



# eXtreme Programming

## Programowanie ekstremalne

Łukasz Dzwoniarek



# XP

Metodologia wytwarzania oprogramowania starająca się skupić na jakości i czasie reakcji na uwagi klienta.

Jako metodyka zwinna kładzie nacisk na częste wydania nowych wersji realizowanego projektu.



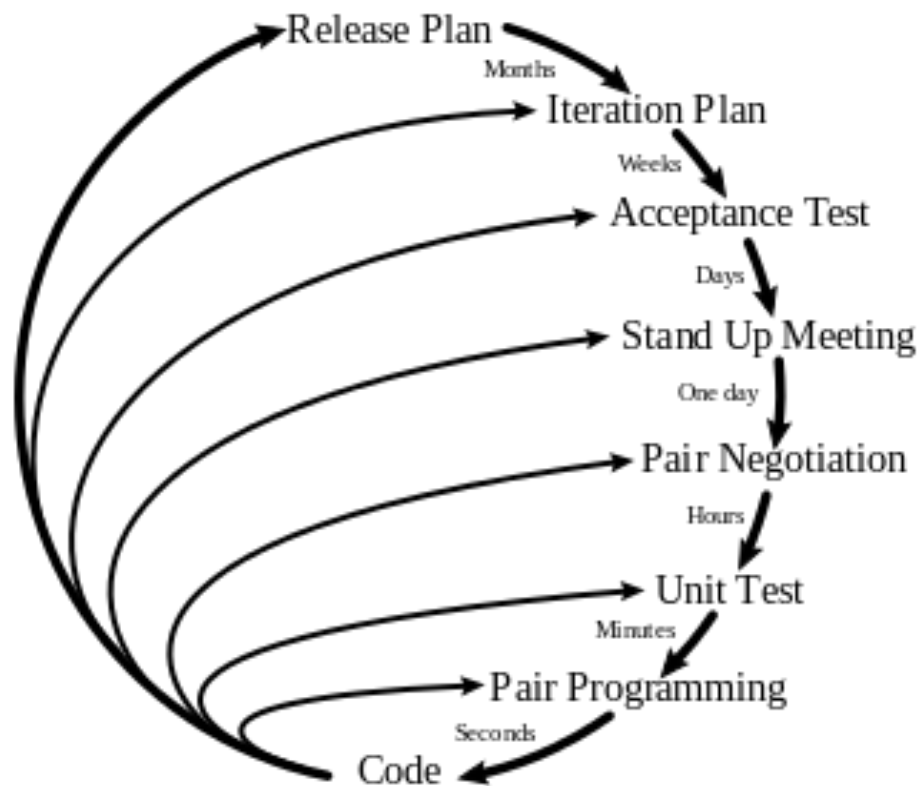
# Historia

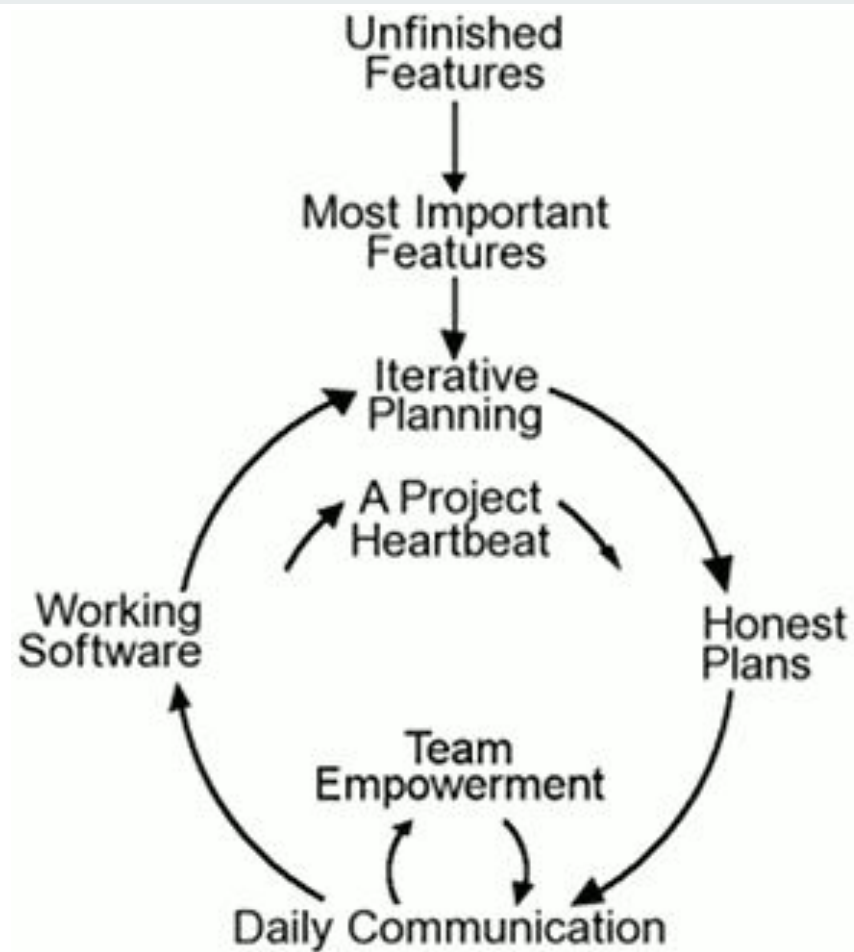
Stworzona przez Kenta Becka, gdy pracował dla Chryslera w marcu 1996, w ramach projektu Chrysler Comprehensive Compensation System.

Została opisana, w książce Extreme Programming Explained, wydanej we wrześniu 1999 roku.

Wiele z technik było już znanych wcześniej np. podczas realizacji programu Mercury przez NASA w 1960, wymagania dla oprogramowania były tworzone równolegle z nim i finalizowane odrobinę przed zakończeniem wytwarzania kodu.

## Planning/Feedback Loops







# Źródła sukcesu

Duży nacisk jest położony na spełnienie potrzeb klienta.

Z powodu krótkich cykli wydawniczych możliwe jest skupienie się na najbardziej naglących potrzebach klienta.

Lepsza motywacja zespołu, który zajmuje się bieżącymi problemami, zamiast myśleć o odległej dacie dostarczenia całego projektu.



# Źródła sukcesu

XP kładzie duży nacisk na pracę zespołową.

W przypadku bardziej klasycznych metod wytwarzania oprogramowanie często dochodzi do rozwarstwienia: klient, menedżerowie, programiści. W przypadku krótkich cykli i małych usprawnień w praktyce jest dużo mniej możliwości na powstawanie takich podziałów.



# 5 usprawnień

komunikacja

uproszczenie

szybkie sprzężenie zwrotne

szacunek

odwaga



# Reguły XP

---



# Planowanie

spisanie historii użytkownika

planowanie wydań (terminarz)

częste wydania

podział projektu na iteracje

# Zarządzaj celami, nie aktywnościami

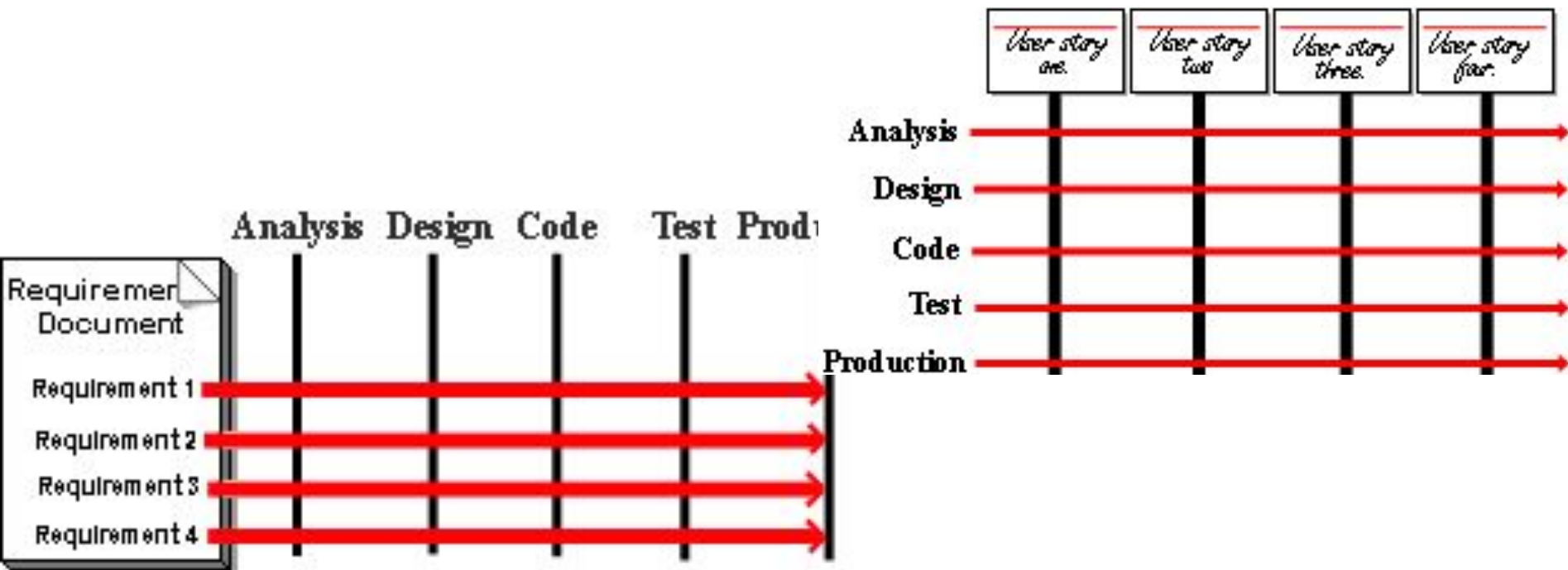
January 2010

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
					1	2
3	4	5	6 <i>Analysis</i>	7	8	9
10	11	12	13 <i>Design</i>	14	15	16
17	18	19	20 <i>Code it</i>	21	22	23
24	25	26 <i>Test it</i>	27	28	29 <i>Demo!</i>	30
31						

January 2010

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
					1	2
3	4 <i>Planning meeting</i>	5 <i>Most important feature</i>	6	7	8 <i>Demo!</i>	9
10	11 <i>Planning meeting</i>	12 <i>Next most important feature</i>	13	14	15 <i>Demo!</i>	16
17	18 <i>Planning meeting</i>	19 <i>Next most important feature</i>	20	21	22 <i>Demo!</i>	23
24	25 <i>Planning meeting</i>	26 <i>Least important feature</i>	27	28	29 <i>Demo!</i>	30
31						

# Zarządzaj celami, nie aktywnościami



## Planowanie cd.





# Zarządzanie

udostępnienie zespołowi odpowiedniego miejsca

ustanowienie racjonalnego tempa

organizacja stand upów każdego dnia

pomiar prędkości projektu

przemieszczanie osób w zespole

jeśli coś nie działa: NAPRAW



# Projektowanie

prostota jest kluczem

utrzymuj rzeczy prostymi najdłużej jak to możliwe

wybór prostych metafor

Class, Responsibilities, and Collaboration (CRC)

spike solutions

nigdy nie dodawaj funkcjonalności wcześniej

ciągła refaktoryzacja



# Prostota jest kluczem

Raz i tylko raz

```
convertToPercent(aFraction)
```

```
convertMetersToCentimeters(aLength)
```





# Prostota jest kluczem

Raz i tylko raz

```
convertToPercent(aFraction) { return aFraction*100; }
```

```
convertMetersToCentimeters(aLength) { return aLength*100; }
```



# Prostota jest kluczem

Raz i tylko raz

```
convertToPercent(aFraction) { return aFraction*100; }
```

```
convertMetersToCentimeters(aLength) { return aLength*100; }
```

```
convertToPercentOrCentimeters(x){ return x*100; }
```



# Kodowanie

Klient jest zawsze dostępny

Kod musi utrzymywać określony standard

Najpierw pisz testy jednostkowe

Cały kod produkcyjny pochodzi z programowania parami

Tylko jedna para może integrować kod w danym momencie

Zbiorowa własność kodu



# Testowanie

Cały kod musi być pokryty testami jednostkowymi

Wszystkie testy jednostkowe muszą przejść przed wydaniem

Gdy zostaje odkryty błąd powstaje odpowiadający mu test

Utrzymywanie i wykonywanie testów akceptacyjnych



# Kiedy powinno się używać XP

Jeśli wymagania ciągle się zmieniają

Nowy typ projektu, który musimy dostarczyć na określony termin

Małe grupy 2-12 osób (Max 30)



# Jak zacząć przygodę z XP

Dobrze jest zacząć z nowym projektem

W praktyce często przechodzi się na XP, gdy z projektem pojawiają się problemy

Kończenie małych fragmentów pomogą przy zmieniających się wymaganiach

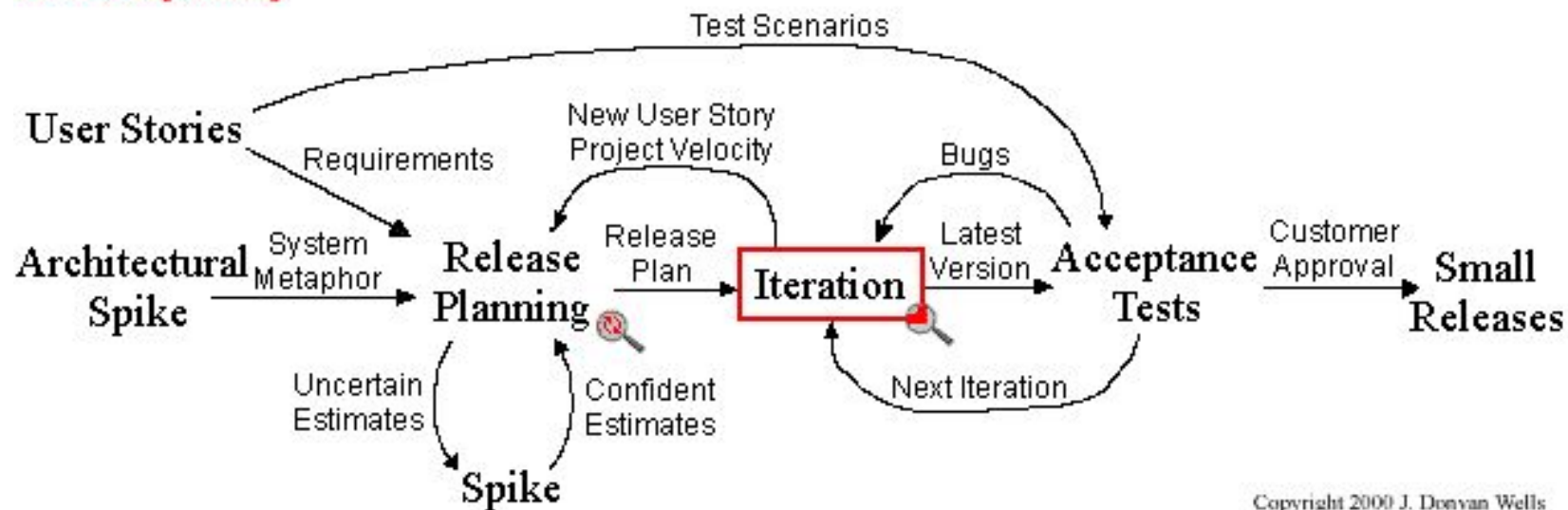
Programowanie parami poprawia jakość kodu i rozeznanie w projekcie programistów

# Paradoks procesu






# Extreme Programming Project







WE'RE GOING TO  
TRY SOMETHING  
CALLED AGILE  
PROGRAMMING.

THAT MEANS NO MORE  
PLANNING AND NO MORE  
DOCUMENTATION. JUST  
START WRITING CODE  
AND COMPLAINING.

I'M GLAD  
IT HAS A  
NAME.

THAT  
WAS YOUR  
TRAINING.



## C3 porażką ?

- projekt rozpoczęto w 1996
- pierwsze wydanie w 1997
- planowane pełne wydanie w 1999
- projekt przerwany w 2000
  - syndrom 90%



# Bibliografia

<http://www.extremeprogramming.org/>

<https://www.martinfowler.com/bliki/C3.html>

<http://wiki.c2.com/?ChryslerComprehensiveCompensation>

<http://c2.com/ppr/>