Avoid software project horror stories

Check the reality value of the estimate first!

Harold van Heeringen
ICEAA 2014

Harold van Heeringen

NESMA board member
Nederlandse Software Metrieken Association
www.nesma.org

President ISBSG
International Software Benchmarking Standards Group
www.isbsg.org

IAC member COSMIC
Common Software Measurement International Consortium
www.cosmicon.com

Senior Software Cost Engineer,
Sogeti Nederland B.V.
www.sogeti.com
Questions for our session today

Software project horror stories – what are they?
The software industry – when will it mature?
Experts – what’s wrong with them?
Why should we do parametric software estimates?
How can we assess the reality value of an estimate?
What should we do to avoid horror stories, or at least decrease the risk of these to happen?

Software industry

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Successful</td>
<td>29%</td>
<td>35%</td>
<td>32%</td>
<td>37%</td>
<td>39%</td>
</tr>
<tr>
<td>Failed</td>
<td>18%</td>
<td>19%</td>
<td>24%</td>
<td>21%</td>
<td>18%</td>
</tr>
<tr>
<td>Challenge</td>
<td>53%</td>
<td>44%</td>
<td>44%</td>
<td>47%</td>
<td>47%</td>
</tr>
</tbody>
</table>

Project resolution results from CHAOS research for years 2004 to 2012.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>84%</td>
<td>72%</td>
<td>79%</td>
<td>71%</td>
<td>74%</td>
</tr>
<tr>
<td>Cost</td>
<td>55%</td>
<td>47%</td>
<td>54%</td>
<td>46%</td>
<td>59%</td>
</tr>
<tr>
<td>Features</td>
<td>64%</td>
<td>63%</td>
<td>67%</td>
<td>74%</td>
<td>69%</td>
</tr>
</tbody>
</table>

Time and cost overruns, plus percentage of features delivered from CHAOS research for the years 2004 to 2012.
Avoid Software Project Horror Stories - Check the Reality Value of the Estimate First!

Impact

Deliver too late: losing business.

Fail/stop: loss of time, money, business and still no solution for the problem that needed to be solved.

Waste of resources that could have been deployed successfully otherwise.

Impact

Deliver too late: losing business.

Fail/stop: loss of time, money, business and still no solution for the problem that needed to be solved.

Waste of resources that could have been deployed successfully otherwise.

Horror stories

Avon Pulls Plug On $125 Million SAP Project

Avon halts its global rollout of an SAP system after a Canadian pilot project prompts re-think

California courts throw hug of relief

A project that was intended to support the system in 11 countries had to be scrapped. Is this a good business practice?

Avon Pulls Plug On $125 Million SAP Project

Avon halts its global rollout of an SAP system after a Canadian pilot project prompts re-think

California courts throw hug of relief

A project that was intended to support the system in 11 countries had to be scrapped. Is this a good business practice?

BBC was ‘complacent’ over failed £100m IT project

The BBC was "far too complacent" in its handling of a failed IT project that cost licence fee payers £98.4m.

The Digital Media Initiative (DMI) was intended to move the BBC away from using and storing video tape.

But it was scrapped, with almost no results, after five years of development.

Related Stories

BBC boss sacked over failed project

BBC's £100m IT failures detailed

These are tax dollars and one of the reasons the whole country was in recession for years.

Failing IT projects cost the government 7 billion USD per year

Projects > 10 million USD only 7% succeeds.

In total, only 30% of IT projects are successful.

In total, only 30% of IT projects are successful.

In total, only 30% of IT projects are successful.

These are tax dollars and one of the reasons the whole country was in recession for years.

Failing IT projects cost the government 7 billion USD per year

Projects > 10 million USD only 7% succeeds.

In total, only 30% of IT projects are successful.

In total, only 30% of IT projects are successful.

In total, only 30% of IT projects are successful.
Avoid Software Project Horror Stories - Check the Reality Value of the Estimate First!

Two main reasons

Unstable user requirements
- Starting the development too early in the project
- Not enough time spent on requirements analysis
- Users not involved or not involved enough

Unrealistic project expectations
- Usually: only expert estimates (optimistic)
- Pressure to lower cost and deliver faster
- End date is not estimated, but a given
- Duration is an important cost driver!

Requirements

Worst in class software development organizations spend 7.5% of the project budget on requirements

<table>
<thead>
<tr>
<th>Req.</th>
<th>Coding and Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5 hours/FP</td>
<td>17.5 hours/FP</td>
</tr>
</tbody>
</table>

Best in class software development organizations spend 28% of the project budget on requirements

<table>
<thead>
<tr>
<th>Req.</th>
<th>Coding and testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0 hours/FP</td>
<td>7.7 hours/FP</td>
</tr>
</tbody>
</table>

More effort spent on requirements increases project success!
Avoid Software Project Horror Stories - Check the Reality Value of the Estimate First!

**Unrealistic expectations**

Software project industry: low maturity
- Low estimation maturity;
- No or little formal estimation processes;
- No or little use of historical data;
- Customers choose suppliers based on price, not reality.

No reality check before finalizing an estimate! So, many unrealistically estimated projects actually start!

Results:
- Many failing projects
- Low customer satisfaction rates

Why do we need realistic estimates?

A realistic estimate is one of the most important **conditions** for a successful project.

The estimate is the **basis** for:
- Business case;
- Planning;
- Proposal (outsourcing: fixed price / date);
- Financial result of the project… and the organization;
- Claiming and releasing of resources;
- Alignment between IT and business / customer;
- Progress reports / dashboards;
- The feeling of the team and the stakeholder.

Without a realistic estimate, the project is **likely to fail**!
Software estimation is hard

It’s hard to accurately estimate software projects:
• Software is hard to measure, because intangible;
• Technical environments change all the time;
• Software companies are not mature enough to measure performance and store the metrics of completed projects;
• Software companies don’t use data of completed projects in new estimates;
• The estimate often has to be finalized before the requirements are fully known;
• It’s hard to estimate whether requirements will change, how much they will change, to factor this in the estimate and to explain this.
• It’s hard to estimate which technical challenges have to be solved during the project;
• …

The effect of underestimation

- Non-linear extra costs
  - Planning errors
  - Team enlargement → more expensive, not faster
  - Extra management attention / overhead
  - Stress: More defects, lower maintainability!!

- Linear extra costs
  Extra hours will be used
Two types of project estimation

Two types of software estimation
- Expert estimate (Bottom-up / WBS)
- Parametric (Top-down / methodical)

Expert Estimate
- Technical specialists
- Bottom-up, effort estimate for activities identified (WBS)

Parametric estimate
- Based on size and historical data
- Use of Parametric models

Expert estimates

Bottom-up, assign effort hours to work items, based on knowledge and experience

Result: expert estimates are optimistic, on average 30% underestimation.

Disadvantages:
- Forgotten activities (testscript reviews, ...);
- No good foundation of the estimate, very subjective;
- The expert is not going to do all the work (who will ?);
- How expert is the expert? (projects are unique);
- Experts don’t take into account duration, team size, etc.;
- Experts don’t assess the reality value, no real use of history.
The effect of underestimation

Parametric estimates

Top-down, based on size, historical data and parametric models

Advantages:
• Objective, repeatable, verifiable, risk;
• Scenario analysis: duration, team size, % confidence.

Disadvantages:
• Requires a certain maturity level of an organization;
• Measurement and analysis of completed projects;
• Investment in expertise and tooling;
• Documentation requirements – must be possible to measure;
• Knowledge about estimation parameters is necessary.

Result: realistic estimate, scenario’s, risk profile.
It-20 - Avoid Software Project Horror Stories - Check the Reality Value of the Estimate First!

Many projects start with optimistic estimates
- Organizations that use only expert estimates;
- Outsourcing: suppliers don’t use parametric estimation;
- Outsourcing: customer selects lowest price;
- Business / Customer: pressure to deliver faster and cheaper;
- Final estimates have to be made based on incomplete requirements;

Software horror stories are a common phenomenon
- But many can be avoided by using parametric estimates
- Or at least perform a reality check of the estimate!!
Basic estimation model

Most important input parameter is **size**.

The two most used size units for software are:

- **Technical size** – slocs
  - Has to be guessed / not possible to measure upfront
  - More slocs is good / bad? Who needs slocs?
  - No standard;
  - Slocs cannot really be compared between technical environments.

- **Functional size** – function points
  - Can be measured upfront (fairly accurately);
  - More function points means more functionality;
  - ISO/IEC standard;
  - Independent of technical environment.
Estimate breakdown

All effort estimates can be broken down to these components:

- Size (Unit of Measure)
- Productivity (effort hours per UoM)
- Adjustments (informed decisions about the specific project)

When this is done, it becomes possible to perform a reality check based on historical data.
- Company data
- Industry data

ISBSG

International Software Benchmarking Standards Group
- Independent and not-for-profit;
- Full Members are non-profit organizations;
- Grows and exploits two repositories of software data (in .xls):
  - New developments and enhancements (> 6000 projects);
  - Maintenance and support (> 1200 applications).

Everybody can submit project data
DCQ on the site / on request (.xls)
Anonymous
Free benchmark report in return
ISBSG industry data

Mission: “To improve the management of IT resources by both business and government, through the provision and exploitation of public repositories of software engineering knowledge that are standardized, verified, recent and representative of current technologies”.

All ISBSG data is
- validated and rated in accordance with its quality guidelines
- current
- representative of the industry
- independent and trusted
- captured from a range of organization sizes and industries

Performing a reality check

Reality check of an estimate
1. Break down the estimate
   - Size
   - Productivity
   - Adjustments
2. Select a relevant peer group in the historical database
3. Analyze the productivity of this peer group
4. Define the ‘reality zone’
5. Assess whether the estimate is in the reality zone
   - If yes, the estimate is probably realistic
   - If no, the estimators have to explain why
**Example**

**Project X**
Technical environment: Java
Effort estimate (by team of experts): 2000 hours
No risk adjustments made

**Breakdown:**
1. Functional size: 411 function points (IFPUG)
2. Productivity: 2000/411 = 4.9 hours/FP
3. Adjustments: 0 hours

---

**Select relevant peer group**

ISBSG ‘New Developments & Enhancements’
Select a relevant ‘peer group’
Data Quality A or B
Count approach: IFPUG 4.x or NESMA
Primary Programming Language = ‘Java’
300 FP < Project Size < 500 FP
Avoid Software Project Horror Stories - Check the Reality Value of the Estimate First!

Data analysis

488 projects selected

Productivity Analysis:

<table>
<thead>
<tr>
<th>Productivity</th>
<th>h/FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum</td>
<td>0.1</td>
</tr>
<tr>
<td>Percentile 10</td>
<td>4.5</td>
</tr>
<tr>
<td>Percentile 25</td>
<td>6.7</td>
</tr>
<tr>
<td>Median</td>
<td>9.8</td>
</tr>
<tr>
<td>Percentile 75</td>
<td>15.4</td>
</tr>
<tr>
<td>Percentile 90</td>
<td>21.6</td>
</tr>
<tr>
<td>Maximum</td>
<td>78.3</td>
</tr>
<tr>
<td>Average</td>
<td>14.2</td>
</tr>
</tbody>
</table>

Reality zone: Percentile 25 – Percentile 75

Estimate project X: 4.9 h/FP
Not realistic !!

Conclusions & recommendation

- Accurate software estimation is hard;
- Optimistic estimates result in failing projects;
- Most organizations only use expert estimates;
- Expert estimates usually result in optimistic estimates;
- Performing reality check using historical data is easy.

Strong recommendation:

Always perform a reality check to decrease the risk of horror stories!!