Metrics and Estimation

Rahul Premraj + Andreas Zeller

These slides are based on Pressman, Chapter 15 “Product Metrics”, Chapter 22 “Metrics for Process and Projects” and Chapter 23 “Estimation”.

Metrics

- Quantitative measures that provide *insight into the efficacy* of the process
- Analyzed and assessed by *software managers*
- Avoids basing judgements solely on subjective evaluation

Lord Kelvin
Lord Kelvin

When you can measure what you are speaking about and express it in numbers, you know something about it.

Lord Kelvin

...but when you cannot measure and express it in numbers, your knowledge is of a meager and unsatisfactory kind.

Lord Kelvin

...it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced to the stage of science.
Tom DeMarco

You can’t control what you can’t measure.

Measures and Metrics

- **Measure**
  Provides a quantitative indication of a product or process attribute

- **Metric**
  A quantitative measure of the degree to which a product or process possesses a given attribute

- **Use measures to obtain metrics**

Measurement Process

1. **Formulation**
   Deriving appropriate measures and metrics

2. **Collection**
   The mechanism used to accumulate required data (i.e., survey / observation / experimental study…)

3. **Analysis**
   Computation of metrics and application of mathematical tools

4. **Interpretation**
   Computation of metrics and application of mathematical tools

5. **Feedback**
   derived from interpretation; passed to software team
Goal / Question / Metric

1. Establish an explicit measurement goal that is specific to the activity or characteristic to be assessed – e.g.,
   • Are function and related data properly compartmentalized?
   • Is the complexity of each component within proper bounds?

2. Define a set of questions that must be answered in order to achieve the goal

3. Identify well-formulated metrics that help to answer these questions

The Metrics Landscape

- Analysis
- Design
- Code

Function Points

- Measure functionality delivered by a system
- Proposed by Albrecht in the 80s.
- ISO recognised software metric to measure software size.
- Based on functionality as perceived by the user and independent of the technology.
Function Points

- **Input:** A set of related inputs as one input.
- **Output:** A set of related outputs as one output.
- **Inquiries:** Each user query type is counted.
- **Files:** Files are logically related data, i.e., data structures or physical files.
- **Interface:** Data transfer to other systems.

<table>
<thead>
<tr>
<th>Information Domain Value</th>
<th>Count</th>
<th>Weighting factor</th>
<th>FPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Inputs</td>
<td>3</td>
<td>4 / 6</td>
<td>9</td>
</tr>
<tr>
<td>External Outputs</td>
<td>2</td>
<td>5 / 7</td>
<td>8</td>
</tr>
<tr>
<td>External Inquiries</td>
<td>2</td>
<td>4 / 6</td>
<td>6</td>
</tr>
<tr>
<td>Internal Logical Files</td>
<td>1</td>
<td>7 / 10 / 15</td>
<td>10</td>
</tr>
<tr>
<td>External Interface Files</td>
<td>2</td>
<td>5 / 7 / 10</td>
<td>10</td>
</tr>
<tr>
<td>Count total</td>
<td></td>
<td></td>
<td>53</td>
</tr>
</tbody>
</table>

To compute function points (FP), use

\[
FP = count\ total \times (0.65 + 0.01 \times \sum (F_i))
\]

- \(F_i\): one of 15 value adjustment factors [0–5]
  1. Does the system require backup?
  2. Are specialized data communications needed?
  3. Are there distributed processing functions?

- All constants determined empirically
Function Points

- Proponents claim:
  - FP is language independent.
  - Size can be derived from problem description.
- Opponents claim:
  - It’s subjective: different people arrive at different estimates.

Function Points

- Counting FP itself takes a long time.
- Difficult to count consistently without extensive training.
- Difficulty of using automated tools.
- Difficulty of counting embedded, highly algorithmic modules, and web systems.

FP Associations

Netherlands Software Metrics Users Association

International Function Point Users’ Group

Finnish Software Measurement Association

and others...
The Metrics Landscape

Design Metrics

- Weighted Methods per Class
- Depth of Inheritance Tree
- Number of Children
- Coupling between Object Classes
- Response for a Class
- Lack of Cohesion in Methods

So-called Chidamber and Kemerer (CK) Metrics – most frequently referenced
Lines of Code

- Simplest and most widely used metric
- Comments and blank lines usually left out.
- Numerous tools available that do this for you :-)!
- Don’t forget to check how accurate they are first!

Disadvantages of LOC

- Size varies with coding style.
- Focuses on coding activity alone.
- Correlates poorly with quality and efficiency of code.
- Penalises higher level programming languages, reuse, etc.

Disadvantages of LOC

- Measures lexical/textual complexity only.
- Difficult to estimate LOC from problem description.
- Does not address issues of structural and logical complexity.
Code Metrics

- Halstead Metrics measure size of vocabulary
- McCabe Metrics measure complexity of control flow as \[ P(V) = \text{edges} - \text{statements} + 2 \times \text{entry points} \]
- No longer appropriate for modern OO systems ⇒ use OO metrics for complexity

The Metrics Landscape

- Analysis
- Design
- Code
- Testing (next week)

All these Metrics!

Once we have obtained all these metrics, what do we do with them?

So-called Chidamber and Kemerer (CK) Metrics – most frequently referenced
Now that we talked about measurement, let’s talk about control.

Estimation

- During project management, one needs to estimate resources, cost, and schedule
- Crucial for project success
- As much an art as it is science…
- …but need not be conducted in a haphazard manner!

What costs are incurred?

- Staff
- Training
- Travel for serious work!
- Hardware
What costs are incurred?

Workspace

Bills to Pay

Recreation

Communication

Why bother to Estimate?

- To establish a budget for a software project.
- To provide a means of controlling project costs.
- To monitor progress against that budget by comparing planned with estimated costs.
- To establish a cost database for future estimation.
- Cost estimation and planning/scheduling are closely related activities.

(c) Ian Sommerville
Parkinson’s Law

Work expands so as to fill the time available for its completion.

Why try to be accurate?

- Under-estimation
  - Loss of money
  - Damage to company’s reputation
- Over-estimation
  - Loss of bids to competitors.
  - Opportunity cost of not doing other projects.
  - Wastage of ideal or untilised resources.

Cone of Uncertainty
Metrics Needed

• Metrics need to be defined.
• They need to be collected.
• Then, validated.
• And lastly, if deemed suitable, used to base estimates upon.

Cost Estimation Techniques

• Expert Judgement
• Algorithmic Techniques
• Case-Based Prediction

Emphasis on Size!

Boehm’s third law
Development effort is a function of product size.

Less = Less = Less
More More More
### Size measures

**Function Points**

<table>
<thead>
<tr>
<th>Information Domain Value</th>
<th>Count</th>
<th>Weighting Factor</th>
<th>FPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Inputs</td>
<td>3</td>
<td>3 / 4 / 6</td>
<td>9</td>
</tr>
<tr>
<td>External Outputs</td>
<td>2</td>
<td>4 / 5 / 7</td>
<td>8</td>
</tr>
<tr>
<td>External Inquiries</td>
<td>2</td>
<td>3 / 4 / 6</td>
<td>6</td>
</tr>
<tr>
<td>Internal Logical Files</td>
<td>1</td>
<td>7 / 10 / 15</td>
<td>10</td>
</tr>
<tr>
<td>External Interface Files</td>
<td>2</td>
<td>5 / 7 / 10</td>
<td>10</td>
</tr>
<tr>
<td>Count total</td>
<td></td>
<td></td>
<td>53</td>
</tr>
</tbody>
</table>

**Design Metrics**

- Weighted Methods per Class
- Depth of Inheritance Trees
- Number of Children
- Coupling between Object Classes
- Response for a Class
- Lack of Cohesion in Methods

**Lines of Code**

- Simplest and most widely used metric
- Comments and blank lines usually left out.
- Numerous tools available that do this for you :-)
- Don't forget to check how accurate they are first!

### Criteria for Size Measure

- Relationship to development effort
- Precision
- Machine countable
- Suitable for early planning

### Estimate from FPs

- Compute function points for new project
- Multiply with organizational average productivity (e.g., 6.5 PM / person month)
- Obtain effort estimate
Expert Judgement

- You approach an in-house or external expert independent consultants or big consultancies such as Accenture, Cap Gemini
- You tell them what needs to be developed.
- He makes an estimate and gives you a number!

Simple... eh?

Note that all three examples are linear... but the ‘diseconomies’ of scale differ. If the distinction is unclear, please refer to a simple explanation at Wikipedia – http://en.wikipedia.org/wiki/Diseconomies_of_scale
Expert Judgement

How do they arrive at an estimate?

Activity Decomposition

| Month | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | Total |
|-------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|-----|
| Requirements | 4 | 4 | 6 | 4 | 6 | 4 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 33 |
| Scenario Development | 3 | 3 | 3 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Data Model Development | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 21 |
| Sequence Diagrams | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Design | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| High Level Architecture | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Performance Model | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Data Base Design | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Detailed Architecture | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Coding & U&U Testing | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Module Design & Coding | 3 | 6 | 9 | 9 | 9 | 7 | 5 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| Data Base Implementation | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| O&M Implementation | 3 | 3 | 3 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Unit Testing | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| System Testing | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Test Case Development | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| Test Environment | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Test Case Execution | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| Project Mgmt & Administration | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| Training & Documentation | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Total Project Staffing | 5 | 18 | 24 | 27 | 27 | 27 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 | 24 |
| Total Project Cost (staff years) | 22.3 |

From Software Measurement and Estimation: A Practical Approach by Linda Laird and Carol Brennan
Expert Judgement

Advantages

• Approachable
• Only resort when historical and quantitative data is absent.
• Can understand your project and development environment.
• Lots of experience (to reuse)
• Can customise solutions
• Can tackle exceptions (e.g., different technology used)

Expert Judgement

Disadvantages

Cost

...an estimate is only as good as the expert’s opinion, and there is no way usually to test that opinion until it is too late to correct the damage if that opinion proves wrong.
Humans are prone to bias, optimism, pessimism, instinct, and desire to win!

... often, can’t explain or don’t want to share!

Today: $2 million

$1 million
Expert Judgement

Disadvantages

Repeatability

Today: $2 million
10 days later: $4 million

Delphi Method

- You take your project to a group of experts.
- Each expert comes up with a list of tasks and estimates.
- Each experts’ list is anonymously distributed amongst others.
- Lists are discussed and refined to make a single big list.
- Several rounds (usually four) take place to further refine the list and arrive at a final estimate.
Rayleigh’s Curve
Putnam-Norden-Rayleigh (PNR) Curve

- Rayleigh curve represents the number of full-time personnel required at any time.
- At project start, small number of engineers needed.
- As project progresses, more work is required that needs more staff. At one point, the number of staff peaks (system testing and product release).
- It then drops down (installation and delivery).
Software Engineering Equation

derived from observing over 4,000 projects

\[ E = \left[ \frac{\text{LOC} \times B^{0.333}}{P} \right] \left( \frac{1}{t^4} \right) \]

- \( E \) = Project effort in person months or person years
- \( \text{LOC} \) = Lines of code estimate for the project
- \( t \) = Length of project measured in months or years
- \( B \) = a "special skills factor"
- \( P \) = a "Productivity Parameter"

Wait! What about these?

1 KLOC = 1,000 LOC

Walston-Felix Model
\[ E = 5.2 \times KLOC^{0.91} \]

Bailey-Basili Model
\[ E = 5.2 + 0.73 \times KLOC^{0.61} \]

Boehm simple Model
\[ E = 3.2 \times KLOC^{1.05} \]

Doty Model for KLOC > 9
\[ E = 5.288 \times KLOC^{1.047} \]

and more...

http://en.wikipedia.org/wiki/Software_equation

Different values of LOC will result in different values of effort.
Maintainability Index

\[
\text{Maintainability} = 171 - 5.2 \ln(V) - 0.23V(G) - 16.2 \ln(L) + 50 \sin\left(\sqrt{2.4}C\right)
\]

Size of vocabulary \quad McCabe complexity \quad Percentage of comment lines


Diversity

And one can come up with very very elaborated curve fitting models…

… but if something works at HP, will it work for you, too? The problem is: Software engineering data is usually very diverse…

projects originate from different countries! (what are the implications of this?)
Projects originate from different companies! These companies are poor examples because they will almost never share their data. However, the point to learn is that their products and processes limit transfer to techniques and knowledge between each other.

and there are many more sources for diversity

so check your data carefully before using it!
The best way to use these models is to *calibrate* them on earlier products (from other or the same companies).

It generally turns out that models that are calibrated on earlier projects are much more effective.
Neural Networks

• Neural Networks (NN) are relatively good predictors of effort.
• But they function like a black-box.
• Extremely time-consuming and difficult to train and optimize.
• Users need to have substantial training themselves.

You *do not* need to learn neural networks for exams.
Case-Based Prediction

- Very favourable for weak-theory domains.
- Less knowledge engineering required.
- Can be shared across a multitude of users and corporations.
- Possible to give justification for proposed solutions.
- Constant and potentially automated learning!
Experts and Analogy

De-Marco-Glass Law

Most cost estimates tend to be low.

Estimation Accuracy

On the whole, we are still no good at this task. Sometimes, we estimate and get lucky :-) Residuals range to thousands of hours.

But some estimate is better than no estimate!
Summary

- Software Engineering Estimation is crucial for the project’s success.
- There are several models proposed.
- None seem to perform well consistently.
- Calibrated models can be very helpful.
- An estimate having some confidence is still better than no estimate at all!