Quality Assurance in Context. Quality Engineering

Lecture 2

Outline

- Quality Assurance in context
  - Handling discovered defect during QA activities
  - QA activities in software processes
  - Verification and Validation perspectives
  - V&V view and Defect-Centered view
- Quality Engineering
  - Activities and process
  - Quality planning& goal setting and strategy formation
  - Quality assessment and improvement
  - Quality engineering in software processes
- Conclusion

Defect handling and related activities

- Defect resolution
  - ensures that each discovered defect is corrected or taken care of through appropriate actions
  - Each corrected or fixed defect needs to be re-verified to ensure failure-free executions under the same execution conditions
- Activities, supporting defect resolution
  - Defect logging
    - ✓ Initial reporting and recording of discovered defect
  - Defect tracking
    - ✓ Monitors and records what happened to each defect after its initial discovery
- Defect discovery and resolution activities
  - Consistent defect interpretation and tracking
    - ✓ Distinguishing execution failure, internal faults and human errors
  - Timely defect reporting

Handling discovered defect during QA activities

- Defect handling/resolution
  - status and tracking
  - causal (root-cause) analysis
  - resolution: defect removal/etc.
  - improvement: break causal chain
- Defect measurement:
  - parallel to defect handling
  - where injected/found?
  - type/severity/impact?
  - more detailed classification possible?
  - consistent interpretation
  - timely defect reporting
- Defect analyses/quality models
  - as follow up to defect handling.
  - data and historical baselines
  - goal: assessment/prediction/improvement
  - causal/risk/reliability/etc. analyses

Defect handling process and tools

- A formalized defect handling process highlights:
  - important activities and associated rules,
  - parties involved, and their responsibilities
- A formalized defect handling process is defined by:
  - the different states associated with individual defect status and transitions among these states due to status changes
  - "new" ➔ "working" ➔ "re-verify" ➔ …… ➔ "closed"
- Software defect handling tools
  - Proprietary
    - ✓ CMVC - Configuration Management Version Control (IBM),
    - ✓ FogBugz, JIRA, Gemini, YouTrack
  - Free software
    - ✓ Bugzilla (www.bugzilla.org)
    - ✓ BugTracker.NET, Trac, Redmine

Defect handling in different QA activities

Which are the 3 classes of QA activities?

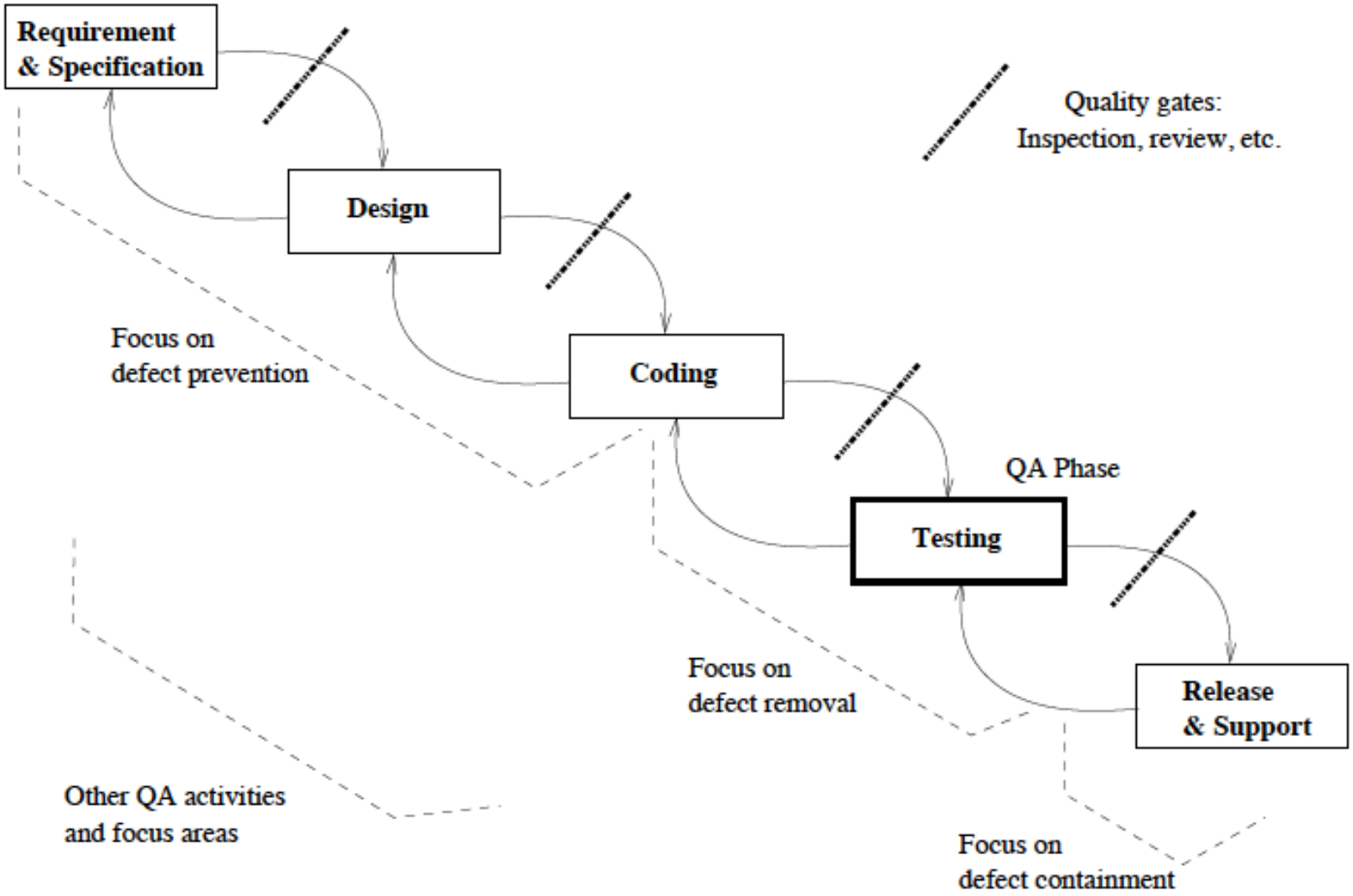
Defect handling in different QA activities

- Defect prevention activities
  - Do not directly deal with defects or the discovered faults
  - Deal with various ways to prevent injection of faults into the software
  - There are little or no discovered faults
- Defect detection and removal
  - Testing and inspection activities are closely related with defect handling
- Defect containment
  - Faults are not typically identified, while their dynamic impact was tolerated and corrected
  - There is little or no discovered faults

QA activities in software processes

- Mega-process:
  - initiation, development, maintenance, termination.
- Development process components:
  - requirement, specification, design, coding, testing, release.
- Process variations:
  - waterfall development process
  - iterative development process
  - spiral development process
  - agile development processes and XP (extreme programming)
  - maintenance process
  - mixed/synthesized/customized processes
- QA important in all processes

QA activities in waterfall process



QA activities in waterfall process 2

- QA throughout process
  - defect prevention in early phases
  - focused defect removal in testing phase
  - defect containment in late phases
  - phase transitions: inspection/review/etc

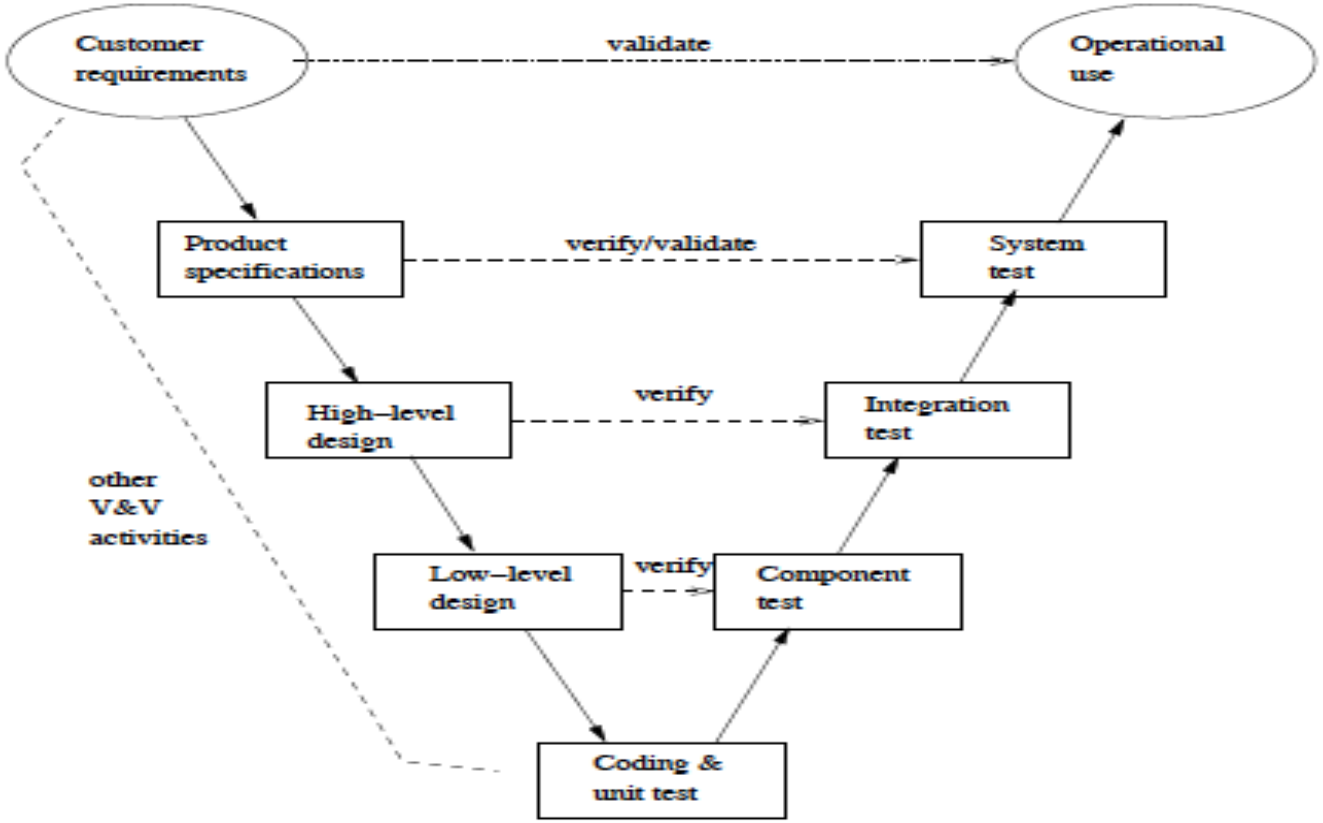
QA in sofware processes

- Process variations and QA:
  - iterative: QA in iterations/increments
  - spiral: QA and risk management
  - XP: test-driven development
  - mixed/synthesized: case specific
  - more evenly distributed QA activities
- QA in maintenance processes:
  - focus on defect handling;
  - some defect containment activities for critical or highly-dependable systems;
  - data for future QA activities
- QA scattered throughout all processes

V&V

- Core QA activities grouped into V&V.
- Validation: w.r.t. requirement (what?)
  - appropriate/fit-for-use/”right thing"?
  - scenario and usage inspection/testing;
  - system/integration/acceptance testing;
  - beta testing and operational support.
- Verification: w.r.t. specification/design (how?)
  - Correct/”doing things right"?
  - design as specification for components;
  - structural and functional testing;
  - inspections and formal verification.

V&V in software process



V&V in V-model above :

- V-model as bent-over waterfall
- left-arm: implementation (& V&V)
- right-arm: testing (& V&V)
- user@top vs. developer@bottom

V&V vs DC-view

- Two views of QA:
  - V&V view
  - DC (defect-centered) view in this course
  - Interconnected: mapping possible?
- Mapping between V&V and DC view:
  - V&V after commitment

  (defect injected already) ➔defect removal & containment focus
  - Verification: more internal focus
  - Validation: more external focus
  - In V-model: closer to user (near top) or developer (near bottom)?

  - Defect prevention – earlier phases
  - Defect reduction – middle or late phases
  - Defect containment – operation phases

DC - V&V mapping

DC-view class	QA activity	V&V view
defect prevention		both, mostly indirectly
	requirement-related	validation, indirectly
	other def prevention	verification, indirectly
	formal specification	validation, indirectly
	formal verification	verification
defect reduction		both, but mostly verification
	testing type	
	- unit & component	verification
	- integration	both, more verification
	- system	both
	- acceptance	both, more validation
	- beta	validation
	inspection type	
	- req. & scenario	validation
	- all other	verification
	analyses, etc.	both, but mostly verification
defect containment		both, but mostly validation
	operation	validation
	design and implementation	both, but mostly verification

Summary

- Defect handling is an integral part of QA activities. These activities can be integrated into software development and maintenance processes as an integral part of the overall process activities, typically in the following fashion:
  - *Testing* is an integral part of any development process, forming an important link in the overall development chain.
  - *Quality reviews or inspections* often accompany the transition from one phase or development activity to another.
  - Various *defect prevention activities* are typically carried out in the early stages.
  - *Defect containment activities* typically focus on the later, operational part of the development process, although their planning and implementation need to be carried out throughout the development process.
- QA activities view
  - V&V view
  - DC-view

Outline

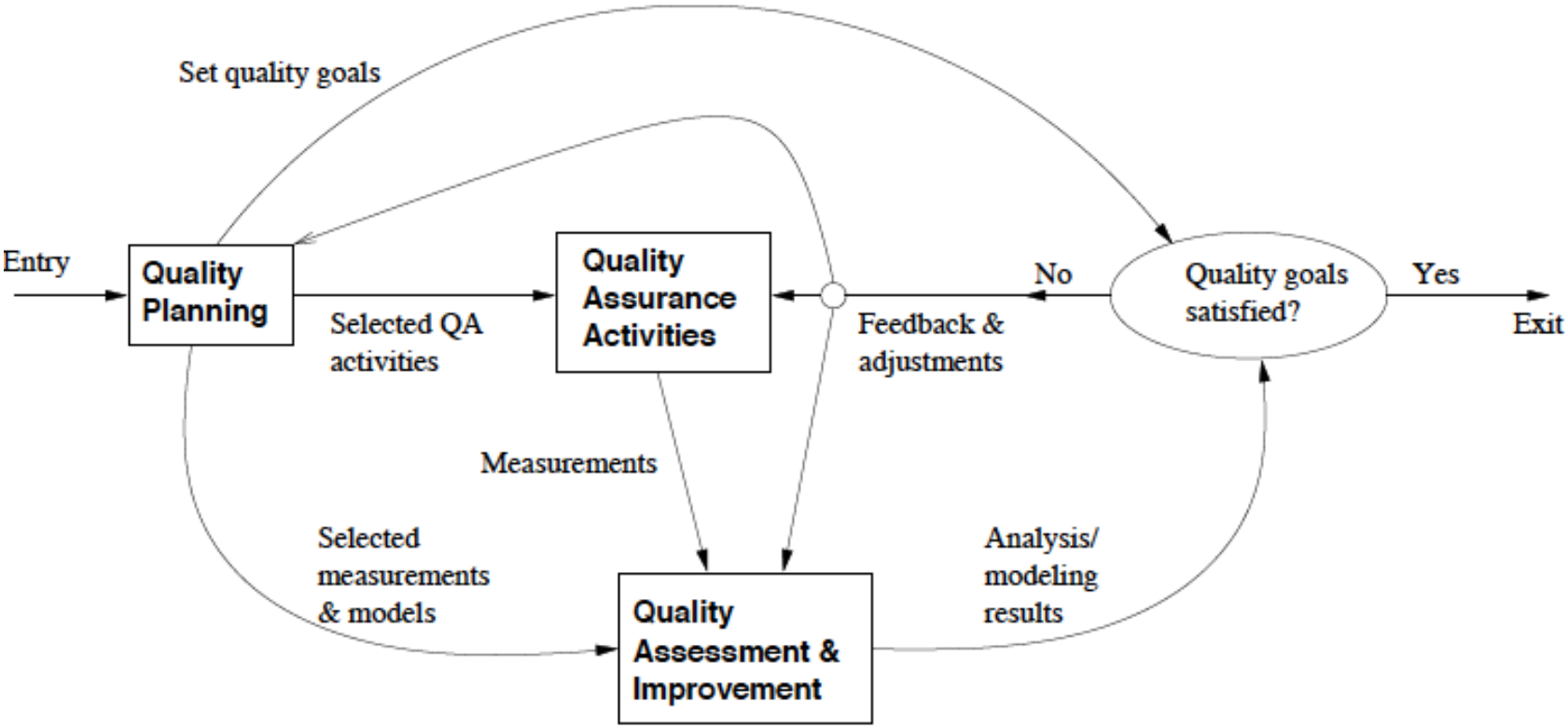
- Quality Assurance in context
  - Handling discovered defect during QA activities
  - QA activities in software processes
  - Verification and Validation perspectives
  - V&V view and Defect-Centered view
- **Quality Engineering**
  - **Activities and process**
  - **Quality planning& goal setting and strategy formation**
  - **Quality assessment and improvement**
  - **Quality engineering in software processes**
- Conclusion

QA to SQE

- QA activities need additional support:
  - Planning and goal setting
  - Management:
    - ✓ when to stop?
    - ✓ adjustment and improvement, etc.
    - ✓ all based on assessments/predictions
- Assessment of quality/reliability/etc.:
  - Data collection needed
  - Analysis and modeling
  - Providing feedback for management
- QA + above

➔ software quality engineering (SQE)

SQE process



- QE process to link major SQE activities:
  - Pre-QA planning;
  - QA;
  - Post-QA analysis and feedback (maybe parallel instead of \post-")

SQE process 2

- Pre-QA activities: Quality planning.
  - Set specific quality goals.
  - Form an overall QA strategy, which includes two sub-activities:
    - ✓ Select appropriate QA activities to perform.
    - ✓ Choose appropriate quality measurements and models to provide feedback,
- In-QA activities: Executing planned QA activities and handling discovered defects
- Post-QA activities: Quality measurement, assessment and improvement
  - The primary purpose of these activities is to provide quality assessment and feedback so that various management decisions can be made and possible quality improvement initiatives can be carried out.
- Feedback
  - short term direct feedback to the QA activities
  - long-term feedback to the overall quality engineering process.

SQE and QIP

- Quality improvement is achieved through
  - measurement, analysis, feedback, and organizational support
- QIP (quality improvement paradigm):
  - Step 1: understand baseline
  - Step 2: change then assess impact
  - Step 3: package for improvement
- QIP support:
  - overall support: experience factory
  - measurement/analysis: GQM (goal-question-metric paradigm)
- SQE as expanding QA to include QIP ideas.

Pre-QA planning

- Pre-QA planning:
  - Quality goal
  - Overall QA strategy:
    - ✓ QA activities to perform?
    - ✓ measurement/feedback planning
- Setting quality goal(s):
  - Identify quality views/attributes
  - Select direct quality measurements
  - Assess quality expectations vs. cost

Setting quality goals

- Identify quality views/attributes
  - customer/user expectations,
  - market condition,
  - product type, etc.
- Select direct quality measurements and models
  - direct: reliability
  - defect-based measurement
  - other measurements
- Assess quality expectations vs. cost
  - cost-of-quality/defect studies
  - economic models: COCOMO etc

Forming QA strategy

- QA activity planning
  - evaluate individual QA alternatives
    - ✓ strength/weakness/cost/applicability/etc.
  - match against goals
  - integration/cost considerations
- Measurement/feedback planning:
  - define measurements (defect & others)
  - planning to collect data
  - preliminary choices of models/analyses
  - feedback & follow up mechanisms, etc.

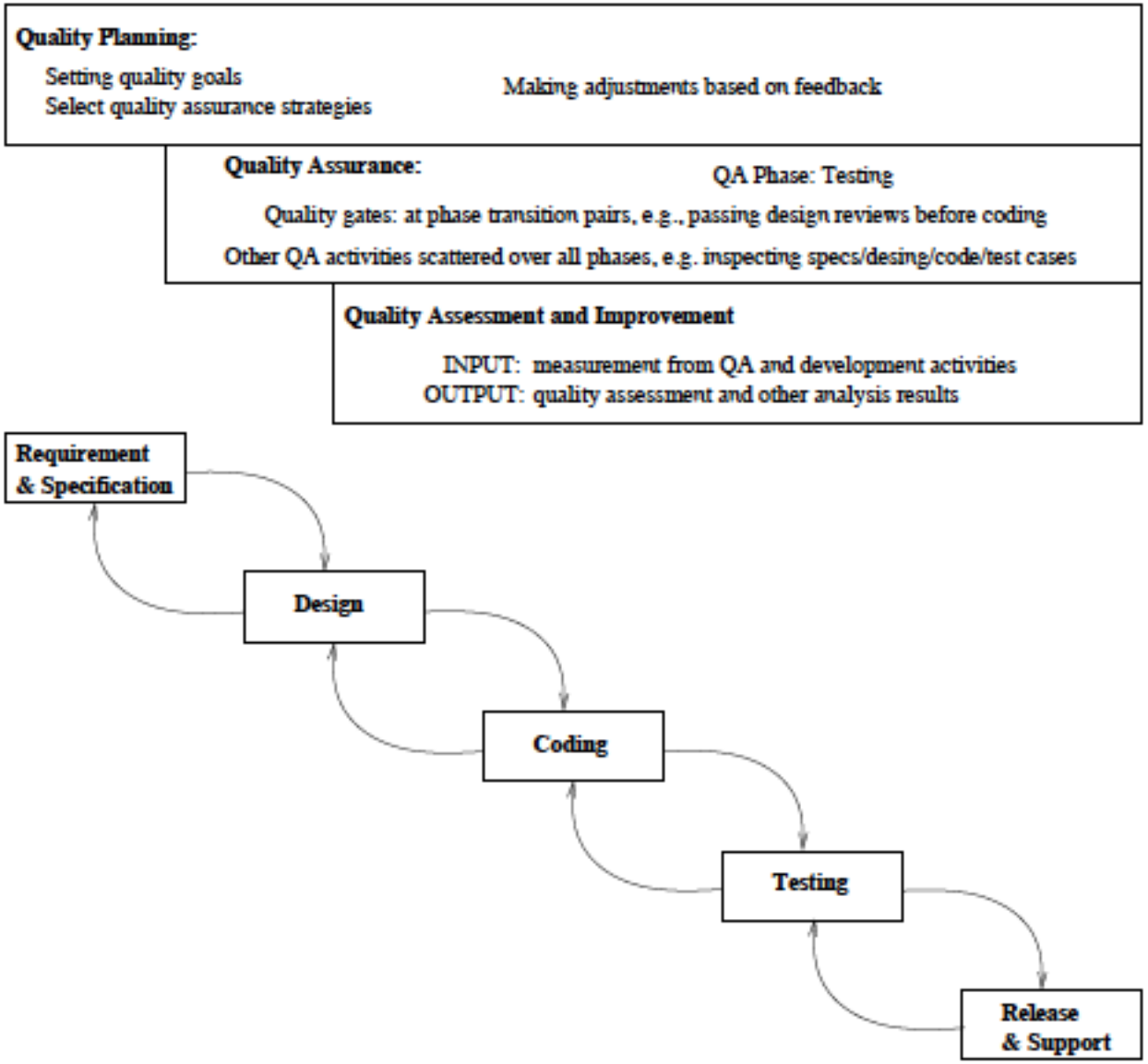
Analysis and feedback

- Measurement:
  - defect measurement as part of defect handling process
  - other data and historical baselines
- Analyses: quality/other models
  - input: above data
  - output/goal: feedback and follow up
  - focus on defect/risk/reliability analyses
- Feedback and follow up:
  - frequent feedback: assessments/predictions
  - possible improvement areas
  - project management and improvement

SQE in Software processes

- SQE activities < development activities:
  - quality planning < product planning
  - QA activities < development activities
  - analysis/feedback < project management
- Fitting SQE in software processes:
  - different start/end time
  - different sets of activities, sub-activities, and focuses
  - in waterfall process: more staged
    - ✓ (planning, execution, analysis/feedback)
  - in other processes:
    - ✓ more iterative or other variations

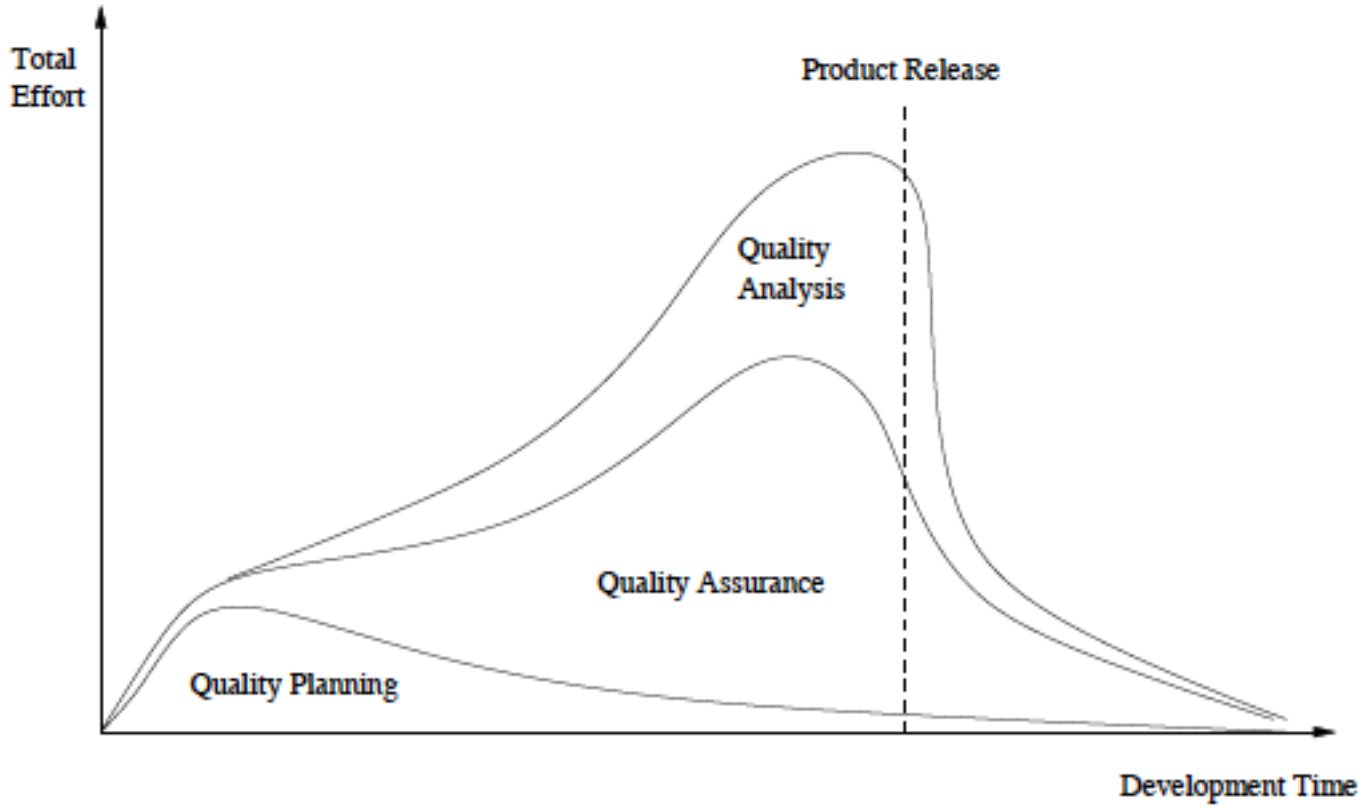
SQE I Waterfall process



SQE effort profile

- QE activity/effort distribution/dynamics:
  - different focus in different phases
  - different levels (qualitatively)
  - different build-up/wind-down patterns
  - impact of product release deadline
    - ✓ (deadline-driven activities)
- Planning: front heavy
- QA: activity mix
  - (early vs. late; peak variability? deadline?)
- Analysis/feedback: tail heavy
  - (often deadline-driven or decision-driven)

SQE effort in Waterfall process



- planning/QA/analysis of total effort
- general shape/pattern only (actually data would not be as smooth)
- in other processes: similar but more evenly distributed

Conclusion

- To manage the QA activities and to provide realistic opportunities of quantifiable quality improvement, we need to go beyond QA to perform the following:
  - Quality planning
    - ✓ set the overall quality goal by managing customer's quality expectations under the project cost and budgetary constraints
    - ✓ select specific QA alternatives and techniques to implement as well as measurement and models to provide project monitoring and qualitative feedback
  - Quality quantification and improvement
    - ✓ through measurement, analysis, feedback, and follow-up activities
    - ✓ The analyses would provide us with quantitative assessment of product quality, and identification of improvement opportunities
    - ✓ The follow-up actions would implement these quality and process improvement initiatives and help us achieve quantifiable quality improvement
- The integration of these activities with the QA activities forms our software quality engineering process, which can also be integrated into the overall software development and maintenance process


Q&A