The Impacts Of Agile Development On The System Engineering Process

17th Annual Systems Engineering Conference
October 27-30, 2014
Springfield, Virginia

Discussion Topics

- Background
- What Is Agile Development
- Benefits
- Impacts on the Traditional Engineering Process
- Solutions
- Summary

Background

- **Driver for Change**
  - Cost Effective Capability Development that Meets the Warfighter's Needs

- **Enablers**
  - **Technology** – Hardware Platforms Are Available on Day One of the Development Effort
  - **Open Architecture** – Standard Interfaces Simplify the Integration of Capabilities as They Are Developed
  - **Commercially Available Libraries** – Leverage Commercial Industry's Research and Development
  - **Agile Development** – Simplify Development by Providing Flexibility at the Detailed Implementation Level

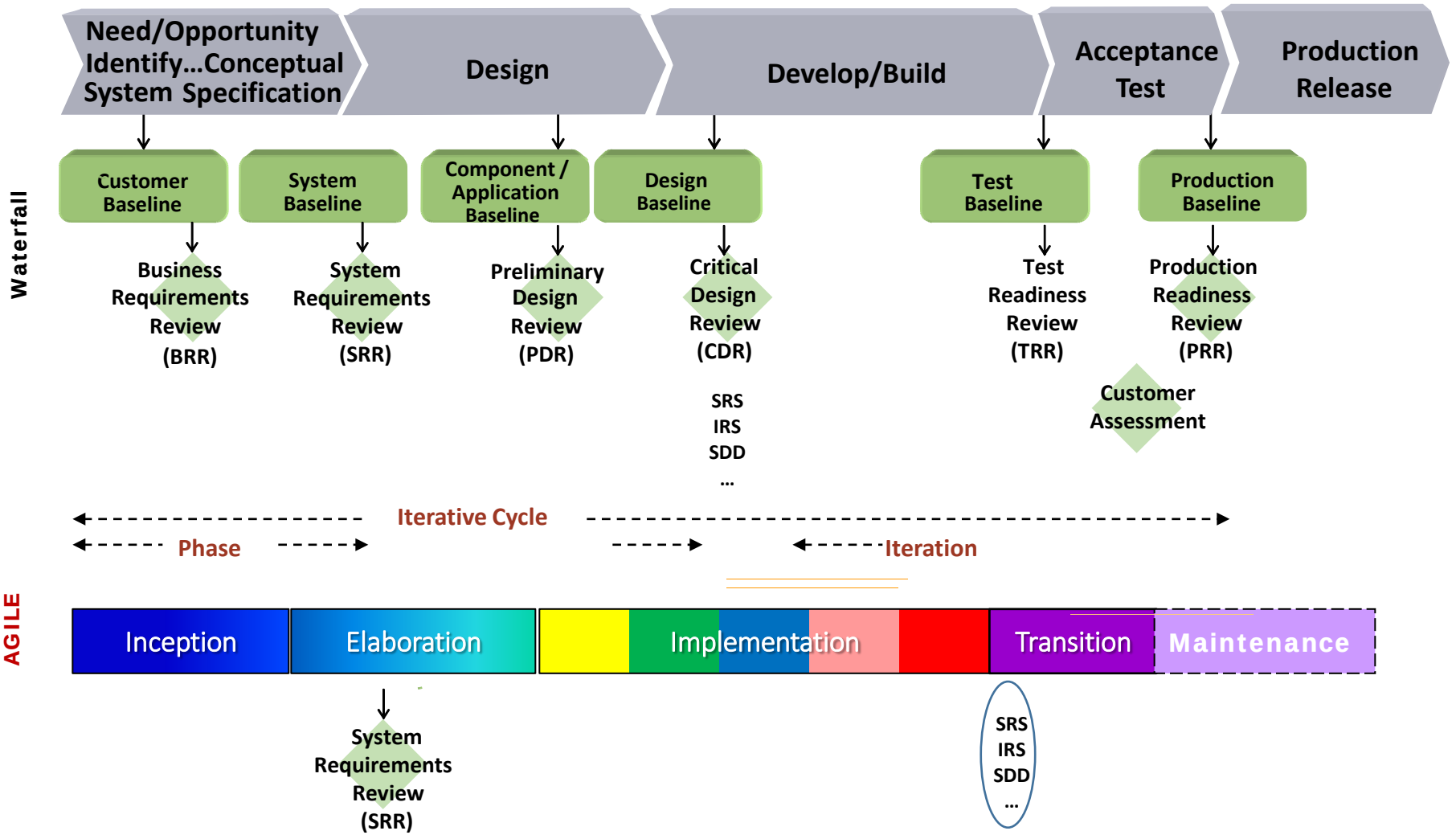
Agile Development

- **Definition**

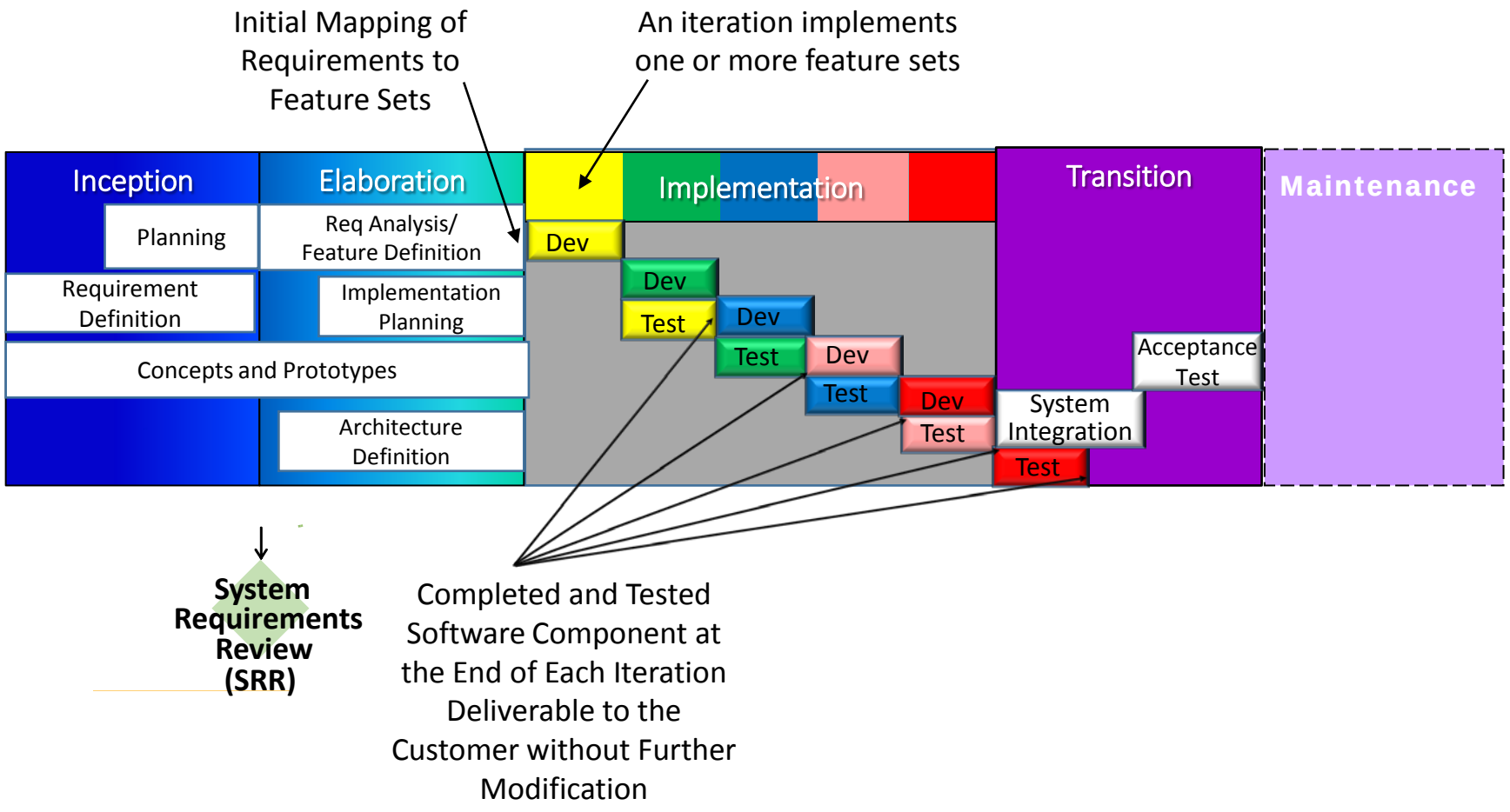
> Agile software development is a group of software development methods in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, continuous improvement and encourages rapid and flexible response to change.

- **The Agile Manifesto is based on 12 principles:**
  - Customer satisfaction by rapid delivery of useful software
  - Welcome changing requirements, even late in development
  - Working software is delivered frequently (weeks rather than months)
  - Close cooperation between business people, the customer, and developers
  - Projects are built around motivated individuals, who should be trusted
  - Face-to-face conversation is the best form of communication (co-location)
  - Working software is the principal measure of progress
  - Sustainable development, able to maintain a constant pace
  - Continuous attention to technical excellence and good design
  - Simplicity—the art of maximizing the amount of work not done—is essential
  - Self-organizing teams
  - Regular adaptation to changing circumstances

System Agile/Traditional Engineering Process Overview



Agile Development Process

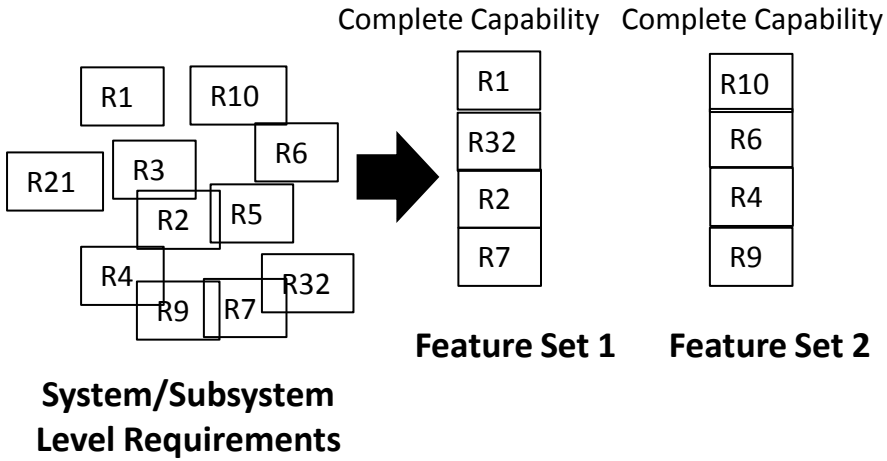


Benefits of Agile Development

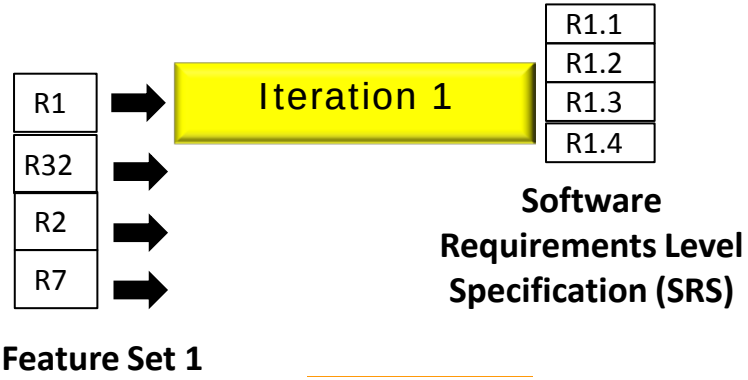
- **Optimal Implementation**
  - Traditional Waterfall Defines the Implementation at CDR thereby Potentially Constraining the S/W Developer in the Use of Commercial Libraries or More Optimal Design Approaches
- **Adaptable to Change**
  - The Iterative Approach Facilitates Change Late into the Development Cycle without Impact
- **Early Assessment of Product Features**
  - Each Iteration Provides a Completed Capability of the System that Can Be Demonstrated to the Customer
- **Continuous Testing**
  - Testing Is Initiated at the End of Iteration 1 Development and Continues without Interruption through the Nth Iteration thus Significantly Reducing the Final Integration and Acceptance Test Time.

Impacts of Agile Development

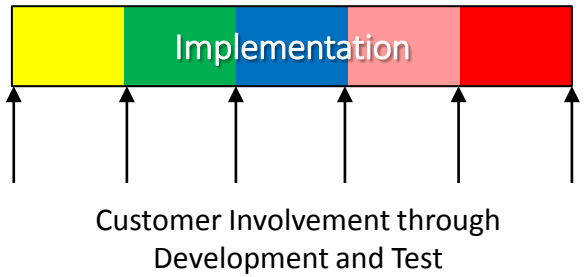
- **Functional Decomposition**
  - Feature Sets Represent Complete Capabilities that Once Built Become Testable Entities and Part of the Final Delivery without Modification



- **Detailed Requirement Definition**
  - SRS Level Requirements Defined During Development Based on Internal Interfaces, Optimal Use of Libraries, etc.



- **Customer Inter-action**
  - Requires Commitment of Time from the Customer for Regular Involvement in the Development and Test Process



Impacts of Agile Development Objectives of PDR and CDR

- **The following are typical objectives of a PDR:**
  - Ensure that all system requirements have been allocated, the **requirements are complete**, and the flow down is adequate to verify system performance
  - Show that the proposed design is expected to meet the functional and performance requirements
  - Show sufficient maturity in the proposed design approach to proceed to final design
  - Show that the design is verifiable and that the risks have been identified, characterized, and mitigated where appropriate
- **The following are typical objectives of a CDR:**
  - Ensure that the "build-to" baseline contains **detailed hardware and software specifications** that can meet functional and performance requirements
  - Ensure that the design has been satisfactorily audited by production, verification, operations, and other specialty engineering organizations
  - Ensure that the production processes and controls are sufficient to proceed to the fabrication stage
  - Establish that planned Quality Assurance (QA) activities will establish perceptive verification and screening processes for producing a quality product
  - Verify that **the final design** fulfills the specifications established at PDR

Issues/Concerns

- **Preliminary Design Review (PDR) and Critical Design Review (CDR) timing**
  - Traditional/Waterfall PDR/CDR occur **before** development starts
    - Not consistent with the Agile model
  - Lack of PDR/CDR limits the customers ability to understand the design and the architecture.
  - Holding PDR/CDR review after all design is done in agile would be at the end of the development cycle also

- **How to communicate requirements and design to a customer that is not present?**
  - Generate high-level designs, (PDR level) and review with the customer before the inception phase
  - Create formal system engineering cycles and give the customer time to review/provide input
  - Provide a light-weight way to show/discuss with customer (who may be remote) progress over an iteration.

Issues/Concerns

- **What Documentation Needs to Be Generated and When? (IRS, IDD, SDD, SRS, SPS)**
  - Build Documentation as a Part of the Development Process
  - Documentation Accurately Reflects that System as Built
  - Periodically provide requirement updates to the customer

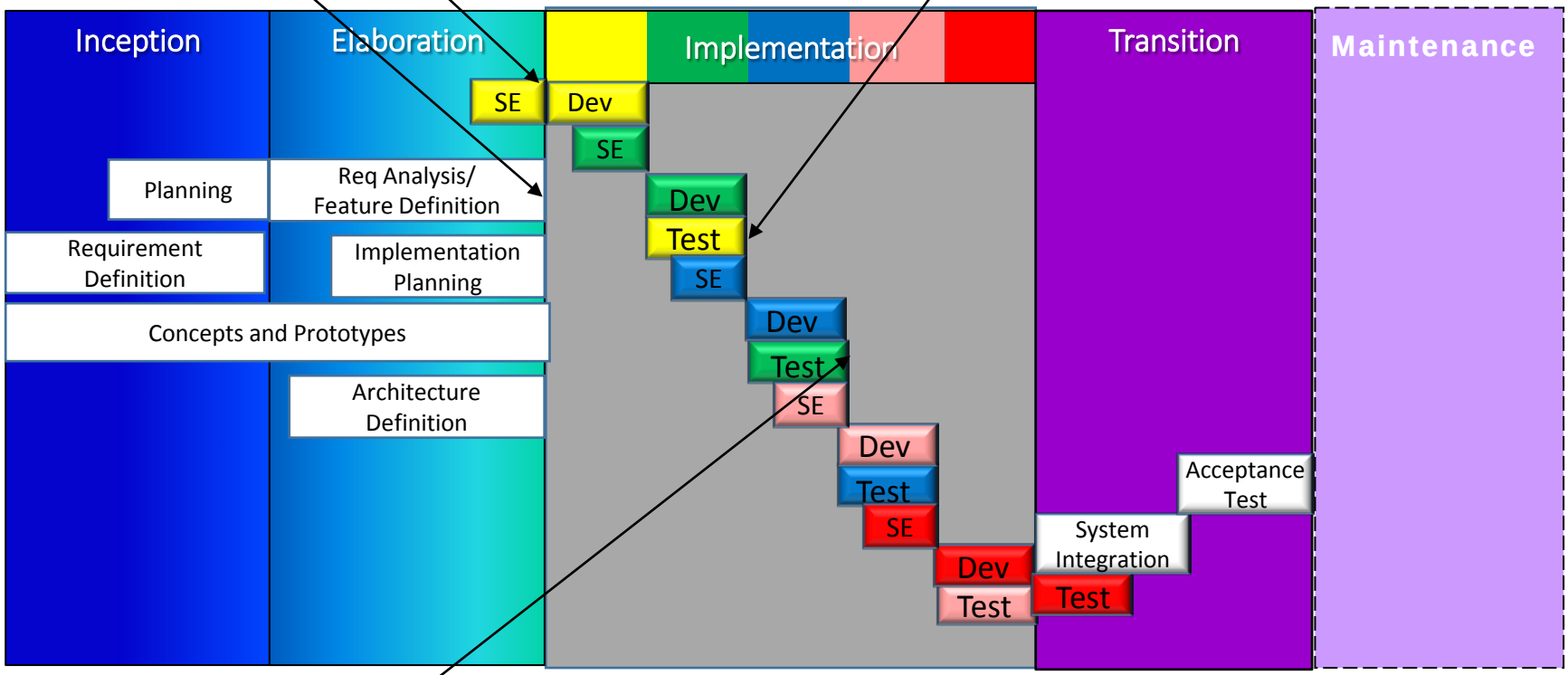
- **How Do I Get My Customer Involved in the Process?**
  - On-site Demonstrations
  - Bi-weekly or Monthly Meetings

Solutions

Initial Mapping of Requirements to Feature Sets

Initial SRS Requirements at the Beginning or Each Iteration

Periodic Demonstrations to Ensure Customer Satisfaction



Customer concurrence on the requirements implemented in the iteration and approval of SRS requirements for the next iteration

Create “Living” Documentation by Updating Each Document at the End of an Iteration Resulting in the Documentation Being a True Representation of the System

Summary

- **Agile Development Provides a Cost Effective Means for Ensuring Customer Expectations Are Met**
- **Agile Development Encourages Innovation by the Developer**
- **Tailoring of the Agile Development Process Can Aid the Transition for the Traditional Waterfall to the Agile Methodology**
  - Creating an Initial Feature Set Baseline
  - Defining an Initial Set of SRS Level Requirements at the Beginning of Each Iteration
  - Creating Traditional Documentation But at Different Points in the Development Timeline to Reflect the Agile Development Approach
  - Introducing Ways of Involving the Customer Throughout the Development Process