



## Dwa zasadnicze pytania

- ❑ Czy wytwarzacie innowacyjne produkty dla swoich klientów ?
- ❑ Czy praca w projekcie jest dla was ekscytująca ?



# Co to jest konstrukcja oprogramowania?

„Pomyśl tylko. Kiedy programujesz, to wciąż projektujesz. Twoim zespołem konstrukcyjnym jest kompilator, linker i interpreter. Kod źródłowy jest tylko pośrednim produktem projektowania!”

*Jack Reeves - informatyk*

Stąd wynikają następujące wnioski:

- W oprogramowaniu konstrukcja jest tak tania, że można uznać ją za darmową.
- Cały wysiłek idzie w projektowanie, więc wymaga utalentowanych i kreatywnych ludzi.
- Proces twórczy trudno zaplanować, więc przewidywalność procesu projektowania jest wątpliwa.
- Budowanie oprogramowania jest prawdopodobnie innym rodzajem działalności niż zwykła inżynieria.

## Agile Software Development - historia

- ❑ Reakcja na popularność „ciężkich” metodyk lat 90-ych
- ❑ Scrum – 1986
- ❑ DSDM, FDD, ASD – 1995
- ❑ Crystal Clear, Extreme Programming – 1996

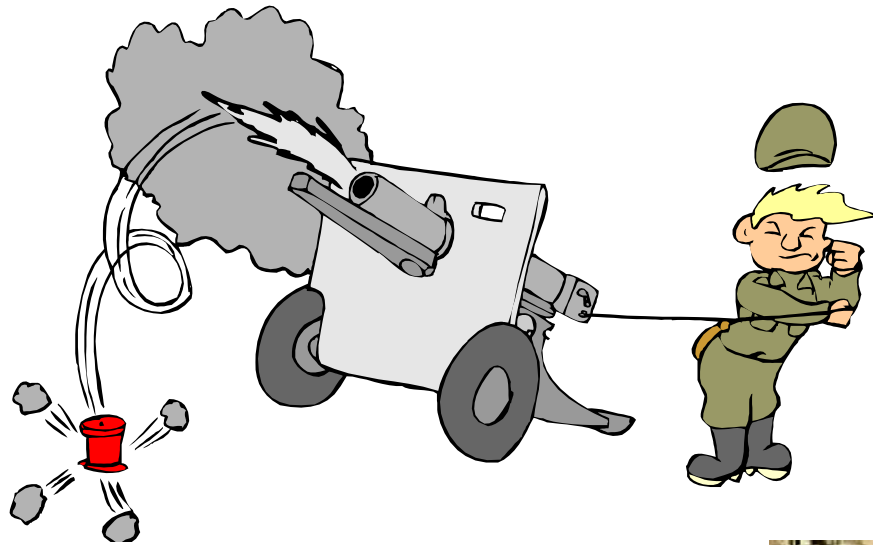
Wiele koncepcji dużo wcześniejszych, np. Toyota Production System (TPS) i wywodzący się z niego „Lean Manufacturing” pochodzi z lat 50ych

2001 – uformowanie się Agile Alliance

# Planowanie w projekcie adaptacyjnym



# Przewidywanie vs. Osiąganie Celu

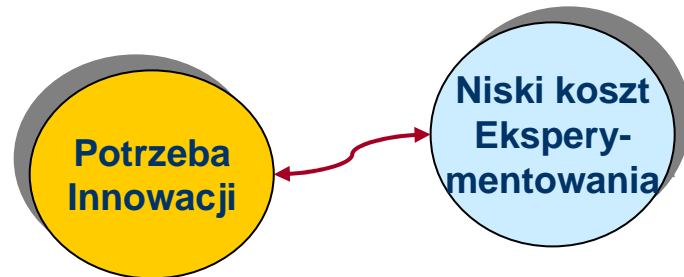


**“W ekstremalnym środowisku postępowanie zgodnie z planem powoduje wytworzenie założonego wcześniej produktu, zamiast produktu, którego się aktualnie potrzebuje.”**

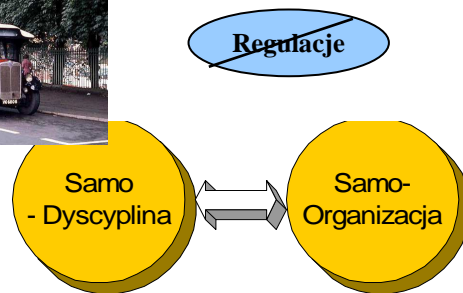


# Istota APM

## Motywatory Biznesowe



## Znajdź właściwych ludzi

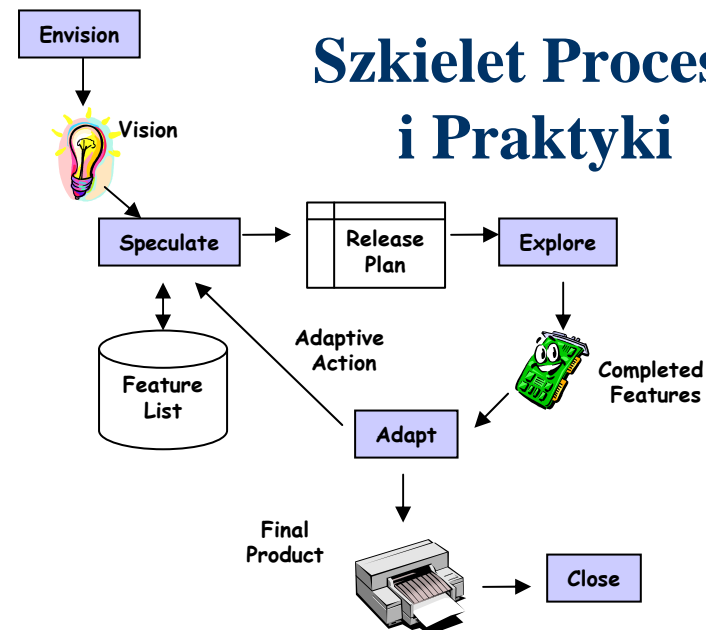


## Podstawowe Wartości i Zasady

The Agile Manifesto

Individuals & Interactions  
 Working Software (Products)  
 Customer Collaboration  
 Responding to Change

## Szkielet Procesu i Praktyki



## Podstawowe wartości (1 z 4)

- Przedkładamy ludzi i interakcje pomiędzy nimi nad procesy i narzędzia
  - Ludzie nie są trybami maszyny
  - Innowacyjność wymaga nieskrępowania

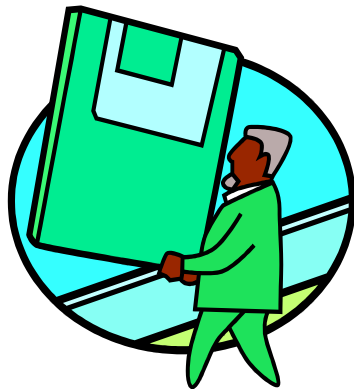


vs.



## Podstawowe wartości (2 z 4)

- Przedkładamy działający produkt nad rozbudowaną dokumentacją
  - Jedyny, uczciwy miernik
  - Dokumenty mogą kłamać ...



vs.



## Podstawowe wartości (3 z 4)

- Przedkładamy kooperację z klientem nad sztywne trzymanie się kontraktu
  - „My” zamiast „My i Oni”
  - Społeczność, wspólne podejmowanie decyzji, szybkość i jakość komunikacji



vs.

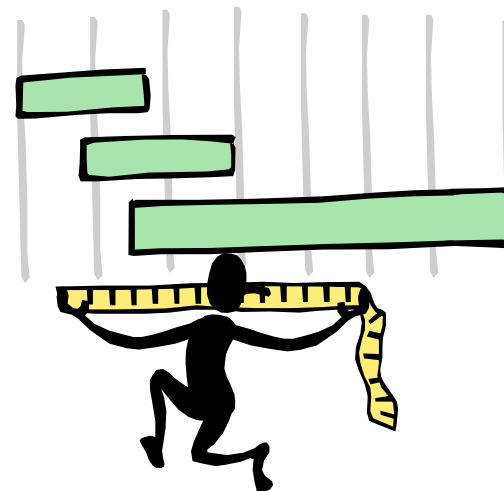


## Podstawowe wartości (4 z 4)

- Przedkładamy umiejętność dostosowywania się do zmian nad „kurczowe” trzymanie się planu
  - Nie negujemy budowania planu
  - Plan vs Spekulacja ...



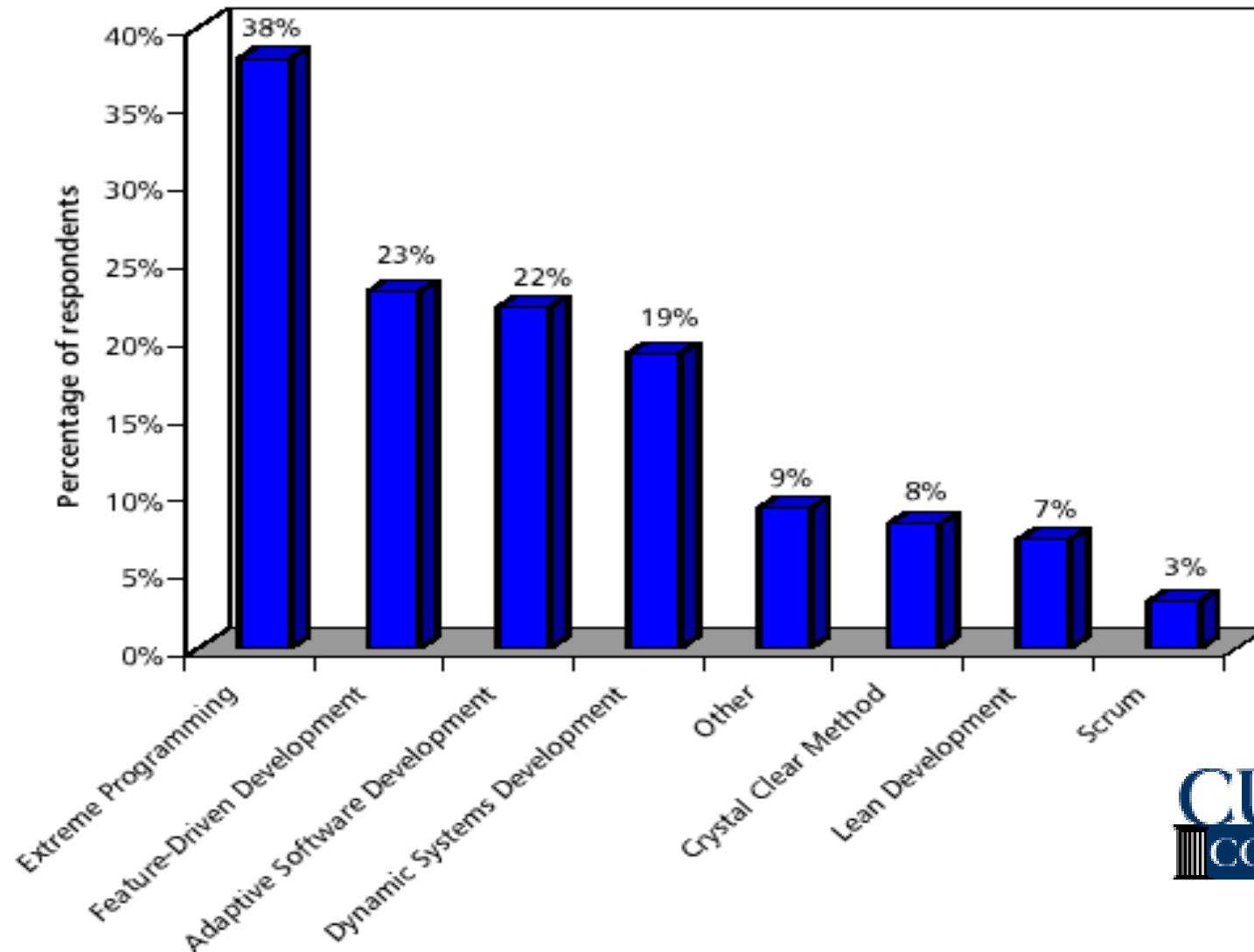
VS.



# Porównanie tradycyjnych i adaptacyjnych praktyk

<b>Adaptacyjne</b>	<b>Tradycyjne</b>
Zorientowane na dostarczanie funkcjonalności	Zorientowanie na podział zadań
Plany są hipotezą, nie przewidywaniem	Plany są przewidywaniem odnośnie przyszłości
Sukces rozumiany jako zdolność adaptacji do zmieniających się warunków w projekcie	Sukces rozumiany jako zgodność z wcześniej założonym planem
Duża precyzja planu dla wczesnych iteracji, bardzo zgrubny charakter planu w dalszej fazie projektu	Szczegółowy plan opracowany jest dla całego projektu
Przyczyny odchylenia od planu są analizowane i dostarczają informacji do zmiany planu kolejnych faz projektu (adaptive action)	Odchylenia od planu są traktowane jako błędy zarządzania i wymagają bezkrytycznej poprawy (corrective action)
Zarządzanie zmianą jest motorem dla procesów innowacyjnych	Zarządzanie zmianą często degeneruje się do biurokratycznych procedur blokujących zmianę
Zorientowane na stworzenie samo-organizującego się, samo-dyscyplinującego się zespołu projektowego	Zorientowane na procedury i techniki kontroli oraz mikrozarządzanie zadaniami projektowymi

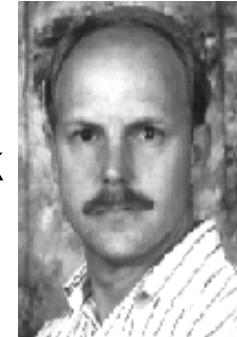
## Metodyki zaliczane do rodziny metodyk „agile”



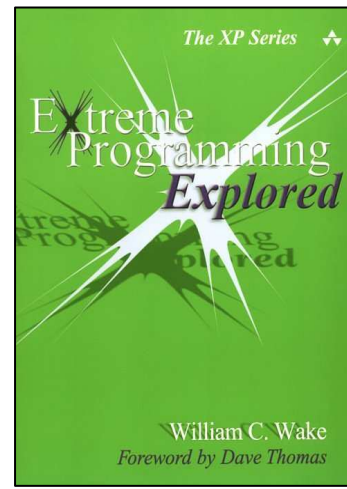
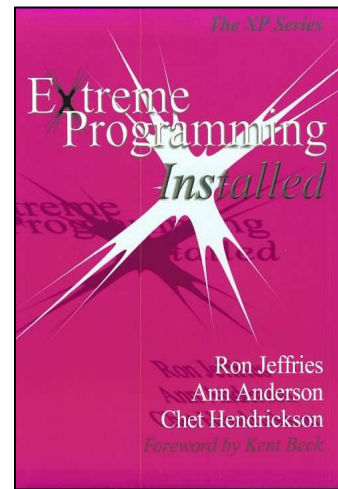
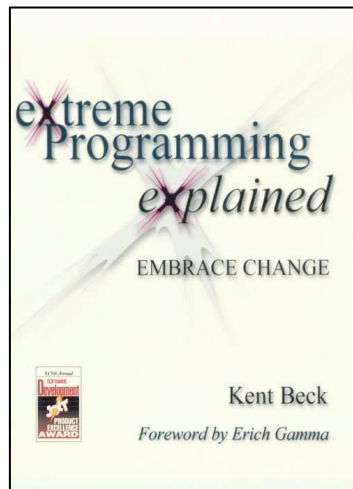
# eXtreme Programming - XP

- ❑ <http://www.extremeprogramming.org>
- ❑ <http://xprogramming.com> - Ron Jeffries
- ❑ <http://xp123.com/> - William Wake
- ❑ <http://www.exoftware.com/>
- ❑ ... i wiele innych ...

**Kent Beck**



**Ron Jeffries**



# Feature Driven Development

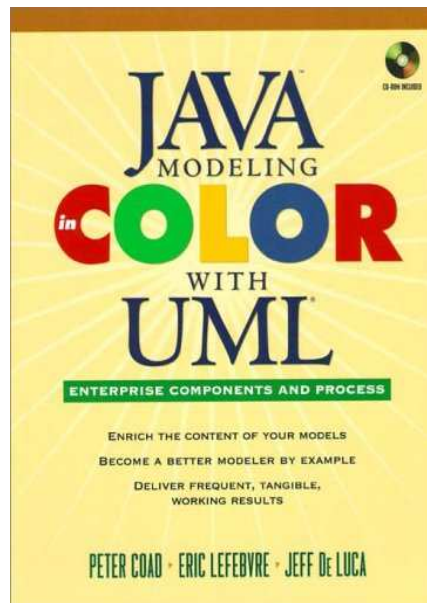
- ❑ [www.nebulon.com](http://www.nebulon.com) – Jeff de Luca
- ❑ [www.togethersoft.com](http://www.togethersoft.com) – Peter Coad
- ❑ [www.featuredrivendevelopment.com](http://www.featuredrivendevelopment.com)



**Jeff de  
Luca**



**Peter  
Coad**



Październik 2002 –  
TogetherSoft kupione  
przez Borlanda

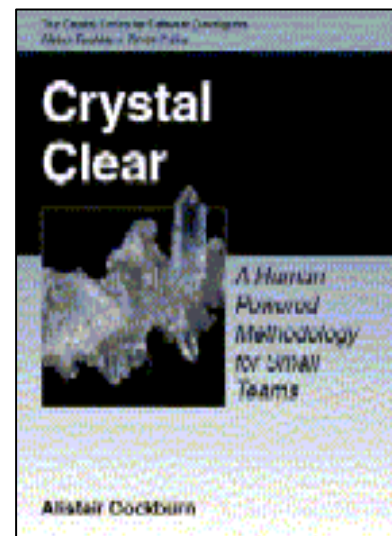
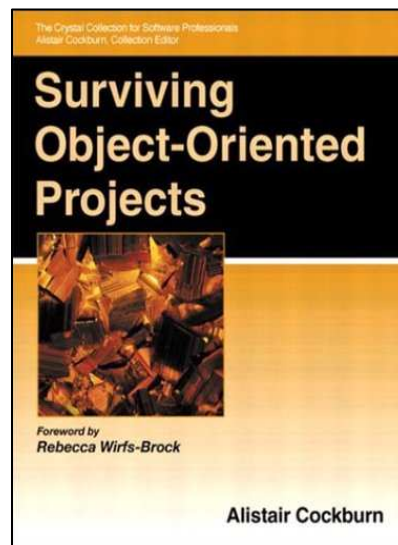
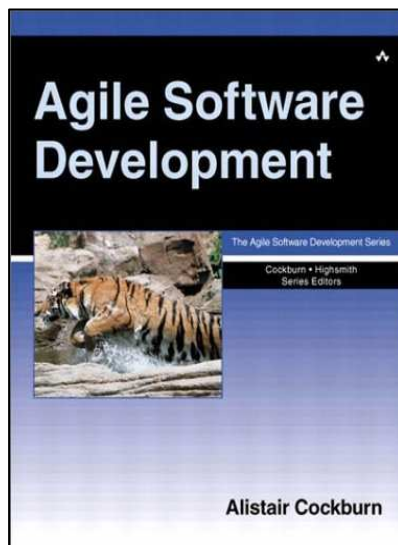
# Crystal Family

- <http://www.crystallmethodologies.org/>
- <http://members.aol.com/acockburn/>

- **Crystal Family**
- **Filozofia „Methodology-per-Project”**



**Alistair  
Cockburn**



## Wstępne wyniki badań

- ❑ Zmniejszenie czasu wytworzenia produktu od 25 do 50%\*\*
- ❑ Zmniejszenie o rząd wielkości liczby błędów \*
- ❑ Wzrost produktywności od 15 do 23%\*\*
- ❑ 40% redukcja kosztów i czasu\*\*\*
- ❑ Lepsza przewidywalność projektu \*

\* Stay, Russell. XP & Enterprise Software Development, Agile workshop, March 2002.

\*\* Reifer, Donald. "How Good Are Agile Methods?" *IEEE Software*. July/August 2002.

\*\*\* Charette, Bob. Eurotel study reported in *Agile Software Development Ecosystems*, Jim Highsmith, Addison Wesley 2002.

## Podstawowe wartości (1 z 4)

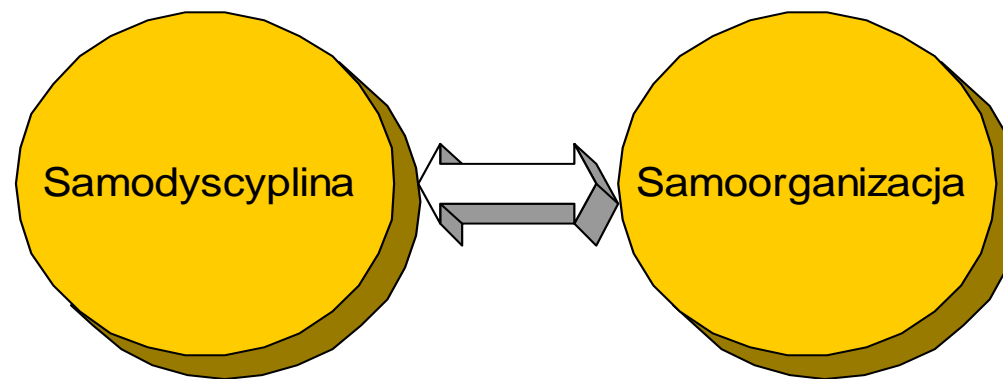
- Przedkładamy ludzi i interakcje pomiędzy nimi nad procesy i narzędzia
  - Ludzie nie są trybami maszyny
  - Innowacyjność wymaga nieskrępowania



vs.



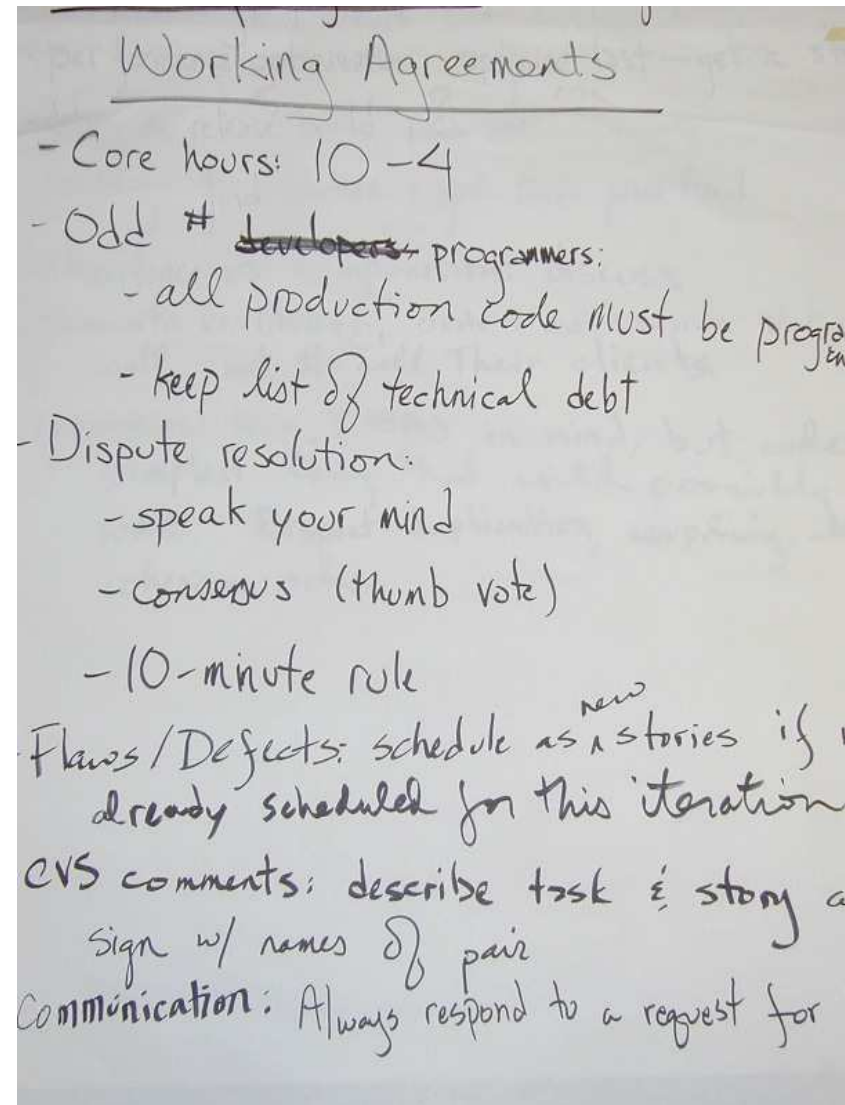
## Budowanie adaptacyjnych zespołów



Równoważenie swobody i odpowiedzialności

## Zasady Wspólnej Pracy

- ❑ Miej respekt dla innych
- ❑ Dbaj o otwartą komunikację
- ❑ Współpracuj w zespole
- ❑ Rozwiązuj konflikty w zespole
- ❑ Wywiązuj się ze zobowiązań
- ❑ Nie pozwalaj innym łamać zasad
- ❑ Toleruj błędy jako element uczenia się





# Wspólne wytwarzanie

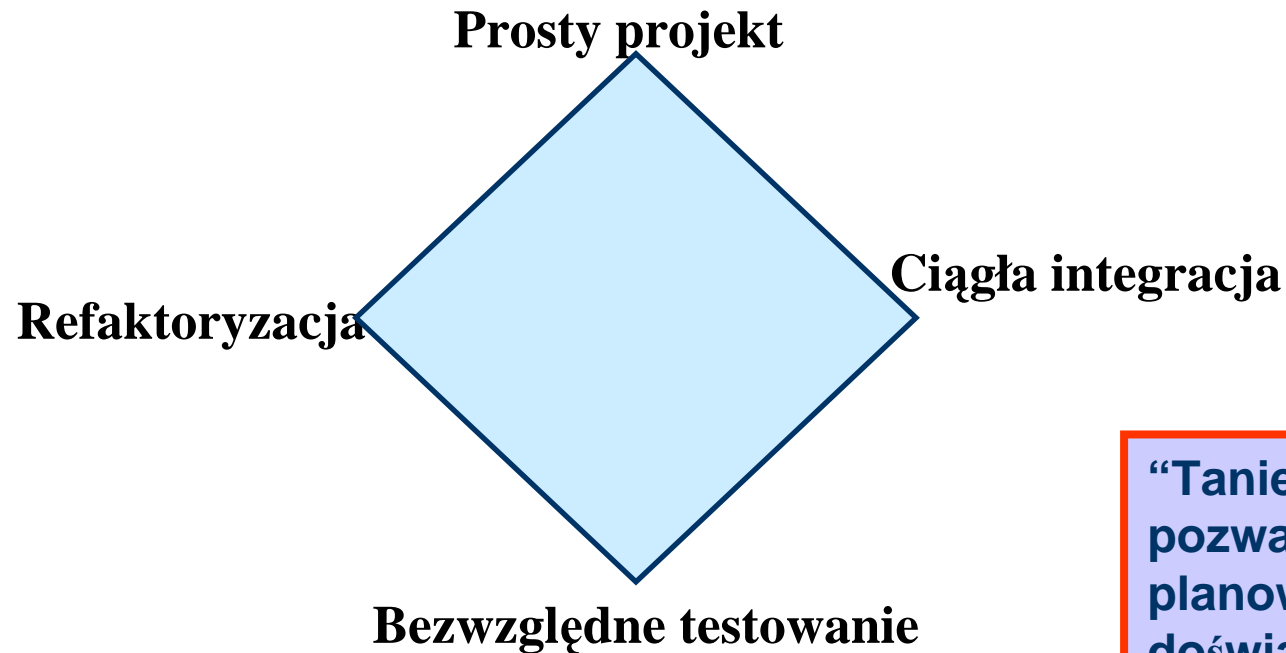




## Bliski kontakt, nawet na odległość



## Praktyka: Iteracje o niskim koszcie

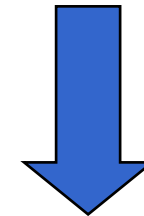
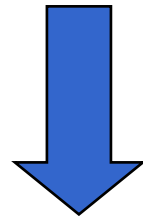


**“Tanie i szybkie iteracje pozwalają zastąpić planowanie doświadczeniem.”**

Rob Austin i Lee Devin  
*Artful Making*

## Zarządzanie oparte na modelu Przywództwo-Współpraca

**Polecenie-Kontrola**



**Przywództwo-Współpraca**

- n Jestem szefem i mówię ludziom co mają robić
- n Jestem tutaj po to żeby pomóc

## Zadania kierownika projektu

- ❑ Skoncentruj zespół na dostarczeniu wyników
- ❑ Przekształć grupę osób w zespół
- ❑ Zwiększ udział poszczególnych osób
- ❑ Zapewnij zespołowi odpowiednie zasoby i usuń przeszkody
- ❑ Przygotuj klientów
- ❑ Utrzymuj właściwy rytm pracy zespołu

## Typowy Projekt ? . . .

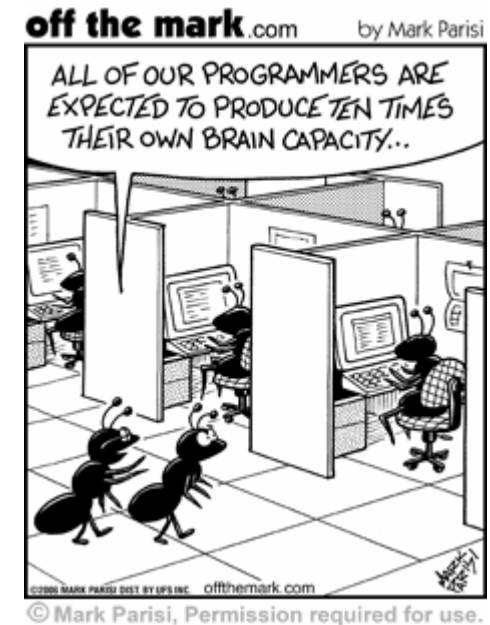
- Duży entuzjazm
- Rozczarowanie
- Panika
- Chaos
- Szukanie winnego
- Ukaranie niewinnego
- Promocja osób nie uczestniczących w projekcie
- Zdefiniowanie celów biznesowych projektu

## **A może w ten sposób...**

- ❑ Wspólna wizja – nie zaczynamy bez celów biznesowych
- ❑ Zaangażowanie całego zespołu, współodpowiedzialność
- ❑ Kierownik po to by nam pomagać
- ❑ Krótkie iteracje – każda zaczyna się od wspólnego planowania, każda kończy się wspólną retrospektywą
- ❑ Codzienne, 15-minutowe spotkania zespołu
- ❑ Stałe tempo prac
- ❑ Bezwzględna walka z niepotrzebną biurokracją

## Prawa Programisty

- ❑ Programista ma prawo do własnych szacunków pracochołności a szacunki te muszą być respektowane przez resztę zespołu
- ❑ Programista ma prawo do uczciwego raportowania postępów prac
- ❑ Programista ma prawo do wytwarzania produktów o wysokiej jakości
- ❑ Programista ma prawo wiedzieć jaka funkcjonalność produktu jest najistotniejsza dla klienta
- ❑ Programista ma prawo w dowolnym momencie zadawać pytania związane z kontekstem biznesowym produktu



***Kent Beck – Prawa Programisty w XP***

## Najważniejsi są ludzie



# Male Googleplexy...



## Czuję się przekonany. Co dalej ?



[www.bms.com.pl](http://www.bms.com.pl)

[bkiepuszewski@bms.com.pl](mailto:bkiepuszewski@bms.com.pl)

