Magazine

Szycie na miarę! Konieczność adaptacji modeli procesów w projekcie informatycznym.

Czy potrzebne są nam modele procesów? (cz.2)

Autor: Rafał Dobrosielski

O autorze: Absolwent Politechniki Gdańskiej, wydziału Elektroniki Telekomunikacji i Informatyki, ukończył również studia podyplomowe z zakresu Inżynierii Oprogramowania i Zarządzania Projektami.

Obecnie dyrektor programowy w Pomorskiej Akademii Optymalizacji Procesów.

Zawodowo, na co dzień, zajmuje się czynnie planowaniem, zapewnianiem i sprawdzaniem jakości w projektach informatycznych.
Jest autorem szkoleń i warsztatów z zakresu zarządzania projektem informatycznym, zarządzania i zapewniania jakości produktów informatycznych oraz adaptacji modeli procesów inżynierii oprogramowania na indywidualne potrzeby projektu.
Interesuje się analizą i modelowaniem procesów biznesowych w organizacjach jak również psychologią biznesu.
www: www.paop.com.pl
Email: rdobrosielski@paop.com.pl

„Tajemnicą, ku której dążył, była prostota, harmonijna prostota, zaskakująca bardziej niż najbardziej skomplikowana magia”.

Władimir Nabokow „Obrona Łużyna”

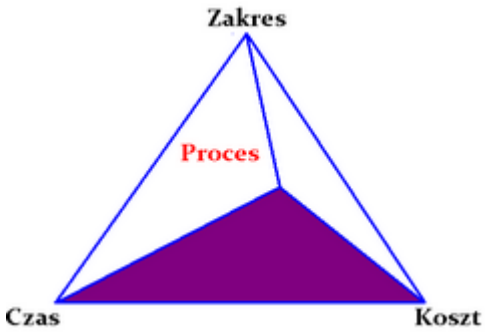
Wstęp

W dotychczasowych publikacjach w ramach cyklu „jakość w projekcie” starałem się, między innymi, przekazać, iż skupienie się na zarządzaniu procesem i czynnościami dość mocno zazębia się z jakością produktu i bardzo często ogranicza ryzyko powstania produktu złej jakości.

Czwarty wymiar trójkąta ograniczeń

Dobrze zdefiniowany proces zwiększa prawdopodobieństwo, iż projekt zostanie ukończony, a korzyści będą większe od poniesionych kosztów.

Alistair Cockburn [16] traktuje proces jako czwarty wymiar tradycyjnego trójkąta ograniczeń, w którym zdefiniowane czas, koszt i zakres projektu wydają się być nierozerwalnie zależne od siebie i jednoznacznie pozycjonują projekt.



Czwarty wymiar trójkąta ograniczeń, proces, jest często traktowany jak magia, która w cudowny sposób pozwala na osiągniecie początkowo niemożliwego rezultatu.

Odpowiednio dobrany proces pozwala nie tylko na zwiększenie zakresu projektu, ale właśnie na osiąganie wyższej jakości końcowego produktu, nie pociągając za sobą zmiany kosztów czy harmonogramu.

Niestety, powyższa zasada ma też odwrotne zastosowanie. Niejednokrotnie ciekawe, pożądane przez rynek projekty zakończyły się niepowodzeniem, gdyż źle dobrany proces znacznie ograniczał komunikację wewnątrz-projektową, podejmowanie szybkich i koniecznych decyzji, potencjał

zespołów wytwórczych, optymalne wykorzystanie zasobów itp.

Czasami, mimo, że projekt zakończył się sukcesem, każdemu z nas zdarzyło się choć raz powiedzieć: „dokonaliśmy tego, dokonaliśmy pomimo narzuconego, ograniczającego nasz potencjał procesu” [17].

Dzieje się tak dlatego, iż organizacje, szczególnie te, które prowadzą setki równoległych projektów wymagają, aby poszczególne zespoły prowadziły projekty wg tej samej metodyki. Dla koordynatorów projektów, zwłaszcza tych zarządzających portfelem projektów, jest to sytuacja najwygodniejsza. Daje im błędne i iluzoryczne poczucie możliwości obiektywnego śledzenia postępów projektów, porównania efektywności poszczególnych zespołów projektowych i oceny menedżerów projektów.

Jedyny, idealny przepis nie istnieje...

Osoby odpowiedzialne za procesy NPI (New Product Introduction) w organizacjach, zwykle przez wiele lat pielęgnują i „ulepszają” ciężkie, o dużym zakresie, metodyki wytwórcze oprogramowania. Historycznie i koncepcyjnie zaczerpnięte są one z procesów produkcyjnych hardware'u. Poszukują oni, dużymi nakładami finansowymi, uniwersalnego przepisu na wytwarzanie oprogramowania opartego (niestety) dodatkowo na modelu kaskadowym, czy też niewiele się od niego różniącego.

Przez długi czas panowało (a w wielu miejscach jeszcze panuje) przekonanie, że im „cięższa”, sprawdzona już w poprzednich projektach metodologia, trzymanie się zdefiniowanych w początkowej fazie projektu (np. dwuletniego) wymagań i planu, tym większa szansa na odniesienie sukcesu (wytworzenia pożądanego przez rynek oprogramowania o zadowalającym poziomie jakości). Jedynym zagrożeniem ma być wzrost kosztów, które musimy notorycznie śledzić. Prowadzenie projektów wg tej samej metodologii w całej organizacji daje też „komfort” pracy kierownikom projektów, których rola sprowadza się do roli tzw. skutecznych egzekutorów planów, uzgodnionych uprzednio z przełożonymi.

Brak chęci usprawniania metodologii, dynamicznego i elastycznego zrównoleglania prac w ramach podzespołów projektowych czy też pomijania nieistotnych i niepotrzebnych produktów pracy bierze się głównie ze strachu przed dokładaniem ryzyka do projektu w jego fazie wytwórczej. Bardzo bezpieczne jest, z punktu widzenia sukcesu osobistego kierownika projektu, wykazanie, że wytworzono (nawet kaskadowo) wszystkie pośrednie produkty pracy, użyto identycznej, jak to zaplanowano na początku projektu, technologii i osiągnięto wszystkie kamienie milowe. Jeśli w takim przypadku wynik projektu jest odrzucony przez zespoły zapewniania jakości czy też rynek, gdyż przyjęte rozwiązanie nie spełnia wymagań wydajnościowych lub nie realizuje potrzeb klienta

czy użytkownika, bardzo łatwo jest wykazać, że winne są nieprecyzyjne wymagania, konkurencja, błędnie wybrany w fazie planowania protokół komunikacyjny... . Można powiedzieć też, że „rynek po prostu się zmienił, ale któż to mógł przewidzieć, sytuacja jest tak dynamiczna a my nie jesteśmy jasnowidzami”... .

Otóż nic bardziej błędnego! Zastosowanie metodologii, która sprawdziła się w projekcie X, prowadzonym przez menedżera Y dysponującego zespołem ZZZ w okresie T, na rynku R, kulturze K i rozproszeniu R wcale nie gwarantuje sukcesu w projekcie o konfiguracji X1, Y1, ZZZ1, T1, K1, R1. Co więcej, nie gwarantuje też sukcesu, gdy tylko jeden z tych zmiennych czynników będzie różny od „oryginalnego”. Ten sam zespół często nie byłby w stanie przeprowadzić tego samego projektu z sukcesem, używając podobnej metodologii ale np. dla innego klienta o innym poziomie zaangażowania i współpracy. Kurczowe trzymanie się zdefiniowanych na początku procesu i wymagań, brak sprzężenia zwrotnego z rynkiem, klientem i użytkownikiem, inercja na zmiany technologii a przede wszystkim na zmienność i wrażliwość czynnika ludzkiego to kluczowe źródła problemów. Organizacje te, co prawda, wytwarzają oprogramowanie odnoszące sukces na rynku, ale jednocześnie charakteryzują się wysokim odsetkiem zamykanych, nikomu już niepotrzebnych projektów, bardzo wysokimi kosztami i poziomem frustracji swoich pracowników, wierzących w sukces, ale których potencjał został źle wykorzystany, ograniczany przez zbyt ciężką metodologię i brak iteracyjnego wytwarzania.

Na tego typu działanie nie mogą sobie pozwolić organizacje mniejsze, uboższe, dla których często jedyny projekt, który z najwyższym wysiłkiem prowadzą, może dopiero stanowić furtkę do ogromnego rozwoju, znalezienia strategicznego inwestora czy też otwarcia nowych rynków i możliwości. Nie mogą sobie również na to pozwolić organizacje działające na rynku wysokich, szybko rozwijających się technologii, wytwarzających dla klientów prowadzących działalność w dynamicznych dziedzinach o wysokim stopniu zmieniających się wymagań, prawa, usług itp. To pozornie „bezpieczne” wytwarzanie oprogramowania staje się przeżytkiem i w sposób naturalny jest wypierane przez rynek poprzez przegraną firm, którym tylko wydaje się, że wydane pieniądze na utrzymanie, respektowanie i egzekwowanie uzgodnionego, zunifikowanego procesu zapewni im sukces.

„Ciężko” nie oznacza bezpiecznie, „lekko” nie oznacza łatwo...

Powyżej starałem się podkreślić, że ryzyko projektu wytwarzanego za pomocą ciężkiej metodyki wcale nie jest mniejsze. Może się okazać, że czas konieczny na wytworzenie produktu zadowalającej jakości jest tak duży, że rynek nie jest w stanie zaczekać na efekty naszej pracy lub też poprawki konieczne do wykonania w produkcie końcowym czy w pośrednich produktach pracy (przy małym stopniu zrównoleglenia prac podzespołów projektowych) mogą być bardzo drogie.

Implementacja i wdrażanie metodyk „lekkich (usprawnionych)” natomiast w ogóle nie oznacza, że wytwarzanie będzie łatwiejsze, a odniesienie sukcesu bardziej pewne. Metodyki „lekkie” nakładają na zespoły konieczność osiągnięcia w takich projektach bardzo wysokiej (samo)dyscypliny i skupienia. Wymaganie co do wysokiego poziomu dyscypliny oznacza, że tzw. „usprawnienia” działają tylko wtedy, kiedy stosujemy się do nich z najwyższą dokładnością i uwagą, natomiast „skupienie” oznacza, że zespół skupia się wyłącznie na sprawach i zadaniach istotnych w danym momencie i w ramach posiadanej obecnie wiedzy [14]. Wyznacznikiem dla priorytetyzacji działań np. programistów czy też testerów może i powinno być np. oszacowane na daną chwile ryzyko projektowe, zidentyfikowane wąskie gardła i działania prowadzące do zwiększania ich efektywności.

Porównując metodyki tradycyjne i zwinne nie możemy poruszać się po płaszczyznach łatwiej<-> trudniej, bezpieczeństwo <-> wysokie ryzyko. Zły wybór metodyki, czy też niewłaściwa jej adaptacja, w każdej z jej rodzajów może doprowadzić do osiągnięcia wysokiego poziomu trudności prowadzenia przedsięwzięcia, rozwiązywania codziennych problemów a tym samym do zwiększenia ryzyka projektowego.

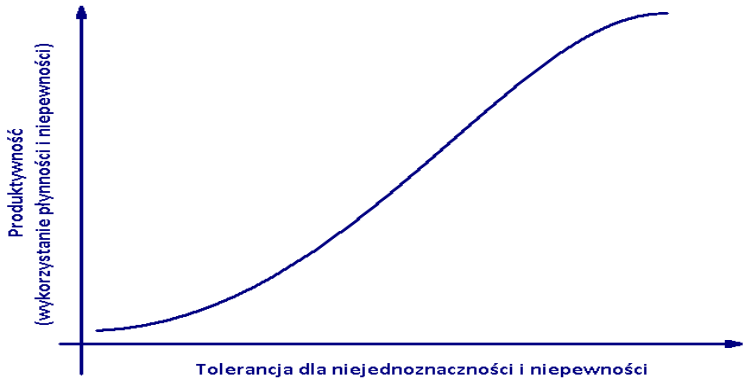
Płaszczyzną do bardziej obiektywnego porównywania i znalezienia istoty różnic pomiędzy wspomnianymi metodykami jest poziom efektywności zespołów projektowych, jaki można uzyskać przy tych samych kosztach oraz umiejętnościach.

Zysk efektywności okupiony tolerancją na niepewność

Gdzie należy szukać źródeł wzrostu efektywności zespołów projektowych przy jednoczesnym „zmniejszeniu” widocznych nakładów i wysiłków koniecznych na zarządzanie nimi?

Otóż wzrost efektywności można uzyskać jedynie zwiększoną tolerancją na tzw. „niepewność chwilową” i niejednoznaczność w projekcie.

Cockburn, w [14] zamieszcza użyteczny wykres, który bardzo prosto konkluduje powyższe rozważania.



Akceptowana tolerancja na poziom niepewności oznacza, jaki poziom niewiedzy w danej chwili są w stanie tolerować ludzie w danym projekcie, aby efektywnie (czy tez właśnie z większą efektywnością) kontynuować przydzielone im prace. Im większy jest akceptowalny poziom tolerancji, tym większy do uzyskania poziom zrównoleglania prac poszczególnych podzespołów projektowych oraz (w uproszczeniu) mniejsze nakłady na dokumentację i komunikację pozawerbalną.

Przy zwiększonej tolerancji na niepewność, można łatwiej implementować i adaptować metodyki zwinne, wytwarzanie iteracyjne i przyrostowe.

Niewątpliwie większy poziom niepewności będzie zaakceptowany przez zespoły o większym doświadczeniu, przewidywalności, elastyczności i umiejętności projektowania bardziej uniwersalnych rozwiązań.

Szycie na miarę... ludzkich możliwości

W części 3 niniejszego cyklu przejdziemy w pełni do klasyfikacji metodologii wytwórczych, ich zalet i wad oraz dostępnych w każdej z nich mechanizmów zapewniania jakości.

Tymczasem, podkreślmy jeszcze raz, że zwiększony poziom efektywności zespołów uzyskiwany w metodologiach usprawnionych (a tym samym możliwość wprowadzenia zmian w organizacji pracy tych zespołów) zależy od wielu czynników ludzkich. takich jak wspomniana już tolerancja na niepewność oraz zdolność do komunikacji (głównie werbalnej), wspólne doświadczenie, zaufanie, poczucie odpowiedzialności, preferencje w stosowaniu odpowiednich technik i metod programowania, specyfika i dostępność klienta, znajomość dziedziny przedmiotu projektu, umiejętność czytania „między wierszami”, wzajemne zrozumienie i wspieranie.

Otóż, nietrudno udowodnić, iż pomimo, że klasyfikowanie zespołów projektowych wg powyższych kryteriów pozwala je podzielić na kilka skończonych podzbiorów, to jednocześnie pozwala też stwierdzić, że każdy z nich jest unikatowy.

Dlatego też przed przystąpieniem do realizacji projektu należy bardzo mocno skupić się, pierwotnie – na wybraniu, a następnie – na adaptacji odpowiedniej metodologii na indywidualne potrzeby zespołu, jakim dysponujemy i zadania, jakie mamy przeprowadzić (w danym miejscu i czasie).

Często zdarza się, iż kierownicy projektów zbyt wcześnie bardzo mocno naciskają na rozpoczęcie prac techniczno projektowych. Jest to tylko pozorna i bardzo szkodliwa oszczędność. Znane powiedzenie przypisywane pisarzom takim jak Blaise Pascal i T. S. Eliot brzmi:

„Gdybym miał więcej czasu, napisałbym Ci krótszy list".

Powinniśmy zatem rozpocząć od oceny, jaką efektywność zespołu można uzyskać w danej sytuacji projektując wraz z nim proste, lubiane i akceptowalne „zasady gry". Bardzo ważne jest też zaplanowanie tzw. zakresu metodologii, czyli określenia które czynności i działania są mniej lub bardziej przez nią formalizowane, a w ramach których pozostawiamy członkom zespołów zupełną swobodę.

Uszyte na miarę... jest chętniej noszone! Miary odpowiednio dobranej metodologii

Kluczem do sukcesu jest, cytowana na początku, poszukiwana harmonia i prostota. Dla zespołu oznacza to głównie tyle, że żaden z jego członków nie czuje się przez proces ograniczany a jego wysiłki nie są głupie i nieużyteczne. Każdy posiada poczucie skupienia wyłącznie na czynnościach i produktach koniecznych, aby wygrać grę a następnie (ewentualnie) móc kontynuować jej kolejną część.

Scott Berkun w [17] podkreśla, że dobry proces będzie pożądany głównie przez tych, którym jest on rzeczywiście potrzebny. Jeśli wprowadzasz zmiany w metodologii, miej pewność, że odpowiadają one na rzeczywiste problemy ludzi czynnie zaangażowanych w projekt – wtedy nie powinieneś mieć większych problemów ze znalezieniem zwolenników do wypróbowania proponowanych zmian, o ile korzyści z wprowadzenia nowej procedury są realne.

Dobra metodologia i zawarty w niej proces posiadają mechanizmy samo-śledzenia. Dostarcza informacji nie tylko o tym, w jakim miejscu znajduje się obecnie projekt, ale także pozwala śledzić, które z kluczowych czynności metodologii są wykonywane, jaki jest ich stan, rezultat oraz nastawienie zespołu.

Którą wybrać z dróg?

W pewnym sensie wytwarzanie oprogramowania może być porównane do rozgrywki szachowej. Nie ma wątpliwości, że w grze w szachy, w danej chwili, najważniejsze jest aktualne ustawienie figur na szachownicy i zagrożenie, czy też problem, który należy rozwiązać. Najważniejszy jest następny ruch.

Jednak wytrawny szachista wie, pamięta i analizuje wszystkie kroki, które doprowadziły do aktualnego stanu na szachownicy. Jeśli ustawienie jest niekorzystne, w następnej partii mistrz będzie w naturalny sposób szukał rozwiązania, aby temu zapobiec. Jeśli natomiast pewna sekwencja ruchów pozwoliła na uzyskanie chwilowej przewagi, będzie ona zapamiętana, pielęgnowana i chętnie użyta następnym razem. Oczywiście mowa tu bardziej o strategii, typie ataku czy pewnych zasadach obrony, niż o dokładnym powtarzaniu ruchów. Każda partia jest inna, każdy dzień i przeciwnik jest inny. Jednak im większe mamy doświadczenie, tym szerszym spektrum potencjalnych rozwiązań i pomysłów będziemy dysponować w danej sytuacji. Jedno jest pewne, posiadanie jednej strategii pozwoli nam wygrać tylko nieliczną ilość bardzo specyficznych partii i to tylko przy tzw. „korzystnych wiatrach", kiedy wszystko układa się po naszej myśli.

Myślę też, iż żaden szachista nie zasiada do kolejnej partii bez pewnego planu, zamysłu na początek gry. Często analizuje i zna doświadczenie przeciwnika, swoje możliwości.

Przyjęcie planu startowego na wytwarzanie oprogramowania w nowym projekcie jest o wiele bardzie skomplikowane, niż inicjalna koncepcja na rozgrywkę szachową. Powodem jest to, że wytwarzanie oprogramowania to skończona gra zespołowa!

Cockburn [14] przez lata obserwacji, analizy i doświadczeń opracował „siedem zasad, które przydają się przy projektowaniu i ocenianiu metodologii:

1. Interaktywna komunikacja twarzą w twarz to najtańszy i najszybszy sposób komunikacji.
2. Zbędny ciężar metodologii jest kosztowny.
3. Większe zespoły potrzebują cięższych metodologii.
4. Większa obrzędowość potrzebna jest projektom o dużej krytyczności.
5. Zwiększenie informacji zwrotnych i komunikacji zmniejsza potrzebę stosowania pośrednich rezultatów.
6. Dyscyplina, umiejętności i zrozumienie przeciw procesom, formalizacji i dokumentacji.

7. Efektywność nie jest najważniejsza w działaniach niebędących wąskim gardłem.”

Celem niniejszego cyklu jest zrozumienie powyższych zasad i umiejętność ich zastosowania – czym będziemy zajmować się w kolejnych artykułach. Pomimo, że Alistair Cockburn był jednym z 17 zwolenników lekkich procesów tworzenia oprogramowania, którzy w 2001 roku ogłosili „Manifest zwinnego wytwarzania oprogramowania”, to dobrze rozumiane powyższe zasady nie zawsze prowadzą do zaprojektowania metodyki zwinnej.

„Pamiętaj, że sukcesu nie generuje żadna magia metodologii ale raczej to, że ludzie wreszcie dostają to czego potrzebują by wykonać swe zadania.” [14]

Literatura

[1] James Bach: „Good Enough Quality: Beyond the Buzzword”, IEEE Computer, Sierpień 1997

[2] Anna Bobkowska: „Inżynieria Oprogramowania”, Studium Podyplomowe Nowoczesne Metody Inżynierii Oprogramowania, edycja 2006-2007

[3] Grady Booch: „Leaving Kansas” IEEE Software. 15(1), Styczeń/Luty 1998

[4] Victor R. Basili: „The goal question metric approach”, (Advanced Computer Studies, Department of Computer Science, University Of Maryland)

[5] John Fodeh: „What's on Your Dashboard” Better Software, October 2007

[6] Janusz Górski: „Zarządzanie projektem informatycznym”, Studium Podyplomowe Nowoczesne Metody Inżynierii Oprogramowania, edycja 2006-2007

[7] Janusz Górski: „Inżynieria oprogramowania w projekcie informatycznym”, Zakład Nauczania Informatyki MIKOM, Warszawa 1999

[8] Jerzy Kaczmarek: „Metryki i zapewnianie jakości”, Studium Podyplomowe Nowoczesne Metody Inżynierii Oprogramowania, edycja 2006-2007

[9] Stephen H. Kan: „Metryki i modele w inżynierii jakości oprogramowania”, Wydawnictwo Naukowe PWN SA, 2006

[10] Adam Korczowski: „Zarządzanie ryzykiem w projektach informatycznych. Teoria i praktyka”, Wydawnictwo Helion 2010

[11] Per Kroll, Philippe Kruchten: „Rational Unified Process od strony praktycznej”, Wydawnictwo Naukowo-Techniczne, Warszawa 2007

[12] Philippe Kruchten: „Rational Unified Process od strony teoretycznej”, Wydawnictwo Naukowo-Techniczne, Warszawa 2007

[13] Zdzisław Szyjewski: „Metodyki zarządzania projektami informatycznymi”, Wyd. PLACET, Warszawa 2004

[14] Alistair Cockburn: „Agile Software Development. The Cooperative Game”, Second edition

[15] Dick Hamlet, Joe Maybee „Podstawy techniczne inżynierii oprogramowania”, Wydawnictwo Naukowo-Techniczne, Warszawa

[16] Process: the 4th dimension, **Humans and Technology**, *Technical Report TR 2003.02, Oct 4, 2003*

[17] Scott Berkun: „Sztuka zarządzania projektami”, Helion S.A., 2006