Quality and Testing

*Is all fun and game until you are need of put it in production.*

—DevOps Borat

Everyone discusses quality and wants to improve it. However, not defining quality has the wonderful effect of allowing quality to be everything, including perks such as full pots of coffee on each table. Many projects have experienced excellent results by using tests as a vehicle for measuring quality. This chapter will discuss quality and tests and show their essential relevance for DevOps.

What Is Quality?

The definition of quality is unique to a given context. Gerald M. Weinberg states that “quality is conforming to someone’s requirements.” Normally, the sponsor of the delivered product (the customer) pays the team to provide the software. In fact, the customer has the right to provide his or her personal definition of quality and determine what he or she wants to spend money on. However, what actually is quality? To bring quality to life, this chapter will introduce some possible attributes of quality. Let’s summarize what attributes people think quality can have by starting with the most obvious and traditional ones. Quality may have the following characteristics:

- Expansive test coverage of unit tests; the more coverage, the better.
- A small number of entries in the bug tracker.
- A small number of entries in the bug tracker given a specific priority ranking of entries.

---

Minimization of accidental complexity, which is complexity that is not inherent to the task to be solved (whereas inherent complexity is essential for solving the specific task).

Compliance with defined metrics that are measured with Checkstyle, PMD, or FindBugs.

Compliance with system runtime quality, including functionality, performance, security, availability, resilience, usability, and interoperability.

Compliance with system non-runtime quality, including modifiability, portability, reusability, and testability.

Excellent stability and capacity of the software.

Free coffee for the whole team at any time.

Additionally, there are more subtle quality attributes, such as the following:

- A good business quality, including costs, schedule, marketability, and appropriateness for the organization.
- A good overall cycle time, as discussed in Chapter 3.

**DEFINITIONS: CAPACITY AND RESILIENCE**

The maximum throughput a system can sustain for a given workload while maintaining an acceptable response time for each individual transaction is its capacity (Michael T. Nygard, *Release It!* The Pragmatic Programmers, 2007, p. 152).

Resilience is the intrinsic ability of a system to adjust its functioning prior to, during, or following changes and disturbances, so that it can sustain required operations under both expected and unexpected conditions. (Hollnagel et.al, *Resilience Engineering in Practice*, Ashgate Publishing, 2011, page xxxvi)

As shown by these few examples, quality can mean many different things. Let’s now group quality attributes into leading and supporting attributes.

**Leading and Supporting Attributes**

Interestingly, people who claim that all aspects of quality are important—such as good test coverage or the stability of the application—are correct. However, the first examples in the list I just provided are supporting attributes, and the final ones, which I labeled in a slightly provocative way as “subtle,” are leading (i.e., necessary) attributes, namely business quality and cycle time.

---