CS445 / SE463 / ECE 451 / CS645

Software requirements specification & analysis

Cost and Time Analysis of Ou's Development of WD-Pic

Summer 2021

Daniel Berry

What I Want To Do

Take actual data about the development of WD-Pic from Lihua Ou’s thesis and estimate

- the time to do an agile development of WD-Pic with the same sequence of modifications that she followed in her actual upfront-RE development, by use of what we know about the cost of modifying code, and
- the time to do a traditional waterfall development of WD-Pic by use of COCOMO.

Then

Compare these two estimates with the time she actually spent and her predicted time for the waterfall.

Actual Time to Develop with Upfront RE

4.9 months to get final version of the UM = RS, after 11 versions

2 months to get the first version (using the planned duration of requirements specification as estimate)

2.9 months to converge from first version to final version with 10 versions

Estimated Time for Agile Development

If Ou had implemented the first version of the manual:

7 months, using the plan, which allocated 7 months to implement

These estimates are usually optimistic. One rule of thumb is double the estimate

14 months, doubled estimate

If you believe that coding costs 10 times writing a spec, then the estimate is 2 months * 10

20 months, factor of 10 estimate

So from 7 to 20 months implementing the first version.

If the agile game is played in which each revision represents a new spike, and you believe that correcting an error during coding costs 10 times correcting it during writing the spec

29 months to fix the defects that the revisions represent.

So, if the agile game were played, the total time to implement would be between:

36 months, 7 + 29

and

49 months, 20 + 29

Estimated Time, by COCOMO, for Waterfall Development

```
9589 Compiler   C
137   ExtEditor   Java
3705 java: Java
1460 ->AttrPane     Java
6023 ->MyDialog    Java
2401 ->PIC    Java
220   ->resources   Java
------------------------------
23535   total  C or Java

0   java:
0   ->help
11285   ->->index.html   HTML
------------------------------
11285   total  HTML
```

```
23535   total  C or Java
11285   total  HTML
------------------------------
34820   total  achieved by applying "wc" to all appropriate
files in the source code directory
-9567   reused  pic
------------------------------
25253   total
```

according to Ou's MMath Thesis

```
24091   total  WD-pic
9567  reused  pic
897   new  pic
------------------------------
34555   total
-9567   reused  pic
------------------------------
24988   total
```

So either way, L = 25 KLOC

E= a * L**b

Since the program has some elements of being organic, i.e., one person and some elements of being semi-detached, i.e., the one person was NOT familiar with the application, use

a = 2.7 (mean(2.4,3.0))

b = 1.085 (mean(1.05,1.12)

E = 2.7 * 25**1.085
  = 2.7 * 32.8673666369
  = 88.7418899196 person-months

BUT it is a one-person project. So maybe it IS organic

a = 2.4

b = 1.05

```
E = 2.4 * 25**1.05
  = 2.4 * 29.3654735772
  = 70.4771365853 person-months
```

```
0   java:
12->images  bitmaps
0   ->help
107   ->->manual_files   bitmaps
-------------------------------
119   total  bitmaps
```

assuming one hour to make each bitmap

119 hours = .66 person-months

Total:
From
71.14 person-months to
89.40 person-months

from 70-90 person months

or between 7 and 9 times the estimate and what it took doing the manual completely before the implementation

Compare: Actual Upfront RE vs Estimated (Agile vs Waterfall)

User's manual as RS, delay implementation until RS is complete:
10 months

Agile Lifecycle:
from
36 months
to
49 months

Typical Waterfall:
70 months
to
90 months

Limitations of This Analysis

We can never compare two methods of developing software accurately.

If we have two different sets of groups, each applying one method to the development of one common system, we cannot be sure that the difference in outcome is caused by the difference in methods and not differences in the sets of groups and differences among the people in the groups. The larger the sets, the more likely that randomization washes away all differences except those between the methods.

Limitations of This Analysis

If we have one set of groups apply two different methods to the same problem, then each learns from the first application and does better in the second application. This is called a learning effect.

If we have one set of groups apply two different methods to two different systems in random order, we cannot be sure that the difference in outcome is caused by the difference in methods and not differences in the systems.

CS445 / SE463 / ECE 451 / CS645

Software requirements specification & analysis

Cost and Time Analysis of Ou's Development of WD-Pic

Summer 2021

Daniel Berry