Estimating without “Points”

(and how you too can be delivered from the tedium of repetitive estimation)

Andy Miller
amiller@objectpartners.com
Abstract

- Repeated estimation can be one of the most unpleasant tasks for any project -- getting it wrong can hurt!
- There is an easy(-ier) method
  - it doesn't/shouldn't take too much time
  - it has a built in feedback loop for increasing it's accuracy
  - it helps the team stay on top of changing requirements and construction progress
Evolution of these methods...

- Used and refined for more than a decade
- Originated in an obsession to “measure things”
- Originated in a desire to make measurements clear to all
- Originated in a belief that

  It is our professional responsibility to be accurate estimators!
  and

  It is our professional responsibility to keep getting better at estimating!
What to Expect Today...

- Lot of ground to cover so...
  - starting with some definitions,
  - an overview of two estimation methods,
  - estimation goals, concepts, principles,
  - guidelines for what to watch out for,
  - ...

- Finishing up with time for questions.
The Life of a Project

♣ people and communication ♣
♣ vision and requirements ♣
♣ estimation and planning ♣
♣ implementation patterns and quality ♣
♣ tools and construction ♣
♣ delivery and deployment ♣
♣ maintenance ♣
Development Process...

assuming an agile-ish/iterative process

- Inception (1st increment)
- Elaboration (2nd increment)
- Construction (1 or more increments)
- Transition (last increment)

- Initial project estimate
- Refined estimates at the end of each iteration
- End of project feedback into estimation metrics
- Final estimation of transition effort
Some Definitions...

**accuracy**

1: freedom from mistake or error : CORRECTNESS

2 a: conformity to truth or to a standard or model : EXACTNESS  
   b: degree of conformity of a measure to a standard or a true value — compare PRECISION 2a

**precision**

1: the quality or state of being precise : exactness

2 a: the degree of refinement with which an operation is performed or a measurement stated  
   b: the accuracy (as in binary or decimal places) with which a number can be represented...

3: relevance
Estimation Goals...

- Start out with accuracy
- Quickly build up to precision
- Everyone understands the data
- Everyone understands the technique
- Easy to do...
  - encourages step wise refinement
- Builds trust between all team members
Early estimation - Do you feel lucky?

- How soon is too soon to provide a “responsible” estimate?
  - Remember that “build trust” goal?

- Always estimate functionality, cost, and time

- At this stage in the game the best we can do is base the estimate on complexity
  - Points are a cop out!
  - Just another way of saying “I don't know”
What Do “We” Care About?

- functionality = requirements
- cost = money
- time = delivery date

- If we can't use “points”...?
  ...how do we deal with uncertainty?
Uncertainty is our friend!

- Uncertainty in the “PROBLEM SPACE”
  - When we are “done” the problem we will have solved is not what we see now

- Uncertainty in the “SOLUTION SPACE”
  - At the beginning of the project we don't know what we're gonna end up building

- Uncertainty in the estimate
Problem and Solution Space Metrics

metric

1 plural : a part of prosody that deals with metrical structure
2 : a standard of measurement
A Problem Space Metric...

- quantitative requirements
  - stories (or use cases, or ...)
  - human actors
  - system actors

- qualitative requirements
  - behavior expectations
  - performance expectations
...A Problem Space Metric...

- **A** = Weighted actors (simple | average | complex)
- **R** = Weighted stories (simple | average | complex)
- **C_t** = Technical Complexity (distribution, performance, efficiency, complexity, reusable, ease of installation, portability, ease of change, concurrency, security, 3rd party integration, user training)
- **C_e** = Environmental Complexity (process familiarity, application experience, OO experience, analyst capability, motivation, requirements stability, staff stability, learning curve)
- **Complexity** = (A + R) * C_t * C_e

A Solution Space Metric...

- Patterns imply the existence of components
  - tool patterns
  - framework usage patterns
  - infrastructure integration patterns
  - ...

- Components
  - Domain objects
  - Processes
  - Interfaces
  - ...

Estimating from Patterns & Components...

- establish patterns
- for each story/scenario paragraph
  - derive a domain model
  - derive a system/service model
  - derive a UI model
- consider...
  - external systems
  - complex processes
  - all actors (human, system, ...)
  - harvest opportunities
  - packaging expectations
  - complexity
  - ...

- brainstorm
  - complexity of system elements
  - complexity UI elements
  - effort per “kind-of” component
  - assumptions and their impact
  - risks and their impact
  - best/expected/worst cases
- sketch out milestones
- set the staff model
- set the cost model
Mapping the problem to a solution...
Mapping the problem to a solution...
Mapping the problem to a solution...
Doing the math...

- Skip the points
- Go straight to the hours
- ... example of a component grid

<table>
<thead>
<tr>
<th></th>
<th>list viewer</th>
<th>list editor</th>
<th>object viewer</th>
<th>object editor</th>
<th>object chooser</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI design / wireframe (one per page)</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>functional jsp and matching form controller</td>
<td>4</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>8</td>
</tr>
<tr>
<td>jsp polishing</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>code chooser</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>service</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>domain object &amp; persistence</td>
<td>-</td>
<td>-</td>
<td>4</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>security rule definition and impl</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>test data definition and view creation (hsql)</td>
<td>-</td>
<td>4</td>
<td>-</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>client side validation</td>
<td>-</td>
<td>1</td>
<td>-</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>server side validation</td>
<td>-</td>
<td>4</td>
<td>-</td>
<td>8</td>
<td>-</td>
</tr>
</tbody>
</table>

*hours per component =* 15, 43, 40, 57, 21
Focus on Iterative Refinement...

- Ask yourself and the team
  - What more do I need to know?
  - What's wrong with the last estimate?
  - How will it be better this time (next time)?
  - Does this estimate reflect the best case or does it show the worst case?
- Be aware of estimate convergence
- Always know the status and the forecast
Estimation Principles...

- Always estimate **functionality**, cost, & time
- First early estimate based on stories
- Additional estimates from design and patterns
- Consider **risk exposure**
- Set the **schedule**, staffing and **costing** models
- Always do a **sniff test**
- Re-estimate only when necessary and even then, **don't waste time on it**: focus on iterative refinement by asking...
Best, Worst, or Expected Case...
Estimate Convergence..
Building Trust?

- Accuracy, more than precision, builds trust
- Repeat these methods from project to project to build trust between development teams and sponsors
  - Repeat these methods from project to project to build trust in your own estimation skills!
- It is this trust that directly influences productivity. When done right these methods will provide the data necessary for management without taking time away from actually constructing the system.
Building Trust...

<table>
<thead>
<tr>
<th>Effort &amp; Size Estimate</th>
<th>Pessimistic Estimate</th>
<th>Optimistic Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Concept</td>
<td>Vision Milestone</td>
<td>Objectives Milestone</td>
</tr>
<tr>
<td>Architecture Milestone</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Refining the Estimate...

- Be prepared to always know the status and the forecast:
  - status != forecast
  - status != history
- Use a card wall for status
- Use a burn chart for forecast