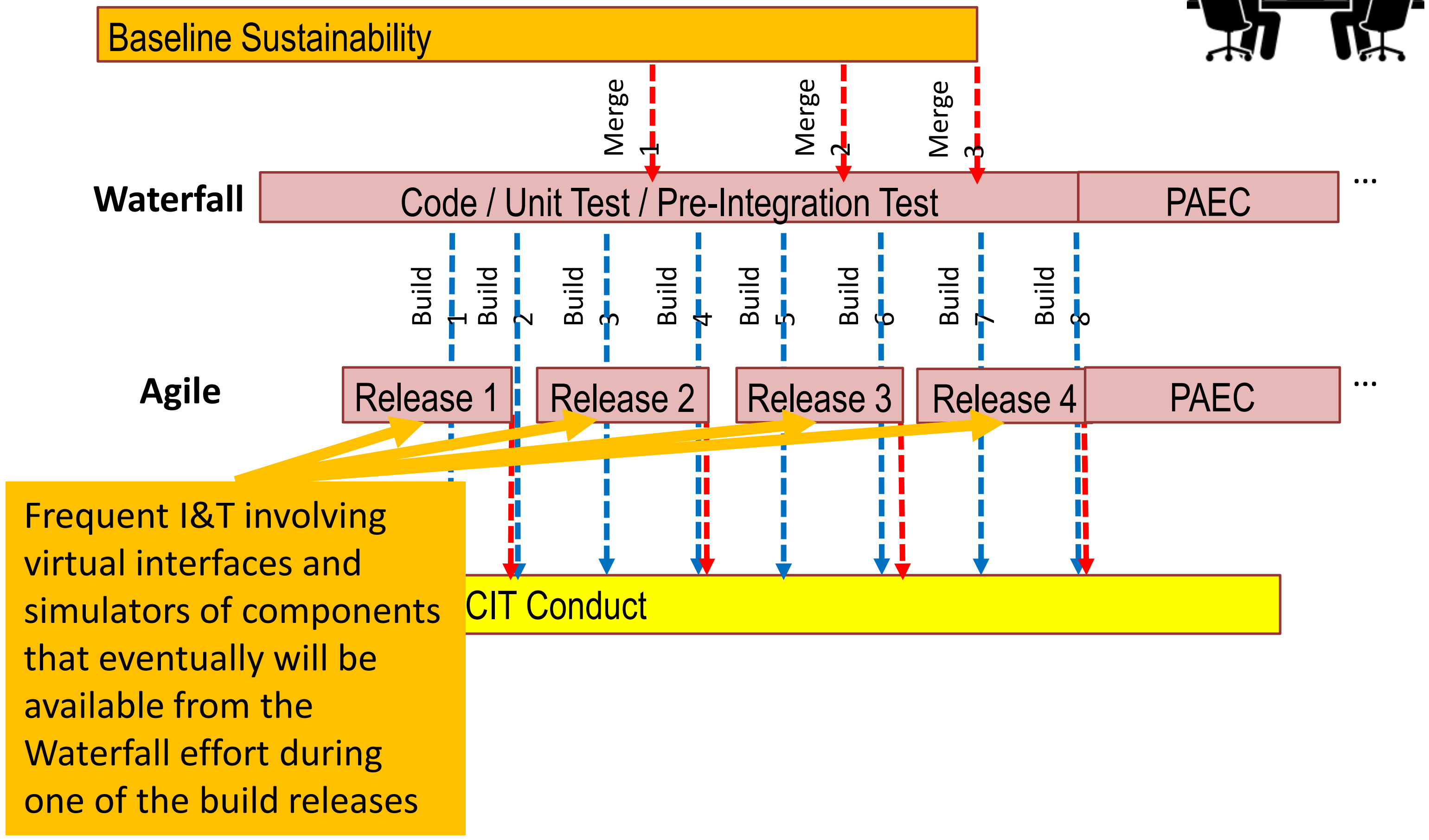
Project A (Traditional Waterfall)



- Command and control (C2) application
- Duration: 39 months (includes schedule extension)
- Software lines of code (SLOC): 178K
- Very experienced development and integration & testing team
- **No DevSecOps pipeline**



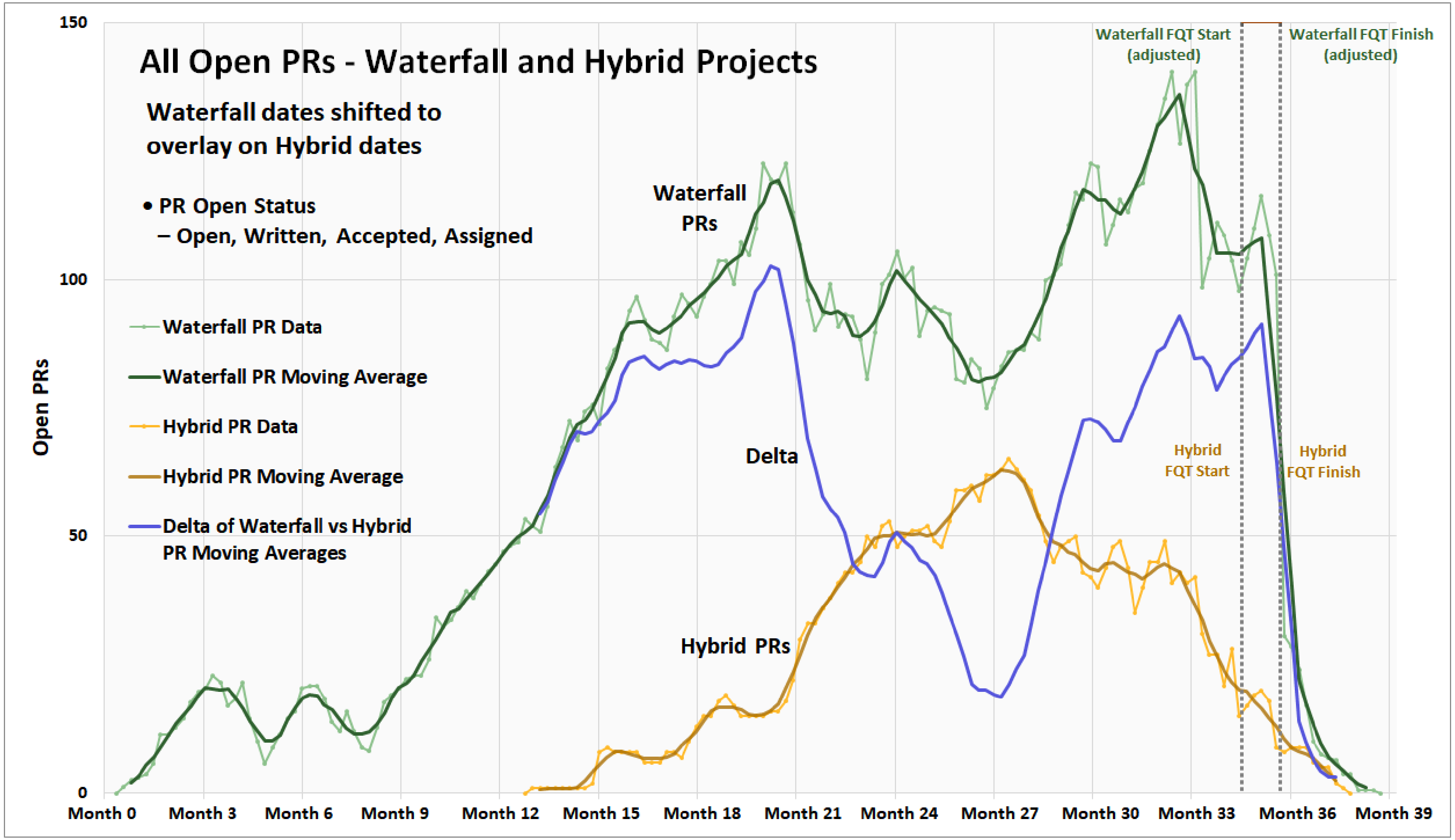
Project B (Hybrid Agile/DevSecOps & Waterfall)



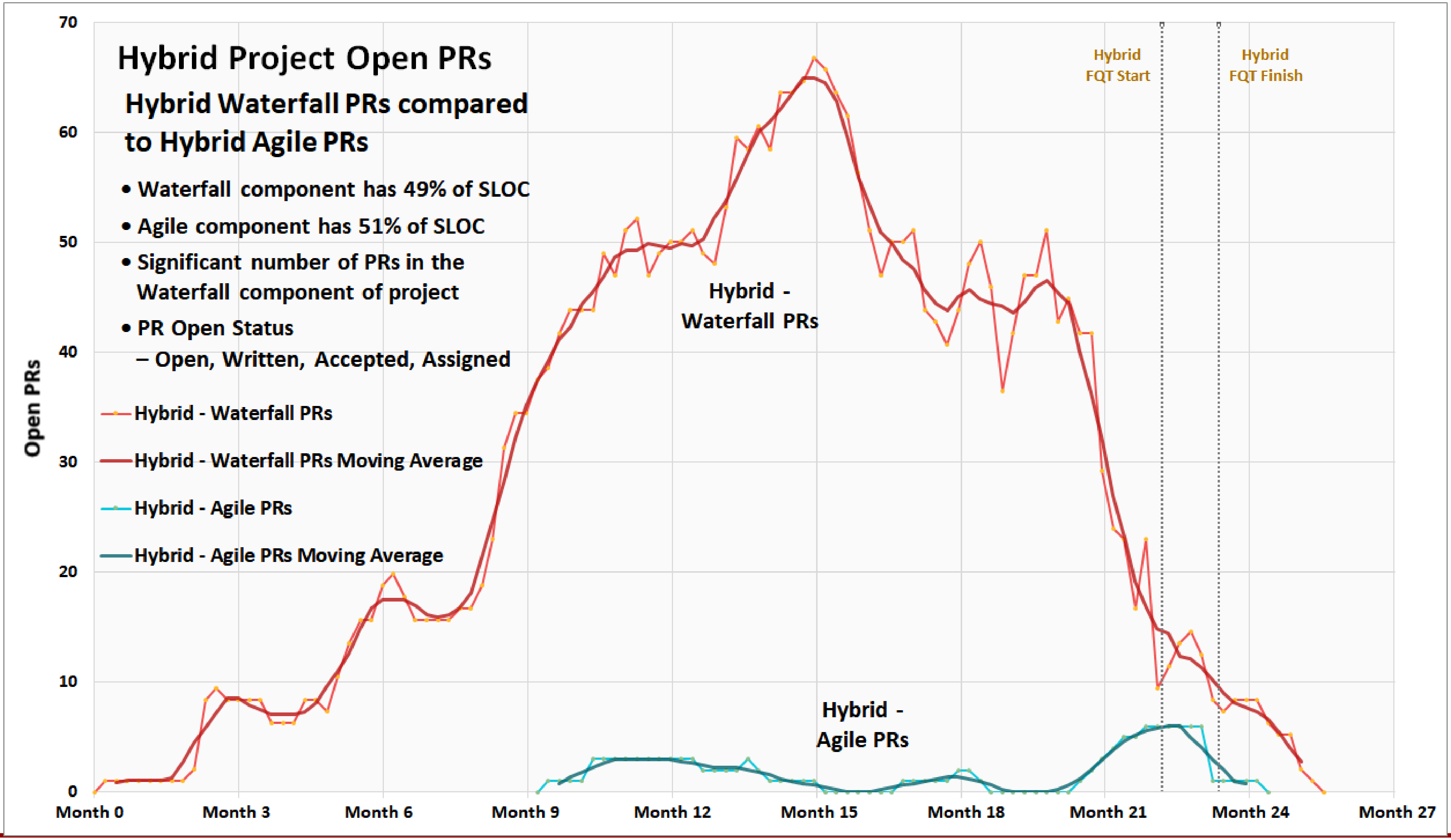
Frequent I&T involving virtual interfaces and simulators of components that eventually will be available from the Waterfall effort during one of the build releases

- Command and control (C2) Application
- Combined Waterfall and Agile effort. Agile effort included a DevSecOps pipeline
- Duration: 25 months
- Software lines of code (SLOC): 113K
- Experience level of waterfall team was high, experience of agile/DevSecOps team was low

Case Study – Comparing Discrepancies Between A & B



Case Study – Comparing Discrepancies (Hybrid)



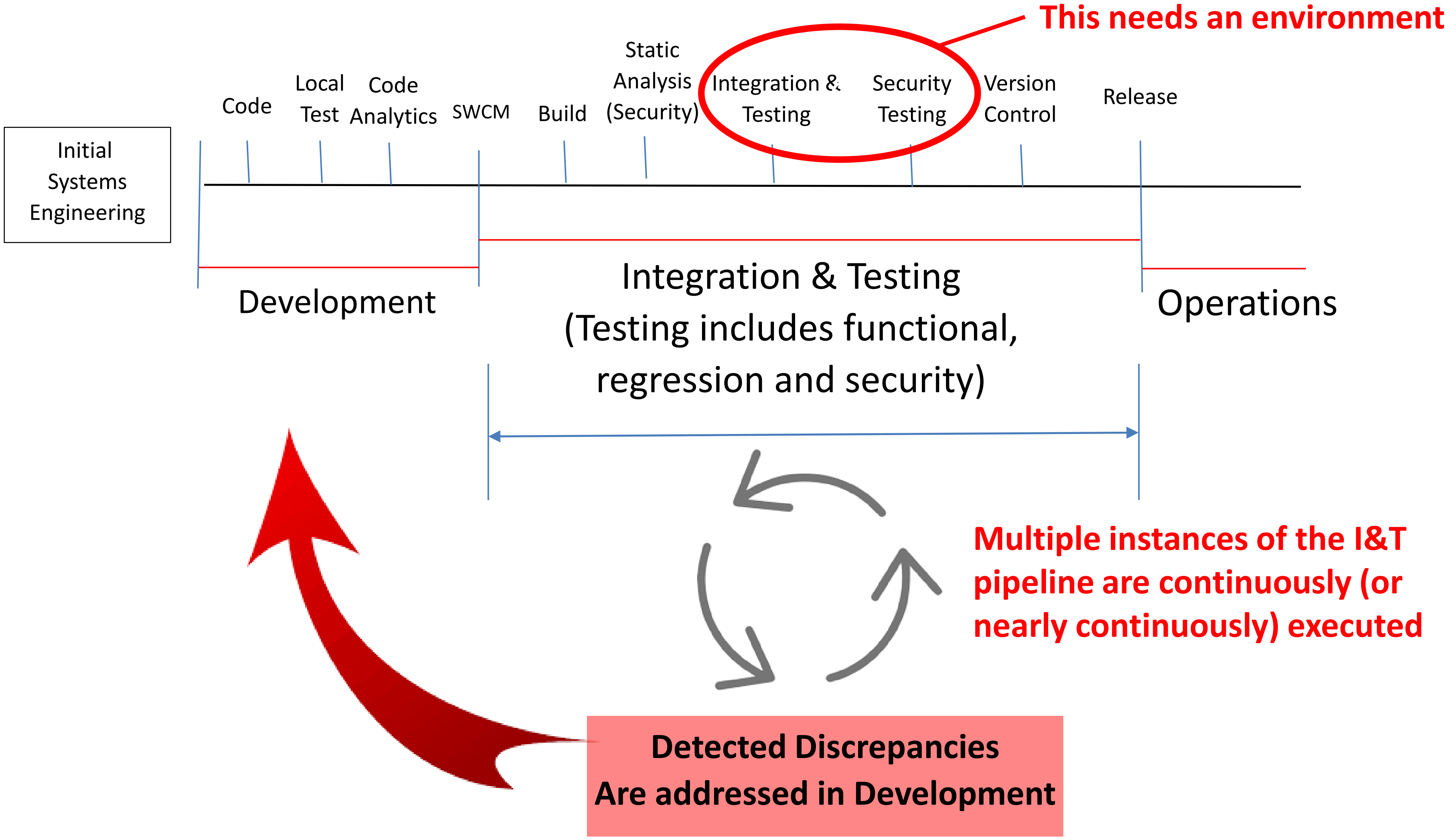
Hybrid Project Open PRs

Hybrid Waterfall PRs compared to Hybrid Agile PRs

- Waterfall component has 49% of SLOC
- Agile component has 51% of SLOC
- Significant number of PRs in the Waterfall component of project
- PR Open Status
  - Open, Written, Accepted, Assigned

Observations/Recommendations

DevSecOps Pipeline



This needs an environment

Initial Systems Engineering

Code

Local Test

Code Analytics

SWCM

Build

Static Analysis (Security)

Integration & Testing

Security Testing

Version Control

Release

Development

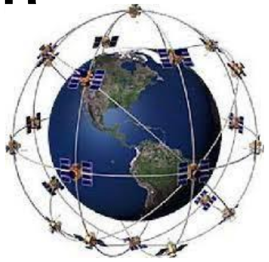
Integration & Testing
(Testing includes functional, regression and security)

Operations

Multiple instances of the I&T pipeline are continuously (or nearly continuously) executed

Detected Discrepancies
Are addressed in Development

Observation: Need for Near Operational Environment

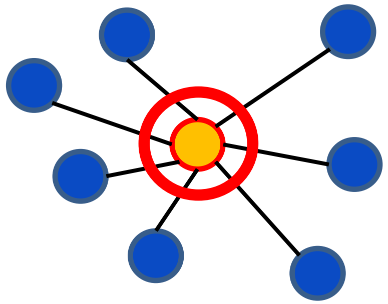


Simulated

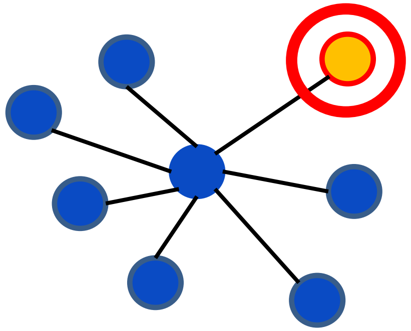
1. Often composed of both actual and simulated systems and interfaces
2. These systems are substitutes for the actual enterprise and allow – at a minimum – an opportunity for continuous integration
3. **Accreditation can be a challenge – especially if requirements are to be officially sold off**
4. **Need to allow time for the development and accreditation of the near operational environment**
5. In a sense, we are faking the deployment of the system. We put the built system on a shelf until the rest of program catches up

Horizontal vs. Vertical Integration & Testing

1. Horizontal I&T: Integration & Testing involves the complete system or a large portion of the enterprise (e.g., a near operational environment)



2. Vertical I&T: Integration & Testing involves a subset of the enterprise.



3. Need some good upfront systems engineering
   - —This does not necessarily entail a detailed design (targeting agile programs)
   - —Want to design a system architecture that minimizes interdependencies between subsystems (may need to refactor an existing design)
   - —The goal here is to undertake a subset of integration and testing while waiting for an actual or near operational environment to become available

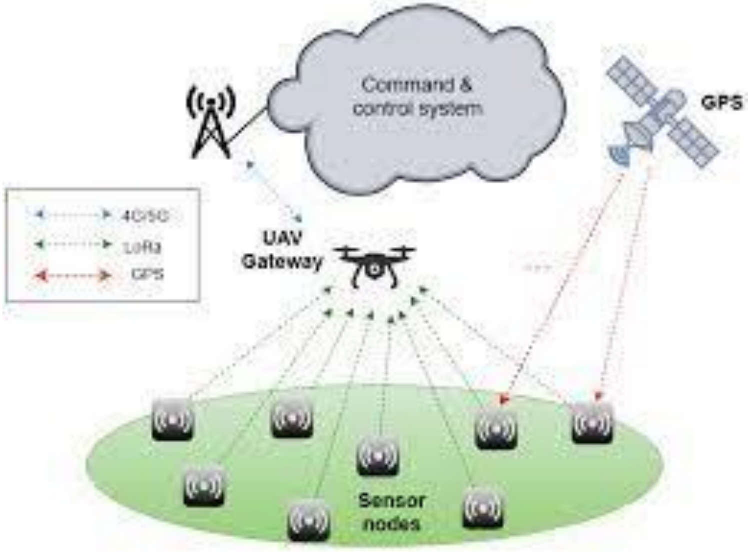
What About Continuous Deployment?

1. End-user engagement: Often this is not possible due to limited availability (end-users and/or an evaluation environment)
   - If possible, hire subject matter experts to be part of the development and I&T team. *<u>Added expense, but will pay off in long run</u>*

2. End-user training: For many applications, frequent engagement with the end-user (e.g., to get feedback) may not be possible due to heavy training requirements
   - Rely on the same end-users, but this will narrow feedback to a very focused group
   - Use the near operational environment for training

Conclusion

- CI/CD can be implemented in large enterprise development efforts
  - Requires some additional resources, ***<u>planning</u>*** and coordination
  - Get started on developing a near operational environment as soon as possible
    - Mitigate IP issues early
  - If necessary, hire subject matter experts from the end-user community
- Need "buy-in" from all the players
  - Customer, all vendors, leadership
- Produce good – even great - performance results


Command &
control system
GPS
4G/5G
LoRa
GPS
UAV
Gateway
Sensor
nodes

Thank You