Aston University
BIRMINGHAM UK

Software Project Management

Unit 2: Measurement, estimation & data analysis

Thais Webber
Richard Lee

Session objectives

- Appreciate the key role of measurement in software project management
- Understand measurement theory and use appropriate scale types to quantify the attributes of software projects
- Explore the limitations associated with measurement in software project management
- Apply measurement principles to your own final year project

Question

- What measurements might you obtain as part of your final year project...

  - Before development

  - During development

  - After development

Before development

- Metrics associated with similar software:
  - Development time
  - SLOC
  - User reviews
- Knowledge elicitation metrics:
  - Requested features
  - Feedback relating to current systems
  - Rankings of features/systems
- Information arising from design:
  - Number of classes/entities/screens
  - Complexity of use cases

During development

- Formal code metrics:
  - Source lines of code (SLOC)
  - Number of modules
- Process metrics
  - Cycle time
  - Position within life cycle
  - Size of backlog
  - Objectives underway/completed
- Test metrics
  - Code coverage
  - % of automated tests

After development

- Test metrics:
  - Defects discovered / resolved
  - Performance metrics
- Usability metrics:
  - Error rate
  - Time spent per page/screen/process
  - Task success rate
- Evaluation-related metrics
  - Objectives met/missed
  - New skills developed

Types of scale

- Nominal
  - Can place data items in groups
- Ordinal
  - Can place items into rank-ordered groups
- Interval
  - Rank ordered groups are an equal distance apart
- Ratio
  - Rank ordered groups are an equal distance apart, with a true zero
- Absolute
  - Subset of ratio – direct measurement only (probably counting something)

What about your project?

Nominal

Ordinal

Interval

Ratio

Absolute

What about your project?

Nominal

To which class does each responsibility belong?

Ordinal

Ranking objectives according to MoSCoW criteria

Interval

Time – monthly development points

Ratio

Number of hours spent on the current module

Absolute

SLOC directly used from third-party sources

Indirect measurement

- Spoilage (SP)

$$SP = \frac{\text{Time to fix post-release defects}}{\text{Total development time}}$$

Indirect measurement

- Spoilage (SP)

$$SP = \frac{\text{Time to fix post-release defects}}{\text{Total development time}}$$

- On what type of **scale** is spoilage?

Indirect measurement

- Spoilage (SP)

$$SP = \frac{\text{Time to fix post-release defects}}{\text{Total development time}}$$

- On what type of **scale** is spoilage?
- Ratio – a value of zero would mean no time fixing post-release defects

Indirect measurement

- Spoilage (SP)

$$SP = \frac{\text{Time to fix post-release defects}}{\text{Total development time}}$$

- What might an atypically large value for SP mean?

Indirect measurement

- Spoilage (SP)

$$SP = \frac{\text{Time to fix post-release defects}}{\text{Total development time}}$$

- What might an atypically large value for SP mean?
- Unusually large amount of time on fixing defects, unusually small amount of time on development, or both

Indirect measurement

$$SP = \frac{\text{Time to fix post-release defects}}{\text{Total development time}}$$

- What might be problematic about this measurement?

Indirect measurement

$$\text{SP} = \frac{\text{Time to fix post-release defects}}{\text{Total development time}}$$

- What might be problematic about this measurement?
  - What do we mean by 'post-release defect'
  - Does the numerator include reporting and assessment as well as fixing time?
  - What is included in total development time? Maintenance, design...
  - What is the measurement for time?
  - What is the start/end time for total development?
  - Is it fair to use this to make comparisons between projects of different types?

Indirect measurement

- p = number of developers
- d = number of days
- s = source lines of code
- c = cost of the whole project

- What additional data could I derive if the above data is known?

Indirect measurement

- p = number of developers
- d = number of days
- s = source lines of code
- c = cost of the whole project

- Effort (e) = p * d
- Productivity inverse = e / s
- Unit effort cost = c / e

Previous exam question

Aston University BIRMINGHAM UK

DISCUSS the limitations of using Source Lines of Code (SLOC) as a measure for estimating software productivity and PROPOSE an alternative approach to overcome them

[5 marks]

Previous exam question

DISCUSS the limitations of using Source Lines of Code (SLOC) as a measure for estimating software productivity and PROPOSE an alternative approach to overcome them

[5 marks]

Ignores code complexity

Ignores variations between programming languages

Easy for a developer to manipulate

Incentivises inefficient code

Metric involving multiplication of number * complexity of classes

Aston University

BIRMINGHAM UK

Software Project Management

Unit 2: Measurement, estimation & data analysis

Thais Webber

Richard Lee