Improving Software Estimation

The **Significance** of **Requirements Quality** on Costs and How to Improve Estimates with Automated Requirements QA

Colin Hammond
28 years Project experience
IFPUG & COSMIC certified
Functional Sizing Evangelist

Joint IT and Software Cost Forum 2021
14 – 17 September 2021
Early Estimates

Why do we estimate cost beforehand?
To know what is realistic and plan accordingly

$1m
9 months

$10m
29 months
What do we estimate?

Software is usually the main cost and schedule driver
Variability 1- 30x
Effort <-> Functional Size
How can good estimate reduce costs

- Scope growth causes disruption & costly rework
- Optimum team size
- Efficient working
- Overstaff. Parkinson's Law

Good early estimation allows us to optimise team size for lowest cost & schedule
Early estimation

Any methodology

Knowable unknowns

1. Initial size estimate
2. “It’s bigger than we anticipated”
3. De-scope for timeline

*1000 FP system will grow typically 2%/m for non-Agile, 10% for Agile

If only we knew this at the start...
Examples - “early scope awareness”

As a visitor I can easily update my profile

As a sales manager I want to consider sales by quarter

We need to synch customer data between these three systems
What makes for good requirements?

**Requirement Quality Attributes**

- Clear
- Concise
- User-oriented
- Testable
- Measurable
- Consistent
- Complete
- Unique
- Valuable
- Design-free
Critical for estimation

Ambiguous?

Have we got it all?

Requirement Quality Attributes

Clear
Concise
User-oriented
Testable
Measurable

Consistent
Complete
Unique
Valuable
Design-free

Can it be sized? Is granular enough?

Automated requirements analysis can help
Automated Analysis

Defect discovery & fix

Automated Requirements Analysis

Functional Sizing

Test Generation

3 big benefits

Improving early “scope awareness”
Automated requirements analysis

Steps

1. Select a CSV of written functional requirements
2. Map the fields & import
3. Press “Analyze”
Automated requirements analysis

Analysing the text of software requirements

This will take about 1-5 seconds per story.
You will only consume credits for requirements that have not been previously analysed.
Automated requirements analysis

Sets of requirements

- Traction OD test
  - Some user stories
  - OWNER access

- Vehicle Embedded System
  - Example Vehicle System

- Education System Upgrades
  - Web administration application

- Jira Server Integration
  - This is just an example that can be deleted whenever you want to.

- REI Sample
  - Example user stories

### MEASURABLE

**Traction OD test**

- 182 of 421 (43%)
- 3558 CFP
- Quality Score: 30.3
- Defects:
  - Ambiguous: 239
  - Missing: 597
  - Duplicated: 78
  - Complex: 21

**Example Vehicle System**

- 13 of 19 (68%)
- 120 CFP
- Quality Score: 38.9
- Defects:
  - Ambiguous: 6
  - Missing: 42
  - Duplicated: 3
  - Complex: 1

**Vehicle Embedded System**

- 182 of 421 (43%)
- 3558 CFP
- Quality Score: 30.3
- Defects:
  - Ambiguous: 239
  - Missing: 597
  - Duplicated: 78
  - Complex: 21

**Education System Upgrades**

- 13 of 19 (68%)
- 120 CFP
- Quality Score: 38.9
- Defects:
  - Ambiguous: 6
  - Missing: 42
  - Duplicated: 3
  - Complex: 1

**Jira Server Integration**

- 13 of 19 (68%)
- 120 CFP
- Quality Score: 38.9
- Defects:
  - Ambiguous: 6
  - Missing: 42
  - Duplicated: 3
  - Complex: 1
What is the value of KNOWING this before investment?
A typical agile user story:

**Add Delivery Details**

As a ... Site visitor

I want ... Add my delivery addresss

So that ... I can receive my goods

Acceptance/Test Criteria ... 

I can click pencil to enter my zip code and full home address

---

**Analyses the text of “Who & What”**

- Determines functional meaning from the text.
- Any phraseology or taxonomy
- Pre-trained for consistent sizing results every time (no setup)
Intelligent, consistent requirements interpretation

First validate the **device**, then validate my **permissions** then, as a **waiter** I can create an **order**

<table>
<thead>
<tr>
<th>Functional Steps</th>
<th>Interpretation</th>
<th>Data Movements</th>
</tr>
</thead>
<tbody>
<tr>
<td>validate <strong>device</strong></td>
<td>Read <strong>device</strong></td>
<td>E: input <strong>device</strong> id &lt;br&gt; R: read <strong>device</strong> from storage &lt;br&gt; X: return <strong>device</strong> or error</td>
</tr>
<tr>
<td>validate <strong>permissions</strong></td>
<td>Read <strong>permission</strong></td>
<td>E: input <strong>permission</strong> id &lt;br&gt; R: read <strong>permission</strong> from storage &lt;br&gt; X: return <strong>permission</strong> or error</td>
</tr>
<tr>
<td>create <strong>order</strong></td>
<td>Create <strong>order</strong></td>
<td>E: input new <strong>order</strong> data &lt;br&gt; R: check if <strong>order</strong> id exists &lt;br&gt; W: write new <strong>order</strong> &lt;br&gt; X: return error or confirmation</td>
</tr>
</tbody>
</table>

Takes 1-5 seconds

Based on a likely interpretation
## Automated CRUD analysis - detects missing requirements

**CRUD and consistency analysis** Find and fix potential inconsistencies, missing and duplicate Stories.

<table>
<thead>
<tr>
<th>Object (226) confirm</th>
<th>Create (93)</th>
<th>Read (85)</th>
<th>Update (96)</th>
<th>Delete (33)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>card</strong></td>
<td>Missing</td>
<td>Missing</td>
<td>Missing</td>
<td>Story/Bug Requirements</td>
</tr>
<tr>
<td><strong>case</strong></td>
<td>Missing</td>
<td></td>
<td>Missing</td>
<td>Duplicate</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Remove Estimated Hours, Remove Estimated Hours</td>
</tr>
<tr>
<td><strong>case detail view</strong></td>
<td>Missing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>case field</strong></td>
<td>Missing</td>
<td></td>
<td>Missing</td>
<td></td>
</tr>
<tr>
<td><strong>chain</strong></td>
<td>Missing</td>
<td></td>
<td>Group Name Chain on Co</td>
<td></td>
</tr>
<tr>
<td><strong>chatter</strong></td>
<td>Missing</td>
<td></td>
<td>Chatter panel should h</td>
<td></td>
</tr>
</tbody>
</table>
Auto-generated Use Case Model Diagrams

Visually Shows:
1. Size
2. Complexity
3. Consistency
4. Completeness
5. Duplication
“Buried functionality” example

User Story
As a visitor I can update my profile

Acceptance Criteria
Check my role and verify I have permissions to update my profile.
Select my company from a dropdown of existing companies
Experience: over 250k requirements analysed in 6 languages

**Requirements:**

- **Quality is low,** very little discipline in requirements writing,
- Many people **bury functionality** in the acceptance criteria, or **omit** it.
- Average user story is: 12 words, 6 CFP (two functional steps)
- Defect potential (in the user story) is about 3 defects
- Typical cost range per story: $400-$12k

**Automation:**

- Performs the heavy lifting of analysis, QA, sizing and test gen
- Can analyse **sets of user stories** - not just individual ones.
- Thorough: up to 380 - 5,000 tests on each user story (average 1,000)
- Finds 50% requirements defects ~ 1-2 per requirement.
- Improves estimation speed and accuracy
- Frees up time to do more value-add work
Automated requirements analysis

Supported Sizing Methodologies

1. COSMIC
2. IFPUG
3. SiFP

Why we Recommend COSMIC

- More precise with *incomplete requirements*
- Simpler: 85% automated almost *no pre-requisite knowledge*
- Proven high correlation to effort

https://cosmic-sizing.org/publications/measurement-guide/
## Reliable Sizing

### Factors that affect a reliable early functional size measurement

<table>
<thead>
<tr>
<th>Factors</th>
<th>Indicative Variability</th>
<th>Rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Requirements Quality</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Completeness of scope</td>
<td>Up to 400%</td>
<td>• We often see only one function mentioned when a full set of CRUD is required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• We almost never see underestimates.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Inconsistent object names can lead to overstatement of Data groups/OOI/ILFs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Ambiguities are often masking understatements of scope.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Functionality is buried in the acceptance criteria</td>
</tr>
<tr>
<td><strong>Requirements Quality</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other attributes</td>
<td>Up to 200%</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Methodology:</strong> IFPUG vs COSMIC</td>
<td>Up to 60%</td>
<td>• The gross FP count between these methodologies is typically in the same order of magnitude.</td>
</tr>
<tr>
<td><strong>Precision of measurement</strong></td>
<td>Up to 20%</td>
<td>• Whether using ScopeMaster or doing it manually a highly skilled vs novice counter rarely more than 20% variance.</td>
</tr>
</tbody>
</table>
Additional Benefits

For DEVELOPER
- Builds a dynamic model

For TESTERS
- Generates traceable test scenarios

For PMO / PROJECT MANAGER
- Objective QA & sizing report

For PROJECT MANAGER
- Scope tracking

For COST ESTIMATOR
- Scope tracking
Takeaways

1. Early, **functional size estimates matter**
2. **Requirements quality** matters to estimation & cost.
3. Need **proactive** effort to reduce unknowns and improve quality
4. **Automation** now consistently estimates functional size for auditable forecasts with insignificant effort.

https://cosmic-sizing.org/publications/measurement-guide/
https://www.iceaaonline.com/scebok/
https://www.scopemaster.com/blog/story-points-compared-with-cfp
https://www.scopemaster.com/solutions/tour/

Colin.Hammond@scopemaster.com