# Agile Development. Filozofia programowania zwinnego


Autor: James Shore, Shane Warden
ISBN: 978-83-246-1614-5
Tytuł oryginału: The Art of Agile Development
Format: 168x237, stron: 480

**Zbiór praktycznych wskazówek dla producentów oprogramowania**

- Jak wdrożyć metodologię programowania zwinnego?
- W jaki sposób zaangażować klientów w projekt?
- Jak kontrolować jakość produktów?

Programowanie zwinne (Agile Development) to obecnie jedna z najpopularniejszych metodologii zarządzania projektami programistycznymi. Metodyka Agile jest szczególnie użyteczna w małych zespołach programistycznych, w których z racji ułatwionej komunikacji nie ma potrzeby tworzenia rozbudowanej dokumentacji. Programowanie zwinne opiera się na iteracyjnej realizacji kolejnych etapów projektu. Kluczem do sukcesu w tej metodzie jest efektywna współpraca między członkami zespołu projektowego.

Książka „Agile Development. Filozofia programowania zwinnego" to przewodnik po programowaniu ekstremalnym, oznaczanym zwykle skrótem XP, które jest jedną z technik wchodzących w skład tej metodyki. Czytając ją, dowiesz się, jak wdrażać metodologię Agile w firmie, na czym polega programowanie ekstremalne i jaką rolę w procesie pełnią poszczególni członkowie grupy projektowej. Nauczysz się budować zespół i określać zakresy zadań osób biorących udział w pracach, planować harmogram udostępniania kolejnych wersji produktu oraz kierować procesem jego tworzenia. Poznasz metody testowania programu i usuwania z niego błędów, zasady pisania dokumentacji oraz reguły prowadzenia spotkań roboczych z klientami.

- Wdrażanie programowania zwinnego
- Techniki programowania ekstremalnego
- Członkowie zespołu XP
- Zarządzanie zespołem
- Angażowanie klienta w proces wytwórczy
- Tworzenie raportów
- Udostępnianie kolejnych wersji systemu
- Standardy pisania kodu
- Testowanie i usuwanie błędów
- Optymalizacja wydajności programu

**Od filozofii do mistrzostwa w zwinnym programowaniu!**

Wydawnictwo Helion
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl

# Spis treści

## Część II Stosowanie XP

## Część III Mistrzostwo w dziedzinie programowania zwinnego

ROZDZIAŁ 1

# Dlaczego zwinne programowanie?

Zwinne wytwarzanie oprogramowania (ang. *agile development*) jest popularne. Wszystkie „fajne" firmy korzystają z tego podejścia: Google, Yahoo, Symantec, Microsoft i tak dalej[1]. Znam firmę, która zmieniła nazwę na Agili-coś tam, aby przyłączyć się do trendu. Organizacja ta zadzwoniła do mnie z prośbą, abym usprawnił ich „zwinne procesy", które po dokładniejszej analizie okazały się niczym więcej jak zlecaniem programowania firmom zewnętrznym w innym kraju niż zwykle. Oczekuję, że w bardzo niedalekiej przyszłości w ofercie dużych firm konsultingowych pojawią się Certyfikowane zwinne procesy i Certyfikowani konsultanci do spraw zwinnego programowania.

Proszę, nie dajcie się wciągnąć w ten bałagan.

W 1986 roku Brooks zapowiedział, że uniwersalne rozwiązania nie powstaną, że do roku 1996 żadna pojedyncza technologia ani technika zarządzania nie doprowadzi do dziesięciokrotnego wzrostu produktywności, niezawodności lub prostoty [Brooks]. I rzeczywiście, nikomu się to nie udało.

Również zwinne wytwarzanie oprogramowania nie jest uniwersalnym rozwiązaniem.

Co więcej, nie zalecam stosowania zwinnego programowania, jeśli ma służyć wyłącznie zwiększeniu produktywności. Zalety tego podejścia — także możliwość częstszego udostępniania oprogramowania — wynikają z *odmiennego* trybu pracy, a nie z *szybszego* wykonywania zadań. Choć według anegdotycznych wzmianek zespoły stosujące zwinne programowanie wykazują się ponadprzeciętną wydajnością[2], efekt ten nie powinien być główną przyczyną wprowadzania tego podejścia. Zespół potrzebuje czasu na opanowanie zwinnego wytwarzania oprogramowania. W trakcie nauki — a może to zająć kwartał lub dwa — członkowie grupy będą pracować wolniej, a nie szybciej. Ponadto nacisk na produktywność czasem zachęca zespół do stosowania uproszczeń i mniejszego rygoru w pracy, co może doprowadzić do *spadku* wydajności.

[1] Źródło: różne raporty użytkowników na konferencjach poświęconych programowaniu ekstremalnemu i zwinnemu programowaniu.

[2] Zobacz na przykład [Van Schooenderwoert], [Mah] i [Anderson 2006].

Zwinne wytwarzanie oprogramowania jest obecnie modne, ale to nie powód, aby stosować to podejście. W trakcie podejmowania decyzji dotyczącej jego wprowadzenia ważna jest odpowiedź na tylko jedno pytanie.

Czy zwinne wytwarzanie oprogramowania pomoże w osiąganiu większych sukcesów?

Kiedy zespół odpowie na to pytanie, będzie wiedział, czy powinien stosować zwinne programowanie.

## Czym jest sukces?

Tradycyjnie *sukces* utożsamiany jest z dostarczeniem produktu na czas, w oczekiwanej cenie i zgodnie ze specyfikacją. Standish prezentuje kilka klasycznych definicji [Standish]:

*Udane*

„Oprogramowanie ukończono na czas, po oczekiwanych kosztach, z mechanizmami i funkcjami zgodnymi z wyjściową specyfikacją".

Mimo popularności tych definicji, czegoś w nich brakuje.

*Z problemami*

„Oprogramowanie jest gotowe i działa, ale przekroczono budżet oraz harmonogram, a mechanizmy i funkcje są uboższe niż w wyjściowej specyfikacji".

*Nieudane*

„W pewnym miejscu cyklu rozwoju rozwój oprogramowania anulowano".

Mimo popularności tych definicji, nie są one w pełni poprawne. Projekt może być udany, nawet jeśli producent nie zarobi na nim ani grosza. Z kolei nawet jeśli przyniesie milionowe zyski, można go uznać za problematyczny.

W magazynie „CIO" znajduje się komentarz do tej osobliwości:

> Nawet te projekty, które spełniają wszystkie tradycyjne kryteria powodzenia — w zakresie czasu, budżetu i specyfikacji — mogą okazać się nieudane, ponieważ produkty nie są atrakcyjne dla użytkowników lub w ostatecznym rozrachunku nie zapewniają firmie korzyści.
>
> […] Podobnie, projekty uznawane za nieudane według tradycyjnych miar z branży IT mogą okazać się sukcesem, kiedy — mimo problemów w obszarze kosztów, czasu lub specyfikacji — system jest uwielbiany przez odbiorców i zapewnia nieoczekiwane korzyści. Na przykład w firmie świadczącej usługi finansowe nowy system […] został dostarczony z sześciomiesięcznym opóźnieniem i kosztował ponad dwa razy więcej, niż zakładano (ostateczny koszt wyniósł 5,7 miliona dolarów). Jednak projekt zwiększył elastyczność organizacji (po 13 miesiącach) i oceniono go jako wielki sukces. Wysokość umarzanych kredytów spadła o 33 miliony dolarów, a krótszy czas potrzebny na zapewnienie korzyści i wyższa wydajność doprowadziły do 50-procentowego wzrostu w liczbie jednocześnie prowadzonych testów strategii ściągania wierzytelności[3].

[3] Nelson R. Ryan, *Applied Insight — Tracks in the Snow*, „CIO Magazine". *http://www.cio.com/archive/090106/applied.html*.

# Poza harmonogram

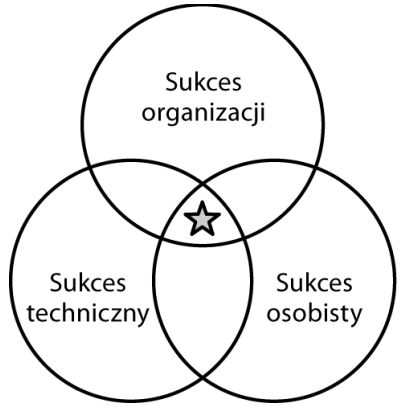
Sukces musi zależeć od czegoś innego niż mieszczenie się w terminie, ale od czego?

Kiedy byłem dzieckiem, cieszyła mnie sama zabawa. Uwielbiałem wyzwania związane z programowaniem. Każde uruchomienie aplikacji traktowałem jak wielkie zwycięstwo. Wtedy nawet program, który nie działał, traktowałem jak pewnego rodzaju sukces, jeśli tylko tworzenie go sprawiało mi radość. Powodzenie postrzegałem głównie w kategoriach *osobistych nagród*.

Kiedy nabrałem doświadczenia, zacząłem pisać bardziej skomplikowane programy i często gubiłem się w ich działaniu. Musiałem porzucić niektóre projekty przed ich ukończeniem. Zacząłem wierzyć, że kluczem do sukcesu jest łatwość konserwacji. Spostrzeżenie to potwierdziło się, kiedy podjąłem pracę i zacząłem współpracować z zespołami innych programistów. Byłem z siebie dumny, kiedy napisałem kod, który był elegancki i łatwy w konserwacji. Sukces oznaczał wtedy *doskonałość techniczną*.

Niektóre projekty kończą się niepowodzeniem mimo dobrego kodu. Nawet doskonale przeprowadzone projekty mogą być nieciekawe dla użytkowników. Doszedłem do wniosku, że zespół projektowy, w którym pracuję, jest częścią większego systemu, obejmującego dziesiątki, setki, a nawet tysiące osób. Produkty mają zapewniać zadowolenie ich wszystkich, a przede wszystkim tych, którzy odpowiadają za wysokość mojej pensji. Dla osób, które opłacają pracę, wartość oprogramowania musi przewyższać jego koszty. Sukces oznaczania *zapewnianie korzyści* firmie.

Przedstawione definicje nie są ze sobą sprzeczne. Ważny jest sukces na wszystkich trzech poziomach (zobacz rysunek 1.1). Bez powodzenia na poziomie osobistym twórca będzie miał problemy z umotywowaniem siebie i pracowników. Bez sukcesu technicznego kod źródłowy w pewnym momencie załamie się pod swym ciężarem. Przy braku powodzenia na poziomie organizacji może się okazać, że zespół nie jest już potrzebny.




*Rysunek 1.1. Oblicza sukcesu*

## Znaczenie sukcesów na poziomie organizacji

Zespoły programistyczne często zaniedbują sukces na poziomie firmy na rzecz łatwiejszego do osiągnięcia powodzenia technicznego i osobistego. Należy jednak pamiętać o tym, że nawet jeśli *dana osoba* nie bierze odpowiedzialności za sukces organizacji, jej przełożeni oceniają zespół na tym poziomie. Wyższa kadra zarządzająca i dyrektorzy wykonawczy prawdopodobnie nie będą zwracać uwagi na to, czy oprogramowanie jest eleganckie, łatwe w konserwacji, a nawet lubiane przez użytkowników. Dla przełożonych liczą się wyniki, czyli zwrot z inwestycji w projekt. Jeśli zespół nie osiąga sukcesów w tym obszarze, menedżerowie podejmą działania, które zapewnią powodzenie.

Niestety, wyższa kadra zarządzająca zwykle nie ma czasu lub odpowiedniej perspektywy, aby w każdym projekcie stosować wyrafinowane rozwiązania. Dlatego częściej używają miecza niż skalpela i słusznie oczekują, że to zespół projektowy zajmie się szczegółami.

Kiedy menedżerowie są niezadowoleni z wyników zespołu, wyciągają miecze. Najbardziej oczywisty obiekt cięć to koszty. Są dwa proste sposoby na ich zmniejszenie: bardzo krótkie terminy, które mają skrócić czas wytwarzania, oraz zlecanie zadań firmom mającym siedziby w państwach o niższych kosztach pracy. Można też zastosować oba te rozwiązania jednocześnie.

Są to jednak nieeleganckie techniki. Krótkie terminy często wydłużają czas pracy zamiast go skracać [McConnell 1999, s. 220], a zlecanie zadań firmom zewnętrznym wiąże się z ukrytymi kosztami [Overby].

Czy krótkie terminy i groźba zlecenia zadań firmom zewnętrznym brzmią znajomo? Jeśli tak, pora na wzięcie przez zespół odpowiedzialności za sukces, i to nie tylko osobisty i techniczny, ale także na poziomie organizacji.

### CO CENIĄ ORGANIZACJE?

Choć wartość niektórych projektów wynika bezpośrednio ze sprzedaży, korzyści organizacji obejmują także czynniki inne niż dochody. Projekty zapewniają wartość na wiele sposobów i nie zawsze można wycenić je w złotówkach i groszach.

Można wyróżnić następujące źródła wartości (obok dochodów i oszczędności)[4]:

- odróżnianie się od konkurencji;
- kreowanie marki;
- wzrost lojalności klientów;
- spełnianie wymagań prawnych;
- oryginalne badania;
- informacje strategiczne.

[4] Oparte częściowo na [Danne i Cleland-Huang].

# Wkraczanie w świat zwinnego programowania

Czy zwinne wytwarzanie oprogramowania pomoże zespołowi w osiąganiu większych sukcesów? Możliwe. Zwinne programowanie koncentruje się osiąganiu celów osobistych, technicznych *oraz* organizacji. Jeśli zespół ma problemy w którymś z tych obszarów, wdrożenie zwinnego wytwarzania oprogramowania może okazać się pomocne.

## Sukces na poziomie organizacji

Zwinne metody pozwalają osiągnąć sukces na poziomie organizacji poprzez koncentrację na zapewnianiu wartości i zmniejszaniu kosztów. Bezpośrednio przekłada się to na wyższy zwrot z inwestycji. Zwinne metody powodują także wczesne ustalanie oczekiwań, dlatego jeśli projekt nie prowadzi do sukcesu organizacji, wiadomo o tym na tyle wcześnie, że można anulować go przed poniesieniem wysokich nakładów.

Zespoły stosujące zwinne programowanie zwiększają wartość dzięki włączeniu w pracę ekspertów biznesowych i skoncentrowaniu wysiłków na podstawowych wartościach, które projekt ma zapewniać organizacji. W projektach realizowanych zgodnie ze zwinnym podejściem jako pierwsze przygotowywane są najbardziej wartościowe funkcje, a zespół często udostępnia nowe wersje, co znacznie zwiększa wartość. Kiedy zmienią się potrzeby biznesowe lub zespół zdobędzie nowe informacje, można zmienić kierunek prac, aby dostosować go do zaistniałej sytuacji. Doświadczone zespoły stosujące zwinne podejście *poszukują* nieoczekiwanych możliwości, które pozwolą ulepszyć plan działania.

Takie zespoły pozwalają zmniejszyć koszty. Po części wynika to z doskonałości technicznej. W najlepszych zwinnych projektach pojawia się tylko kilka błędów miesięcznie. Mniejsze jest także marnotrawstwo, co jest efektem wczesnego anulowania słabych projektów i zastępowania kosztownych praktyk rozwoju prostszymi. Komunikacja w zwinnych zespołach jest szybka i precyzyjna, a praca jest możliwa nawet w przypadku nieobecności kluczowych osób. Członkowie regularnie kontrolują proces i nieustannie usprawniają kod, dzięki czemu konserwacja oprogramowania i wprowadzanie w nim poprawek są łatwiejsze.

## Sukces techniczny

Programowanie ekstremalne, czyli zwinna metoda, na której koncentruję się w tej książce, szczególnie dobrze nadaje się do zapewniania sukcesu technicznego. Programiści stosujący XP współpracują ze sobą, co pomaga im kontrolować drobne szczegóły istotne w doskonałych produktach, a jednocześnie gwarantuje, że każdy fragment kodu przejrzą przynajmniej dwie osoby. Programiści nieustannie integrują kod, co umożliwia zespołowi udostępnienie oprogramowania w każdym momencie, kiedy ma to sens biznesowy. Cały zespół koncentruje się na całkowitym ukończeniu jednej funkcji przed przystąpieniem do pracy nad następną. Zapobiega to nieoczekiwanym opóźnieniom w momencie udostępniania produktu i umożliwia swobodne zmienianie kierunku.

Programowanie ekstremalne obejmuje, obok struktury rozwoju, zaawansowane praktyki techniczne, które prowadzą do osiągnięcia doskonałości technicznej. Najbardziej znaną techniką jest wytwarzanie sterowane testami. Pomaga ono pisać kod, który działa dokładnie tak, jak programiści tego oczekują. Zespoły stosujące XP tworzą także proste, nieustannie zmieniające się projekty, które można łatwo zmodyfikować przy zmianie planów.

## Sukces osobisty

Sukces osobisty jest, no cóż, osobisty. Zwinne programowanie może nie spełniać wszystkich wymagań w obszarze sukcesu osobistego. Jednak po przyzwyczajeniu się do tej techniki użytkownik, niezależnie od zajmowanego stanowiska, prawdopodobnie odkryje wiele jej zalet.

*Kierownictwo i wyższa kadra zarządzająca*
: Te osoby docenią długowieczność oprogramowania i koncentrację zespołu na zapewnianiu wysokiego zwrotu z inwestycji.

*Użytkownicy, udziałowcy, eksperci z dziedziny i menedżerowie produktu*
: Te osoby docenią wpływ, jaki mają na kierunek rozwoju oprogramowania, a także koncentrację zespołu na dostarczeniu użytecznych i wartościowych programów oraz wysoką częstotliwość udostępniania wersji.

*Menedżerowie produktu i projektu*
: Menedżerowie docenią możliwość zmiany kierunku prac w obliczu nowych potrzeb biznesowych, zdolność zespołu do podejmowania i spełniania zobowiązań oraz wyższe zadowolenie udziałowców.

*Programiści*
: Programiści docenią wyższą jakość pracy wynikającą z podniesienia jakości technicznej, większego wpływu na szacunki i harmonogramy oraz niezależności zespołu.

*Testerzy*
: Testerzy docenią traktowanie ich jak pełnoprawnych członków zespołu, możliwość wpływu na jakość na wszystkich etapach projektu oraz bardziej wymagającą i mniej schematyczną pracę.

Praca w zwinnych zespołach to jeden z najprzyjemniejszych okresów w mojej karierze. Wystarczy, że wyobrażę sobie przyjacielską atmosferę w zespole, który współpracuje w celu zaprojektowania i udostępnienia produktu o trwałej wartości, gdzie każdy członek grupy z entuzjazmem przyczynia się do skutecznego działania całości. Wyobrażę sobie branie odpowiedzialności za dany obszar — techniczny, biznesowych lub związany z zarządzaniem — i zaufanie reszty zespołu do moich zawodowych sądów i umiejętności. Jakże przyjemne jest rozwiązywanie problemów projektowych i obserwowanie poprawy jakości.

Zwinne programowanie zmienia obraz branży. Rozwój i udostępnianie oprogramowania w nowy sposób wymaga dużo pracy i pomyślunku. Jednak jeśli zespół stosuje omawiane podejście spójnie i przestrzegając wszelkich zasad, może uzyskać niezwykłe wyniki. Będzie regularnie wytwarzał naprawdę wartościowe oprogramowanie i każdego tygodnia demonstrował postępy. Ponadto rozwój oprogramowania stanie się przyjemniejszy niż kiedykolwiek wcześniej.

Wszyscy gotowi? Zaczynamy.