340CT
Software Quality and Process Management

Dr. Yih-Ling Hedley
Email: aa0817@coventry.ac.uk

Seven Core Metrics (Recap)

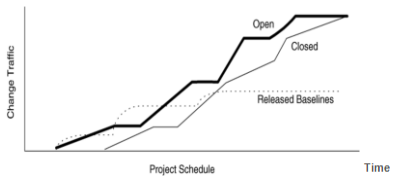
- **Quality Indicators**
  - **Change traffic and stability** (change traffic over time)
  - **Breakage and modularity** (average breakage per change over time) (note: breakage defined as the average extent of change)
  - **Rework and adaptability** (average rework per change over time)
  - **Mean time between failure (MTBF) and maturity** (defect rate over time)

Quality Indicators 1

- **Change traffic and stability**
  - Change traffic is the **number of software change orders opened and closed** over the life cycle.
  - with the work and progress metrics, it provides insight into the stability of the software and is **convergence toward stability**.
  - The change traffic relative to the release schedule provides insight into **schedule predictability**.

Activity

Q: What is the state of the following project (showing the software change order opened and closed) in terms of change traffic and stability?



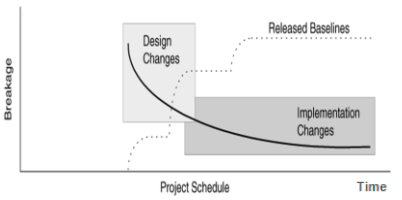
A: The change order (opened and closed) is converged towards the end, which demonstrates a healthy project with stability (the opposite is divergence indicating instability).

Quality Indicators 2

- **Breakage and modularity**
  - Breakage defined **as the average extent of change**, which is the amount of software baseline that needs **rework**.
  - **Modularity as the average breakage trend over time.** Modularity is a measure of breakage localisation, with a lower value being better.

Activity

Q: What is the state of the following project in terms of breakage and modularity?



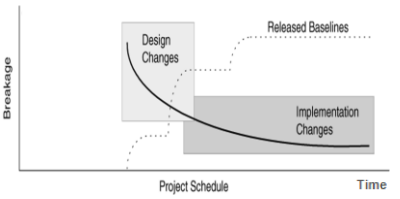
A: The average of breakage (the amount of software baseline that needs rework) trend over time is decreasing and stable, which demonstrates a healthy project with low modularity towards the end.

Quality Indicators 3

- **Rework and adaptability**
  - **Rework as the average cost of change**, which is the effort to analyse, resolve, and retest all changes to software baselines.
  - **Adaptability as the rework trend over time.** Adaptability quantifies the **ease of change**, with a lower value being better.

Activity

Q: What is the state of the following project in terms of rework and adaptability?



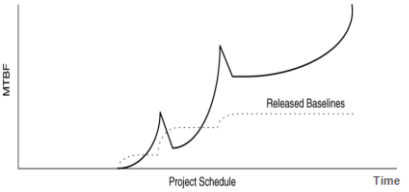
A: The rework (average cost of change) trend over time is decreasing or stable, which demonstrates a healthy project with low adaptability (the ease of change trend over time) towards the end.

Quality Indicators 4

- **Mean time between failure (MTBF) and maturity**
  - MTBF is the **average usage time between software faults**. In rough terms, MTBF is computed by dividing the test hours by the number of type 0 and type 1 SCOs (Software Change Orders).
  - Maturity is defined as the MTBF trend over time.

Activity

Q: What is the state of the following project in terms of mean time between failure (MTBF) and maturity?



A: As longer time (i.e. MTBF, average usage time between software faults) is needed to detect next faults towards the end of the project, it demonstrates a healthy project with maturity towards the end.

Quality Indicators: Metrics 1

- **Metrics**
  - **Modularity**: a measure of breakage localisation
  - **Adaptability**: a measure of the ease of change.
  - **Maturity**: a measure of the trustworthiness of the software, with trust increasing through extended usage.
  - **Maintainability**: a measure of the required productivity needed for maintenance.

Software Change Orders

- **Software Change Orders (SCOs)**
  - SCO types:
    - type 0 for critical defects: reworks due to errors
    - type 1 for normal defects: reworks due to low quality
    - work
    - type 2 for improvements: reworks due to going for better quality
    - type 3 for new features: customer change requests

Quality Indicators: Metrics 1.1

- **Modularity = B/N**, average breakage due to N (number of rework), reflects the inherent ability of the integrated components to localize breakage. **(lower value is better)**
  - Breakage (B) for Open Rework: cumulative SLOC to rework
- **Adaptability = E/N**, average effort per N, how "easy" was it to change N things. **(lower value is better)**
  - Rework Effort (E): cumulative effort spent fixing.
- **Maturity = UT/(SCO0+SCO1)**, mean time between failures or defects (MTBF). **(larger value is better)**
  - Usage Time (UT): hours of operation under realistic usage scenarios.

Quality Indicators: Metrics 1.2

- **Maintainability = (scrap ratio)/(rework ratio)**, maintenance productivity, ratio of productivity of maintenance to productivity of development. The smaller the better.
  - Scrap Ratio = B/$SLOC_T$, percentage of product scrapped (or reworked).
  - Rework Ratio = E/Development_Effort, percentage of effort spent in rework.

Note:
1. $SLOC_T$ - Total SLOC: estimated total size of the software under development.
2. Rework Effort (E): cumulative effort spent fixing.

Quality Indicators: Example 1

- The below shows the data recorded during a software development project. The software was finished with 10,000 SLOC and required an effort of 250 person-days to develop. Determine the Modularity, Adaptability, and Maturity of the software.

Type	value	Broken SLOC	Effort to Fix
0	20 defects	500	5 person-days
1	100 defects	5000	30 person-days
2	20 defects	300	15 person-days
3	10 features	500	20 person-days
Usage Time (UT)	720 hours	N/A	N/A

Quality Indicators: Example 1.1

- **Modularity = B/N = 5800/140**
  = 41.4 SLOC/defect
  - B = cumulative broken SLOC due to N (number of rework) = 500 + 5000 + 300 = 5800 SLOC
  - N = C0+C1+C2 = 20 + 100 + 20 = 140 defects

Quality Indicators: Example 1.2

- **Adaptability = E/N = 50/140**
  = 0.36 person-days/defect
  - E = cumulative effort spent fixing N = 5+30+15=50 man-days
  - N = 20 + 100 + 20 = 140 defects
- **Maturity = UT/(C0+C1)**
  = 720/(20+100) = 6 hours/defect

Quality Indicators: Example 1.3

- **Maintainability = (scrap ratio)/(rework ratio) =**
  = 0.58 / 0.2 = 2.9
  - Scrap Ratio = B/$SLOC_T$ = (500 + 5000 + 300) / 10,000 = 5800/10,000 = 0.58
  - Rework Ratio = E/Development_Effort
    = (5+30+15) / 250 = 50 / 250 = 0.2

Quality Indicators: Case Study

- **CertWare workbench:** part of a project by NASA (National Aeronautics and Space Administration) Langley Research Center and Kestrel Technology LLC (2011-2012) http://nasa.github.io/CertWare/collateral/SafetyCaseMetrics.pdf
  - CertWare provides supporting models for a prototype extensible workbench for safety cases, with a service-based APIs.
  - CertWare workbench metrics, used to collect **management and quality indicators**, are based on those proposed by Walker Royce

YL Hedley

Case Study: CertWare