Łukasz D. Sienkiewicz

Koordynacja procesu wytwarzania oprogramowania Scrum z uwzględnieniem wpływu usług realizowanych przez dostawców zewnętrznych

Ekonomiczne Problemy Usług nr 106, 59-75

2013

ZESZYTY NAUKOWE UNIWERSYTETU SZCZECIŃSKIEGO

NR 781 EKONOMICZNE PROBLEMY USŁUG NR 106 2013

ŁUKASZ D. SIENKIEWICZ
Uniwersytet Ekonomiczny we Wrocławiu

KOORDYNACJA PROCESU WYTWARZANIA OPROGRAMOWANIA SCRUM Z UWZGLĘDNIENIEM WPŁYWU USŁUG REALIZOWANYCH PRZEZ DOSTAWCÓW ZEWNĘTRZNYCH

Wprowadzenie

Celem tego opracowania jest zaproponowanie modelu i uporządkowanych wskazówek wspomagających skoordynowanie nowo powstałego procesu na przykładzie dużej firmy w ściśle określonym środowisku. Niniejsze opracowanie stanowi poszerzenie rozwiązania sprawdzonego w praktyce i opisanego w publikacjach[1], a rozważania opisane w nim mogą usprawnić wytwarzanie oprogramowania w środowisku pracującym zgodnie z manifestem Agile w organizacji sieciowej używającej firm trzecich (tj. zewnętrznych i wewnętrznych dostawców usług).

Koordynację wspomnianych procesów udało się uzyskać dzięki:

– identyfikacji umiejscowienia omawianych procesów w środowisku pracującym zgodnie z wytycznymi manifestu Agile;

[1] L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum for Third Party Services and Network Organizations*, IEEE Digital Library, Szczecin 2011; L.D. Sienkiewicz, *Scrumban – the Kanban as an addition to Scrum software development method in a Network Organization*, „Business Informatics" 2012, nr 24 (2), s. 73–81; *idem*, *Collaboration between the Scrum and third party services in the network organization*, „Business Informatics" 2012, nr 23 (1), s. 59–66.

- zaproponowaniu holonicznego modelu[2], bazującego na podejściu proponowanym przez metodykę Scrum;
- użyciu modelu 3C[3] w celu identyfikacji relacji i oddziaływań występujących pomiędzy uczestnikami omawianych procesów;
- dodaniu kilku nowych reguł i usunięcie niektórych artefaktów z metodyki Scrum.

Nowo powstały proces jest kompilacją dwóch procesów: wytwarzania oprogramowania Scrum i procesu dostarczania wyników pracy przez firmy trzecie (*3rd party service providers*).

Pomimo iż powszechnie przyjmuje się, że metodologia Agile została zdefiniowana w celu realizacji projektów (głównie w obszarze szerokorozumianego IT), to metody wchodzące w skład wspomnianej metodologii (np. Scrum, ASD, XP) mają charakter procesowy. Z tego powodu w tym artykule Scrum jest traktowane jako zbiór wytycznych dla procesu wytwarzania oprogramowania, umiejscowiony w ściśle określonym środowisku pracującym zgodnie z wytycznymi manifestu Agile[4]. Omawiana organizacja ma charakter sieciowy, co determinuje wpływ wyników pracy usługodawców zewnętrznych na wyniki pracy procesu wytwarzania oprogramowania.

W dalszej części jako przykład usługi realizowanej przez firmę trzecią przedstawiono proces tłumaczenia tekstów z języka angielskiego na 40 innych języków ze szczególnym uwzględnieniem sposobu i formy dostarczania wyników. Niniejsze rozwiązanie prezentuje innowacyjne podejście do wytwarzania oprogramowania w organizacjach sieciowych. Zaproponowany model

[2] L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum…*; P. Mella, *The Holonic Perspective in Management and Manufacturing*, „International Management Review" 2009, nr 5 (1), s. 19–30; L.A. Maciaszek, *Modeling and Engineering Adaptive Complex Systems, Challenges in Conceptual Modeling*, Tutorials, Posters, Panels and Industrial Contributions to the 26th International Conference on Ceonceptual Modeling – ER, 2007, nr 83, s. 31–38.

[3] L.D. Sienkiewicz, *Collaboration between the Scrum…*; *Classes of Collaborative Networks, Encyclopedia of Networked and Virtual Organizations*, L.M. Camarinha-Matos, H. Afsarmanesh, No.1 A–F, red. G.D. Putnik, M.M. Cunha, IGI Global, New York 2008, s. 193–198; L. M. Camarinha-Matos, H. Afsarmanesh, *Collaborative Networks –Value creation in a knowledge society*, w: *Knowledge Enterprise, Intelligent Strategies in Product Design*, Manufacturing, and Management, Shanghai 2006; L.M. Camarinha-Matos, H. Afsarmanesh, N. Galeano, A. Molina, *Collaborative networked organizations – Concepts and practice in manufacturing enterprises*, „Computers & Industrial Engineering" 2009, nr 57 (1), s. 46–60.

[4] http://agilemanifesto.org (1.05.2013).

wyewoluował z metodyki Scrum, będącej reprezentantem metodyk zwinnych, i metody Kanban[5], wywodzącej się z koncepcji *Lean Software Development*[6].

1. Podobne prace

Od kiedy H. Takeuchi i I. Nonak przedstawili Scrum jako nowe podejście do wytwarzania oprogramowania[7], metoda ta stała się flagowym przykładem tego, że mała liczba ściśle określonych zasad wystarczy, aby podążać za pryncypiami manifestu Agile[8]. Niestety, ściśle określone role i zasady nie sprawdzają się w środowisku sieciowym. Z tego powodu w artykule tym zaproponowano dodatkową rolę (tj. dostawcę usług/firmę trzecią – S), jak i kilka reguł wspomagających adaptację metodyki Scrum do warunków panujących w organizacjach sieciowych.

H. Fuks i inni, a także C. Lucena i inni przedstawili podejście bazujące na modelu 3C (tj. kooperacji, koordynacji, komunikacji) w kontekście budowania systemów kolaboracyjnych[9]. Autorzy badali model 3C „poprzez szczegółowe analizy każdego z elementów, a następnie poprzez badanie studium przypadku aplikacji typu *learningware* zbudowanej na tym modelu. Zaprezentowana interpretacja modelu 3C wydaje się zgodna pod względem liczby i rodzajów zależności zachodzących pomiędzy rolami występującymi w metodyce Scrum a innymi interesariuszami („Uczestnicy otrzymują informacje zwrotne na temat ich działań od innych uczestników, co podnosi świadomość istotności pomiędzy uczestnikami procesu”[10]). W niniejszym opracowaniu zostały

[5] L.D. Sienkiewicz, *Scrumban – the Kanban as an addition*...

[6] M. Poppendick, T. Poppendick, *Lean Software Development: An Agile Toolkit*, Addison-Wesley Professional, 2003.

[7] T. Hirotaka, N. Ikujiro, *The New New Product Development Game*, „Harvard Business Review” 1986, nr 64.

[8] http://agilemanifesto.org (1.05.2013); K. Schwaber, *Agile Project management with Scrum*, USA: Microsoft Press, Redmond, Washington 2004; K. Schwaber, M. Beedle, *Agile Software Development with Scrum*, USA: Prentice Hall, Upper Saddle River, New Jersey 2002.

[9] H. Fuks, A. Raposo, M. Gerosa, C. Lucena, *Applying the 3C Model to Groupware Development*, „International Journal of Cooperative Information Systems” 2005, nr 14 (2–3), s. 99–328; *The 3C Collaboration Model. Encyclopedia of E-Collaboration*, C. Lucena, H. Fuks, A. Raposo, M. Gerosa, M. Pimentel, IGI Global, Texas 2008, s. 637–644.

[10] H. Fuks, A. Raposo, M. Gerosa, C. Lucena, *Applying the 3C Model*...

zidentyfikowane rodzaje współoddziaływania poszczególnych ról w sieci w ramach procesu kolaboracji[11].

Idea kolaboracji była również rozwijana w wielu publikacjach przez L.M. Camarinha-Matos i H. Afsarmanesh[12]. Opisali oni koncept nawiązujący do środowiska produkcyjnego, gdzie dokonano klasyfikacji kolaborujących organizacji sieciowych. Tak przedstawiony podział pozwala na łatwe zidentyfikowanie zależności występujących między bytami w organizacji sieciowej (w tym dokumencie rozpatrywanej jako organizacja holoniczna). W przeszłości również P. Mella badał holoniczną formę organizacji i zarządzania[13]. Zbadał sześć różnych przykładów holonicznych sieci występujących w środowiskach produkcyjnych. W nawiązaniu do tych badań w niniejszym opracowaniu model przedstawiający uczestników metodyki Scrum w organizacji sieciowej został przedstawiony jako sieć holoniczna, która „składa się z autonomicznych firm ulokowanych w różnych miejscach, scharakteryzowanych poprzez różne role i operacje i połączonych przez holoniczną sieć, prawdziwą lub wirtualną. Zazwyczaj zorientowaną na osiąganie wspólnych celów poprzez współdzielenie zasobów, informacji i kluczowych kompetencji[14]".

D.S. Hovorka i K.R. Larsen zaprezentowali badania nad wpływem organizacji sieciowej na możliwości wprowadzenia praktyk proponowanych w manifeście Agile do organizacji[15]. W badaniach użyli specjalnie zaprojektowanego studium przypadków, którego celem było „zbadanie interakcji pomiędzy strukturą sieciową, społecznym oddziaływaniem, podobieństwem organizacji (homofilią) i zdolnościami absorpcji w dwóch dużych organizacjach IT"[16]. Niestety, zaproponowany przez nich model APM (*Adoption Practice Model*) wymaga międzyorganizacyjnej sieci umożliwiającej wprowadzenie metod zwinnych.

---

[11] C. Lucena, H. Fuks, A. Raposo, M. Gerosa, M. Pimentel, *The 3C Collaboration Model*...

[12] L.M. Camarinha-Matos, H. Afsarmanesh, *Classes of Collaborative Networks*...; L.M. Camarinha-Matos, H. Afsarmanesh, *Collaborative Networks*...; L. M. Camarinha-Matos, H. Afsarmanesh, N. Galeano, A. Molina, *Collaborative networked organizations*...

[13] P. Mella, *The holonic Perspective in Management*...

[14] *Ibidem*.

[15] D.S. Hovorka, K.R. Larsen, *Enabling agile adoption practics through network organization*, „European Journal of Information Systems" – *Including a special section on business agility and diffusion of information technology 2006*, nr 15 (2), s. 159–168.

[16] *Ibidem*.

W oparciu o badania i modele zaproponowane przez wspomnianych badaczy w niniejszym dokumencie została przedstawiona bardziej szczegółowa analiza wybranej metodyki zwinnej, nie skupiająca się tylko na przedstawieniu jej jako zbioru dobrych praktyk. Zaproponowana szczegółowa analiza jednej wybranej metody wytwarzania oprogramowania (tj. Scrum) i usługi dostarczanej przez dostawcę zewnętrznego (na wybranym przykładzie tłumaczeń, jako usługi oferowanej przez firmę trzecią) ma na celu umożliwienie lepszego dopasowania do realiów organizacji sieciowej.

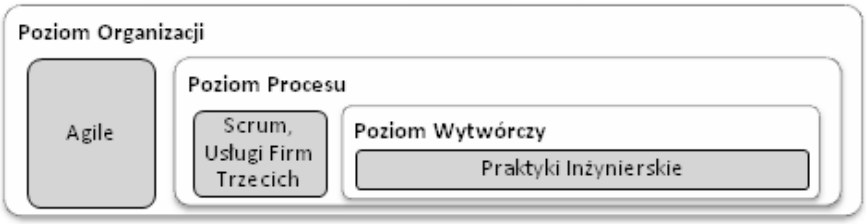
2. Identyfikacja procesów podlegających koordynacji

W kolejnych punktach zostały przedstawione w sposób szczegółowy procesy podlegające koordynacji. Takie przedstawienie procesów ma na celu formalną identyfikację zakresu ich działań, wyróżnienie ich podstawowych celów, przedstawienie formy dostarczanych wyników oraz identyfikację interesariuszy.

2.1. Wytwarzanie oprogramowania zgodnie z wytycznymi metodyki Scrum

Ponieważ metodyka Scrum została już wielokrotnie opisana[17], w opracowaniu tym zostały przytoczone tylko te jej aspekty, które mogą mieć znaczący wpływ na koordynację omawianych procesów.

[17] http://agilemanifesto.org…; T. Hirotaka, N. Ikujiro, *The New Product Development*...; K. Schwaber, *Agile Project*...; K. Schwaber, M. Beedle, *Agile Software*...; *Agile Management for Software Engineering*, D.J. Anderson, Upper Saddle River, New Jersey 07458: Prentice Hall Professional 2004; A. Cockburn, *Selecting a project's methodology*, „IEEE Software" 2000, nr 4 (17), s. 64–71; P. Deemer, G. Benefield, *The Scrum Primer: An Introduction to Agile Project Management with Scrum*, Good Agile version 1.04, 2007; V. Guntamukkala, J.H. Wen, M. Tarn, *An empirical study of selecting software development life cycle models*, „Human System Management" 2006, nr 25 (4), s. 268–278.



Rys. 1. Scrum i usługi firm trzecich w środowisku pracującym zgodnie z wytycznymi manifestu Agile

Źródło: opracowanie własne na podstawie: L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum…*

W celu poprawnego ujęcia umiejscowienia metodyki Scrum w organizacji istotne jest prawidłowe zidentyfikowanie obszarów jej działania, odwołując się do koncepcji *Three Level Framework*[18], zaadaptowanej do organizacji o charakterze zgodnym z wytycznymi manifestu Agile. Poprzez środowisko Agile na poziomie organizacyjnym (*Organizational Level*) rozumie się wszystkie te działania, które nie zostały ujęte w wytycznych metodyki Scrum (np. HR, finanse, księgowość, *top management* itp.), a są niezbędne dla istnienia organizacji. Scrum, jako metoda wytwarzania oprogramowania została umiejscowiona na poziomie procesowym (*Process Level*[19]), co przedstawiono na rysunku 1. Poziom ten reprezentuje zbiór powtarzalnych zasad i artefaktów (np. cykle, dzienniki, role itp.) używany przez zespoły wytwarzające oprogramowanie. Trzeci poziom, nazwany tutaj poziomem wytwórczym (*Engineering Practices*) ilustruje wyodrębnione z poziomu procesowego techniki i metody używane przez inżynierów w trakcie codziennej pracy (np. programowanie parami, przegląd zmian, sesje demonstracyjne, CI, TDD itp.), bez których nie można osiągnąć celów procesowych z wyższego poziomu[20].

Ponieważ Scrum jest metodyką wywodzącą się z metodologii Agile, za wartość najistotniejszą i praktycznie nienegocjowalną z punktu widzenia procesu przyjmuje się jakość produktu (tutaj sprecyzowaną jako niezawodność zarówno zaimplementowanych algorytmów, jak i wizualnej części aplikacji).

[18] G.A. Rummler, A.P. Brache, *Improving Performance – How to Manage the White Space in the Organization Chart*, Jossey Bass Inc., San Francisco 1995.

[19] L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum…*

[20] L.D. Sienkiewicz, L.A. Maciaszek, *Collaboration between the Scrum…*

2.2. Proces tłumaczenia i dostarczania wyników pracy tłumaczy

Opisywany proces jest umiejscowiony w dużej międzynarodowej organizacji, o charakterze sieciowym, gdzie do realizacji zamierzonych działań wykorzystuje się pośrednio i bezpośrednio wyniki pracy tłumaczy ulokowanych w różnych strefach czasowych. Całość realizowana jest jako proces wewnętrzny, koordynowany przez przedstawiciela firmy wykonującej zlecenie (tj. lidera zlecenia). Proces jest wykonywany zgodnie z wytycznymi usługobiorcy, a jego celem bezpośrednim jest dostarczenie usługi tłumaczenia (na 40 języków) tekstów napisanych w języku angielskim z uwzględnieniem kontekstu ich użycia, a dostarczonych w ściśle sprecyzowanym formacie (tj. pliki *.resx). Usługa jest realizowana przez międzynarodową firmę zrzeszającą tłumaczy z całego świata. Każde zamówienie jest realizowane jako „krótki projekt", gdzie podwykonawcami poszczególnych tłumaczeń są wybrani tłumacze wyselekcjonowani z bazy zrzeszonych tłumaczy.

Ponieważ proces ten stanowi uzupełnienie do istniejącego procesu wytwarzania oprogramowania, w poniższym opracowaniu skupiono się tylko na tych elementach, które mają kluczowe znaczenie dla skoordynowania obu procesów.

Cały przebieg procesu przedstawia się następująco:

- Klient zgłasza zapotrzebowanie na wykonanie usługi tłumaczenia. Razem ze zgłoszeniem dostarczany jest oryginalny tekst w języku angielskim, lista języków, na które ma być dokonane tłumaczenie, lista słów niepodlegających tłumaczeniu oraz opis kontekstu użycia tłumaczonych tekstów.
- Firma świadcząca usługę dokonuje wyceny zlecenia wraz z szacowanym czasem dostarczenia – suma wartości tłumaczenia na poszczególne języki jest liczona według następującego wzoru: liczba słów razy stała stawka za 250 słów wyznaczana dla każdego języka osobno.
- Klient dokonuje akceptacji kosztów.
- Firma świadcząca usługę zleca tłumaczenia wybranym tłumaczom.
- Firma świadcząca usługę odbiera wyniki od poszczególnych tłumaczy.

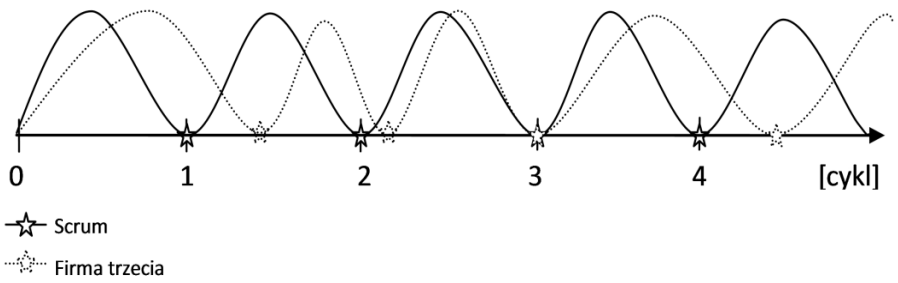
– Firma świadcząca usługę dostarcza wyniki do zleceniodawcy lub wskazanemu odbiorcy (np. zespołowi wykonawczemu).
– W procesie wyodrębnione zostały trzy role: klient będący usługodawcą, firma wykonująca zlecenie tłumaczenia, czyli usługobiorca, oraz tłumacze będący podwykonawcami usługobiorcy.

3. Wytwarzanie oprogramowania w środowisku pracującym zgodnie z wytycznymi manifestu Agile z uwzględnieniem wyników pracy firm trzecich

Powyżej zostały omówione procesy na bazie, których kreowany jest proces wytwarzania oprogramowania umiejscowiony w międzynarodowej korporacji z obszaru IT (urządzenia mobilne), o charakterze sieciowym. Firma, w której wdrożony został skoordynowany proces, zatrudnia około 17 tys. pracowników, ulokowanych w różnych krajach na wszystkich kontynentach, między innymi w Polsce. Organizacja ta na co dzień korzysta z usług dostarczanych przez firmy trzecie lub usługodawców wewnętrznych (firmy trzecie w sieci).

Omawiana firma zajmuje się głównie *outsourcingiem* zasobów ludzkich (tj. programistów, testerów itp.) i jest silnie ukierunkowana na korzystanie z wyników pracy dostawców usług wewnętrznych (tj. dział IT, księgowość, inne działy/komórki), jak i zewnętrznych (tj. serwisy pogwarancyjne, biura tłumaczeń, zewnętrzni eksperci itp.). Pracownicy firmy wykonującej zlecenie są ulokowani w placówce w Polsce (tj. zespół wykonujący pracę i koordynator procesu). Przedstawiciel klienta jest ulokowany w Finlandii i kontaktuje się z Polską poprzez narzędzie Microsoft Communicator. Do codziennej pracy zespół wykonawczy wykorzystuje IDE (*Integrated Development Environment*) firmy Microsoft. Cały zakres pracy jest przechowywany we współdzielonym narzędziu Scrumworks.

Wytwarzanie oprogramowania odbywa się w sposób cykliczny przyrostowy. Do kolizji pomiędzy procesami dochodzi zawsze wtedy, gdy wyniki tłumaczeń (tj. wszystkie 40 języków) nie zostaną dostarczone dokładnie przed kolejnym planowaniem następnego cyklu (*Sprint Planning*), co przedstawiono za pomocą symbolu ê na rysunku 2.



Rys. 2. Dostarczanie wyników pracy procesu wytwarzania oprogramowania i tłumaczeń

Źródło: opracowanie własne.

Celem bezpośrednim koordynowanego procesu jest poprawienie niezawodności i bezpieczeństwa użytkowania telefonów komórkowych ze szczególnym uwzględnieniem dostarczenia usługi aktualizacji oprogramowania i usuwania danych osobistych przechowywanych w pamięci urządzenia.

3.1. Ograniczenia warunkujące realizację procesu

Z racji tego, że Scrum jest metodyką wywodzącą się z metodologii Agile, za wartość najistotniejszą i praktycznie nienegocjowalną z punktu widzenia procesu przyjmuje się jakość dostarczanego produktu (tutaj sprecyzowaną jako niezawodność zarówno algorytmów działania aplikacji, jak i interfejsu użytkownika).

Przyjęte podejście determinuje wysoki poziom jakości dostarczanego produktu i jednocześnie dopuszcza elastyczność zakresu (tj. możliwe są ustępstwa co do liczby elementów dostarczonych w poszczególnych cyklach).

W omawianym przykładzie czas kolejnych dostarczeń produktu jest zdefiniowany jako dwa tygodnie, a koszt i budżet poniesiony na realizację procesu wytwarzania oprogramowania, podobnie jak i koszt wykonania tłumaczeń, nie stanowią przedmiotu dalszych rozważań, gdyż są przeniesione na inny poziom organizacji.

3.2. Wskazanie uczestników i relacji występujących między nimi

Gdy rozpatruje się umiejscowienie procesu w organizacji sieciowej, niezbędne jest poprawne zdefiniowanie zależności występujących pomiędzy elementami sieci. Z tego powodu do dalszej analizy wszystkie byty (tj. uczestnicy) procesu będą identyfikowane jako całość i część, czyli jako holony, a struktury holonów wraz z ich hierarchią będą nazywane holarchiami, zgodnie z teorią holonów opracowaną przez A. Koestlera[21].

Ponieważ teoria holonów, holarchii i holizm są opisane w literaturze[22], w dalszej części tego opracowania ich szczegółowy opis został pominięty.

W artykule[23] został przedstawiony trójwarstwowy model holarchii, gdzie role zdefiniowane w procesie wytwarzania oprogramowania zgodnie z wytycznymi metodyki Scrum są umiejscowione w pierwszych dwóch warstwach, a nowo wyodrębniona rola dostawcy usług została umiejscowiona w warstwie trzeciej. Role dodatkowe, takie jak inni interesariusze czy zarząd, zostały pominięte, gdyż nie odgrywają na tym poziomie organizacji znaczącej roli w całym procesie.

Ponieważ warstwy w holarchii są autonomiczne, umożliwia im to dostosowywanie się do nowych okoliczności i zmian w środowisku, z tego powodu doskonale nadają się do przedstawienia zależności występujących pomiędzy uczestnikami procesu (tj. holonami) i tym samym pozwalają w łatwy sposób zidentyfikować te obszary metodyki Scrum, które powinny zostać poprawnie skoordynowane lub zmienione w celu dostarczenia optymalnego rozwiązania.

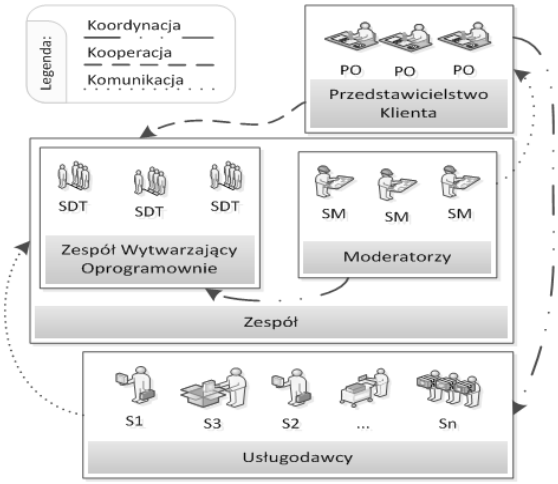
W zaproponowanym modelu (rys. 3) wszystkie zależności pomiędzy warstwami są skierowane w dół, a przekazywanie wyników pracy czy informacji odbywa się drogą komunikacji zwrotnej. Takie rozgraniczenie pozwala efektywnie sterować przepływem pracy i zapobiega powstawaniu cyklów, co znacznie zmniejsza złożoność prezentowanego modelu. Szczegółowe omówienie kolejnych warstw nie stanowi przedmiotu rozważań, dlatego zostało pominięte. Szczegółowy opis wraz wynikami praktycznego zastosowania

[21] A. Koestler, *The Ghost in the Machine*, Penguin Group, London 1967.

[22] P. Mella, *The holonic Perspective*...; A. Koestler, The Ghost…; F. Capra, *The Turning Point. Science Society and the Rising Culture*, Flamingo, New York 1982, s. 27; L.A. Maciaszek, *Architecture – Centric Software Quality Management, Web Information Systems and Technologies*, WEBIST 2008, LNBIP 18 2009, Springer-Verlag, s. 11–26; K. Wilber, *A brief history of everything*, Shambhala Publications, Boston 2000.

[23] L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum*...

wspomnianego modelu został zaprezentowany w dokumencie *Adapting Scrum for Third Party Services and Network Organizations*[24].



Rys. 3. Model relacji występujących między uczestnikami procesu wytwarzania oprogramowania w organizacji stosującej zwinne metodyki

Źródło: opracowanie własne na podstawie: L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum*...

Skróty użyte na rysunku 3 oznaczają: PO (*Product Owner*) – przedstawiciel klienta, SM (*Scrum Master*) – koordynator procesu, SDT (*Software Development Team*) – zespół wytwarzający oprogramowanie, S (*Service Provider*) – dostawca usług.

Na modelu przedstawionym na rysunku 3 wyodrębnionych zostało kilka typów relacji występujących pomiędzy uczestnikami procesu (tj. kooperacja, koordynacja, komunikacja), które zgodnie z „3C Collaboration model" są składowymi kolaboracji[25]. Rozróżnienie typów relacji pozwala nam lepiej zrozumieć formę współpracy i ograniczenia z nią związane.

[24] L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum*...

[25] L.M. Camarinha-Matos, H. Afsarmanesh, *Encyclopedia of Networked*...; L.M. Camarinha-Matos, H. Afsarmanesh, *Collaborative Networks*...; L.M. Camarinha-Matos, H. Afsarmanesh, N. Galeano, A. Molina, *Collaborative networked*...; H. Fuks, A. Raposo, M. Gerosa, C. Lucena, *Applying the 3C Model*...

Tak skonstruowany model wspiera budowanie zaufania i multikulturowość. Wpływ S na wyniki pracy zespołu jest znaczący i nie powinien być pomijany (np. terminy zaproponowane przez S są zupełnie nieskoordynowane z cyklami pracy SDT). Może to powodować znaczące opóźnienia w pracy pozostałych uczestników.

3.3. Identyfikacja działań i kluczowych elementów skoordynowanego procesu

Analiza modelu przedstawionego w punkcie 3.2 i odniesienie do modelu „3C Collaboration" relacji pomiędzy uczestnikami wymuszają dokonanie pewnych zmian i dostosowań oryginalnej metodyki Scrum. W dalszej części artykułu zostanie zwrócona uwaga na te aspekty, które diametralnie różnią się od tradycyjnego podejścia proponowanego w ramach metodyki Scrum:

- Wykonalność zadań zamiast estymacji czasowych: z powodu nieregularnych dostaw od dostawców usług sugeruje się, aby porzucić estymowanie zadań na nadchodzący cykl, a w zamian skupić się na priorytetyzowaniu zadań. Dzięki zmianie podejścia z planowania na priorytetyzowanie diametralnie zmniejsza się liczba niedotrzymanych zobowiązań (na płaszczyźnie PO – SDT).
- Spotkanie raportujące zamiast spotkania *Sprint Review*: ponieważ regularne spotkania określone przez metodykę Scrum angażują bardzo dużą ilość zasobów ludzkich, sugeruje się, aby w spotkaniu raportującym uczestniczyli tylko: przedstawiciel klienta (tj. PO) i przedstawiciel zespołu (np. Scrum Master lub członek zespołu). Takie spotkanie powinno się odbywać częściej niż regularne *Sprint Review*, np. co tydzień. Tak zmodyfikowane podejście zapewnia lepszy przepływ informacji pomiędzy zespołem a przedstawicielem klienta.
- Planowanie na żądanie zamiast regularnego planowania cyklu (*Sprint Planning*): ponieważ zaleca się porzucenie estymacji czasowych na rzecz priorytetów, tym samym sugeruje się, aby zmniejszyć liczbę spotkań planistycznych do minimum, tym samym traktując je jako „wydarzenie na żądanie" (np. gdy klient potrzebuje pomocy przy zorganizowaniu dziennika zadań).

Zaproponowane zmiany są podyktowane zaadaptowaniem holonicznego modelu. Dzięki takiemu podejściu udało się zredukować zależności pomiędzy poszczególnymi warstwami (nie ma cykli informacyjnych oraz została ustabilizowana liczba zadań zaplanowanych na cykl itd.). Takie ujęcie zależności sprawia, że zespół wytwarzający oprogramowanie, tak samo jak moderatorzy procesu, jest w pełni zaangażowany w dostarczanie wyników pracy.

Ponieważ oryginalnie metodyka Scrum ma tylko jedną metrykę, a dokładnie wskaźnik prędkości dostarczania wyników pracy (velocity), do zaproponowanego modelu zostały opracowane dodatkowe wskaźniki, których użyteczność została sprawdzona w publikacji L.D. Sienkiewicza i L.A. Maciaszka[26]:

– Niezawodność: mierzy, czy zespół dostarcza to, do czego się zobligował. Wyznacza się poprzez obliczenie stosunku zadań, co do których zespół się zobowiązał (c_i), do rzeczywiście wykonanych zadań (d_i). Wartość tego wskaźnika może być przedstawiona jako procentowy stosunek skuteczności wyliczanej na każdy cykl (R_i):

$$R_i = \frac{c_i}{d_i} * 100\%$$

– Produktywność: może służyć do mierzenia prędkości projektu. Oblicza się poprzez zsumowanie liczby naprawionych błędów (b_i) i nowo zaimplementowanych wymagań (s_i). Wartość produktywności (P_i) powinna być wyliczana w każdym cyklu:

$$P_i = b_i + s_i$$

– Skuteczność: może służyć do mierzenia skuteczności usługi testowania. Metryka (E_i) pozwala zmierzyć łączny procent błędów, które wyciekły do klienta (3). Wyliczona wartość wskaźnika pozwala monitorować skuteczność wewnętrznej usługi testowania (a_i) poprzez wyliczenie stosunku pomiędzy wszystkimi znalezionymi błędami do błędów znalezionych przez firmę zewnętrzną, trzecią (e_i).

$$E_i = \frac{a_i - e_i}{a_i} * 100\%$$

[26] L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum*...

Wszystkie zaproponowane wskaźniki powinny być wyliczane dla każdego z cykli osobno. Monitorowanie odchyleń wspomnianych wskaźników jest kluczowe i odpowiednie reagowanie na zmiany ich wartości przyczynia się do wzrostu zadowolenia klienta, co potwierdzają wyniki[27].

Podsumowanie

W ciągu ostatnich 30 lat zaproponowano wiele różnorodnych podejść do wytwarzania oprogramowania, począwszy od *Code and Fix*[28], poprzez model kaskadowy, spiralny, wczesne prototypowanie, programowanie ekstremalne, Scrum itd. Ponieważ wymienione pozycje są przedstawicielami trzech różnych podejść: *heavyweight*, *middleweight* i *lightweight*, trudno jest jednoznacznie stwierdzić, czy jedno podejście jest lepsze od drugiego. Historia pokazuje, że inne podejście jest dobre dla jednego projektu, a zupełnie inne dla innego. Jednakże na bazie wyników badań[29] można przyjąć, że jednym z najskuteczniejszych rozwiązań dla projektów rozpoczynających się od podstaw (*from scratch*) są zwinne metody, takie jak: Scrum, eXtreme Programming itd.[30], których adaptacyjność (zwinność) jest bardzo duża. Ponieważ metodyka Scrum zastosowana zgodnie z wszystkimi wytycznymi nie sprawdza się, gdy wyniki pracy jednego zespołu zależą od wyników pracy firm trzecich (np. biur tłumaczeń, usług testowania, wsparcia technicznego, wsparcia IT, itd.), ważne jest, aby tak uelastycznić metodę, by sprawdzała się w dynamicznie działających organizacjach sieciowych. Koordynacja procesów omówiona w tym dokumencie jest dobrym tego przykładem.

---

[27] L.D. Sienkiewicz, L.A. Maciaszek, *Adapting Scrum*...

[28] V. Guntamukkala, J.H. Wen, M. Tarn, *An empirical study*...; W. Royce, *Managing the development of large software systems*, „IEEE Computer Society Press", Los Angeles 1970, s. 1–9.

[29] K. Schwaber, *Agile Project*...; K. Schwaber, M. Beedle, *Agile Software*...; A. Cockburn, *Selecting a project's methodology*...; J. Nandhakumar, J. Avision, *The fiction of methodological development: A field study of information system development*, „Information Technology and People" 1999, nr 2 (12), s. 176–191; M. Lindvall, V. Basili, B. Boehm, P. Costa, K. Dangel, F. Shull, R. Tesoriero, L.A. Wiliams, M.V. Zelkowitz, *Empirical Findings in Agile methods*, Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile methods – XP/Agile Universe 2002, Springer-Verlag, London 2002, s. 81–92.

[30] K. Schwaber, *Agile Project*...; K. Schwaber, M. Beedle, *Agile Software*...; A. Cockburn, *Selecting a project's methodology*...

Można zaobserwować, że zwinne metodyki są zazwyczaj przedstawiane jako zbiór, w którego skład wchodzą zasady i pryncypia mające wspomóc interpretację manifestu Agile[31]. Z tego powodu jest bardzo trudno jednoznacznie zdefiniować, czym jest metodologia Agile, a tym samym trudno jest jednoznacznie stwierdzić, jak jej metody zaadaptować do organizacji sieciowej. Dlatego też rozważania zawarte w tym artykule mogą stanowić swoisty przyczynek do dalszych badań i analiz.

Literatura

Wilber K., *A brief history of everything*, Shambhala Publications, Boston 2000.

Anderson D.J., *Agile Management for Software Engineering*, Prentice Hall Professional, New Jersey 2004.

Schwaber K., *Agile Project Management with Scrum*, Microsoft Press, Redmond, Washington 2004.

Schwaber K., Beedle M., *Agile Software Development with Scrum*, Prentice Hall, New Jersey 2002.

Cockburn A., *Selecting a project's methodology*, „IEEE Software" 2000, No. 4 (17).

Camarinha-Matos L.M., Afsarmanesh H., *Classes of Collaborative Networks. Encyclopedia of Networked and Virtual Organizations*, No. 1 A–F, eds. G.D. Putnik, M.M. Cunha, IGI Global, New Yor 2008.

Hovorka D.S., Larsen K.R., *Enabling agile adoption practics through network organization*, „European Journal of Information Systems – Including a special section on business agility and diffusion of information technology" 2006, No. 15 (2).

Rummler G.A., Brache A.P., *Improving Performance – How to Manage the White Space in the Organization Chart*, Jossey Bass Inc., San Francisco 1995.

http://agilemanifesto.org.

Fuks H., Raposo A., Gerosa M., Lucena C., *Applying the 3C Model to Groupware Development*, „International Journal of Cooperative Information Systems" 2005, No. 14 (2–3).

Nandhakumar J., Avision J., *The fiction of methodological development: A field study of information system development*, „Information Technology and People" 1999, No. 2 (12).

Lean Software Development: An Agile Toolkit, M. Poppendick, T. Poppendick, Addison-Wesley Professional, 2003.

[31] http://agilemanifesto.org…

Maciaszek L.A., *Modeling and Engineering Adaptive Complex Systems, Challenges in Conceptual Modeling*, Tutorials, Posters, Panels and Industrial Contributions to the 26th International Conference on Ceonceptual Modeling – ER 2007, No. 83.

Maciaszek L.A., *Architecture-Centric Software Quality management*, Web Information Systems and Technologies*, WEBIST 2008, LNBIP 18*, Springer-Verflag 2009.

Sienkiewicz L.D., *Collaboration between the Scrum and third party services in the network organization*, „Business Informatics" 2012, No. 23 (1).

Sienkiewicz L.D., Maciaszek L.A., *Adapting Scrum for Third Party Services and Network Organizations*, IEEE Digital Library, Szczecin 2011.

Sienkiewicz L.D., *Scrumban – the Kanban as an addition to Scrum software development method in a Network Organization*, „Business Informatics" 2012, No. 24 (2).

Camarinha-Matos L.M., Afsarmanesh H., *Collaborative Networks – Value creation in a knowledge society, in Knowledge Enterprise*, Intelligent Strategies in Product Design, Manufacturing, and Management, Shanghai 2006.

Camarinha-Matos L.M., Afsarmanesh H., Galeano N., Molina A., *Collaborative networked organizations – Concepts and practice in manufacturing enterprises*, „Computers & Industrial Engineering" 2009, No. 57 (1).

Lindvall M., Basili V., Boehm B., Costa P., Dangel K., Shull F., Tesoriero R., Wiliams L.A., Zelkowitz M.V., *Empirical Findings in Agile methods,* Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile methods – XP/Agile Universe 2002, Springer-Verlag, London 2002.

Deemer P., Benefield G., *The Scrum Primer: An Introduction to Agile Project Management with Scrum*, Good Agile version 1.04.2007.

Mella P., *The holonic Perspective in Management and Manufacturing*, „International Management Review" 2009, No. 5 (1).

Koestler A., *The Ghost in the Machine*, Penguin Group, London 1967.

Capra F., *The Turning Point. Science Society, and the Rising Culture*, Flamingo, New York 1982.

Lucena C., Fuks H., Raposo A., Gerosa M., Pimentel M., *The 3C Collaboration Model, Encyclopedia of E-Collaboration*, IGI Global, Texas 2008.

Hirotaka T., Ikujiro N., *The New New Product Development Game*, „Harvard Business Review" 1986, No. 64.

Guntamukkala V., Wen J.H., Tarn M., *An empirical study of selecting software development life cycle models*, „Human System Management" 2006, No. 25 (4).

Royce W., *Managing the development of large software systems*, IEEE Computer Society Press, Los Angeles 1970.

COORDINATION OF SCRUM SOFTWARE DEVELOPMENT PROCESS WITH TAKING INTO ACCOUNT THE INFLUENCE OF THE OUTCOMES PROVIDED BY THIRD PARTY SERVICE PROVIDERS

Summary

Number of scientific publications and press releases demonstrate that organizations are adopting the Scrum software development method with success in almost all areas. Nevertheless, traditional Scrum method is not sufficient for managing work in Network Organizations where third party service providers may know nothing about the Scrum. This paper describes the findings of a field study that explores the Scrum in Network Organizations. As an example has been described the process of coordination (Logistic point of view) the Scrum software development process and non-Scrum process of delivering the outcomes of third party vendor (i.e. translations from translators). The identification of relationship between both processes, allow to propose new role (i.e. Service Provider) in addition to core Scrum roles and some extensions and changes in core Scrum artifacts, what helps in adapting the Scrum method to work in Network Organization where changes and high competition are the cornerstone of the entire process. The solution proposed in this paper may be used with success in other cases when Scrum team goals depend on outcomes of non-Scrum process.

Translated by Łukasz D. Sienkiewicz