M. Bartyzel, M. Sieraczkiewicz

Kiedy Agile nie zadziała

Jakie błędy najczęściej popełnia się stosując metody zwinne?

Już ponad dziesięć lat minęło odkąd metody zwinne zaczęły podbijać świat. Przez ten czas rozwiązania z tego obszaru uległy znacznemu rozwojowi i wielu zmianom. Jednocześnie narosło wiele mitów. Jak wygląda w praktyce stosowanie koncepcji ze świata Agile? Kiedy proponowane podejście się sprawdza a kiedy nie? Czynimy założenie, że znasz podstawowe założenia dotyczące metod zwinnych.

Wprowadzenie metody z rodziny Agile nie jest w ogóle łatwe. Problemem zazwyczaj jest to, że kadra zarządzająca powierzchownie dowiadując się o co chodzi w tym podejściu, odbiera nową metodę jako obietnicę - od teraz w magiczny sposób wszystko zacznie lepiej działać. Nieważne, że mamy kiepskich ludzi i hołdujemy zasadzie "każdego specjalistę można zastąpić skończoną liczbą studentów". Nieważne, że kompletnie nie dba się o zarządzanie wiedzą i umiejętnościami w zespole, bo nigdy nie ma na to czasu. Nieważne, że osoby realizujące projekty są przerzucane między projektami - bo przecież chodzi o interdyscyplinarność i każdy się powinien we wszystkim orientować. Nieważne, że terminy ustala się apriori, z do końca niejasnych przyczyn lub zakłada się, że to jedyny sposób, żeby przekonać klienta, że jesteśmy profesjonalni, a w terminie się "jakoś zmieścimy". Nieważne, że nie znajdujemy czasu na to, aby utrzymywać architekturę i kod, bo przecież nie ma czasu na tego typu zbędne czynności, których efektu nie widzi klient.

Nieważne, bo przecież Agile obiecuje, że teraz będzie już tylko pięknie. Dokleimy etykietkę to tego co robimy teraz - niech się to nazywa np. "Scrum", zrobimy trochę więcej chaosu, bo przecież chodzi o zarządzanie chaosem - zatem każdy robi co chce i jak chce, przestajemy cokolwiek dokumentować. Przecież stosujemy Agile, wszystko się samozorganizuje.

I dlatego najczęściej Agile nie działa, bo ludzie, którzy po omacku go stosują lub kierownictwo, który wstępnie akceptuje korzyści z danego podejścia, stosują bardzo uproszczone myślenie - "Ok. Robimy Scrum. Ale tak naprawdę 8 z 10 elementów nam nie pasuje, więc wybierzemy sobie, to które pasują (dwa elementy), a resztę będziemy robić tak jak do tej pory (czytaj wyżej).

I tak wszystko będzie tak jak było, albo gorzej.

Zatem uwaga: W takim przypadku nie należny, nawet nie WOLNO zacząć stosować Agile. Agile to zmiana, ogromna zmiana, to nie etykietka, która rozwiąże wszystkie problemy (w tym przypadku prawdopodobnie lepiej się sprawdzi butelka alkoholu).

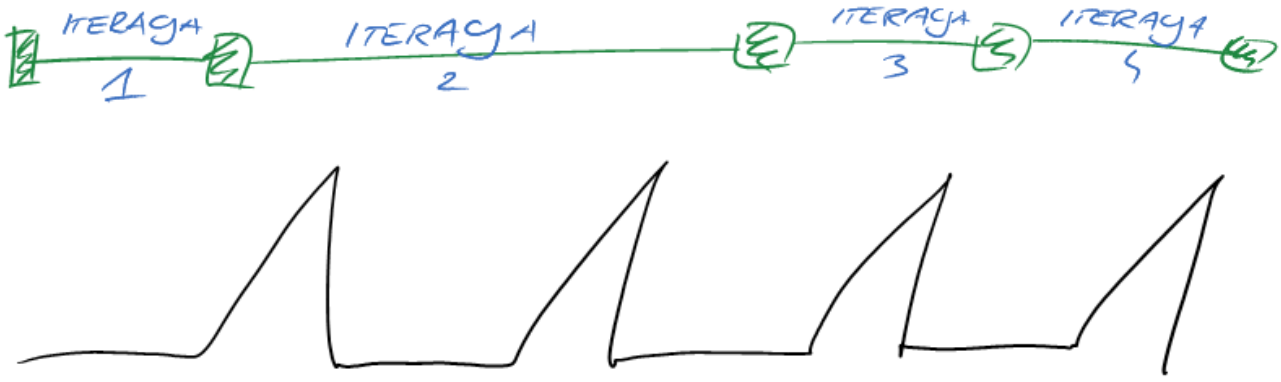
SUKCESY I PORAŻKI

Mieliśmy wiele okazji widzieć, uczestniczyć i procować z zespołami, które pracują zwinnie (przynajmniej tak deklarują). Oto kilka historii, większość z nich powiązana ze Scrumem.

Pierwszy przykład

Pierwszy z nich to projekt, którego głównym celem było stworzenie dodatkowego, choć istotnego modułu służącego do zarządzania obiegiem dokumentów firmy. Wspomniany moduł był częścią większego systemu, który to

Rysunek 1. Nierówne iteracje



został oparty o gotowe rozwiązanie służące do zarządzania procesami biznesowymi. Zespół był kilkuosobowy wdrażający nową technologię, nie używaną wcześniej w praktyce.

Początek projektu był bardzo intensywny. Wybrano Scrum Mastera, zdefiniowano Backlog, stworzono zgrubny plan na kilka Sprintów, zespół określił zadania i zabrał się do pracy. Pierwszy miesiąc był bardzo poprawny. Udało się zachować termin 4-tygodniowego Sprintu. Powstała też pierwsza wersja produktu, choć bardzo uboga, to można było zobaczyć jak działa i klient mógł ocenić efekty, z których był zadowolony.

Nierówne iteracje

Następne dwa tygodnie nie były już tak dobre. Okazało się, że odbyły się Sprinty o różnych i dziwnych długościach – 3 i 5 tygodni. Zespół uznał, że to będzie dobre rozwiązanie, aby zakończył „całe" funkcjonalności w czasie Sprintu. Tym samym naruszono istotną zasadę – równych długości iteracji – która sprawia, że mamy bardzo wyraźny punkt odniesienia dla wykonanych prac. Jasno można ocenić, jak duże zostały uczynione postępy. Przy ruchomych długościach iteracji, to co dzieje się w projekcie, staje się płynne i subiektywne.

Nieregularność codziennych spotkań

Ponadto Scrum Master stwierdził, że codzienne spotkania to w zasadzie strata czasu. Zatem odbywały się one najpierw co 2 dni, później co 3 dni, później raz na tydzień, później sporadycznie, aż zwyczaj umarł, zaś członkowie zespołu przestali wiedzieć, co się dzieje w projekcie poza fragmentem, nad którym pracują. Duch zaangażowania zginął. Nieregularność spotkań bardzo często w efekcie sprawia, że przestają się one odbywać, traci się naturalnie powstający pęd działania, aż do momentu, kiedy zespół przestaje mieć poczucie zintegrowania w obliczu wspólnego celu.

Nie odbywające się retrospekcje

Scrum Master zrezygnował również z przeprowadzania retrospekcji, nie widząc zbytnio korzyści, które mógłby z nich odnieść.

Po co są retrospekcje? Poetycko mówiąc, po to aby na chwilę się zatrzymać i podumać o tym, co się wydarzyło. Zaś mówiąc pragmatycznie należy przeanalizować to co się wydarzyło w projekcie:

- jakie działania się powiodły - w celu znajdowania praktyk, które warto rozpowszechniać w zespole i kontynuować w dalszej części projektu;
- jakie działania się nie powiodły, jakie błędy popełniliśmy, po to aby zastanowić się, jakie praktyki czy też konkretne działania należy wdrożyć, aby w przyszłości problemy danego typu nie powtarzały się.

Co się dzieje, kiedy nie ma retrospekcji? Nic się nie dzieje. Niewiele zmienia się. Retrospekcja to stymulacja, to trzeźwe i obiektywne spojrzenie na to, co się stało, aby wykorzystać możliwość uczenia się. W ten sposób zwielokrotniamy efekty naszego doświadczenia. Retrospekcja to esencja zwinności, gdyż wyciągnięta z niej nauka jest wstępem do wprowadzania zmian. To powinna być ostatnia praktyka, z której się rezygnuje.

Drugi przykład

Kolejny przykład, który chcemy przedstawić wyglądał nieco inaczej. Projekt dotyczył przepisania dużego, istniejącego już systemu służącego do zarządzania grupą powiązanych firm. W tym przypadku prawdziwym wyzwaniem na poziomie umiejętności była technologia. Przed rozpoczęciem projektu 20 osobowy zespół tworzył głównie rozwiązania oparte od technologie 4GL i posiadał umiejętności programowania proceduralnego. I to było jedno z największych wyzwań w tym projekcie – stworzenie systemu w nowej dla większości osób technologii i nowym, obiektowym paradygmacie programowania.

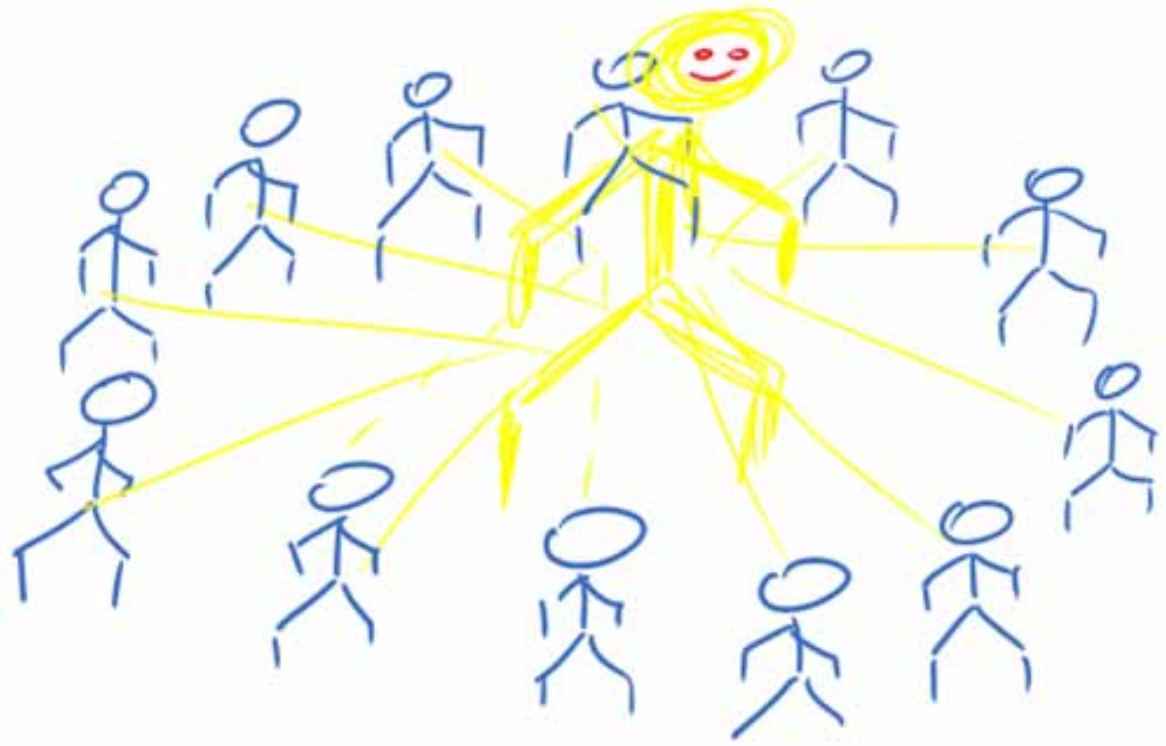
W tym przypadku również wprowadzono Scrum zamiast ciężkiej tradycyjnej metody, która od dawna uwierała.

Nierówny zespół i brak mentoringu

Z 20 osób stanowiących zespół tylko 5 znało jako tako nowe technologie, głównie z wiedzy nabytej hobbystycznie. Około połowa osób była bardzo mało doświadczona (mniej niż 3 lata w komercyjnych projektach), a kolejnych kilka do tej pory zajmowała głównie funkcje analityczne, gdzie teraz miały wrócić do prac programistycznych. Start projektu od strony technicznej był bardzo trudny. Wiedza i doświadczenie było zdobywane metodą prób i błędów. Ze względu na brak umiejętności bardzo szybko powstawały specjalizacje typu „my zajmujemy się tylko dostępem do danych", „my zajmujemy się front-endem". Zespół zaczął rozdzielać się na podzespoły specjalizacji technicznych. Osoby, które uczyły się technologii, popełniały bardzo dużo błędów i znacząco opóźniały prace. Nie było dobrze.

W tym przypadku pojawił się jeden z głównych problemów związanych z metodami zwinnym, równie kłopotliwy w przypadku każdego typu procesu wytwórczego – nierówny, małokompetentny zespół. Agile stawia bardzo silne założenie, że zespół powinny stanowić osoby doświadczone, które posiadają potrzebne kompetencje. Powyżej przedstawiony zespół nie spełniał tego warunku.

Jak sobie poradzić z tak idealistycznym założeniem? Nie należy brać, aż tak dosłownie tego stwierdzenia – w pełni kompetentny zespół to ideał, który jest trudno i rzadko osiągalny w realiach biznesowych. Dobry kierownik musi dbać o to, aby przynajmniej 60-70% członków zespołu było doświadczonych. Ponadto należy zadbać o zorganizowany proces mentoringu, czyli bezpośredniej współpracy dwóch osób – doświadczonej i niedoświad-



Rysunek 2. Brak lidera

czonej, aby przyspieszyć proces nabywanie odpowiednich umiejętności. Mentorem może być członek zespołu lub w przypadku, gdy zespół nie może pozwolić sobie na obciążenie kluczowych osób w projekcie, można skorzystać z pomocy z zewnątrz – innego zespołu, innego działu lub zewnętrznego konsultanta.

Zarządzanie wiedzą

Ponadto niezwykle ważnym zadaniem lidera jest zorganizowanie i zarządzanie procesem wymiany wiedzy. Poniżej znajdziesz kluczowe działania w tym obszarze:

1. Przygotuj program wdrażania nowej lub niedoświadczonej osoby do projektu:
   - jakimi umiejętności powinna się posługiwać (np. wzorce projektowe, test-driven development, refaktoryzacja, tworzenie czystego kodu);
   - w jakie technologie powinna być wprowadzona (np. Maven, NHibernate, NUnit, JUnit, Entity Framework, JSF, EJB, Symfony);
   - jakich założeń musi przestrzegać (np. standardu kodowania, przyjętej architektury – mantry architektonicznej, struktury organizacyjnej projektu, zasad dodawania nowych funkcjonalności);
   - kto, w jaki sposób i jak długo będzie wprowadzał w zadane zagadnienia.
2. Zorganizuj cykl wewnętrznych szkoleń. Warto dbać o ich systematyczność – mogą się odbywać co 2-4 tygodnie. Nie muszą być idealnie przygotowane, kluczowe jest, aby przekazać użyteczną wiedzę, która jest wykorzystywana w projekcie.
3. Dokonuj regularnej retrospekcji technicznej – analizy stosowanych rozwiązań projektowych, propagacji dobrych praktyk i wygaszania nieskutecznych lub szkodliwych rozwiązań. Warto taką analizę poprzedzać regularnie wykonywanymi przeglądami kodu.

Brak wyraźnego lidera

Co się działo w samym projekcie? Po pewnym czasie ludzie zaczęli pracować jako samotne wyspy. Mimo że odbywały się spotkania, dzielono zadania, nie zaistniał duch zespołu. W zasadzie po pewnym czasie większość osób ze swoimi zadaniami zaczęła dryfować w sobie tylko znaną stronę. Lider nie potrafił asertywnie zmobilizować ludzi do współpracy, nie reagował odpowiednio na pojawiające się problemy, nie potrafił odbudować zaangażowania we wspólne działanie.

Piękno i moc metod zwinnym polega na ich mocnym osadzeniu w ludziach. To ludzie są tutaj kluczowi, nawet najlepsze technologie, narzędzia czy proces, bez ludzi, którzy współpracują ze sobą i są zaangażowani, nie spowoduje, że projekt się uda. Każde przedsięwzięcie oparte na ludziach wymaga lidera, który będzie potrafił stworzyć wizję, przekuć ją w plan działania, zjednoczyć ludzi i nadać sens temu co robią oraz bezustannie wspierać w realizowaniu wizji. Wbrew powszechnie panującej opinii samoorganizujący się zespół to nie grupa osób działających samopas w niezintegrowanej formie. Lider potrzebny jest po to, aby połączyć wszystkie elementy układanki oraz zagrzać do walki.

Rezygnacja z formalizmów

Kropkę nad i w negatywnym sensie w tym przypadku stanowiła całkowita rezygnacja z formalizmów. Stosowana dotąd metodyka wymuszała bardzo dużą ilość

miejscami niepotrzebnej dokumentacji. W projekcie zaistniał efekt odbicia – zespół, który przez wiele lat wykonywał wiele żmudnych i często niepotrzebnych prac, teraz przechodząc w drugą, równie niebezpieczną skrajność – całkowitą zrezygnował z formalizmów. Co przy tego typu wielkości projektu efekcie spowodowało brak trwałej wiedzy o projekcie i zgody co do tego, co zostało ustalone.

Trzeci przykład

Dla odmiany przedstawimy teraz historię sukcesu i określimy, co przede wszystkim spowodowało powodzenie przedsięwzięcia. Tym razem mieliśmy do czynienia z dużym projektem i dużym zespołem. Kilkadziesiąt zaangażowanych osób.

Agile na poziomie firmy a nie tylko zespołu technicznego

Scrum wdrożony od początku do końca i którym została przesiąknięta cała organizacja. Już pierwsza wizyta w firmie wywoływała niezaprzeczalne wrażenie – każda osoba uczestnicząca w projekcie była przesiąknięta zasadami metod zwinnych – rozumiała znaczenie komunikacji, współpracy z zespołem i klientem oraz rozumiał sens stosowanego procesu.

Human infection i wizja

Nic tak nie zaraża jak czyjś entuazjazm. W przypadku omawianego projektu ten entuzjazm wisiał w powietrzu, a warto przypomnieć, że projekt był kilkudziesięcioosobowy, zaś sam Scrum funkcjonował już od 3 lat. Zatem nie był to efekt świeżości. Ludzie nawzajem zarażali się pasją. Ponadto wszystkie osoby znały wizję propagowaną przez lidera - widziały dokąd zmierza projekt, w jakim jest momencie, jakie plany są z nim związane, jakie jeszcze zmiany czekają zespół, jak ma wyglądać współpraca osób w projekcie.

Rysunek 3. Rezygnacja z formalizmów

Silny lider

To co wyraźnie wyróżniało tę organizację, a w zasadzie ten projekt, to bardzo silny lider, który dbał o rozpowszechnianie wizji, o rozpowszechniania przyjętych zasad zwinnych, który pomagał rozwiązywać problemy i usuwał przeszkody pojawiające się w projekcie. Ponadto lider dbał o to, co mu się skutecznie udawało, by przestrzegać przyjętych zasad. Zostało zdefiniowane i egzekwowane Definition of Done, codzienne spotkanie odbywały się regularnie, wprowadzano standardy w projektach, zacieśniano współpracę między zespołami a klientem.

Było to jak dotąd najbardziej wzorowe wdrożenie zwinności do projektu, jakie mieliśmy okazję widzieć.

Przykład czwarty

O ile powyższy przykład wydaje się książkowy czy wręcz nierealny, o tyle najbliższy projekt jest przykładem jak przy mało sprzyjających warunkach można wprowadzać zwinność z powodzeniem. Projekt realizowany w organizacji, w której odgórnie nałożonych było sporo założeń formalnych, narzucających wysoki poziom formalizmu. Stosowany był bardzo ciężki proces oparty o PMI, który musiał pozostać taki jaki był ze względu na wymagania formalne. Zespół 18 osób.

Infekcja oddolna

Zespół zaczął wdrażać oddolnie pojedyncze praktyki. Najpierw wprowadzono codzienne spotkania, które wyraźnie ożywiły współprace z zespole i zwiększyły integrację osób.

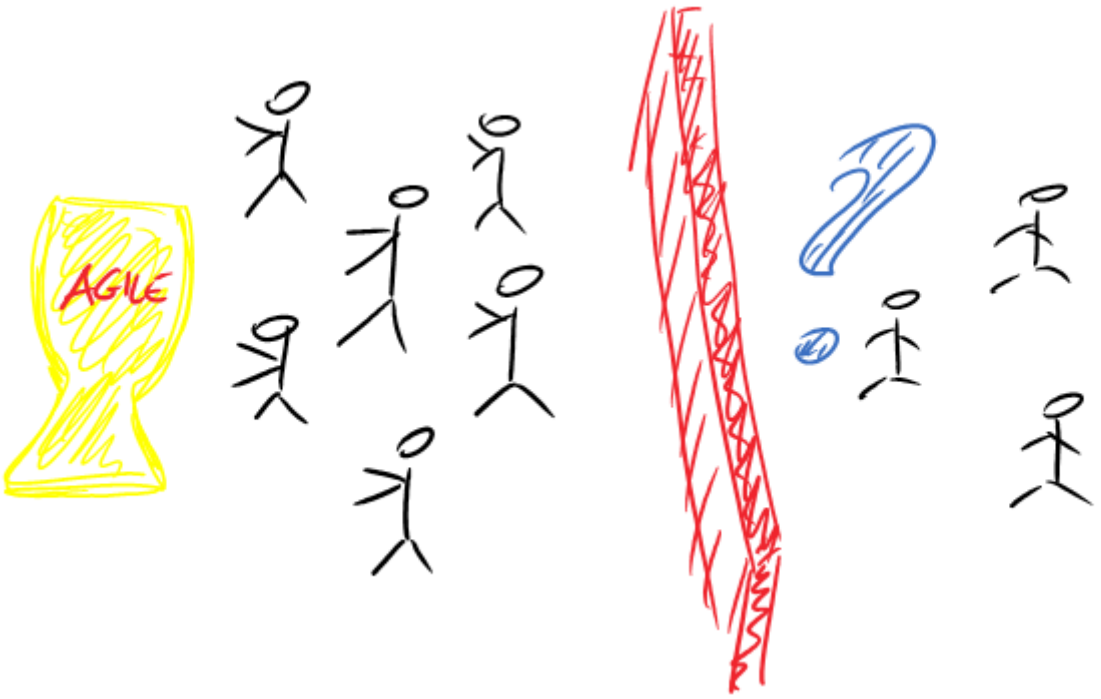
Stopniowo coraz krótsze iteracje

Następnie z kilkumiesięcznych milestonów zrobiono najpierw trzymiesięczne, potem dwumiesięczne, a ostatecznie miesięczne iteracje. Zdecydowanie szybciej zaczęto wykrywać problemy wynikające z integracji prac w projekcie oraz zmniejszyła się ilość przypadków, kiedy programista na kilka tygodni (miesięcy) tonął w problemie. Poprawianie znalezionych błędów również zajmowało mniej czasu (programiści poprawiali kod, który napisali miesiąc temu, a nie kilka miesięcy temu).

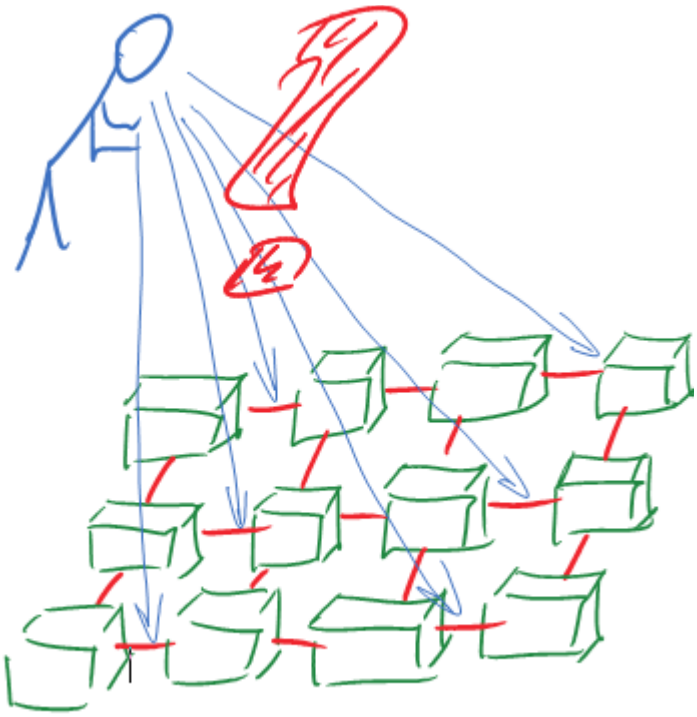
Doskonałość techniczna

Omawiany projekt miał ponad dziesięcioletnią historię. Wiele rozwiązań było naznaczonych ograniczeniami technologicznymi lat 90 (np. pozostałości po ograniczeniach długości nazw, kodowanie danych w jednej komórce w tabeli w celu zaoszczędzenia ilości miejsca). W efekcie projekt był bardzo niespójny. Ponadto na przestrzeni tych lat projekt został doświadczony wieloma technologiami i pomysłami architektonicznymi. Istna wieża Babel.

Rysunek 4. Proces zwinny funkcjonuje tylko na poziomie zespołu technicznego



W tym przypadku potrzebne było strategiczne podejście do ustabilizowania i rozwoju architektury. Najpierw zdefiniowano kluczowe elementy systemu (core systemu), jako główne elementy, które warto refaktoryzować. Dość autonomiczne, ale słabo napisane części systemu zostały odizolowane poprzez Anticorruption Layer. Wprowadzono mantrę architektoniczną – spójny opis architektury i definicji odpowiedzialności w systemie oraz wprowadzono plan wdrożenia powolnego przekształcania rozwiązań do nowej definicji architektury. W efekcie refaktoryzowania istotnych części systemu zaczęto wprowadzać testy jednostkowe, które w ostatecznie przekształciły się w stałą praktykę.



Rysunek 5. Niedojrzałość rozwiązań korporacyjnych

Przykład piąty

Po dwóch dobrych przykładach czas na antyprzykład, gdzie zwinność została wprowadzona tylko z nazwy, tylko dlatego że tak chciał zleceniodawca, zagraniczny partner, który wdrożył u siebie Scrum.

Powierzchowne zmiany

Z uwagi na oczekiwania zleceniodawcy firma pospiesznie zaczęła wprowadzać zmiany, masowo przeszkolono większość osób. Liderów i kilku programistów nazwano Scrum Masterami. Ze względu na wymagania zleceniodawcy wysłano ich na szkolenie certyfikowane. Wprowadzono codzienne spotkania, które nie odbywały się codziennie, a jak już się odbywały to trwały godzinę lub półtorej i przekształcały się w luźne pogadanki, nie stanowiąc źródła wymiany informacji o pracach i pojawiających się problemach.

Opór ludzki

Osoby uczestniczące w projekcie nie były w żaden sposób przygotowane do nadchodzących zmian. Zostały postawione przed faktem dokonanym. Natychmiast pojawił się opór i myślenie „zarząd znowu wymyśla nową metodę realizacji projektów a wszystko będzie po staremu". Zabrakło lidera oraz wyraźnej wizji, która precyzowałaby, po co zmiany zostały wprowadzane, jaki to będzie miało wpływ na ludzi, na projekt i co będzie się działo w ciągu najbliższych kilku miesięcy. Wśród członków zespołu nie

było żadnej motywacji do wprowadzania zmian, a co chwilę pojawiały się argumenty, dlaczego to nie zadziała.

INNE CZĘSTO POPEŁNIANE PROBLEMY

Brak klienta lub brak współpracy z klientem

Często mówi się, że rola tzw. Product Owner'a to rola mityczna. Trudno osiągalna w rzeczywiści. Oczywiście każda firma ma swoją rzeczywistość i może ona mniej lub bardziej sprzyjać ścisłej współpracy. Często jednak problem wynika z tego, iż zespół nie wie jak przekonać klienta do ściślejszej współpracy. Najczęściej brakuje umiejętności wskazania klientowi korzyści z zacieśnienia współpracy i strat płynących z braku bezpośredniego kontaktu. Co w efekcie objawia się jako następny przedstawiony problem:

Rysunek 6. Brak wiedzy i doświadczeń

Proces zwinny funkcjonuje tylko na poziomie zespołu technicznego

I co ciekawe ten wariant częściowo się sprawdza, tzn. zespół sporo zyskuje przez wprowadzenie praktyk na poziom zespołu (np. codziennych spotkań, definition of done, retrospekcji, planowania, podziału zadań). Jednak bardzo często powoduje izolowanie się od klienta, a w efekcie nawet obawę przed kontaktem z klientem. Ponieważ feedback jest bardzo rzadki, toteż jest duże prawdopodobieństwo, że będzie trzeba wprowadzać dużo zmian w tym, co zostało już wykonane.

Niedojrzałość rozwiązań korporacyjnych

Rozwiązania zwinne były od początku skrojone dla małych zespołów, z czasem zaczęto wymyślać metody skalowanie zespołów oraz pracy rozproszonej. Ogromnym wyzwaniem jest wprowadzenie metod z rodziny Agile w środowisku korporacyjnym, gdyż zazwyczaj organizacyjnie struktury tego typu są mało przyjazne dla podejść adaptacyjnych oraz dopiero w ostatnich latach kształtują się i są testowane rozwiązania na dużą skalę.

Brak wiedzy i doświadczeń

Główne wyzwanie w przypadku takich metod jak Scrum polega na tym, że jest to tylko szkielet, a wiele praktyk nie jest określonych. W efekcie nie wiadomo jak to powinno wyglądać w praktyce. Przykładową trudną umiejętnością jest inkrementacyjne tworzenie rozwiązań – małymi krokami rozwijamy aplikację dokładając do tego co już istnieje, zawsze posiadając działające oprogramowanie. W efekcie zespoły zaczynają twierdzić, że to podejście nie działa i następuje erozja procesu. Kluczem do sukcesu jest w tym przypadku pozyskanie doświadczonych liderów lub wsparcia z zewnątrz.

PODSUMOWANIE

„Miało być tak łatwo…" często słyszymy w zespołach, które wprowadzają podejście zwinne. Otóż należy sprostować powszechne przekonanie – metody zwinne są proste, ale nie są łatwe we wdrożeniu. Zazwyczaj potrzebna jest duża świadomość jaka jest zasada działania poszczególnych technik i kiedy się one sprawdzają, a kiedy nie. Żeby świadomie rezygnować z rekomendowanych praktyk trzeba mieć duże doświadczenie i znać konsekwencje zaprzestania ich użycia. Agile to również nie jest etykietka, to sposób myślenia, sposób życia, polegający na ciągłej zmianie opartej o uczenie. Żadna organizacja nie stanie się zwinna tylko i wyłącznie poprzez wprowadzenie lub usunięciu kilku ról czy nazw. Z drugiej strony końcowy zysk jest zazwyczaj duży i choć zmiana nie nastąpi z dnia na dzień, to warto strategicznie podejść do problemu i stopniowo przekształcać to, co już istnieje, w model zwinny. Wierzymy, że kilka powyższych przykładów będzie ostrzeżeniem przed tym, czego nie należy robić lub źródłem inspiracji do tego, jak podejść do danego problemu.

Michał Bartyzel,
Mariusz Sieraczkiewicz
m.bartyzel@bnsit.pl,
m.sieraczkiewicz@bnsit.pl

Trenerzy i konsultanci w firmie BNS IT. Badają i rozwijają metody psychologii programowania, pomagające programistom lepiej wykonywać ich pracę. Na co dzień Autorzy zajmują się zwiększaniem efektywności programistów poprzez szkolenia, warsztaty oraz coaching i trening.