

Reflective Software Engineering

Module 07: Size and Time Estimation

(1)



Objectives

Understand why size and time estimation is useful in reflective software engineering.

Understand the conceptual process of deriving a size estimate and time estimate using historical data.

(2)



Why estimate?

Software engineering has a long history of poor to non-existent size/time estimation.

Why break with tradition now?

(3)



In a world of accurate size and time estimates...

Software projects would be delivered on time.

Software projects would be delivered on budget.

Software engineers would go home in evenings and weekends.

Software engineers would make more money.

Less software startups would fail.

(4)



Benefits of size estimation

Successful size estimation:

- supports successful planning
- supports successful time estimation
- indicates a successful design stage

Unsuccessful size estimation:

- provides useful feedback on design stage and size estimation procedure,
- leading to improved future design stages and size estimation procedures,
- leading to future successful size estimation

(5)



Benefits of time estimation

Successful time estimation:

- Tells you when, where, and on what to work in order to finish the project on time.
- Creates time to work on other things.
- Helps you avoid making impossible commitments.

Unsuccessful time estimation:

- Helps diagnose problems in size estimation
- Helps improve future time estimates from size estimates
- Helps diagnose problems in work behavior or workplace environment.

(6)



Estimation from the development perspective

Conceptual Design



Planning



Implementation



Postmortem

Conceptual design:

- Purpose is to produce a size proxy for planning.

Planning:

- Uses proxy to build size and time estimates which constrains schedule.

Implementation:

- Build the system

Postmortem:

- Analyze success/failure of design and estimates.

(7)



Conceptual Design

During conceptual design, a primary goal is to produce a "proxy", or substitute, for the goal size representation of your system.

Examples:

- If we are estimating technical report size, our goal may be a size estimate in words, but our "proxy" is the table of contents outline.
- If we are estimating Java source size, our goal may be a size estimate in LOC, but our "proxy" might be the number of methods in our conceptual design.

(8)



Example Java Proxies

Simple Structural Proxies:

- Number methods, classes, packages
- We use this proxy in this course.

Annotated Structural Proxies:

- Use JavaDoc to add "weight" to methods, classes, packages by counting JavaDoc comment size.

UML Proxies:

- Number methods, classes, packages, plus inheritance/component relationships.

(9)



Example tech report proxy

Simple Structural:

- Number of chapters, sections, subsections.

Annotated structural:

- Number of chapters, sections, subsections plus number of figures

(10)



Planning Part I: Proxy -> Size Estimate

The conceptual design creates a proxy for the size of your work product.

Next, you must translate this proxy into a size estimate. Two approaches:

- The "identity" translation, if your proxy unit is the same as your size unit.
 - Ex: "methods" is both proxy and size unit
- Some other translation which converts proxy units to size units.
 - Ex: 10 sections * 500 words/sec. = 5K words

(11)



Planning Part II: Size estimate -> Time estimate

Once you have produced a size estimate,

- Examples: 5678 words, 42 methods, 14 sections, 12 classes, 67 function points, 56 powerpoint slides, 123 HTML pages, etc.

The next task is to produce a time estimate.

Time estimation depends upon the presence of at least one historical data point that relates your size unit (words, methods, sections, etc) on a prior project to the length of time the project required.

(12)



Planning Part II: Examples

Given a new size estimate, and a historical rate value on similar projects (size/time), generate new time estimate

- $100 \text{ LOC} * 10 \text{ LOC/hour} = 10 \text{ hours}$

Given a new size estimate, and a set of historical size/time pairs, use a curve-fitting algorithm to generate new time estimate

- linear regression
- logarithmic regression

You can also divide the total time among phases.

(13)



Implementation

This is the easy part :-).

It's quite important to keep track of the time you spend during implementation.

Also, you need to compute the total "new" work product created at the end of implementation.

- Use LOCC Total or Diff in this class.

The result is your "actual" size and "actual" time.

(14)



Postmortem

Postmortem is where you compare your estimates to your actual data.

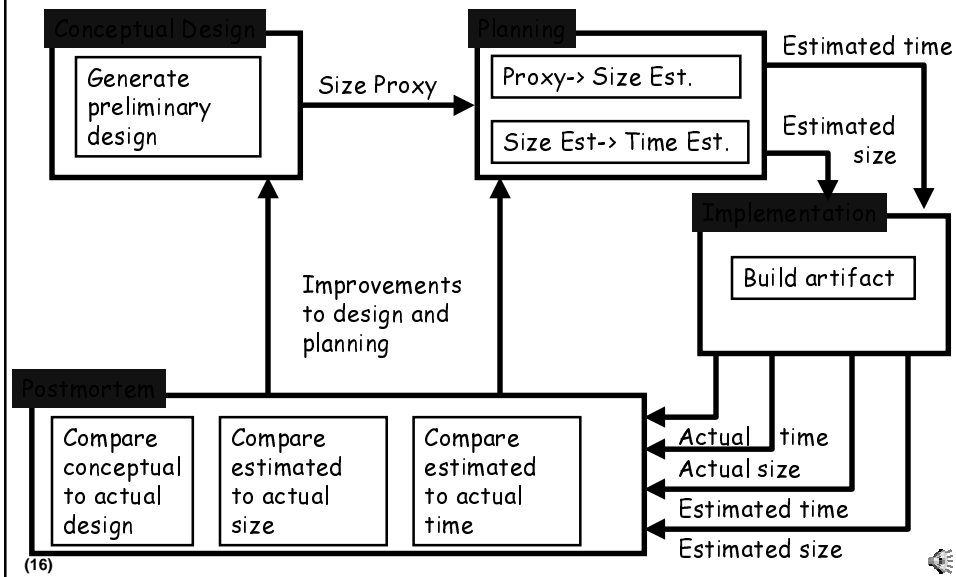
Differences between estimates and actuals can help you improve:

- your conceptual design
- the choice of size proxy
- the translation of proxy to size estimate
- the choice of mapping from size to time
- the partitioning of total time among phases

(15)



The Complete Estimation Cycle



(16)



End of Module 07

**Proceed to Module 07b
to learn about estimation
using the Leap toolkit.**

(17)

