Testowanie mutacyjne

Czyli jak dobre w rzeczywistości są Twoje testy?

Marcin Zajączkowski
m.zajaczkowski@gmail.com

Warszawa, 2013-07-06

confitura'13

Ja technicznie

Java architect TDD practitioner
Team mentor Clean code developer
Software Craftsmanship
Evangelist
FOSS developer Linux enthusiast
IT Trainer Code quality freak Blogger

Plan prezentacji

- Czym jest i jak działa testowanie mutacyjne?
- Dlaczego jest nieznane i rzadko stosowane?
- PIT – narzędzie, które działa
- Mutanci w akcji – pokaz praktyczny
- Zastosowanie w Twoim projekcie

http://www.townofdarienny.com/

http://www.oldwestlawmansforgottenmemoir.com/memoir_bringsV13.html

WANTED

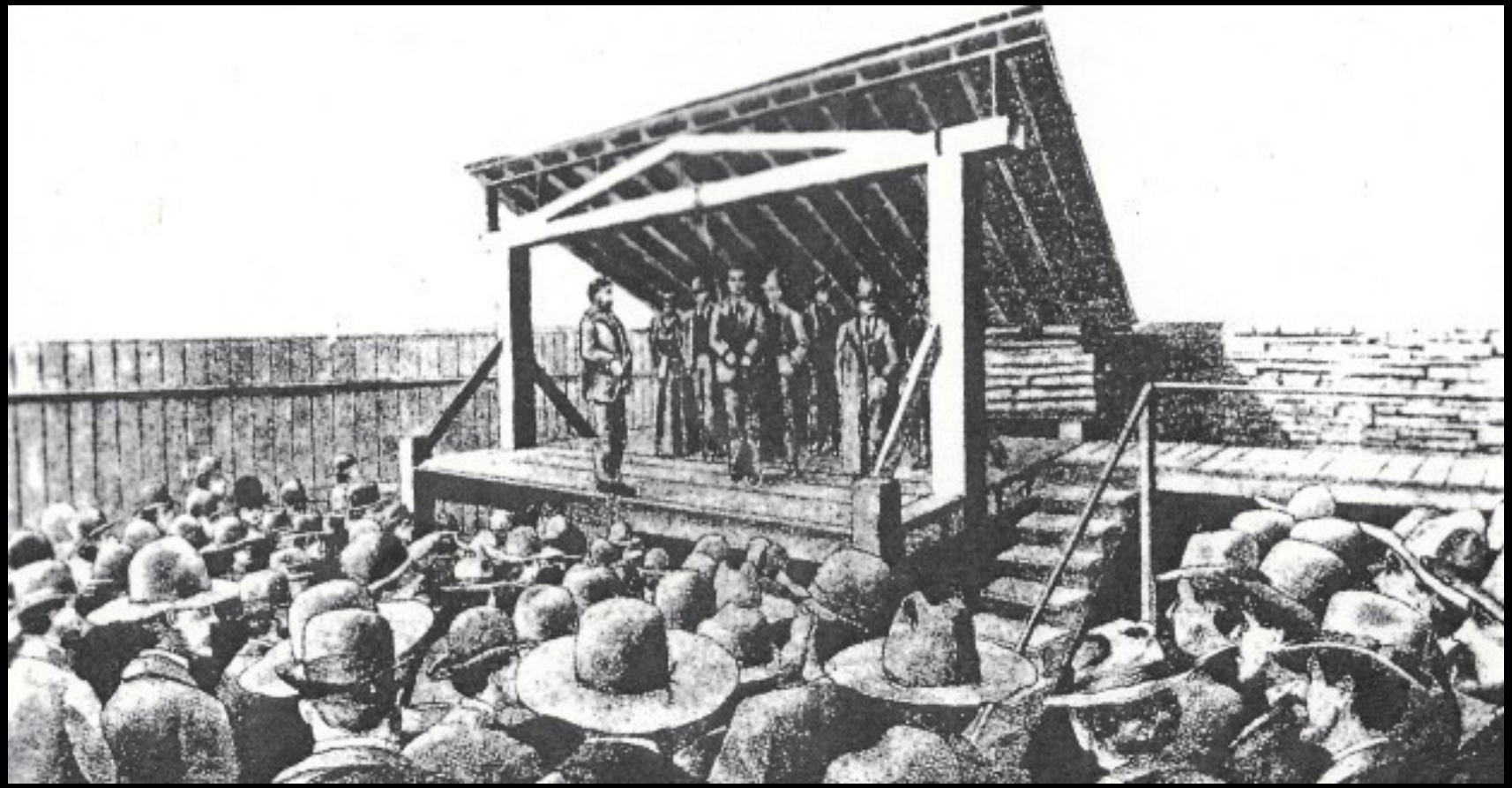
DEAD OR ALIVE

BILLY THE KID

WILLIAM BONNEY

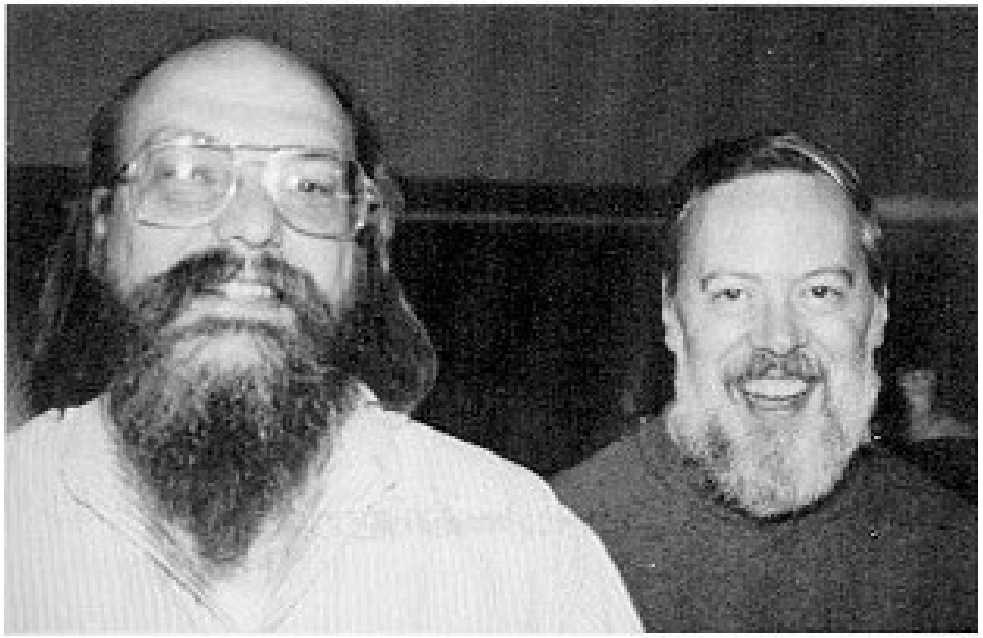
$5,000 REWARD

NOTIFY- Marshall Pat Garrett



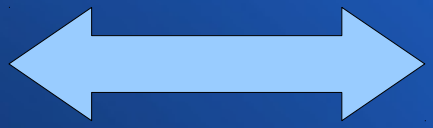
http://www.nps.gov/fosm/historyculture/executions-at-fort-smith-1873-to-1896.htm

https://secure.flickr.com/photos/7402220@N02/491093210/



http://publish.illinois.edu/libraryitnews/2012/06/

Analogie



Projekt	Miasto
Błędy w kodzie	Przestępstwa
Testy automatyczne	Szeryfowie
Pokrycie kodu	Ścieżki patrolowe
Mutanci w kodzie	Prowokacje

Pomysł z analogią do przestępczości, policji i jej kontrolowania zaczerpnięty od Chrisa Rimmera

Testowanie mutacyjne

- Uszkodzenie wybranej linii kodu produkcyjnego (wprowadzenie mutacji)
- Sprawdzenie, czy jakikolwiek test automatyczny to wykryje (czy przestanie przechodzić)
- Mutacje, które przetrwały (nie zostały zabite) są potencjalnymi błędami, które nie zostałyby wykryte przez testy

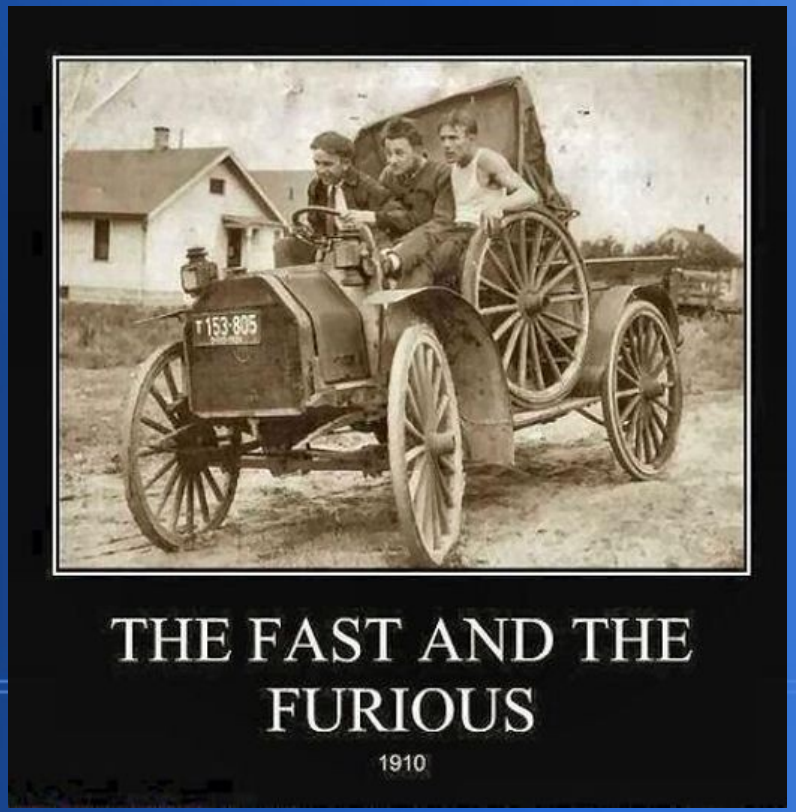
Problemy z narzędziami

- Mała liczba narzędzi dla Javy (wiele już nieutrzymywanych)
- Długi czas wykonywania
- Wymagana modyfikacja kodu produkcyjnego
- Nieskończone pętle
- Przepełnienie stosu

http://bestclipartblog.com/27-tools-clip-art.html/tools-clip-art-2

PIT – szybkie mutowanie

- Manipulacja bajtkodem
- Mutowanie tylko linii ze standardowym pokryciem
- Wykonywanie tylko powiązanych testów
- Zrównoleglenie wykonania
- Analizy przyrostowe



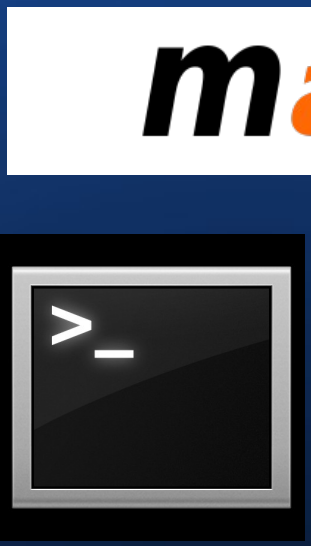
http://carhumor.net/blast-from-the-past/

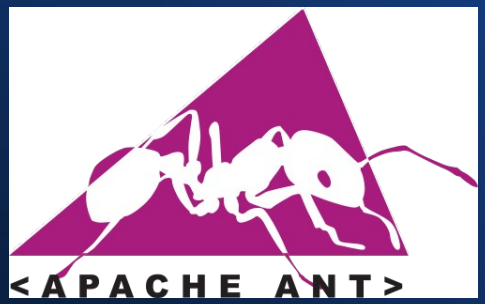
PIT – wiele mutacji

- Conditionals Boundary Mutator
- Negate Conditionals Mutator
- Math Mutator
- Increments Mutator
- Invert Negatives Mutator
- Return Values Mutator
- (Non) Void Method Calls Mutator
- I nie tylko...

1 - http://blog.spoongraphics.co.uk/tutorials/create-a-cute-furry-vector-monster-in-illustrator
2 - http://blog.spoongraphics.co.uk/tutorials/create-a-cute-vector-monster-from-a-pencil-sketch

PIT – bogaty ekosystem





TestNG

Spock

Zdjęcia - strony domowe przedstawionych projektów

PIT – mocne strony

- Szybki
- Z dużymi możliwościami
- Dobra integracja z innymi narzędziami

Alternatywy

- Javalanche – mały ekosystem
- µJava – ograniczony dostęp do kodu narzędzia
- Jester – obecnie nie utrzymywany
- Jumple – obecnie nie utrzymywany
- Judy – obecnie nie utrzymywany,
  produkt z Wrocławia

Mutanci w akcji

http://www.adolescentadulthood.com/2013/01/23/how-did-the-teenage-mutant-ninja-turtles-get-their-names/

Co można zyskać?

- Informację, jak dobre w rzeczywistości są Twoje testy
- Miejsca w kodzie, które nie są należycie testowane
  - Dokładniej niż przy “zwykłym” pokryciu kodu
- Lepszą jakość kodu
- Mniej błędów wykrytych na produkcji
- Zadowolenie z pracy
- ... (i inne korzyści z pisania testowalnego kodu)

Kiedy zastosować?

- Pisany od zera projekt nastawiony na jakość
- Wysokie pokrycie kodu, ale mimo to wciąż błędy na produkcji, które mogłyby (i powinny) być wykryte przez testy
- Wątpliwość w jakość testów
  - Wymaganie w HLD – 95% pokrycia kodu przy jednoczesnej realizacji przez zespół, który wcześniej nie pisał testów automatycznych
- Niedosyt mimo wysokiego pokrycia kodu

Dostosuj swoją aplikację

- **Pisz testy**
- Pisz szybkie jednostkowe testy (nie tylko wolne integracyjne)
- Oddzielaj szybkie testy jednostkowe od wolnych integracyjnych
  - Bądź w stanie uruchamiać tylko wybraną grupę testów

Czy ktoś tego używa
w “aplikacjach enterprise”?

- Tak :-)

- The Ladders
- Jumi
- Może Ty?

http://www.mysciencework.com/fr/MyScienceNews/10027/de-l-in-opportunite-des-open-spaces-dans-les-labos

Posumowanie korzyści

Sprawdzenie skuteczności testów automatycznych

Bardziej niezawodny kod

Mniej problemów w pracy

Więcej czasu na ciekawe rzeczy

Satysfakcja z wykonywanej pracy

Dziękuję za uwagę

Marcin Zajączkowski
m.zajaczkowski@gmail.com

http://blog.solidsoft.info/