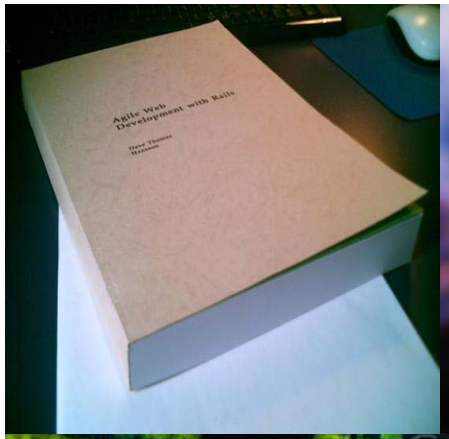
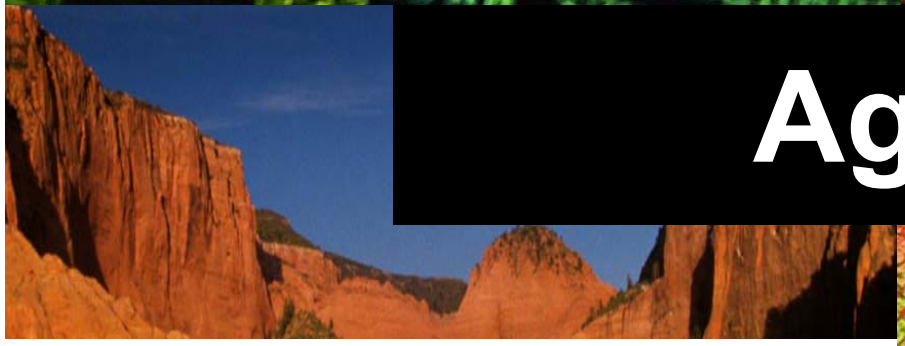
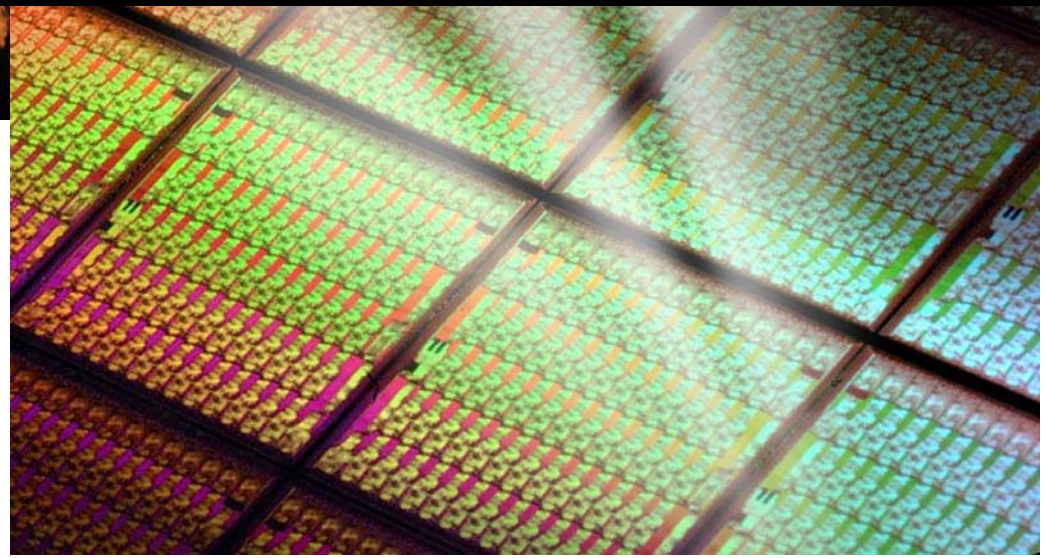


Agile Development





Marcin Kucięba
marcin.kucieba@sabre.com

Sabre Holdings™

Agile Development

- Dotychczasowe podejście
- Konieczność zmian
- Agile Manifest
- Praktyki Agile
- Dlaczego Agile?
- Agile resources & books

Sabre Holdings

Software development dotychczas

- Waterfall – tradycyjne podejście w procesie wytwarzania oprogramowania

Sekwencyjność - brak możliwości zmiany wcześniejszych decyzji !

Sabre Holdings

Konieczność zmian

- Brak efektywności dotychczasowego podejścia
- Wysoki współczynnik projektów zakończonych porażką

  Według raportu CHAOS w 1998 roku:
  26% projektów zakończonych zostało z sukcesem
  28% zakończonych zostało porażką
  46% projektów przekroczyło budżet bądź planowany termin zakończenia

- Zderzenie narzuconej metodyki z „rzeczywistością projektu"
- Brak możliwości reagowania na zmiany
- Niska jakość dostarczanego oprogramowania

Sabre Holdings

Początki zmian

- Extreme Programming
- SCRUM
- DSDM
- Adaptive Software Development
- Crystal
- Feature-Driven Development
- Pragmatic Programming

Sabre Holdings

Manifest Agile

11 Lutego 2001 w Wasatach w stanie Utah 17 ludzi spotkało się aby określić wspólny mianownik nowych „procesów” związanych z wytwarzaniem oprogramowania oraz stworzyć podstawowe, wspólne dla tych „procesów” zasady software developmentu

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas

Sabre Holdings

Manifest Agile

- **Interactions with individuals** over processes and tools.
- **Creating working software** over comprehensive documentation.
- **Customer collaboration** over contract negotiation.
- **Responding to change** over following a plan

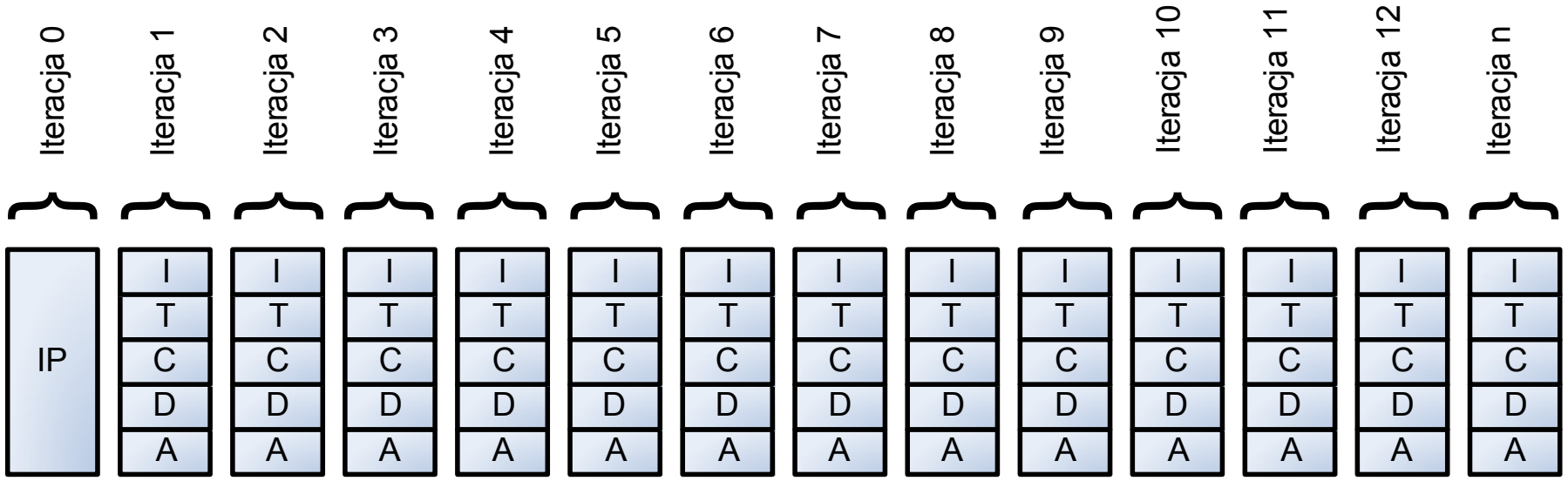
Sabre Holdings

Praktyki Agile

- Planowanie
  - Iteracje
  - User stories
  - Obecność klienta
  - Planowanie iteracji
  - Prezentacja aplikacji
  - Project velocity
  - Release planning
- Design
  - Simple design
  - YAGNI
  - Refactoring
- Kodowanie
  - Standardy kodowania
  - Test driven development
  - Continious integration
  - Wspólny kod
  - Pair programming
  - Zespół nie pracuje *overtime*
- Testowanie
  - Unit testy
  - Analiza jakości kodu
  - Testy integracyjne
  - Testy akceptacyjne
  - Automatyzacja testów

Sabre Holdings

Iteracje



IP – Initial Planning
A – Analiza
D – Design
C - Kodowanie
T– Testowanie
I– Integracja

- Zawsze niezmienna długość
- Pełny cykl zadań developerskich w każdej iteracji

Sabre Holdings

Planowanie iteracji

- Zawsze na początku każdej iteracji
- Klient decyduje co będzie realizowane w danej iteracji
- Aktywny udział wszystkich członków zespołu
- Estymacja –zadanie wyłączne developerów
- Podział zaplanowanych *user stories* na zasadach *sign up*
- Planujemy tylko tyle, ile jesteśmy w stanie zrobić w iteracji

Sabre Holdings

User stories

- Opisują zamknięty kawałek funkcjonalność
- Muszą posiadać wartość dla klienta
- Muszą mieścić się w iteracji
- Nie skupiają się na aspektach technologicznych projektu
- Są demonstrowalne
- Są szacowane w bezjednostkowych punktach
- Wymagają implementacji we wszystkich warstwach systemu

Sabre Holdings

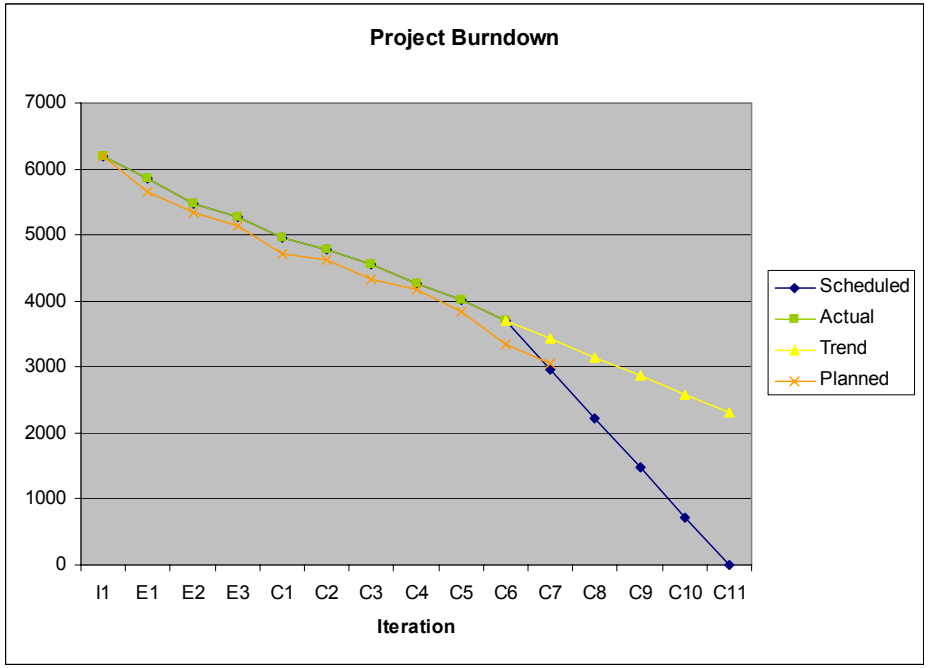
Prezentacja aplikacji

- Każdy z developerów prezentuje zaimplementowane *user stories*
- Wspólna ocena wykonania *user stories*
- Regularny *feedback*
- Możliwość śledzenia postępu prac przez klienta

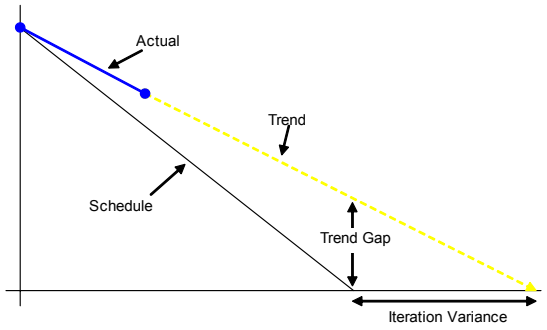
Sabre Holdings

Project velocity

- Backlog – zbiór *user stories* do wykonania
- Tracking - analiza postępu prac
- Velocity – średnia ilość zrealizowanych punktów
- Burndow chart – charakterystyka postępu prac w czasie
- Metryki – narzędzie umożliwiające predykcje



Trend Gap Last Iteration	2309
Total Deliverable Work:	6184
Gap %:	**37%**
% of Iterations Remaining:	**27%**
Trend Units of Work per Iteration:	279
Needed Units or Work per Iteration:	741
Per Iteration Variance:	**166%**
Iteration Variance	9
Iteration Variance %	**60%**



Sabre Holdings

Release planning

- Etapowe wdrażanie systemu
- Minimalizacja ryzyka związanego z uruchomieniem systemu
- Zwiększone szanse na sukces poprzez zarządzanie czynnikiem *time to market*
- *What you get is what you see* - klient wie i widzi, co system oferuje użytkownikom
- Klient decyduje co i kiedy oddać użytkownikom

Sabre Holdings

Simple design

- Nie robimy dokładnego *up front design*
- Zawsze stosuj *simple design principles*
  - High cohesion, Low copuling, Single responisbility, Open/Close, Liskov's Substitution
- Proste kod łatwiej zmienić
- Prosty kod łatwiej zrozumieć
- Prosty kod łatwiej utrzymywać
- Implementacja designu, który jest prosty zajmuje mniej czasu
- Pamiętaj o tym, że ktoś będzie używał twojego kodu
- Unikaj zbędnej komplikacji i *overdesignu*
- *YAGNI – You are not going to need this*

Sabre Holdings

Test Driven Development

- Najpierw implementujemy testy, potem klasy
- Testy definiują zakres implementacji oraz funkcjonalność klas i komponentów
- Testy wspomagają *simple design*
- Testy stanowią dokumentacje użycia klas i komponentów
- Tworzone klasy i komponenty są łatwiejsze w użyciu

Sabre Holdings

Continious Integration

- Cały zespół pracuje na wspólnym kodzie
- Projekt posiada automatyczny proces budowania aplikacji

  ant, maven
- Build integracyjny kompiluje projekt i uruchamia wszystkie testy unitowe i sprawdza jakość kodu
- Build integracyjny uruchamiany jest po oddaniu każdej zmiany
- Status buildu jest komunikowany natychmiast wszystkim członkom zespołu
- Zmiany muszą być oddawane często

Sabre Holdings

Unit testy

- Unit testy są częścią developmentu i tworzone są przez developerów
- Wszystkie testy powstają przed implementacją
- Testy umożliwiają refactoring
- Każda metoda publiczna klasy powinna posiadać test
- Unit testy powinny pokrywać jak największą ilość kodu – pokrycie testami powinno być mierzone
- Aplikacja nie może być „zreleasowana” jeżeli któreś z klas nie posiadają testów
- Unit testy strzegą zaimplementowanej funkcjonalności przed przypadkowym uszkodzeniem podczas przyszłej implementacji

Sabre Holdings

Analiza jakości kodu

- Wysoka jakość kodu jest jednym z priorytetów Agile Development
- Sprawdzanie jakości powinno być częścią continious integration
- Narzędzia wspomagające analizę jakości
  - Checkstyle (standardy kodowania)
  - Cobertura (code coverage)
  - PMD (statyczna analiza kodu)
  - JDepend (analiza zależności)

Sabre Holdings

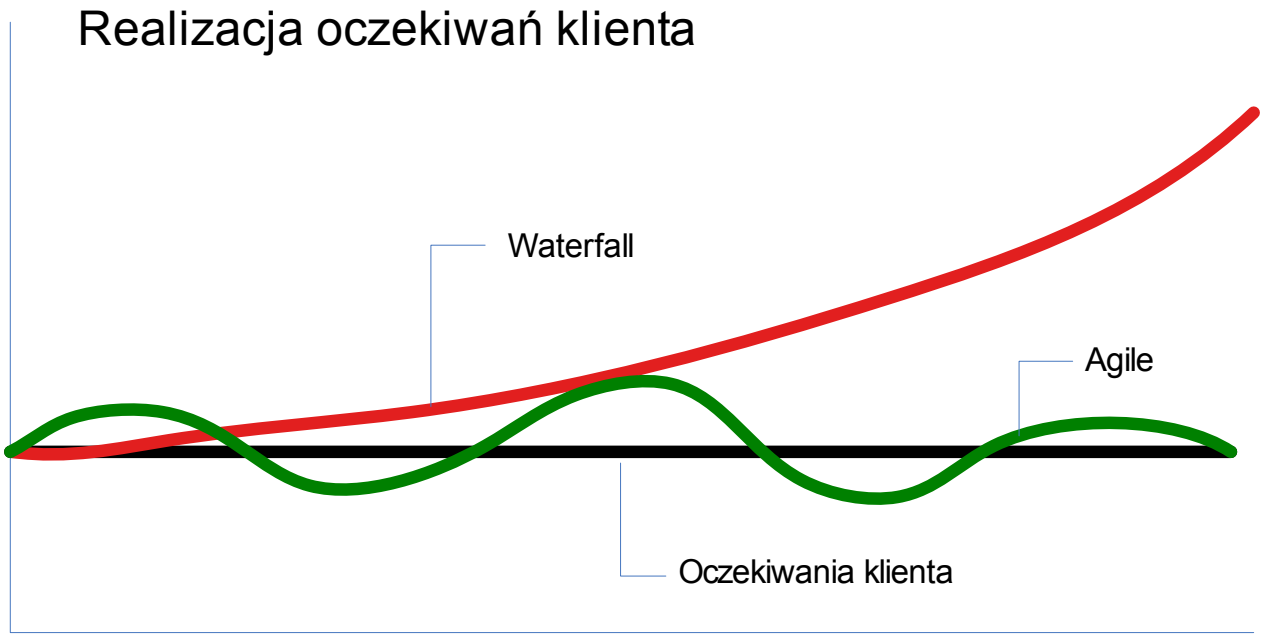
Praktyki Agile

- Stanowią narzędzia w rękach zespołu
- Wspierają podstawowe zasady wyrażone w manifeście
- Obejmują wszystkie aspekty związane z tworzonym oprogramowaniem
  - Tworzenie kodu, organizacja projektu, zarządzanie, testowanie
- Są od siebie zależne wspierając się nawzajem
- Wyznaczają dyscyplinę i organizują prace w projekcie

Sabre Holdings

Dlaczego Agile?

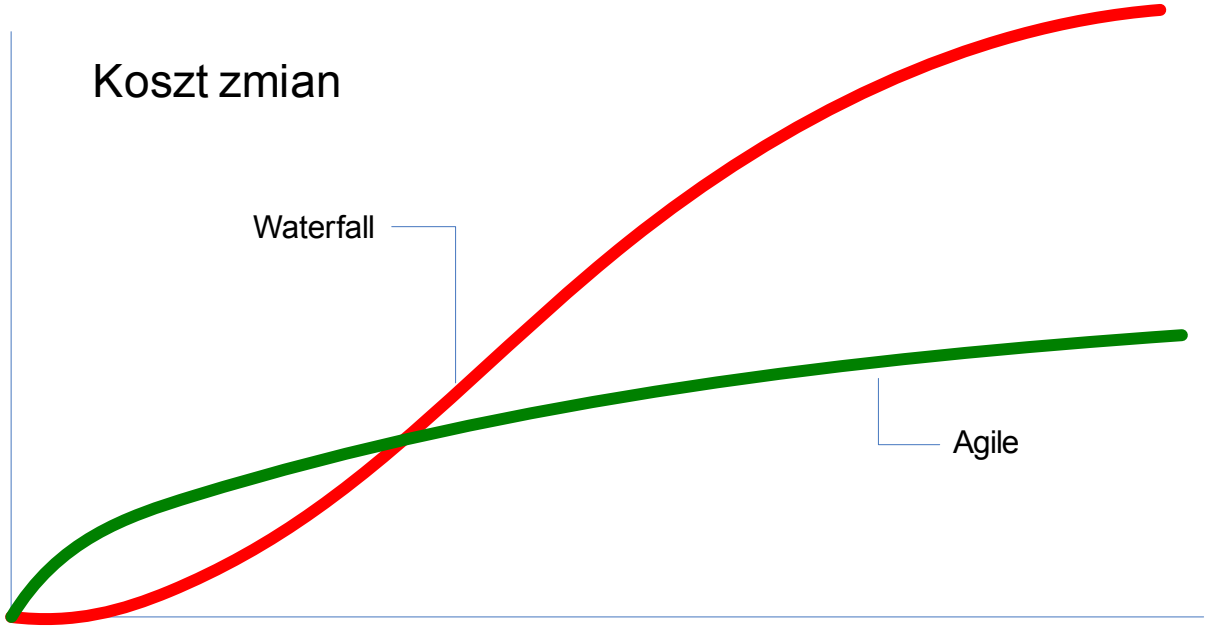
- Zapewnia większą jakość dostarczanego softwaru
- Dzięki feedbackowi klienta produkt nie rozmija się z oczekiwaniami
- Pozwala reagować na zmiany w trakcie realizacji projektu
- Pozwala klientowi na bieżąco kształtować produkt



Sabre Holdings

Dlaczego Agile?

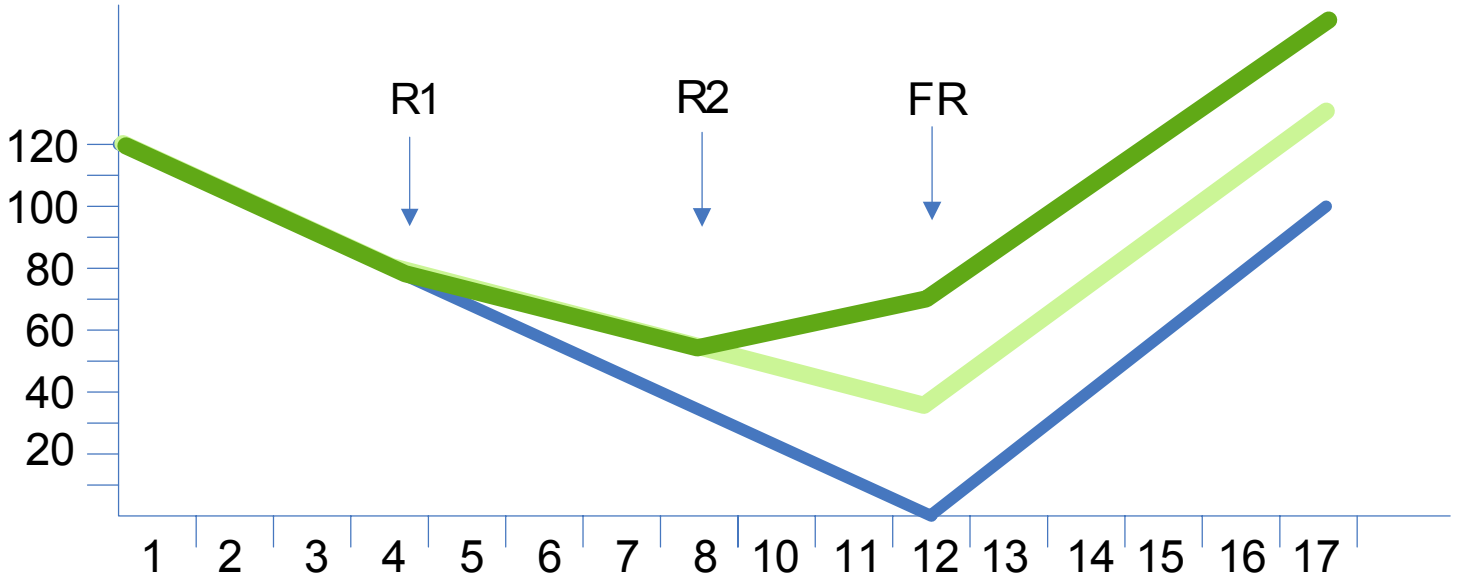
- Dzięki simple design i obecności testów łatwo jest wprowadzać zmiany
- *Maintanance* systemu jest tańszy w porównaniu z dotychczasowym podejściem



Sabre Holdings

Dlaczego Agile?

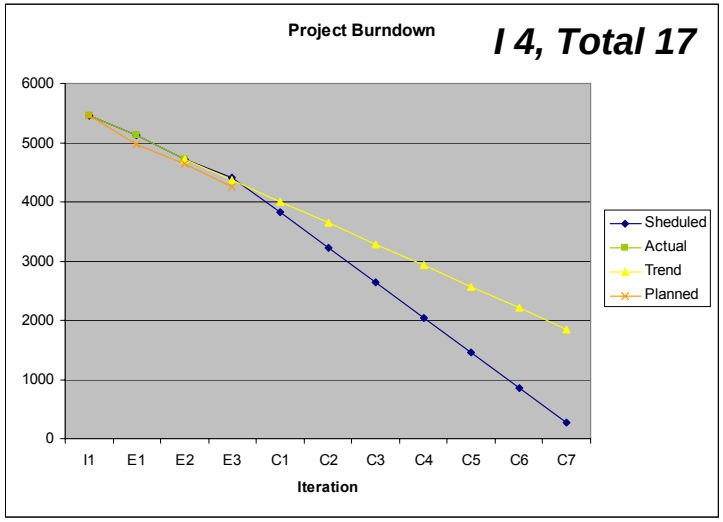
- Pozwala na bieżąco zarządzać *release* planem dzięki czemu klient ma większe elastyczność budżetowania projektu

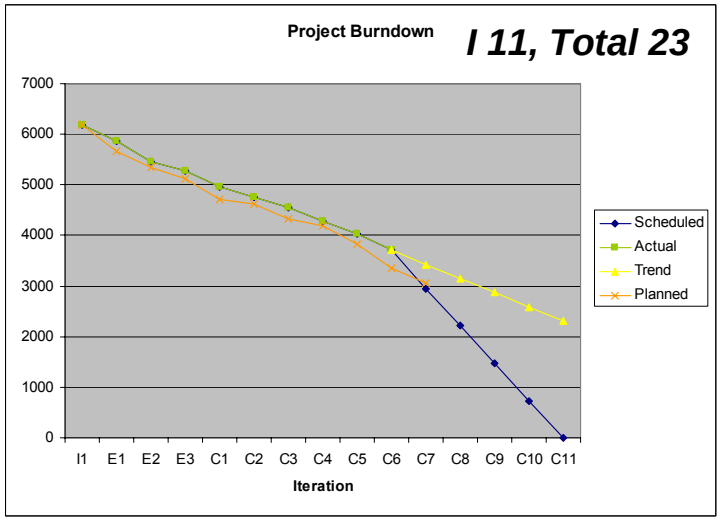


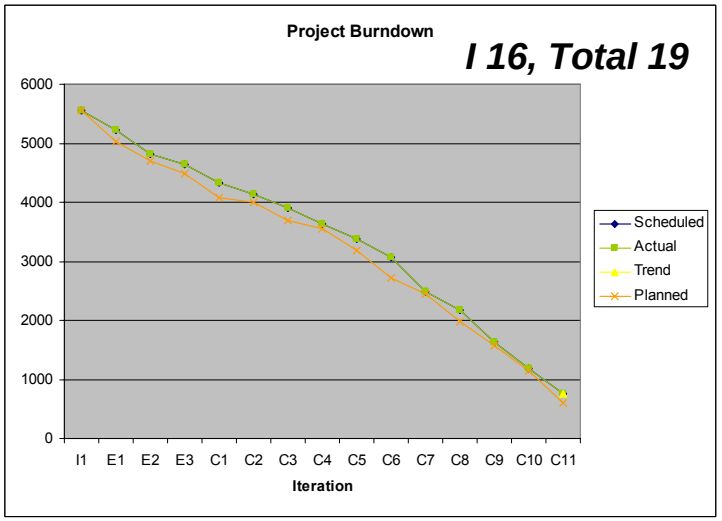
Sabre Holdings

Dlaczego Agile?

- Większa kontrola realizacji prac
- Szacunek na podstawie dotychczasowych doświadczeń zamiast planowania przyszłej realizacji
- Natychmiastowa szacunkowa weryfikacji wstępnych estymacji







Sabre Holdings

Dlaczego Agile

- Od pierwszej iteracji system jest gotowy do wdrożenia dzięki *continious integration*
- Unit testy zwiększają zaufanie do działania systemu
- Działające oprogramowanie motywuje zespół
- Samodzielność zespołu owocuje większym zaangażowaniem

Sabre Holdings

Agile resources

- Agile Development
  - http://www.agilemanifesto.org/
  - http://www.agilealliance.org/

- Extreme Programming
  - http://www.xprogramming.com/
  - http://www.extremeprogramming.org/
  - http://www.jera.com/techinfo/xpfaq.html

Sabre Holdings

Agile books

Agile Software Development, Principles, Patterns, and Practices ***Rober C. Martin***

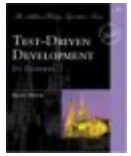
Applying UML and Patterns
Craig Lerman

Agile and Iterative Development: A Manager's Guide
Craig Lerman

Extreme Programming Explained: Embrace Change
Kent Beck, Cynthia Andres

A Practical Guide to eXtreme Programming

David Astels, Granville Miller, Miroslav Novak



Test Driven Development: By Example
Kent Beck

Refactoring: Improving the Design of Existing Code

Martin Fowler, Kent Beck, John Brant, William Opdyke

Sabre Holdings