CS 428 Creating Your Test Plan

WINTER 2020

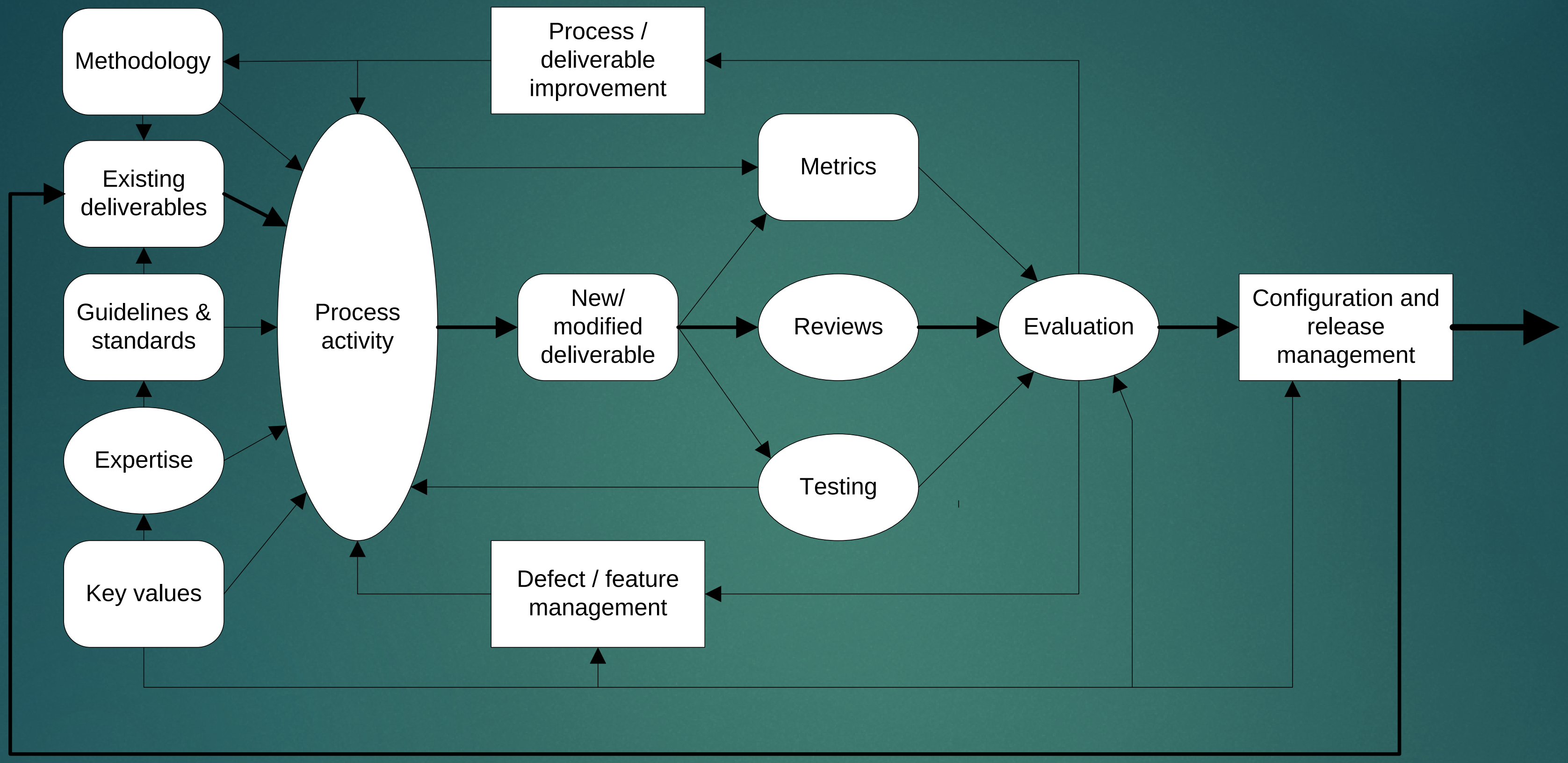
BRUCE F. WEBSTER

- As mentioned in prior lectures, SQA is the neglected stepchild of IT
- **SQA applies** not just to source code but **to all deliverables** in a software project, such as those you have created already or will create (org chart, requirements, PERT/Gantt charts, requirements, architecture, design, test plans/data, etc.)
- “Quality assurance occurs throughout the **entire** software development process, and **each person** involved in that process has an impact on the quality of the resulting application. This is critical to understand: **quality assurance is not a separate activity done by a separate organization**. While a given individual on a development team may oversee QA for that team, **all other members of that team are equally responsible for quality assurance**.” – me, 1996

What is software quality assurance?

- **Reliability**: the system must carry out its functions without causing unacceptable errors or having an unacceptable downtime.
- **Performance**: the system must complete its various operations within timespans acceptable to the client.
- **Functionality**: the system must offer sufficient usable features to meet the client's needs.
- **Compatibility**: the system must interact effectively with existing IT systems, including appropriate external systems under the control of other entities.
- **Lifespan**: the system must continue to offer acceptable reliability, performance, and functionality over a sufficient period of time to warrant the cost to the client, including in many cases having the ability to grow with the client.
- **Deployment**: the vendor must deliver and deploy the system, and the client receive its benefits (reliability, performance, and functionality), in a timeframe acceptable to the client.
- **Support**: the system must have the capability to be upgraded and repaired over time.
- **Cost**: the cumulative expense of developing, deploying, upgrading, and maintaining the system must appear to be justified in the eyes of the client.

Core quality values in software


Methodology
Process / deliverable improvement
Existing deliverables
Metrics
Guidelines & standards
Process activity
New/ modified deliverable
Reviews
Evaluation
Configuration and release management
Expertise
Testing
Key values
Defect / feature management

SQA should be tightly integrated into the entire software lifecycle

- **Methodology**: defining the steps and deliverables of your process
- **Key values**: defining what is important for your product and for your company
- **Expertise**: having people who know what they're doing (TEPES)
- **Requirements and guidelines**: having consistent standards for all deliverables (not just source code)
- **Metrics**: automated, objective values that provide useful information you can use to judge progress and make improvements
- **Reviews**: ensuring that all deliverables have had human judgment passed on them by multiple persons in open discussion
- **Testing**: ensuring reliability, performance, and functionality of deliverables
- **Configuration management**: controlling the time stream
- **Defect/feature management**: prioritizing effort
- **Process/deliverable improvement**: learning from your mistakes (Armour)

Key Aspects of SQA

- Unit/class testing (focus on smallest modules)
- Cluster testing (groups of related units/classes)
- Integration testing (putting everything together)
- System testing (trying it out in a production-like environment)
- Performance/stress testing (trying it out w/a real-world load on it)
- User acceptance/usability testing (do the users actually understand it?)
- Regression testing (re-test entire system after changes made)

Some types of software testing

- Should tie back to requirements and design
- Should check for **reliability**, **performance**, **functionality**
- Should indicate what tests are being done and when they are done (or repeated)
- Should indicate what constitutes success for each test
- Should include some form of user-acceptance testing
- Get feedback, input from entire team
- First draft due by midnight Saturday (02/22), but will likely be revised through the rest of the semester

Building your test plan

FOR NEXT WEEK (02/24)

- By midnight Saturday (02/22)
  - Test plan up on team wiki
  - Status report up on team wiki
  - Individual: do another podcast (#6)
- By class next week (02/24)
  - Read *Mythical Man-Month*, chapters 16-17
  - Read *Peopleware*, Part II (chapters 7-13)
  - Read *Accelerate*, chapter 9
  - Read Webster #6
- Remember: **first demo in four weeks** (03/16)
- Remember: **midterm in five weeks** (03/23)