Project Management by Functional Capability

NDIA CMMI Technology Conference and User's Group November 15, 2007

Software Engineering Institute
Carnegie Mellon

Fred Schenker
Software Engineering Institute

CSCI

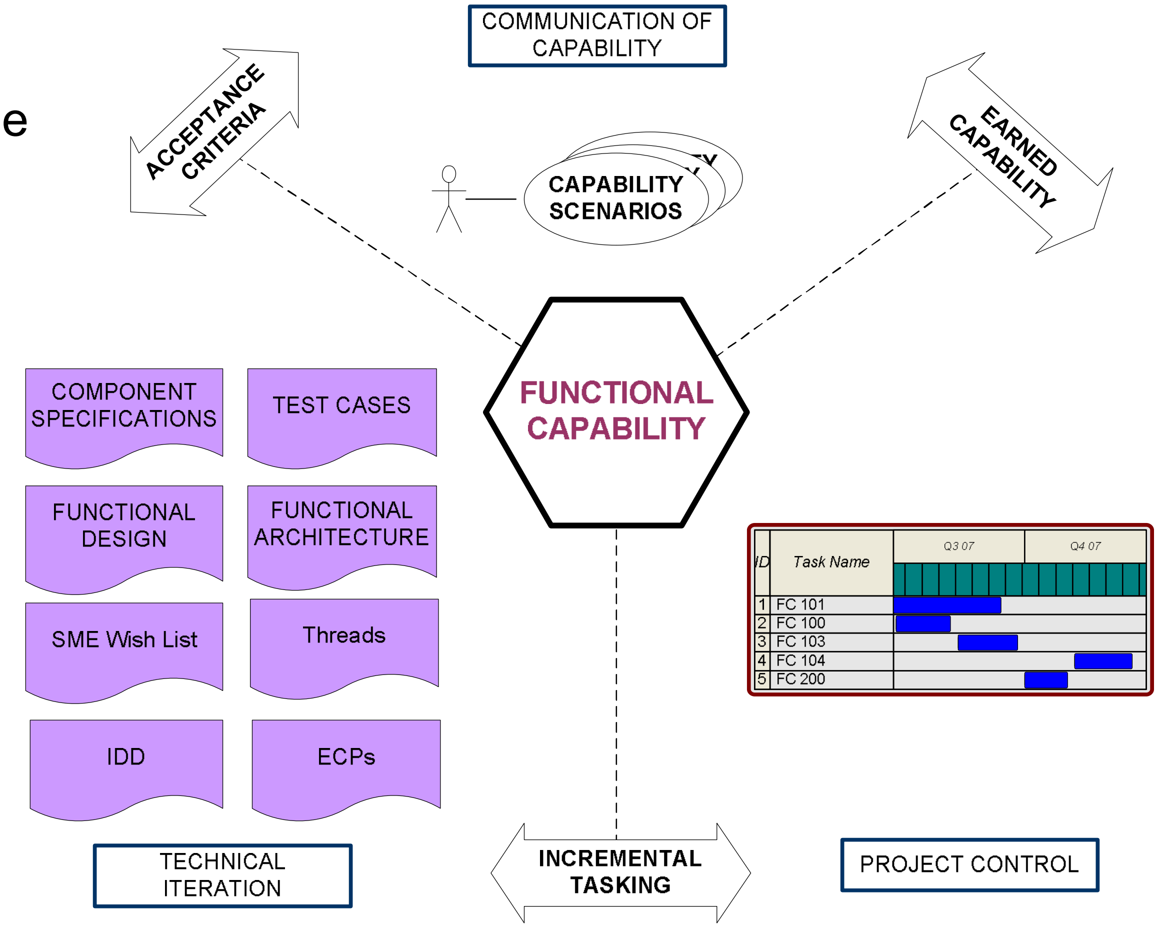
Bob Jacobs
Computer Systems Center Inc.

resentation

- To introduce Functional Capabilities (FCs) as a “useful” mechanism for managing work in a complex product development environment
  - An efficient way to communicate functionality to the user, the developer, and other stakeholders
  - A structure of discrete artifacts and flows that define product development lifecycle activities
    - logical design
    - system analysis, design and implementation
    - testing
  - A scheme for planning, tasking, and tracking work
  - An effective generator of artifacts for CMMI
- To share experiences gained from initial deployment of this project management process

ability – Context

- Consider your *Program* to be a large amount of *functionality,* expressed as *capabilities*
- *Functional* decomposition will provide increments of work to be accomplished, resulting in incremental *capability*
- We are proposing *functional capabilities* as a project management scheme to help deliver:
  - Å the right product
  - Å delivered on time and within budget



- Problem Statement
- SIAP
- Program Performance
- Functional Capability Overview
- Functional Capability Elaboration
- CMMI Mapping
- Summary

nent

- Product developers routinely fail to execute their projects
  - GAO Report 05/301, 2005
  - Defense Acquisition Performance Assessment, 2006
- How do acquirers gain insight into their project’s performance?
  - Does developer CMMI ML significantly affect project performance? If not, why not?
- How do contractors know they are producing what their customer wants?
- Do we need a different project context for Systems of Systems (SoS)?
  - CMU/SEI-2006-TR-017, Systems of Systems: “Scaling Up the Development Process”

munication of Capability

- Capability must be expressed in user terms... What they want
  - Joint Capabilities Integration and Development System (JCIDS) is not sufficient
  - systems engineers need more expressive methods for requirements capture and development
- What they will get
  - “System” specifications (to drive developers) that users can relate directly to capabilities
- And how they know they are getting it
  - Earned value expressed in terms of capability, i.e., “earned capability”
    - performance-based earned value
    - assessment of functionality bow wave

Development Practices

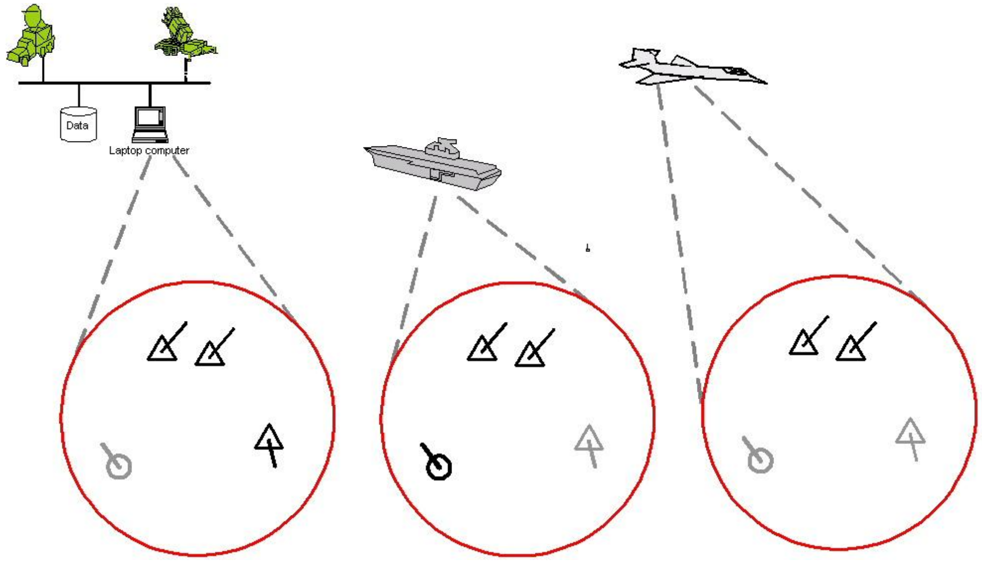
- SoS: Collaborating systems developed by collaborating system acquisition teams
  - ï highly autonomous systems and teams
- Process challenges in:
  - organizational ownership, responsibilities, and technical team interactions
  - systems:
    - boundary definition
    - legacy systems and continuous technology evolution
    - continuous capability evolution
  - project definition, measurement, and reporting mechanisms
  - project execution processes
- Practical process methods are needed

ingle Integrated Air Picture

Å FCs developed from experiences in SIAP

ð SIAP is a Software Intensive System

ð FCs should apply to SoS in general



Å SIAP Capability

ð **user viewpoint:** common, correct, complete, continuous, timely track situation presentation

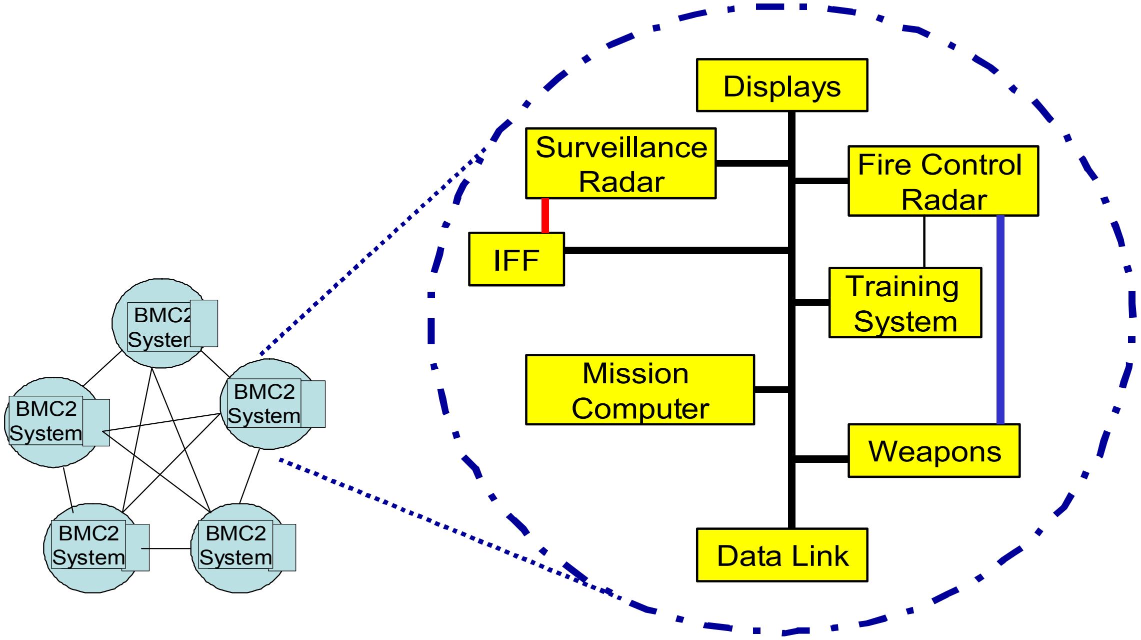
—**system viewpoint:** state of data consistency among distributed, replicated data stores, for objects of peer interest

DISCLAIMER: This presentation makes no statement concerning current SIAP engineering practices.

ity Material Challenge

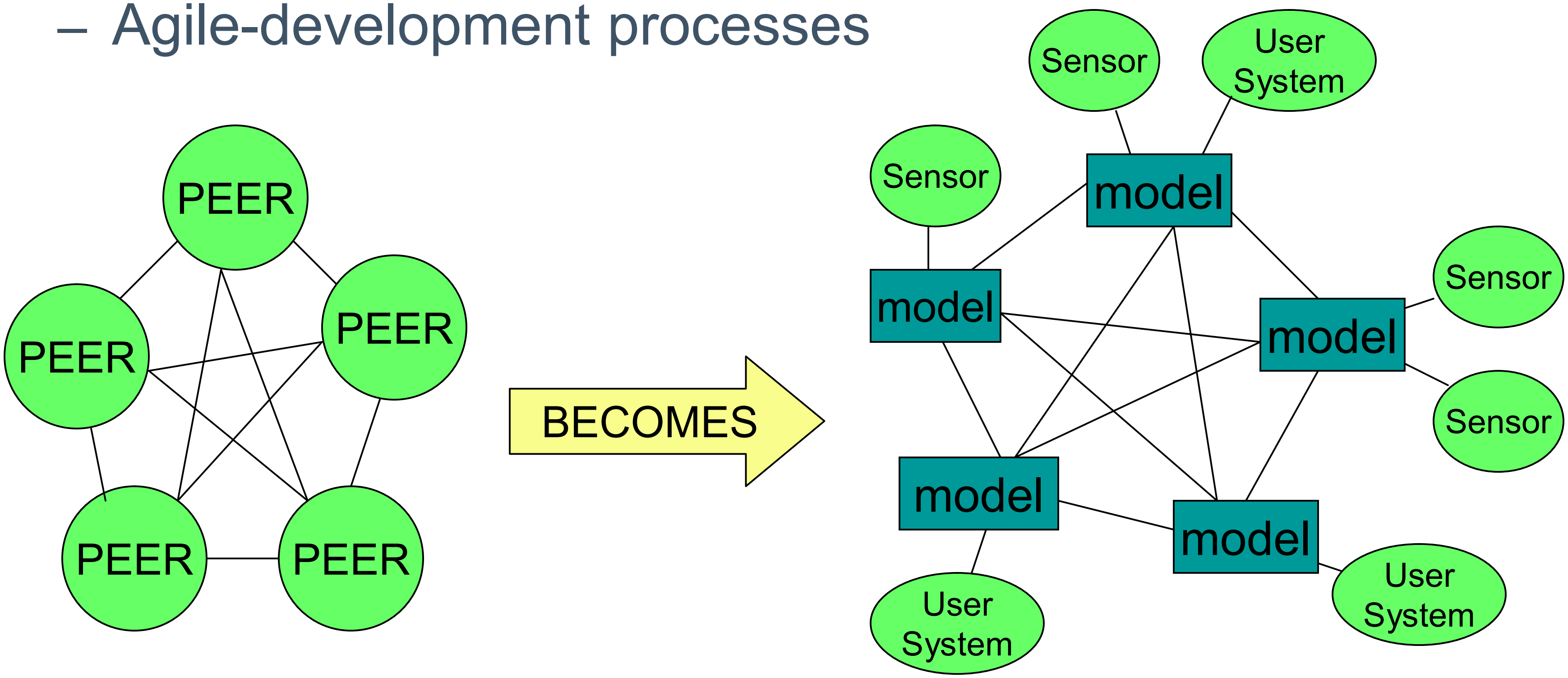
Å SIAP requires interactions of networked peers, each an operational node hosting multiple integrated systems

Å Network connections are weak, with ad hoc, dynamic configurations



ity Material Solution

- Executable Object Model transformable to code, with core required functionality
- Agile-development processes



Unpredictable Heterogeneous Set of Systems

Predictable, Logically Homogeneous Federation

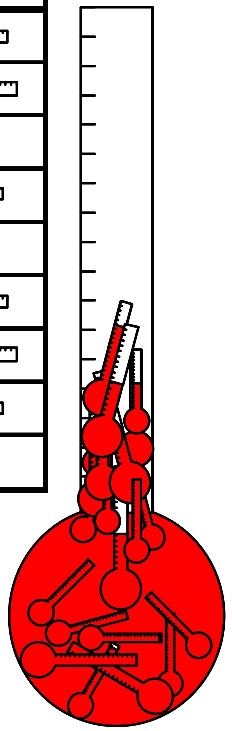
f Capability

- Functional Capabilities express *functional* requirements
  - manageable abstraction level for SoS
  - meaningful to user and developer
- An FC identifies a value-chain
  - tangible artifacts
  - framework for measuring program process performance
- An FC represents value that can be earned against a planned-performance baseline
  - an example of Performance-Based Earned Value®

ability – Earned Capability (Value)

FC #	Description	# Req	# Use Cases	# Scenarios	# IPT Affected	Pol. Vis.	Total	Status
FC 1		5	3	1	1	Hot	26	🔴
FC 2		49	8	3	3	Hot Hot	88	🔴
FC 2.1		18	2	2	1	Hot	24	🟡
FC 2.2		22	4	1	1	Hot Hot	34	🟡
FC 2.3		9	2	3	2	Medium	14	🟡
FC 3		13	6	2	2	Medium	39	🔴
FC 4		45	9	4	3	Hot	81	🔴
FC 4.1		33	6	2	2	Hot	46	🟡
FC 4.2		12	3	2	1	Medium	22	🟢

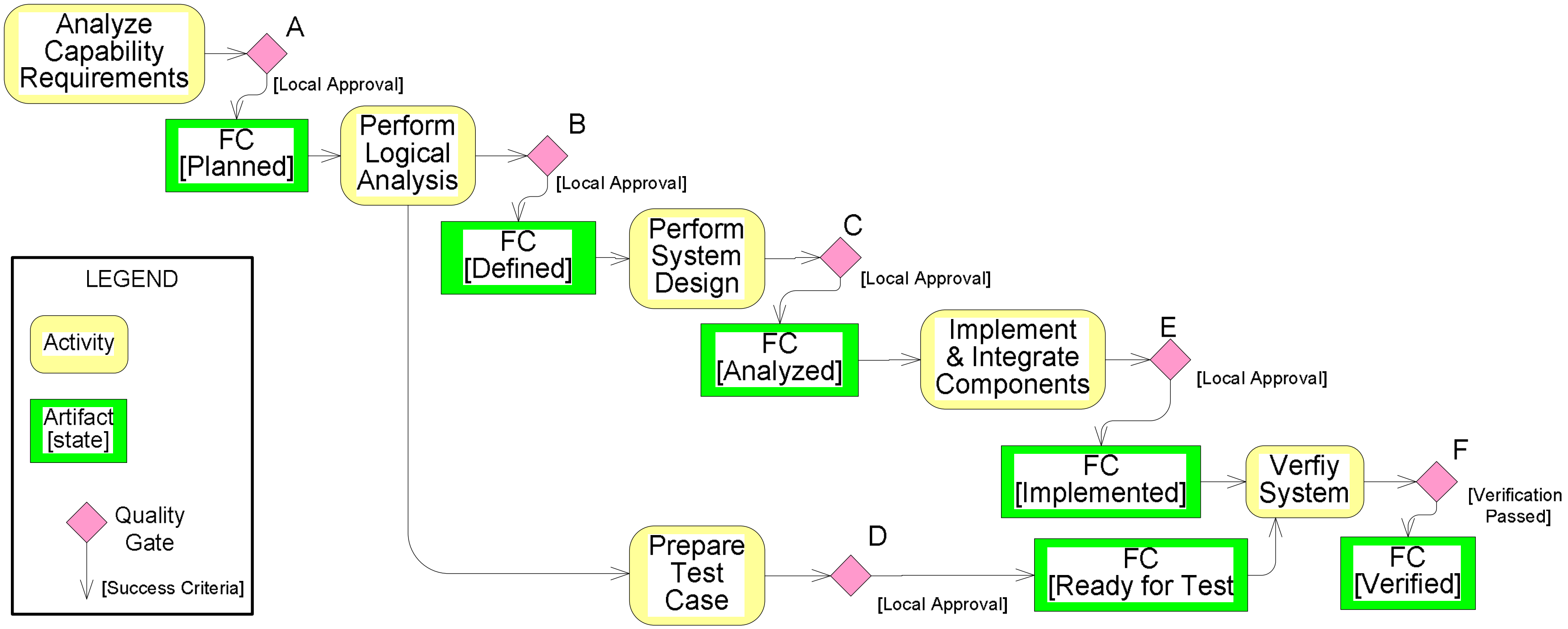
- Establish relative size measures for each capability
- Establish dependencies between capability projects
- Establish the approved list of capability (or value)
- Release work as appropriate and accrue “value” against the project capability “baseline” at Management reviews
- Measure project lifecycle task duration and effort to refine estimation process and establish project historical parametric data
- Capability can be “re-scoped”, but deviations from the baseline are easily recognizable as the “bow-wave” of functionality



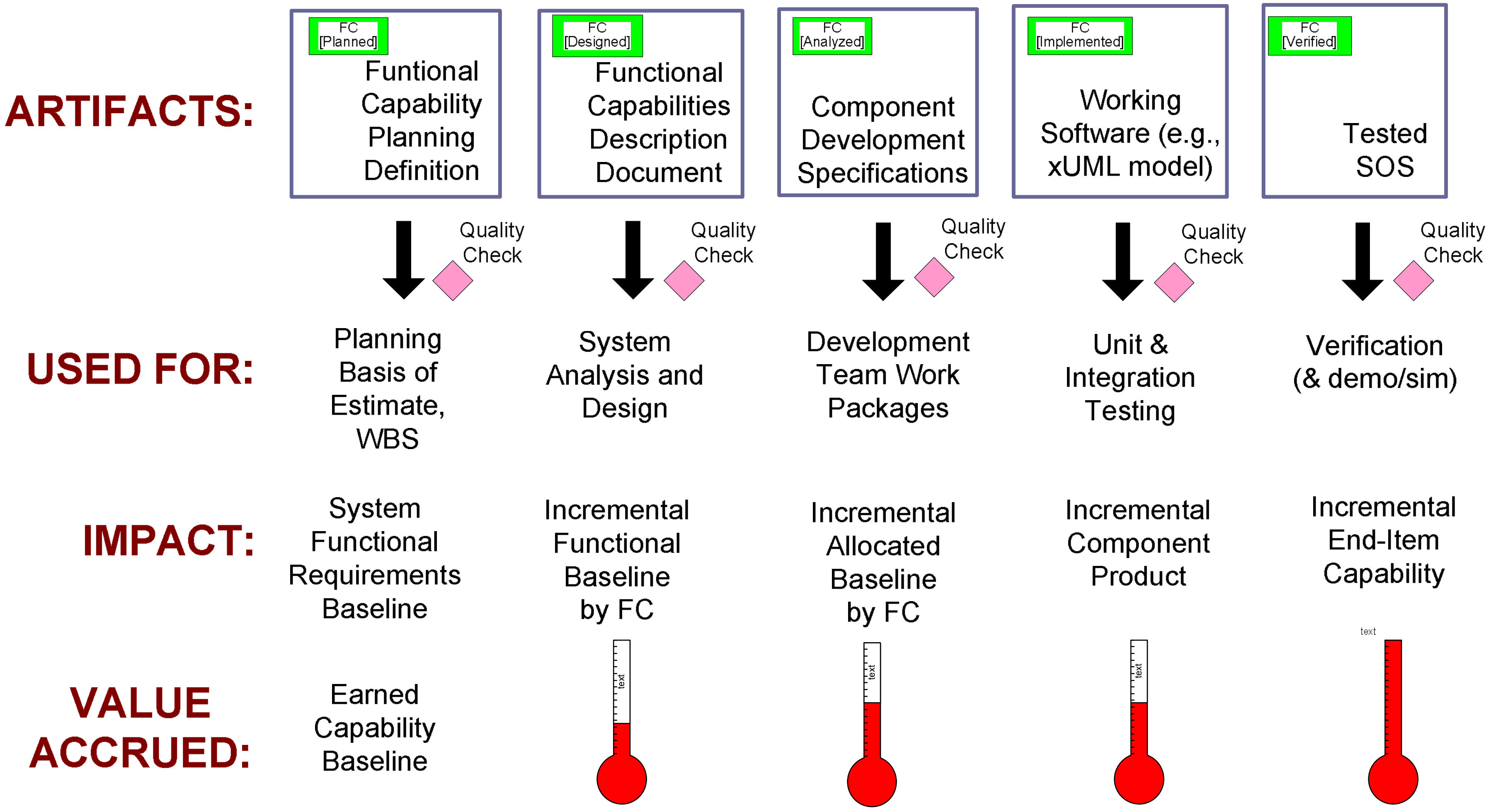
Capabil-o-meter

ability Life Cycle

- Each FC advances through lifecycle phases, representing states of completion, defined by artifacts
- Artifacts are reviewed at Quality gates, providing evidence of value

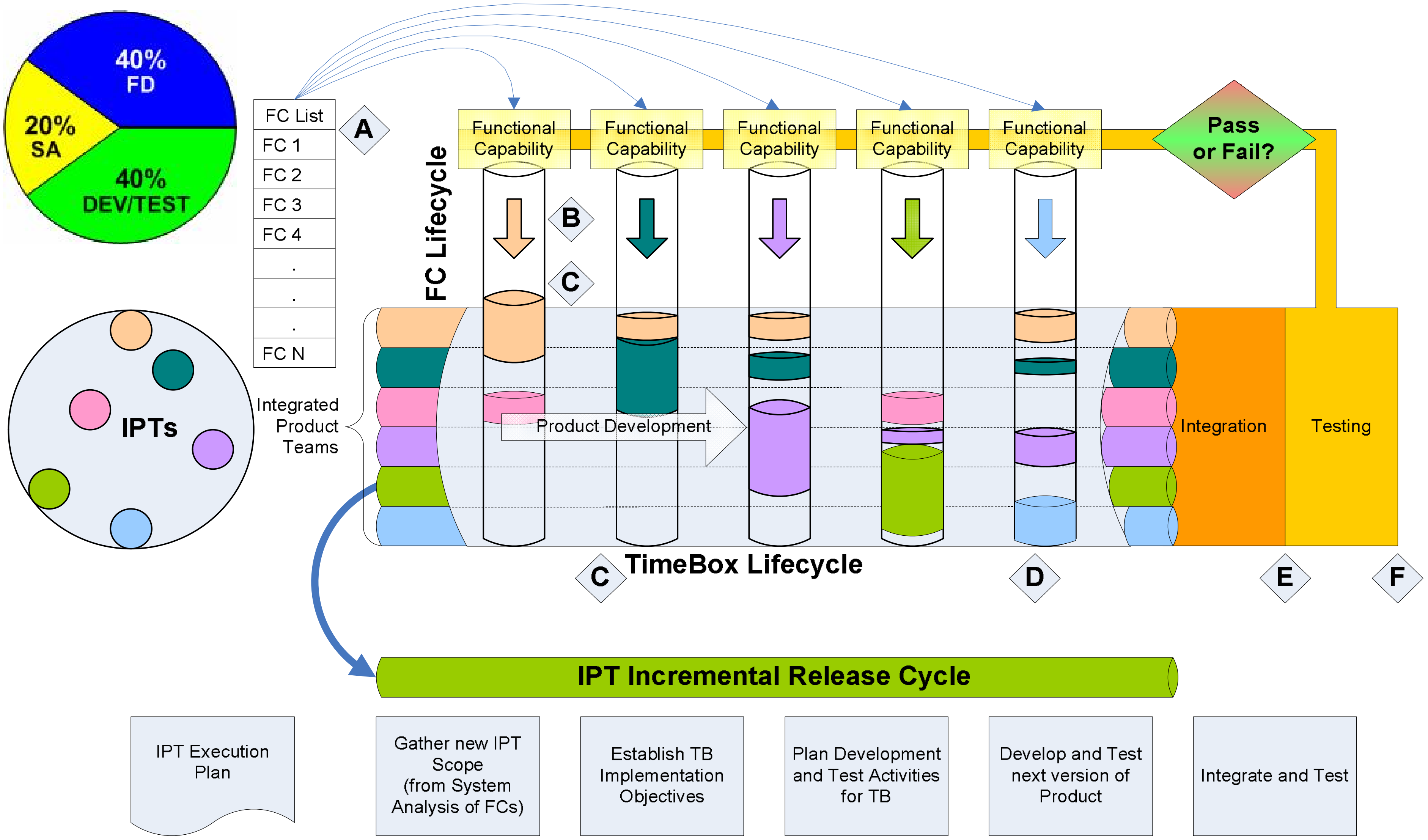


d Value



ARTIFACTS:	Funtional Capability Planning Definition	Functional Capabilities Description Document	Component Development Specifications	Working Software (e.g., xUML model)	Tested SOS
USED FOR:	Planning Basis of Estimate, WBS	System Analysis and Design	Development Team Work Packages	Unit & Integration Testing	Verification (& demo/sim)
IMPACT:	System Functional Requirements Baseline	Incremental Functional Baseline by FC	Incremental Allocated Baseline by FC	Incremental Component Product	Incremental End-Item Capability
VALUE ACCRUED:	Earned Capability Baseline				

ability – Overview



40% FD

20% SA

40% DEV/TEST

IPTs

Integrated Product Teams

FC List

FC 1

FC 2

FC 3

FC 4

.

.

.

FC N

A

FC Lifecycle

Functional Capability

Functional Capability

Functional Capability

Functional Capability

Functional Capability

Pass or Fail?

B

C

Product Development

Integration

Testing

C

TimeBox Lifecycle

D

E

F

IPT Incremental Release Cycle

IPT Execution Plan

Gather new IPT Scope (from System Analysis of FCs)

Establish TB Implementation Objectives

Plan Development and Test Activities for TB

Develop and Test next version of Product

Integrate and Test

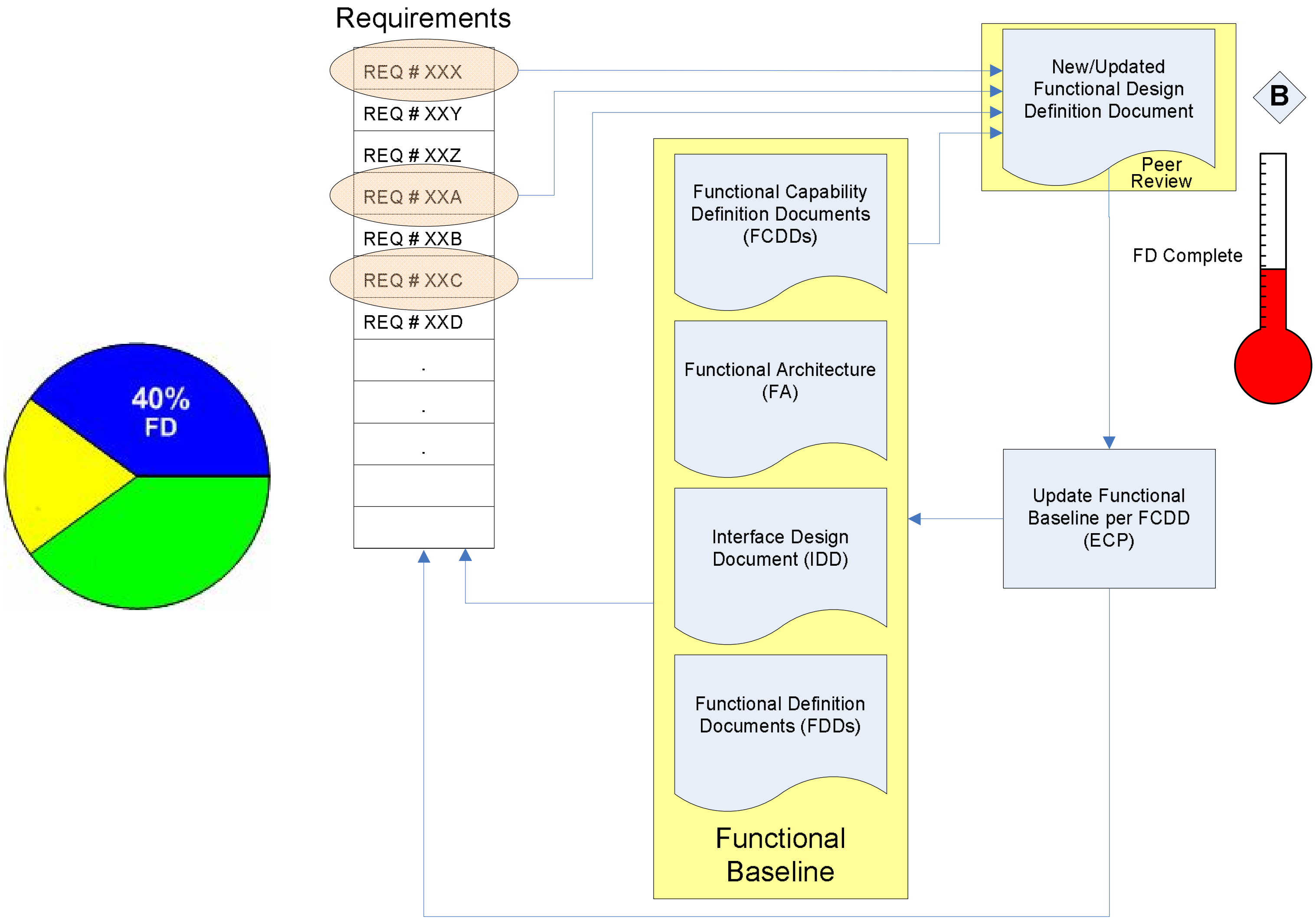
ability – Planning Definition

- Early in the Program Lifecycle, Functional Capability planning definitions are needed:
  - Based on End-to-End mission scenarios
  - No more than one or two pages per FC
  - Preliminary allocation of requirements
  - High-level textual description
  - Basis of estimates for effort, resource, and schedule planning (use cases, complexity, requirements, etc.)
  - Use historical data where possible (and practical)
  - Establish FC priority and FC-FC dependencies
- Use the planning definitions to establish Earned Capability baseline and to scope project deliverables and dates

ability – Functional Definition

- Refine the scenarios to specify the capabilities
- Finalize allocation of functional requirements to the notional FC
- Elaborate the FC
  - Create a contextual description of the functionality
  - Create sequence diagrams, use cases, behavior diagrams
  - Ensure the allocated requirements are explained adequately in the context of the functionality
  - Provide criteria for FC acceptance
- Validate the FC
  - Peer review
  - Customer review
  - Management review (Q-Gate)

ability – Functional Definition



Requirements

REQ # XXX

REQ # XXY

REQ # XXZ

REQ # XXA

REQ # XXB

REQ # XXC

REQ # XXD

.

.

.

New/Updated Functional Design Definition Document

Peer Review

B

FD Complete

Functional Capability Definition Documents (FCDDs)

Functional Architecture (FA)

Interface Design Document (IDD)

Functional Definition Documents (FDDs)

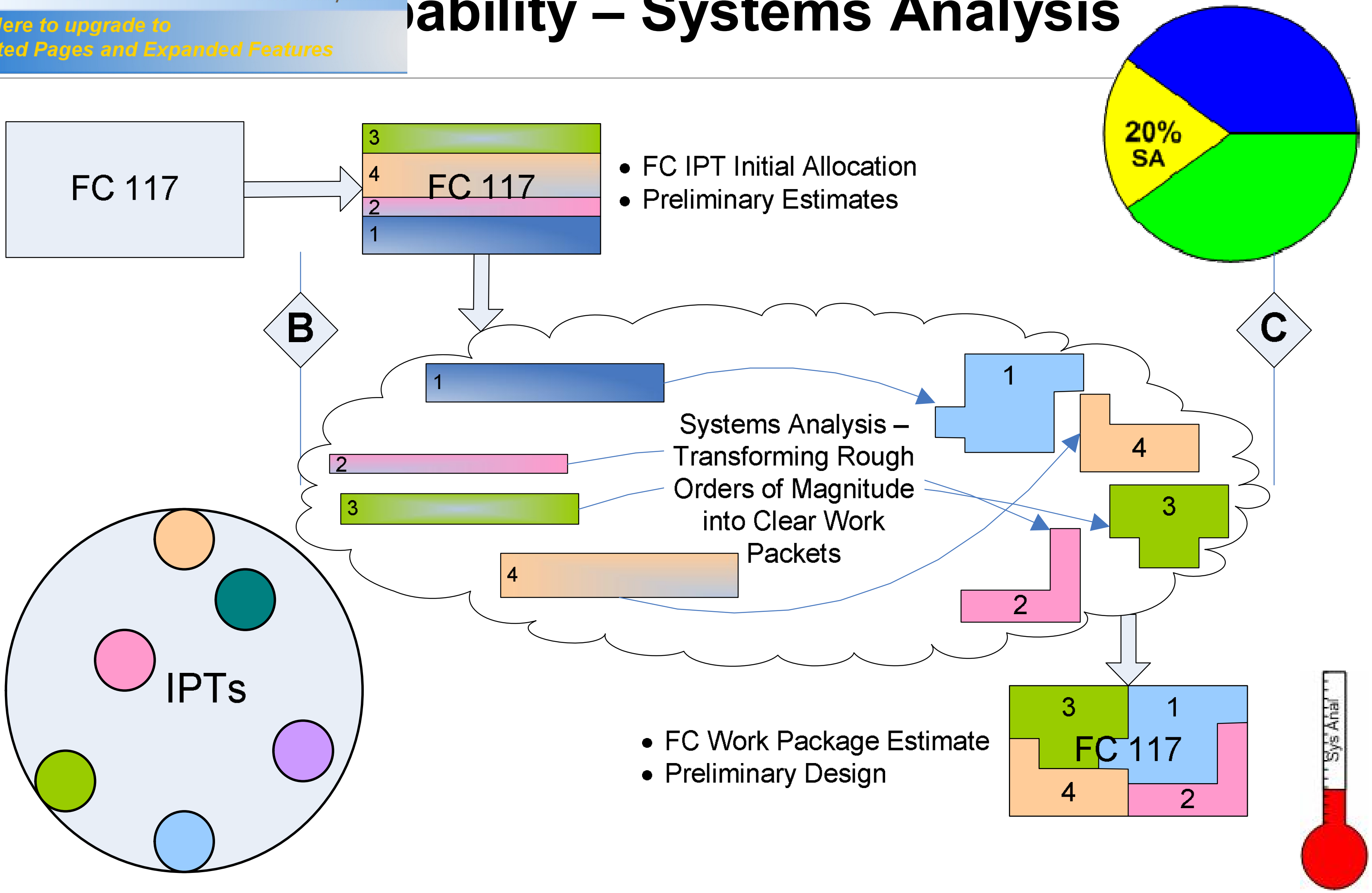
Functional Baseline

Update Functional Baseline per FCDD (ECP)

pability – Systems Analysis

- Start with validated functional design
- Allocate functionality to legacy components
  - Identify and analyze design alternatives as necessary, especially for risk mitigation
  - Update existing / create new design documentation, component specifications
  - Create work packages to implement the new designs
  - Update previous estimates of effort and schedule
  - Identify task dependencies, establish need for commitments for inter-component deliverables
- Validate the Analysis
  - Peer review
  - Customer review
  - Management review (Q-Gate)

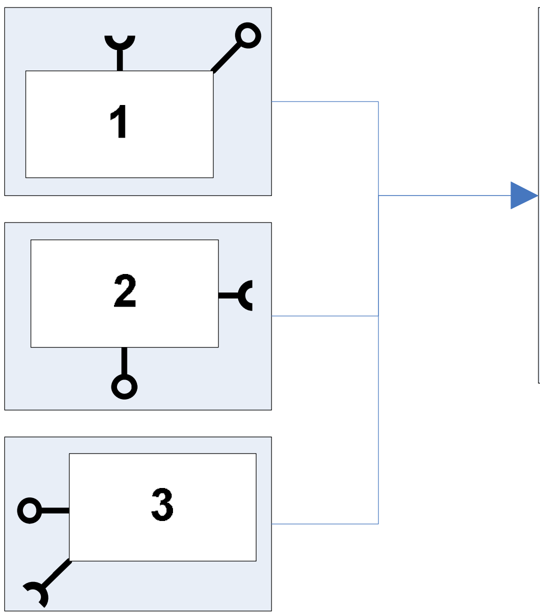
ability – Systems Analysis

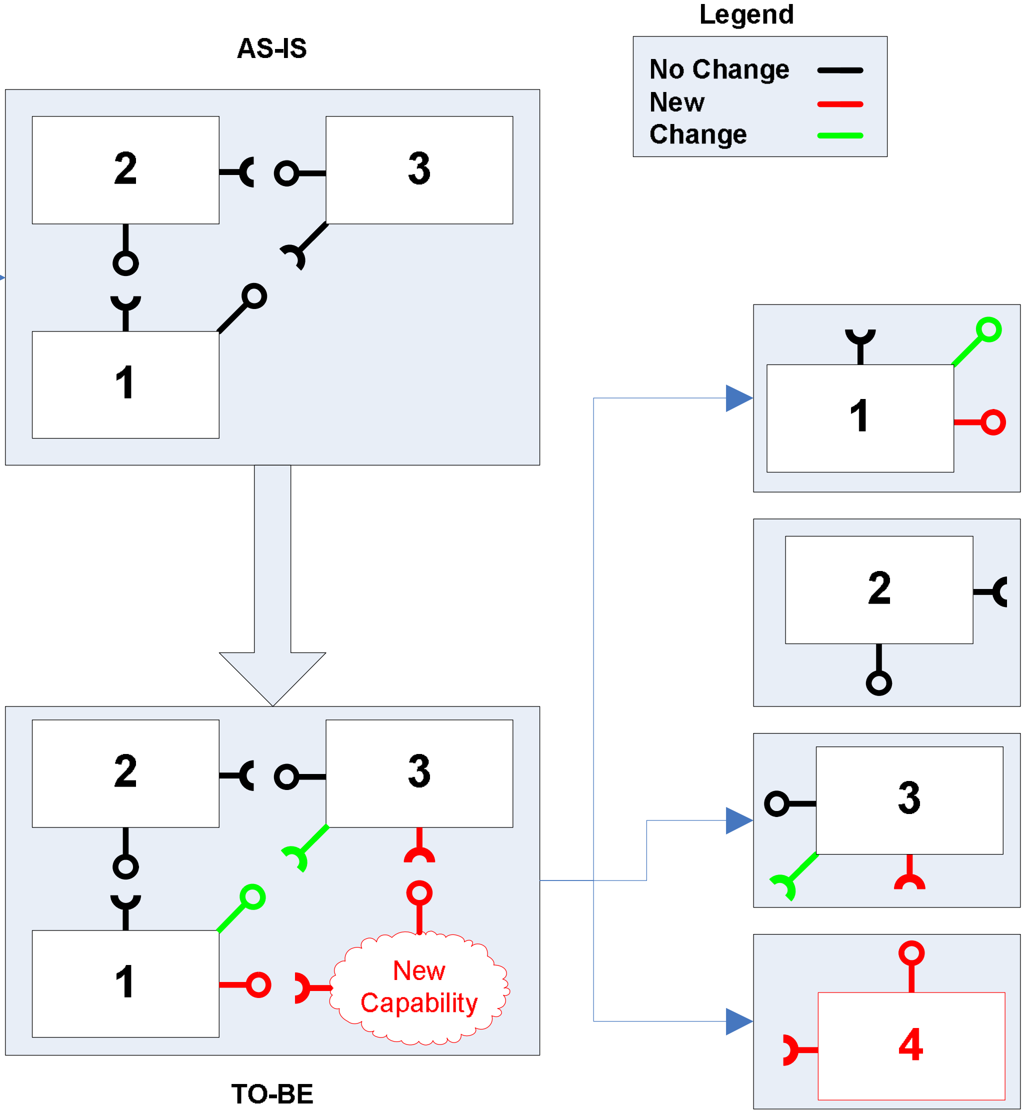

20%
SA
FC 117
3
4
FC 117
2
1
B
C
1
2
3
4
Systems Analysis – Transforming Rough Orders of Magnitude into Clear Work Packets
1
4
3
2
IPTs
3
1
FC 117
4
2
Sys Anal

- FC IPT Initial Allocation
- Preliminary Estimates

- FC Work Package Estimate
- Preliminary Design

ability – Systems Analysis





Para. #	Title	Affected Area	Type of Change	Work Ref. #
1.1				
1.2				
1.3				
2.1				

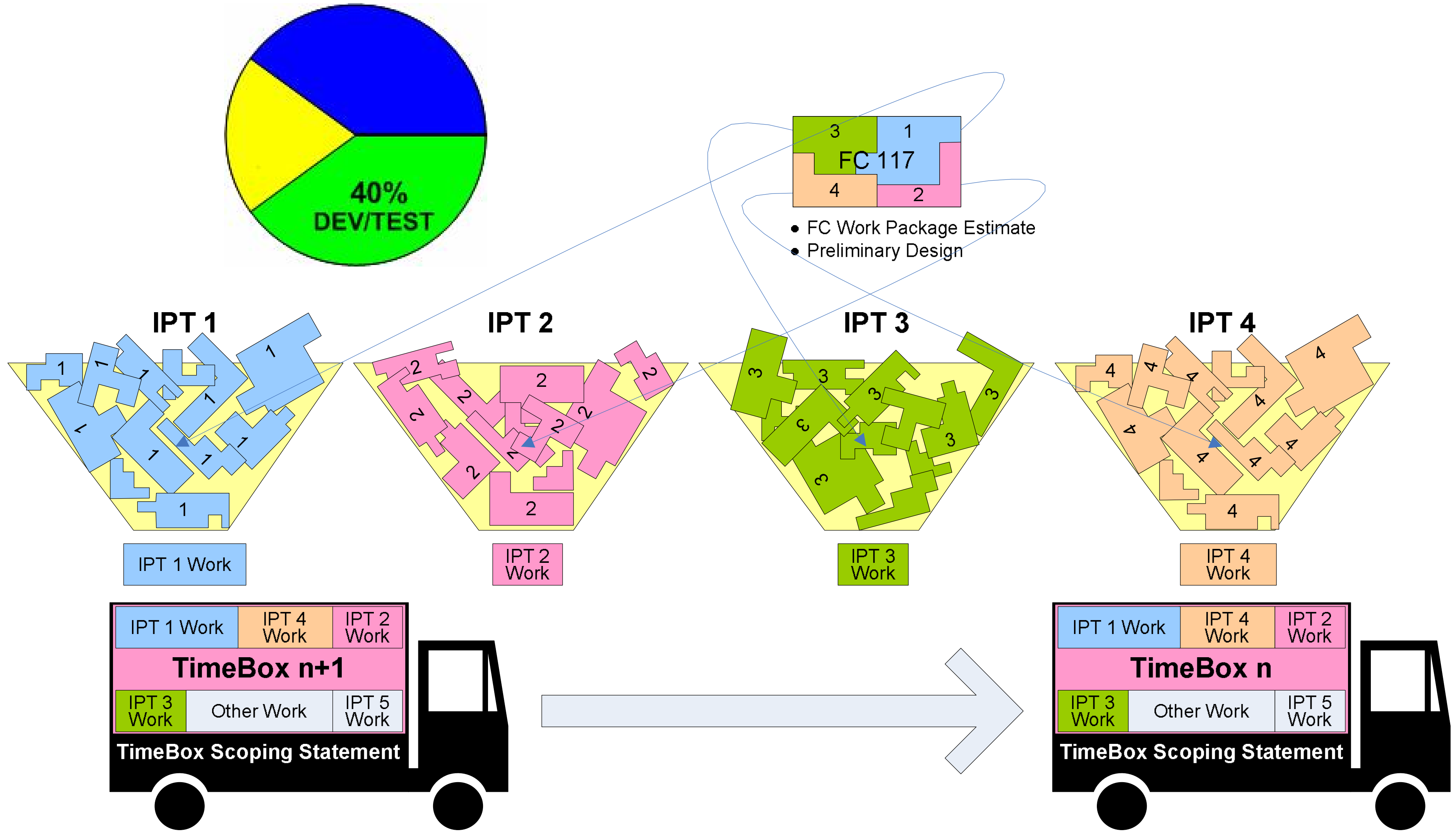
ability – Test Preparation

- Start with Functional Capability Definition Document requirements acceptance criteria
  - Review the acceptance criteria
    - New scenarios that need to be instantiated
    - New requirements that need to be verified
    - Legacy requirements that have been further clarified
  - Develop/modify test cases based on the criteria
  - If necessary, create new scenario (data set)
  - Identify need for additional test tools, and develop those tools
- Validate the Test Preparation
  - Peer review test cases and scenarios
  - Management review (Q-Gate)

ability – Dev. & Int.

- Start with validated System Analysis
- Coordinate the tasks so that the Functional Capability is achieved
  - Identify and negotiate commitments between development teams
  - Establish development goals for the next increment of production (TimeBox)
  - Execute tasks in accordance with the plan
  - Perform verification tasks and pass on to integration
- Integrate the new products
  - Check interfaces, build new integrated product
  - Verify new build (smoke test)
- Validate the Development and Integration
  - Management Review (Q-Gate)

ability – Dev. & Int.

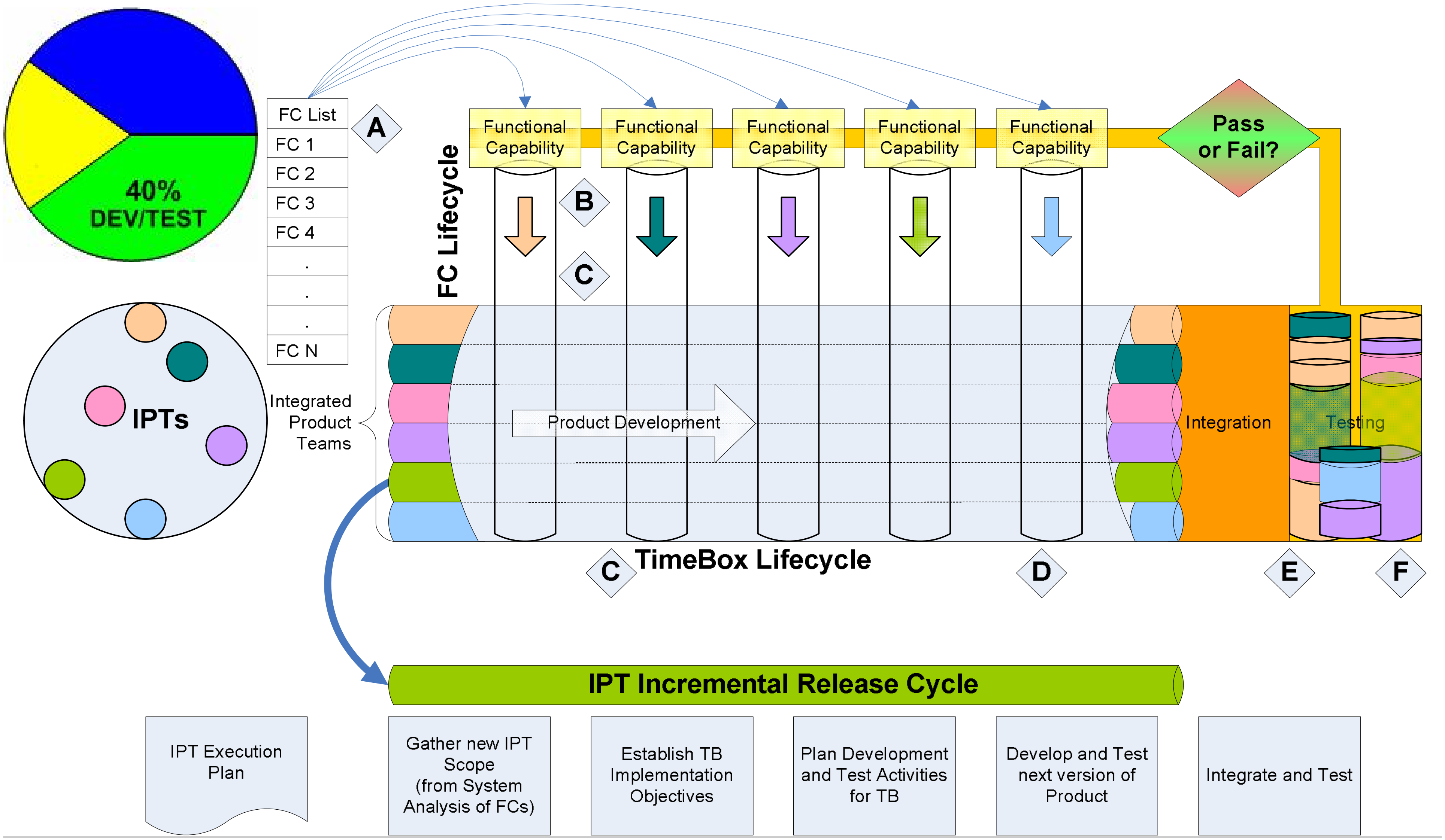


- FC Work Package Estimate
- Preliminary Design

ability – System Test

- Start with stable production build
  - Regression test (with new test cases)
  - Log bugs/defects
  - Perform SoS simulated testing (if possible)
  - Evaluate performance bottlenecks; potential SoS issues
  - Produce test report
- Validate the results
  - Management review (Q-gate)

ability – System Test



CMMI

Q: So what does this have to with CMMI anyway?

This is the CMMI User’s Conference, right?

A1: If you adopt the Functional Capability lifecycle, you get a lot of CMMI credit…

A2: If you managed your projects this way you could use CMMI practices (esp. M&A) to help you

- Produce what your customers want
- Make sure your contractor is performing

pabilities – CMMI Mapping 1

- Project Planning (SG 1, SG 2, SG 3)
  - Estimation of FC scope (size, complexity, effort, priority)
  - Standard FC WBS
  - Defined FC lifecycle
  - FC implementation risks
  - Stakeholder identification and involvement (FC prioritization)
  - FC Implementation Budget and Schedule (FC Owners ≈ CAMs)
  - Summation of FC Planning Definitions (Baseline Plan)
  - Commitments established between IPTs
- Project Monitoring and Control (SG 1)
  - Defined project milestones (Q-Gates)
  - “Earned” Capability to calibrate program performance

abilities – CMMI Mapping 2

- Requirements Development (SG 1, SG 2, SG 3)
  - Stakeholder “needs” documented (or referenced) in FCDD, and validated via peer review
  - Context for requirement implementation and acceptance criteria provided in FCDD
    - Basis for product component and interface requirements
    - Definition of required functionality
    - Basis for requirements validation
  - Use cases documented in the FCDD (Operational concepts and scenarios)
- Technical Solution (SG 1, SG 2, SG 3)
  - Alternative solutions documented in FCDD and propagated through System Analysis of FC
  - FCDD represents documentation of Functional design

abilities – CMMI Mapping 3

- Requirements Management (SG 1)
  - FCDD helps to develop an understanding of requirements
  - FCDD to Requirements trace useful for identifying impact of changes
- Verification (SG 1, SG 2, SG 3)
  - Requirements Verification acceptance criteria defined in FCDD
  - Defined artifacts represent obvious opportunities for Peer Review
- Validation (SG 1, SG 2)
  - Defined artifacts are used to interpret, communicate and validate product design
  - Product lifecycle defines artifacts, essential for planning validation activities

abilities – CMMI Mapping 4

- Integrated Project Management (SG 2)
  - FC Definition Document provides basis for management of stakeholder involvement, dependencies, and identification (and resolution) of coordination issues
- Measurement and Analysis (SG 1, SG 2)
  - FC baseline represents program commitment
  - Tracking of FC progress connects tasks execution to management information needs
- Quantitative Project Management (SG 1, SG 2)
  - FC baseline represents the program’s performance objective
  - Tracking of FC progress helps to determine whether the program’s objectives for performance are being satisfied, and are used to identify appropriate corrective actions

- Functional Capability provides a useful framework for managing projects
  - In a complex environment (SoS)
  - As a significant contributor of value-adding artifacts
  - As a starting point for introducing quantitative methods into the project management process
  - As a means of communicating capability, both desired and earned
  - As an effective means to deliver relevant technical and project management content to external stakeholders
  - As a method of assessing the “bow-wave” on a project, and calibrating the reported earned value

Thank you for your attention!!

ation

Fred Schenker
Software Engineering Institute
4500 5th Avenue
Pittsburgh, PA 15213-3890

412-268-9145
ars@sei.cmu.edu

Robert Jacobs
Computer Systems Center Inc.
6225 Brandon Ave.
Springfield, VA 22150

703-866-4000
bjacobs@csci-va.com