Optymalizacja Automatycznych Testów Regresywnych

W Organizacji Transformującej do Agile

Adam Marciszewski
adam.marciszewski@tieto.com

Agenda

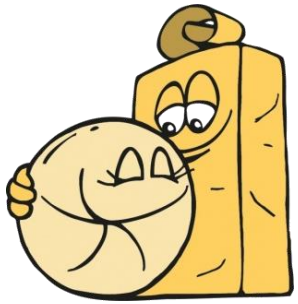
- Kontekst projektu
- Typowe podejście
- Wyzwania
- Cel
- Założenia
- Opis metody krok po kroku
- Testy jako analogia do próbkowania sygnałów
- Odkrycia
- Wynik Optymalizacji
- Materiały źródłowe
- Pytania i Odpowiedzi

Kontekst projektu

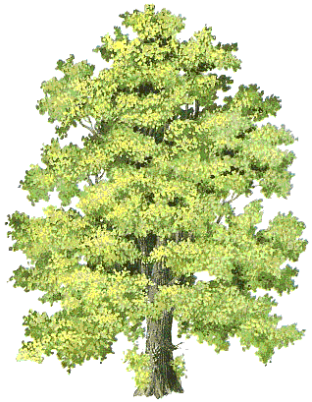
- Rozproszony geograficznie projekt, realizowany na rzecz globalnego klienta.
- Projekt przejęty od innego wykonawcy.
- Klient podjął decyzję by transformować do Scrum
- Bardzo złożony produkt (kilkanaście podsystemów, kilka różnych technologii)
- Wymagania dostępne tylko dla nowych cech

Typowe podejście

- Analiza Strategii Testów i Planu testów
- Konfrontacja wymagań i przypadków testowych
- Ocena pokrycia wymagań/funkcjonalności testami
- Ocena ryzyka
- Optymalizacja
- …wydawałoby się: bułka z masłem



- … a tu jednak Lipa! ☹



Wyzwania

- Brak wymagań
- Brak wiedzy na temat pokrycia systemu testami
- Czas wykonania automatycznej regresji > 30h
- Duża ilość przypadków testowych (>1000)
- Duże ryzyko przedostania się błędów do środowiska klienta
- Brak wiedzy na temat skuteczności testowania
- Niska skuteczność detekcji błędów

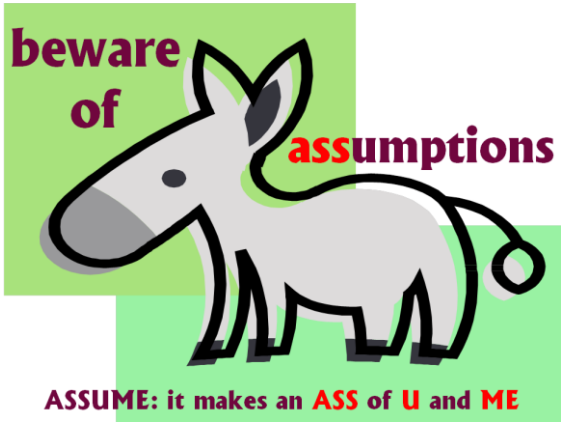
Cel

- Czas wykonania regresji < 12 h
- Skrócenie czasu regresji nie może obniżyć skuteczności wykrywania błędów
- Skrócenie czasu regresji nie może zredukować pokrycia systemu testami
- Obszary obarczone większym ryzykiem będą testowane intensywniej
- Obszary kluczowe dla klienta będą testowane intensywniej
- Zmienna granulacja przypadków testowych

Założenia

Przystępując do optymalizacji przyjęto następujące założenia:

- Klient otrzymał wszystkie wymagane cechy
- Jeśli nie dostarczono jakiejś cechy, a klient tego nie zauważył od kilku lat – to tak naprawdę nie potrzebował tej funkcjonalności ;)
- Dostarczone cechy spełniają wymagania funkcjonalne
- Walidacja była poprawna
- Problemy po stronie weryfikacji



Opis metody - krok po kroku

Faza 1 – Analiza i Optymalizacja zbioru testów regresywnych

Faza 2 – Optymalizacja techniczna

Faza 3 – Monitorowanie i dostrajanie (minimalizacja ryzyka)

Faza 4 – Start!

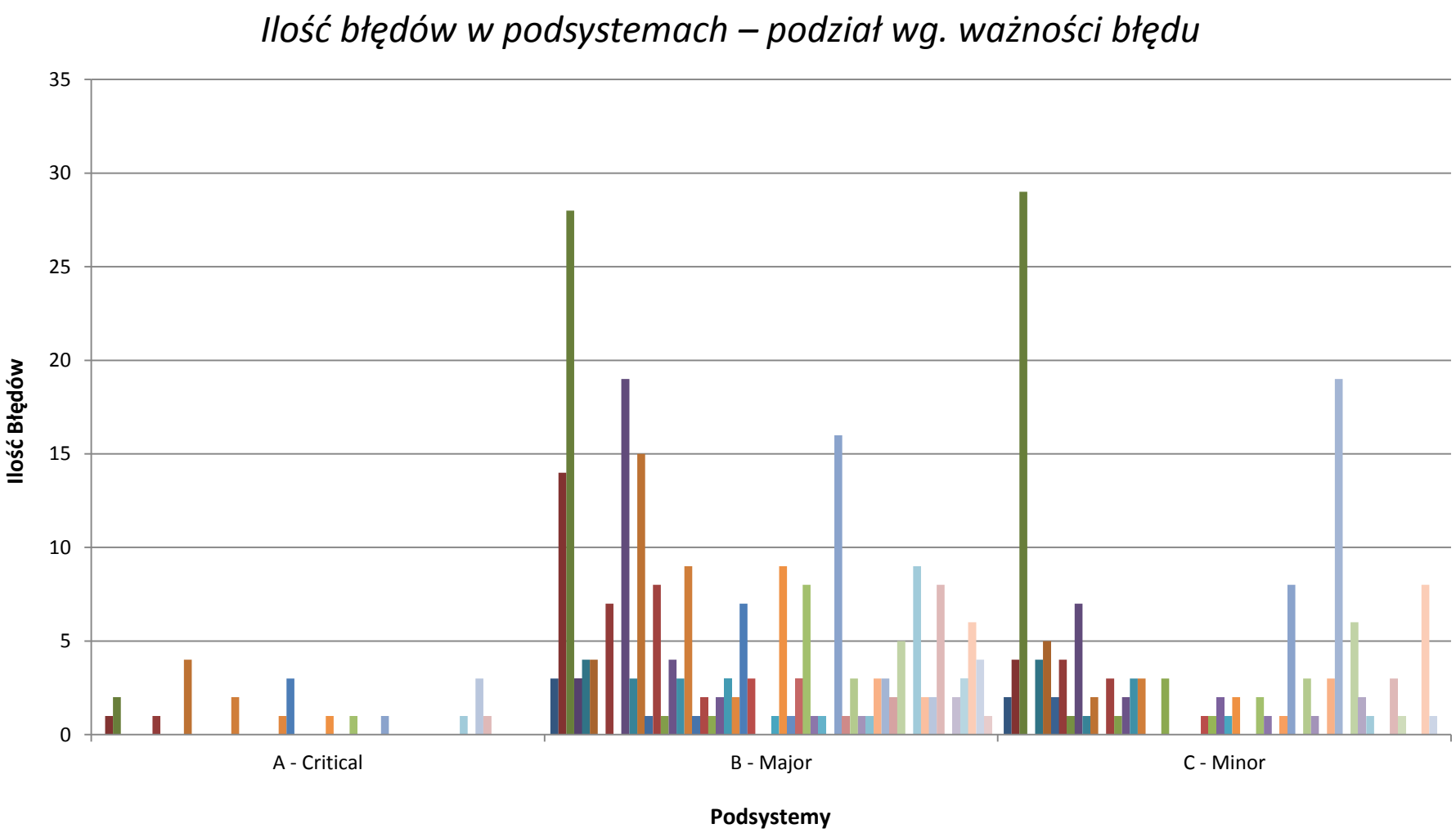
Opis metody – Faza 1

Faza 1 – Analiza i Optymalizacja zestawu testów regresywnych

- Identyfikacja funkcjonalności sytemu metodą (*White-box, Statycznie)*
- Ocena pokrycia systemu testami *(Veryfikacja testów)*
- Ocena rozkładu błędów w testowanym systemie
- Identyfikacja krytycznych obszarów systemu
- Ocena Efektywności testów (*ang. Defect Detection Percentage)*
- Obliczenie Miary Błędu *(ang. Defect Measure)*
- Dobór zestawu testów w oparciu o:
  - Testy w oparciu o ryzyko *(Risk-Based Testing)*
  - Testy w oparciu o korzyści klienta *(Benefit-Based Testing)*
  - Zmienną granulację przypadków testowych

Faza 1 - Rozkład błędów (1/2)

Faza 1 - Rozkład błędów (2/2)

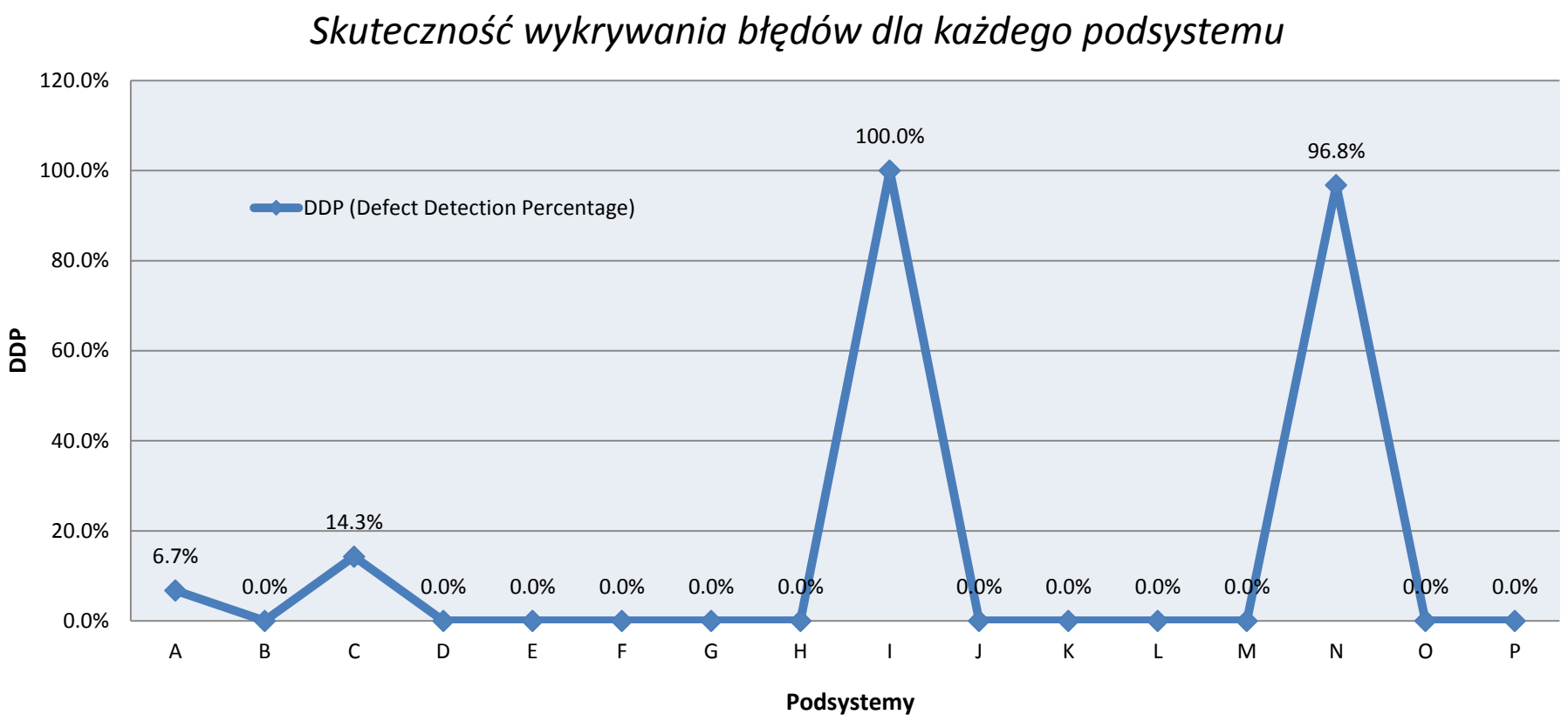


Faza 1 - Efektywność testowania

- Efekt pestycydów
- Efektywność testów regresywnych, jako skuteczność wykrywania błędów
- Skuteczność wykrywania błędów *(ang. DDP – Defect Detection Percentage)*

$$DDP = \frac{\text{Ilość błędów wykrytych przez nasze testy}}{\text{Ilość wszystkich wykrytych błędów}} \text{ x } 100\ \%$$

Faza 1 - Efektywność testowania



Faza 1 - Miara Błędu (1/2)

- Wiedza o ilości i rozkładzie błędow jest niewystarczająca
- Ryzyko oraz koszt wystąpienia błędu łatwiej jest oszacować wprowadzając Miarę Błędu
- Miara błędu *(ang. DM – Defect Measure)*

  Wagi błędu w zależności od ważności błędu, np.:

  Krytyczny = 10

  Średni = 5

  Niski = 1

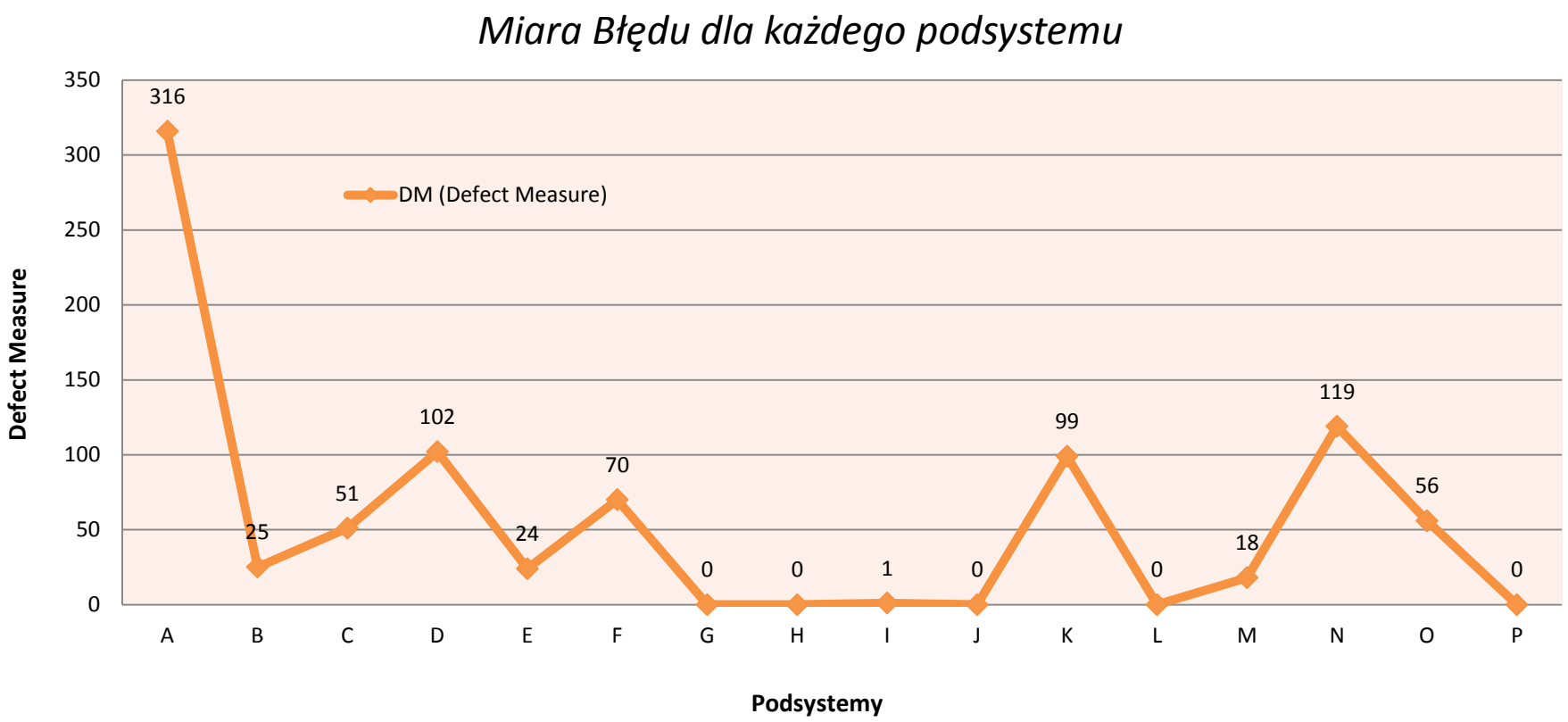
DM = 10*H + 5*M + L

H – ilość błędów krytycznych

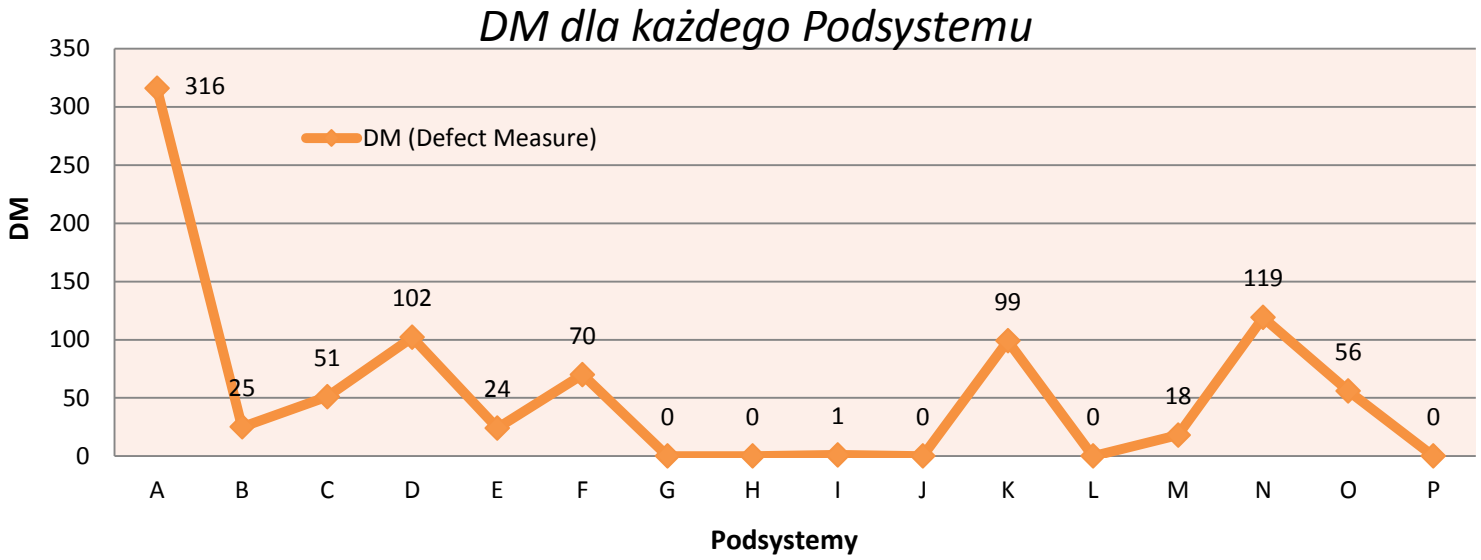
M – ilość błędów o średniej ważności

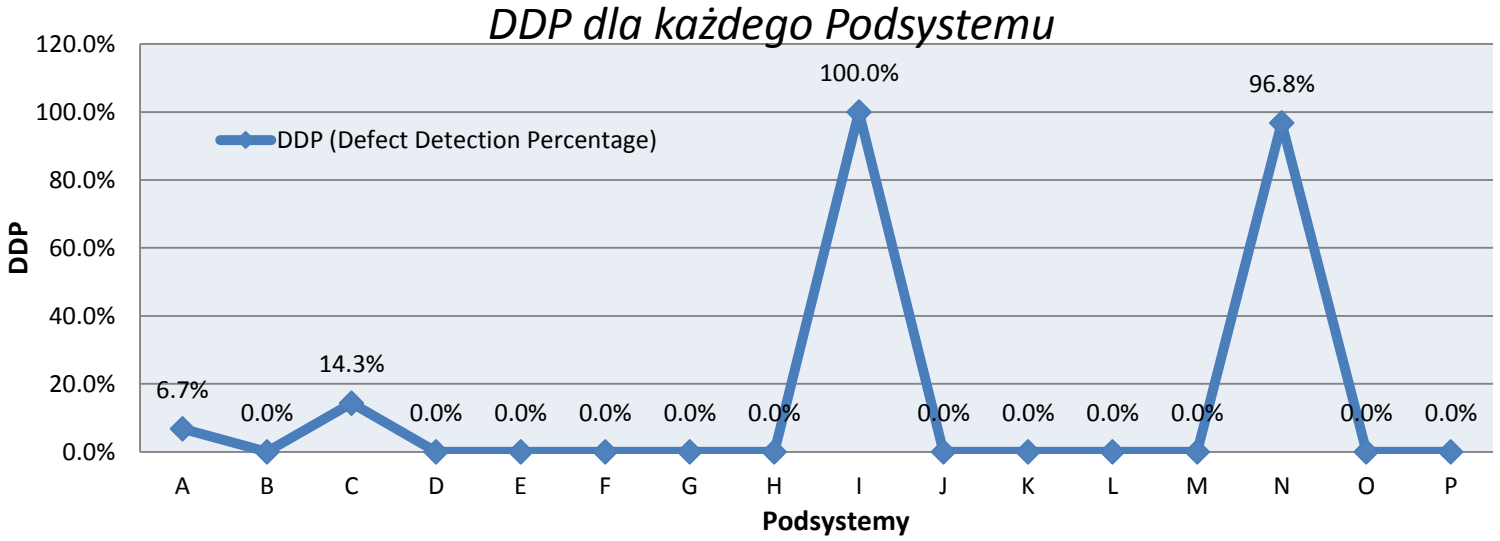
L – ilość błędów o niskiej ważności

Faza 1 - Miara Błędu (2/2)



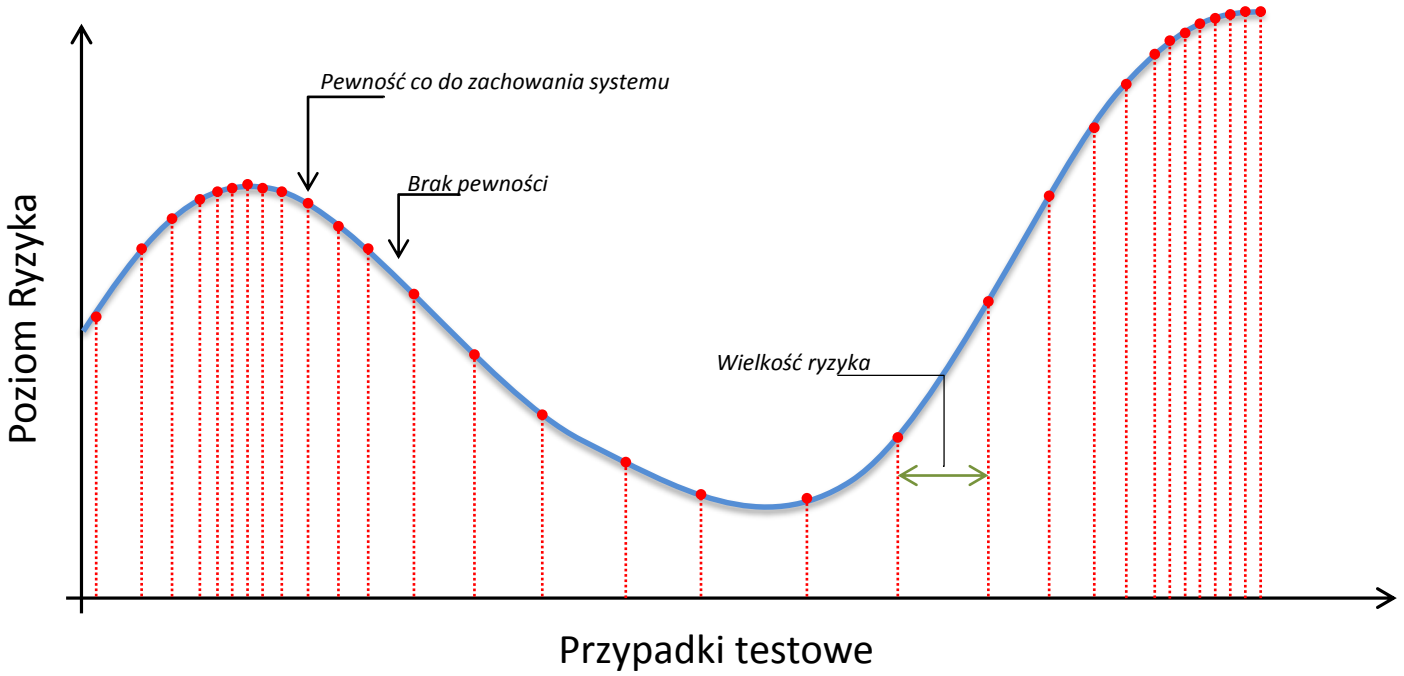
DDP - a - DM


DM dla każdego Podsystemu
DM (Defect Measure)
DM
Podsystemy
350
300
250
200
150
100
50
0
316
25
51
102
24
70
0
0
1
0
99
0
18
119
56
0
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P


DDP dla każdego Podsystemu
DDP (Defect Detection Percentage)
DDP
Podsystemy
120.0%
100.0%
80.0%
60.0%
40.0%
20.0%
0.0%
6.7%
0.0%
14.3%
0.0%
0.0%
0.0%
0.0%
0.0%
100.0%
0.0%
0.0%
0.0%
0.0%
96.8%
0.0%
0.0%
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P

Δ Granulacja przypadków testowych

Testy jako analogia do próbkowania sygnałów analogowych.



Rozmawiając z udziałowcami lub menadżerami o testowaniu warto przedstawiać informację w formie ryzyka związanego z daną decyzją.

Odkrycia

- Skuteczność wykrywania błędów = 25%
- 25% defektów było wykryte przez 6% z 1000 przypadków testowych
- Pozostałe 94% z 1000 przypadków testowych nigdy nie wykryło żadnego błędu – co nie oznacza, że tych błędów tam nie było
- Czas potrzebny na wykonanie pojedynczej kampanii regresji > 31h
- 40% tego czasu zajmowały testy tylko 1-go podsystemu
- W tym czasie znaleziono tylko 7% błędów dotyczącego tego podsystemu
- Regresja wykrywała błędy tylko w 4 z kilkunastu podsystemów
- Brak kontroli wersji testów regresywnych (brak powtarzalności procesu)
- Część funkcjonalności w ogóle nie była testowana
- Część funkcjonalności interfejsu posiadało fałszywą implementację lub jej brak

Opis metody – Fazy 2, 3 i 4

Faza 2 – Optymalizacja techniczna

- Automatyzacja brakujących przypadków testowych
- Optymalizacja skryptów testowych
- Usprawnienia narzędzi testowych
- Zrównoleglenie wykonania testów

Faza 3 – Monitorowanie i reagowanie (minimalizacja ryzyka)

- Okres pilotażowy dla zoptymalizowanych testów
- Zoptymalizowana regresja (dzienna)
- Stara regresja (weekendowa)
- Porównywanie wyników i skuteczności

Faza 4 – Start (wypływamy na głęboką wodę)

- Wdrożenie zoptymalizowanej regresji
- Poprzedni zestaw testów regresywnych odchodzi na zasłużoną emeryturę

Wynik Optymalizacji

- Czas regresji po optymalizacji 8h
- Zrównoleglenie skróciło czas do 3h
- Wzrost skuteczność detekcji błędów (25% → 70%)
- Wzrost wydajności testowania
  - Krótszy czas dostarczania informacji zwrotnej
  - Więcej funkcjonalności testowana w krótszym czasie
- Mniej przypadków testowych przy wyższym pokryciu systemu testami

Materiały źródłowe

1. Lee Copeland – *„A Practicioner's Guide to Software Test Design"*
2. Tom Gilb – *„Competitive Engineering"*
3. Hans Shäfer – *„Risk and Benefit Based Testing"*
4. Juile Gardiner – *„Testing that Serves the Project"*

Pytania i Odpowiedzi

"Go ask your search engine."

Thank You

You Made My Day