

Evidence Based Scheduling

A glowing green glass with a complex, web-like structure inside, set against a dark background with light streaks and bokeh.

Adam Dudczak
(adudczak@gmail.com)

I Spotkanie dyskusyjne Poznań Java User Group

28 maja 2008

Plan prezentacji

- 1 Grunt to dobry plan...?
- 2 Evidence Based Scheduling
- 3 Jak to działa?
- 4 Rady wujka Joela
- 5 Zakończenie

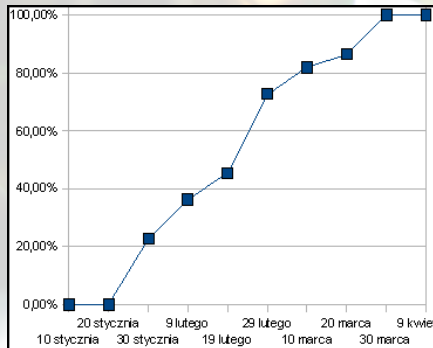
Dlaczego to takie trudne?

- Po co tracić czas, to i tak się nie sprawdzi.
- Jak przewidzieć to czego nie da się przewidzieć?
- A może dorzucimy animacje, ajaxy i przepisemy od początku w Ruby?
- ... daj spokój, przecież i tak nie będzie czasu na testy.

Evidence Based Scheduling

- Evidence Based Scheduling umożliwia tworzenie harmonogramów, które się sprawdzają.
- Integralna część cyklu wytwarzania oprogramowania wspieranego w systemie FogBugz.
- Evidence Based Scheduling to:
 - prosta, ogólna metoda i da się ją zastosować bez FogBugz'a.
 - zbiór porad i praktyk o których warto pamiętać zawsze!

Evidence Based Scheduling



- Pozwala określić najbardziej prawdopodobną datę zakończenia prac.
- Jedyne co jest potrzebne, to dane dotyczące przewidywanego i rzeczywistego czasu wykonania zadań przez poszczególnych programistów.

Przede wszystkim dekompozycja

Dekompozycja, dekompozycja, dekompozycja

- Określenie maksymalnego czasu trwania zadania np. 16h.
- Im dokładniejszy opis poszczególnych zadań tym większa szansa na dobre oszacowanie.
- Wiedza o pewnych aspektach implementacji już na etapie wydzielania zadań – planują programiści, a nie kierownictwo!

Jak dużo czasu potrzeba?

Śledzenie czasu wykonania zadań

- Należy oszacować czas wykonania zadania i zestawić to z rzeczywistym wynikiem.
- Reinstalacja Windowsa, niespodziewane spotkanie, gadatliwy kolega, nieprzespana noc – **czy to też się liczy?**
- Skuteczność oszacowań

Numer zadania	1	2	3
Czas przewidywany	4	5	3
Czas rzeczywisty	10	4	6
Skuteczność oszacowania	0.4	1.25	0.5

Jak dużo czasu potrzeba?

Śledzenie czasu wykonania zadań c.d.

- Przykładowe rozkłady skuteczności szacowania:
 - $\{1, 1, 1, 1\}$ – idealny
 - $\{12, 0.1, 1.5, 4.7\}$ – zły
 - $\{0.4, 1.25, 0.5, 0.6\}$ – normalny
- Im dłuższy staż tym lepsze oszacowania – wyrzucić stare dane!
- Jeżeli programista nie ma swojej historii oszacowań należy stworzyć dla niego historie z dużym rozrzutem skuteczności szacowania (przykład „zły”).

Z wizytą u wróżki...

Analiza możliwych scenariuszy

- Szacowanie przy użyciu **metody Monte Carlo**
- Przykład, Programista: **Bolek**
 - Wektor skuteczności oszacowań Bolka: {0.6, 0.5, 0.5, 0.7, 0.95 ...}
 - Zbiór zadań do wykonania: {2, 5, 12, 16, 31}

Numer zadania	2	5	12	16	31	
Oszacowanie Bolka [h]	4	8	2	8	16	
Losowa skuteczność	0.6	0.5	0.6	0.6	0.5	
Prawdopodobny czas [h]	6.7	16	3.3	13.3	32	71.3

Z wizytą u wróżki...

Analiza możliwych scenariuszy c.d.

- Wynik pierwszej rundy symulacji MC dla Bolka to : **71.3h**.
- Po uwzględnieniu : urlopów, dni roboczych i zależności kolejnościowych między zadaniami dostaniemy prawdopodobną datę zakończenia prac Bolesława.
- Powtarzamy taką symulację wiele razy i w ten sposób uzyskujemy wykres obrazujący prawdopodobieństwo zakończenia prac w funkcji daty.

Data zakończenia	04.05.2008	18.05.2008	1.06.2008
Liczba wystąpień daty	4	10	12

- Jak widać to może być dość żmudne... ale od tego są właśnie komputery.

Rady wujka Joela

- Dekompozycja i szacowanie czasu wykonania zadania wymaga wiedzy nt. implementacji – to zadanie dla programistów, a nie dla kierownictwa.
- Czas konieczny na naprawianie błędów w napisanym kodzie powinien być doliczany do zadania w ramach którego ten kod został napisany.
- Trzymaj się swoich oszacowań – nie ulegaj presji zarządzających projektem.
- Wyrzuć to co niepotrzebne – presja czasu może być pożyteczna.

Czytanki

- 1 <http://www.joelonsoftware.com/items/2007/10/26.html>
– „Evidence Based Scheduling” – Joel Spolsky
- 2 <http://www.fogcreek.com/FogBugz/docs/60/topics/schedules/Evidence-BasedScheduling.html> –
„Evidence-Based Scheduling” - Fog Creek Software

Pytania?

„Evidence Based Scheduling”

Spotkanie dyskusyjne **Poznań Java User Group**
adudczak@gmail.com

Prezentacja powstała w oparciu o artykuł „**Evidence Based Scheduling**”
którego autorem jest Joel Spolsky. Tytułowa ilustracja pochodzi z
<http://flickr.com/photos/ckaroli/2050812134/>.