

NR 1/2014

# REQ MAGAZYN

KWARTALNIK

MARIAZ INŻYNIERII  
WYMAGAŃ I MARKETINGU

NA STYKU BIZNESU I IT  
WYMAGANIA W AGILE



TRUDNI INTERESARIUSZE W PROCESIE  
DEFINIOWANIA WYMAGAŃ

WIEDZA I DOŚWIADCZENIE, CZYLI NA  
CZYM WARTO BUDOWAĆ KARIERĘ

ISSN 1234-5678  
1234567 1234567

INŻYNIERIA

WYMAGAŃ

INŻYNIERIA

OPROGRAMOWANIA

ZARZĄDZANIE

PROJEKTAMI

# Szanowni Państwo

- Przedstawiamy Państwu pierwszy numer REQ Magazyn - kwartalnika poświęconego inżynierii wymagań, inżynierii oprogramowania oraz zarządzania projektami.
- Magazyn jest odpowiedzią na liczne pytania stawiane podczas cyklicznych spotkań analityków, programistów, kierowników projektów oraz coraz częściej Klientów, którzy chcą zrozumieć cały proces wytwarzania oprogramowania.

Zadaniem magazynu jest poszerzanie wiedzy połączoną z propagowaniem dobrych praktyk w każdym z wcześniej wymienionych obszarów. Kładziemy największy nacisk na rolę analizy biznesowej i systemowej oraz jej uczestnictwo na każdym z etapów wytwarzania produktu końcowego. Mamy nadzieję, że dzięki temu jakość produktów oferowanych Klientom będzie coraz lepsza.

W pierwszym numerze zastanawiamy się jak rozmawiać z tzw. „trudnym Klientem”, jak wygląda pierwszy etap analizy oraz czy udaje się skutecznie połączyć inżynierię wymagań z marketingiem. Poszukujemy odpowiedzi na odwieczne pytania: „Co jest ważniejsze: certyfikaty czy doświadczenie?” oraz „Czy biznes i IT to sprzymierzeńcy czy skazane na siebie strony w projekcie”. Z uwagi na coraz większą popularność metod zwinnych w projektach pokazujemy co oznacza dla całego zespołu projektowego wdrożenie takiego podejścia oraz jak odnaleźć swoje miejsce w nowej „zwinnej” rzeczywistości.

## Zapraszam do lektury!

Monika Perendyk

Prezes Stowarzyszenia Inżynierii Wymagań

Redaktor Naczelnego



REDAKCJA

## Opracowanie i redakcja:

**Redaktor naczelna:  
Monika Perendyk**

**Zespół redakcyjny:**  
Włodzimierz Dąbrowa  
Artur Kiełbowicz  
Rafał Stańczak

## **Projekt graficzny i skład:**

**Prawa autorskie**  
Wszystkie opublikowane artykuły są objęte prawem autorskim autora lub przedsiębiorstwa. Wykorzystywanie w innych publikacjach, kopiowanie lub modyfikowanie zawartości artykułu bez pisemnego upoważnienia autora jest zabronione.

**Facebook**  
<http://www.facebook.com/reqmaqazy>

**Reklama**  
czasopismo@wymagania.org.pl

**Współpraca**  
Osoby zainteresowane współpracą w zakresie publikacji prosimy o kontakt: [czasopismo@wymagania.org.pl](mailto:czasopismo@wymagania.org.pl)



STOWARZYSZEN  
INŻYNIER  
WYMAGA



wymagania.org.pl

# Spis

## Treści

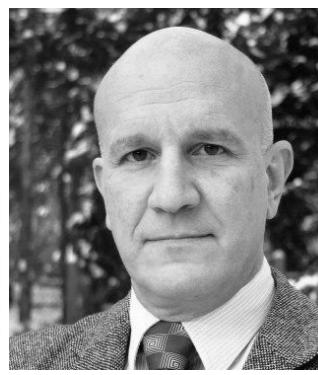
### Działy stałe

- 2 Słowo wstępu
- Inżynieria Wymagań**  
Bogdan Bereza  
Rynkowa inżynieria wymagań:  
długo wyczekiwany mariąz inżynierii wymagań i marketingu
- 6 Bożena Rumak  
Trudni interesariusze w procesie definiowania wymagań
- 10 Hanna Wesołowska  
Wiedza czy doświadczenie, czyli na czym warto budować karierę
- Inżynieria Oprogramowania**  
Ewa Brzeska  
18 Na styku biznesu i IT:  
od potrzeby biznesowej do gotowego produktu – rozważania w kontekście  
inżynierii oprogramowania
- Zarządzanie projektami**  
Krystian Kaczor  
24 Wymagania w Agile

### Działy ruchome

- 32 Relacja z wydarzenia:  
II Kongres Profesjonalistów IT





Bogdan  
Bereza

# Inżynieria Wymagań

## Rynkowa inżynieria wymagań

Długo wyczekiwany mariaż inżynierii wymagań i marketingu

### Pozycjonowanie tematu: IT a marketing

**N**a początek – konieczne wyjaśnienie terminologiczne. „Rynkowa inżynieria wymagań”, czyli MDRE (Market-Driven Requirements Engineering) – to nazwa, która łatwo może wprowadzić w błąd. Nie chodzi w niej tylko o to, aby inżynieria wymagań brała pod uwagę potrzeby klientów, rynku – bo to jest zasada obowiązująca uniwersalnie.

Rynkowa inżynieria wymagań dotyczy rynku otwartego, na którym istnieje wielu potencjalnych klientów, ale żaden z nich jeszcze nie kupił ani nie zamówił naszego produktu. Pozyskiwanie wymagań nie może polegać w tej sytuacji na wyypytywaniu, wszystko jedno jakimi metodami, konkretnych organizacji, czy osób, bo nie wiadomo, kogo konkretnie pytać, kto ostatecznie kupi nasz produkt i będzie się nim posługiwał. W takiej sytuacji, proces pozyskiwania wymagań odbywa się nie klasycznymi metodami inżynierii wymagań, na przykład opisanymi w normie ISO/IEC/IEEE 29148 (<http://www.computer.org/portal/documents/82129/160549/IEEE+29148-2011.pdf>), lecz metodami dziedziny mającej mało wspólnego z IT – marketingu.

Potencjalne, inne nazwy tej branży to „zarządzanie produktem”, ale zwyczajowo stosuje się ją już wobec innych działań, lub może „marketingowa inżynieria wymagań”. „Eksplorację” zauważyszli już – prawem kaduka – nasi koledzy, testerzy „eksploracyjni”. Niechże więc zostanie „rynkowa inżynieria wymagań”.

Według definicji (<http://pl.wikipedia.org/wiki/Marketing>), marketing, to „handel aktywny, wychodzący naprzeciw potrzebom klienta, próbujący odgadnąć skryte potrzeby klienta, usiłujący te

potrzeby uświadamiać oraz pobudzać, a nawet kreować i zaspokajać je.” Według tego samego źródła, w skład marketingu wchodzą „badanie rynku, kształtowanie produktu, oddziaływanie na rynek, ustalanie ceny, sprzedaż”.

Związek inżynierii wymagań z marketingiem można uprościć, traktując dział, czy agencję marketingową, zajmującą się badaniami rynku i „pobudzaniem, a nawet kreowaniem, potrzeb” po prostu jako zastępczego klienta, jako surogat klienta. „Oni” – czytaj marketing – mieliby swoimi sposobami określić, czego potrzebują potencjalni klienci, a „my” – czytaj dział IT – przełożymy to na nasz język, na „wymagania” i zbudujemy.

Nie muszę mówić, że taki model współpracy (tradycyjnego) działu marketingu z (tradycyjnym) działem IT jest niewystarczający nawet we współpracy z pojedynczym klientem. Zwycięski model relacji biznes – IT to taki, w który obie strony współpracują ze sobą, zaś IT jest nie tylko zasobem, dostarczającym rozwiązania informatyczne, niczym wodociąg dostarcza wodę,

ale aktywnym, współodpowiedzialnym partnerem tego procesu.

Nawet, jak pisze Adam Kolawa w swojej książce „Skokowy wzrost wydajności” (ang. The Next Leap in Productivity), IT może stać się motorem rozwoju biznesu, przejmować inicjatywę.



Rysunek 1. Profesor Tony Gorscheck – próba graficznego ujęcia MDRE ([www.gorscheck.com](http://www.gorscheck.com))

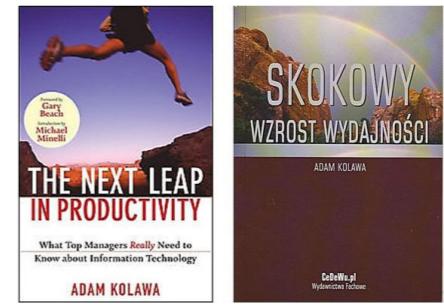
### Historia nauczycielką życia

Czytając fascynującą książkę o historii firmy „Apple”<sup>1</sup>, wciąż myślałem, że to tak naprawdę jest opowieść o rynkowej inżynierii wymagań, choć oczywiście ta nazwa w tekście nie pojawia się ani razu. Książka traktuje o meandrach, jakimi ta firma poruszała się od lat 70-ych do dzisiaj, wypuszczając na rynek produkty będące raz wielkim sukcesem, raz klęską. Autor opowiada, jak wielki Steve usiłował wstrzelić się w potrzeby rynku, jak tworzył zestawy wymagań wobec kolejnych produktów, jak je zmieniał.

Nie sposób oprzeć się wrażeniu, że wbrew ugruntowanej tradycji, inżynier wymagań, zajęty ich pozyskiwaniem, i marketingowiec badający rynek, i Steve, właściciel „Apple”, robią w istocie to samo. Nawet można w badaniu rynku dostrzec kawałki analizy biznesowej, gdy analizuje się domniemany tryb pracy potencjalnych klientów.

Ta narzucająca się prawda, zgodna zresztą z okresem inżynierii wymagań przez standard IEEE/ISO 29148 jako „dziedziny interdyscyplinarnej”, bardzo oporne toruje sobie drogę do naszych głów. Zgodnie z uświetoną tradycją, inżynier wymagań gardzi marketingowcem, marketingowiec inżynierem (jak w komiksach „Dilbert”), a tak zwany analityk biznesowy naśmiewa się z obydwu. Inżynier wymagań, wedle folkloru branżowego, chodzi – niczym programista – w dżinsach i niezbyt czystym T-shircie, analityk, w garniturze z krawatem lub bez, ociera się o najwyższe kręgi biznesu i nawet kupując mleko myśli w BPMN-ie, zaś marketingowiec, w stroju hipsterskim, projektuje kosztowne i kłamliwe kampanie reklamowe.

Jeśli tak nawet bywa, to na pewno tak być nie powinno, skoro chcemy działać skutecznie. Tak głosiłem, ale prozelitów wielu nie uzyskałem. Odsiecz przyszła mi z nieoczekванego kierunku



Rysunek 2. Rewolucyjne koncepcje Adama Kolawy

ku, bo ze świata akademickiego. Tam, od kilku lat, badania w zakresie inżynierii wymagań objęły nową dziedzinę, nazwaną MDRA – Market-Driven Requirements Engineering. Czyli metody, techniki i procedury, służące do pozyskania i opisania wymagań dla produktów, których użytkownikiem jest odbiorca masowy.

### Wymagania, które znajduje marketing

Pamiętacie, jak kilkanaście lat temu część Ericssona, produkującą telefony komórkowe, połączyła się z firmą Sony? Nieważne, że później ten związek się rozpadł, ale jego początek był wyraźnym symptomem zasady, że tradycyjna inżynieria wymagań (Ericsson), wielce przywiązana do swego inżynierskiego, informatycznego rodowodu, w praktyce nie umiała nawet dostrzec, a co dopiero zagospodarować wielu atrybutów jakości / typów wymagań (Sony). Dziś ten rodzaj atrybutów jakości często nazywa się modnie „doświadczeniem klienta” (user experience, UX) i docenia jego wagę w odniesieniu do pewnych typów produktów. Inna nazwa wobec tych niejasnych wymagań, to „charyzma” (ang. charisma, James Bach).

Podstawowa rzecz, którą dobrze, coraz lepiej rozumie marketing, a która wciąż słabo dociera do świadomości działów badawczo-rozwojowych oraz produkcji, w tym do działów IT, to fakt, że doświadczenie klienta zależy w przeważającym stopniu nie od cech produktu, a od szeregu innych czynników, leżących poza samym produktem.

Eksperyment Radosława Hofmana, opisany m.in. w jego artykule Behavioural economics in software quality engineering ([http://www.academia.edu/5515168/Behavioral\\_economics\\_in\\_software\\_quality\\_engineering\\_v06\\_ESE](http://www.academia.edu/5515168/Behavioral_economics_in_software_quality_engineering_v06_ESE)), udostępniony szerszej publiczności w tekście „Oprogramowanie a emocje” (<http://www.computerworld.pl/artykuly/366523/Oprogramowanie.a.emocje.html>), szokował. To samo oprogramowanie, na którym dwie równoważne grupy wykonywały, niezależnie od siebie, testy, i znalazły w nim w przybliżeniu identyczne błędy, zostało przez te grupy subiektywnie ocenione dramatycznie różnie (przez jedną grupę bardzo dobrze, przez drugą bardzo źle), zależnie od indoktrynacji, której zostały





Rysunek 3. MDRE w wydaniu firmy Apple

poddane przez potajemnych współpracowników eksperymentatora. Włos się jeży na inżynierskiej głowie: czyżby w ogóle nie warto było inwestować w jakość i niezawodność, bo ocena rynku i tak zależeć od lajków, szeptanego marketingu sieciowego i działań niekompetentnych technicznie, ale hałaśliwych specjalistów od reklamy?

Rozwiążanie tego pozornego paradoksu leży w znacznie ścisłej, niż dzisiaj, współpracy marketingu oraz inżynierii wymagań. Marketing dba, a przynajmniej powinien zadbać o tak zwane zarządzanie doświadczeniem klienta (customer experience management, CEM). Według zasad CEM, na opinię klienta o produkcie, wpływ większy niż sądziliśmy, wywierają nie cechy samego produktu, ale następujące czynniki:

- Jakość procesu sprzedaży,
- Jakość obsługi klienta,
- Wygoda posługiwania się produktem (interakcja, użytkowniczość, HCI),
- Niezawodność,
- Łatwość naprawy, obsługi, modyfikacji; jakość i dostępność serwisu,
- Charyzma, moda,
- Opinie w prasie, w Internecie,
- Opinia o samej firmie,
- Brak lub obecność czynnika „rozbitych okien”, takich jak na przykład uwikłanie firmy w skandale, niezły klientom proces reklamacji, złodziejskie praktyki i warunki sprzedaży,
- Wygoda i przejrzystość fakturowania (wobec usług, których to dotyczy),
- Warunki panujące w punktach obsługi klienta,
- Zrozumiałość i przejrzystość instrukcji obsługi, ich odpowiednia lokalizacja,
- Estetyka.

Pominięcie tych czynników może spowodować, że technicznie dobry produkt poniesie klęskę, a gorszy zwycięży. Szkoda czasu i pieniędzy na wytwarzaniu produktów wysokiej jakości, które zniszczy kłamiąca reklama, aroganckie praktyki sprzedaży, złe działanie serwisu. Szkoda też czasu na wytwarzaniu produktów kiepskiej jakości, które po temu trzeba z trudem wciskać niezbyt zadowolony

klientom, na których ponosi się straty.

Te zależności doskonale opisuje tak zwany model Kano<sup>2</sup>, bardzo słabo znany w IT. Dzieli on wymagania na trzy grupy: na (1) wymagania zwykłe, na (2) „zachwycające” i na (3) „rozwściekacze”.

Poziom zadowolenia klienta jest z grubsza liniowo zależny z poziomem spełnienia wymagań zwykłych. Im lepiej produkt je realizuje, tym wyższe jest zadowolenie klienta.

Zachwycające to te, których niespełnienie nikomu specjalnie nie doskwiera, ale kiedy choć w części są zrealizowane, wywołują efekt wielkiego WOW. Te wymagania są ważne dla takich produktów IT, których zawodność i zwykłe bugi są do zaakceptowania, ale atrybut trafnie nazwany przez Jamesa Bacha „charyzmą”, choćby dostępny tylko trochę, zaraz wywołuje lawinę lajków. Do nich zaliczają się gadżety, telefony, gry, wszelkie fejsbuki, twitery oraz instagramy, a nawet zupełnie normalne aplikacje internetowe: sklepy, portale informacyjne i plotkarskie, i te pokazujące pogodę.

Natomiast dla systemów wbudowanych/krytycznych dominują rozwściekacze: cechy, których płynnego działania wręcz nie zauważamy, ale najmniejsze potknienie ciężko przeżywamy, albo i nie przeżywamy, jeśli zawiedzie na przykład ABS w hamulcach, czy system sterowania ruchem kolejowym, czy rozrusznik serca. One są jak powietrze,



Rysunek 4. Brak charyzmy – telefon firmy Ericsson z połowy lat 90-ych

2 Np. [http://en.wikipedia.org/wiki/Kano\\_model](http://en.wikipedia.org/wiki/Kano_model)

Rysunek 5. Czy już nie liczy się jakość, tylko hałas w mediach społecznych?

dla tego ci, którzy je testują, są ciisi, solidni, mało mówią i dobrze pracują, i mało mają czasu na uleganie mistycznym modom typu XP, TDD, eksploracja i inne. Kto się nie lansuje, tego nie ma?

## Obszary MDRA

MDRA – pod tą nazwą gromadzą się różne prace akademickie<sup>3</sup>, stawiające sobie za cel badanie, jakie są zasady i mechanizmy rynkowej inżynierii wymagań. Badania nie wykazują, moim zdaniem, jakichś radykalnych różnic w porównaniu z inżynierią wymagań dla systemów robionych na zamówienie, poza obszarem pozyskiwania wymagań. Generalnie, jeśli chodzi o pozostałe etapy proce-

3 Dobre podsumowanie: <http://www.lunduniversity.lu.se/lup/publication/776217> („Market-driven requirements engineering processes for software products - a report on current practices“).



Rysunek 6. Noriaki Kano i jego model

**Bogdan Bereza**, uczestnik w różnych rolach (programista – tester – kierownik) dziesiątek projektów IT, zarówno tradycyjnych jak i agile) w Szwecji, w Polsce i w innych krajach. Doświadczony konsultant i doradca oraz trener. Prowadzi szkolenia w zakresie IT, zarządzania oraz psychologii biznesu od ponad 20 lat. Autor wielu książek i artykułów. Między innymi, trener szwedzkiej „Agile Academy”.



Bożena  
Rumak

## Inżynieria Wymagań

# Trudni interesariusze w procesie definiowania wymagań

### Proces definiowania wymagań

W procesie tworzenia oprogramowania bardzo ważną rolę pełni etap definiowania wymagań. W inżynierii oprogramowania wyróżnione są następujące procesy związane z tym obszarem:

- określanie i analizowanie wymagań (rozpoznawanie dziedziny, zbieranie i klasyfikacja wymagań, usuwanie sprzeczności, nadawanie priorytetów, sprawdzanie wymagań),
- specyfikowanie wymagań (biznesowe, systemowe, funkcjonalne, niefunkcjonalne)
- zatwierdzania wymagań (zwartych w dokumentacji wymagań).

Efektywna realizacja tych procesów ma kluczowe znaczenie dla przebiegu całego procesu tworzenia oprogramowania.

W czasie swojej pracy zawodowej pracowałam lub nadzorowałam prace analityków w wielu projektach różnej wielkości, co dało mi dużo materiału do wyciągnięcia wniosków.

Zaobserwowałam, że analitycy, którzy dysponują co najmniej podstawową wiedzą teoretyczną z inżynierii oprogramowania definiują wymagania skutecznie i efektywnie.

Zauważałam również, że z każdym rokiem oczekiwania odbiorcy wobec analizy biznesowej i kompetencji analityków są coraz większe. Postanowiłam nie traktować tych problemów jak przysłowiowej 'kwadratury koła' lecz przetestować w praktyce, czy dobranie odpowiednich technik zbierania wymagań oraz wykorzystanie kilku podstawowych kompetencji analityków przyniesie oczekiwany skutek.

### Interesariusze

Analitycy spotykają się w projektach informatycznych z interesariuszami, czyli przedstawicielami jednostek organizacyjnych, którzy biorą czynny udział w realizacji projektu lub są żywotnie zainteresowani pomyślnym albo niepomyślnym zakończeniem projektu.

Bardzo często analitycy uważają, że część interesariuszy utrudnia im sprawne definiowanie i dokumentowanie wymagań.

W celu zweryfikowania tych opinii przyjęłam następujące założenia:

1. Dostawcy chcą zrozumieć potrzeby odbiorców i przyszłych użytkowników (zależne od ich kultury, języka, umiejętności interpersonalnych) i zbudować system, który spełni ich potrzeby.
2. Potrzeby odbiorców/użytkowników ulegają zmianie w czasie (odbiorcy chcą dysponować coraz lepszą ofertą, oczekiwania użytkowników rosną w związku z dynamicznym rozwojem rynku informatycznego).
3. Odbiorca działa wg zasady IKIWI SI tzn. I'll Know It When I See IT (będę wiedział, gdy zobaczę). Zasada ta jest dużym wyzwaniem dla analityków, gdyż ostateczna wersja systemu powstaje najczęściej po zakończeniu analizy biznesowej, zaś klient:

- zwraca z potwierdzeniem potrzeb biznesowych do czasu prezentacji prototypu aplikacji lub
- w przypadku projektu, w którym nie jest zaplanowany prototyp dopiero, wtedy gdy widzi działającą aplikację zastanawia się nad swoimi potrzebami.

W niniejszym artykule skoncentrowałam się na wnioskach z większych projektów (powyżej 100 wymagań biznesowych), w których wystąpiły następujące zjawiska:

- odbiorca przedstawiał swoje wstępne potrzeby na wysokim stopniu ogólności,
- po stronie odbiorcy nie było jednoznacznie wskazanych osób odpowiedzialnych za podejmowanie decyzji,
- w trakcie procesu identyfikacji wymagań biznesowych następowała zmiana podstawowych założeń biznesowych lub prawnych.

Zjawiska te występują dość często w praktyce i są prawdziwym wyzwaniem dla analityków. Zaletą jest przede wszystkim to, że realizacja procesu analizy wymagań w takich warunkach umożliwia znaczny rozwój kompetencji analitycznych, a projekty są naprawdę bardzo interesujące.

### Rozpoznawanie dziedziny i zbieranie wymagań

Większość z analityków miała do czynienia z takimi przypadkiem jak ten opisany poniżej.

Odbiorca w dokumencie przekazanym do dostawcy przedstawił krótki opis swojej potrzeby biznesowej: Przygotowanie oprogramowania wspierającego proces sprzedaży kredytów gotówkowych przez agentów kredytowych.

Podczas pierwszego spotkania z odbiorcą analityk zidentyfikował m.in. następujące wymagania:

- system powinien umożliwić rejestrację wniosku kredytobiorcy przez agenta kredytowego,
- system powinien umożliwić przeprowadzanie oceny zdolności kredytowej kredytobiorcy.

Analityk na tym samym spotkaniu zidentyfikował "ukryte" wymagania:

- w systemie powinny być obsługiwane tylko kredyty gotówkowe dla osób fizycznych,
- kredyty mogą być udzielane w walucie pln lub euro, które w znacznym stopniu ograniczyły zakres projektu.

W przypadku, gdy potrzeba odbiorcy sformułowana jest zbyt ogólnie w celu zidentyfikowania "ukrytych" wymagań konieczne jest dostosowanie technik zbierania wymagań do wielkości projektu i typu klienta.

Najważniejsze są w takim przypadku następujące kompetencje analityka biznesowego:

- podstawowa wiedza merytoryczna z analizowanego obszaru.
- umiejętność analitycznego myślenia,
- umiejętności komunikacyjne (w tym: umiejętność zadawania pytań w sytuacji stresowej),
- dobranie stylu działania i wypowiedzi do osób uczestniczących w projekcie,
- umiejętność obserwacji zachowań odbiorcy i wnioskowania.

W trakcie dyskusji z analitykami i kierownikami projektu często poruszamy temat wiedzy merytorycznej z analizowanego obszaru. Mam nadzieję, że trudno wymagać od analityka, aby miał wiedzę ze wszystkich obszarów analizy. Moje doświadczenie wskazuje jednak, że bez podstawowej wiedzy (zdobytej wcześniej lub nabyciej bezpośrednio przez spotkaniem z klientem) trudno będzie osiągnąć oczekiwane efekty w krótkim czasie oraz prawidłowo zidentyfikować "ukryte" wymagania klienta.

Dodatkowo bardzo dobre efekty osiągają podczas zbierania wymagań analitycy, którzy posiadają umiejętności prezentacji, negocjacji oraz facylityzacji. W ostatnich latach widać wyraźną poprawę tych kompetencji zarówno po stronie dostawców, jak i odbiorców rozwijających informatycznych.

Wysoki poziom efektywności osiągają ci analitycy, którzy potrafią w trakcie spotkań obserwować zachowania odbiorców i dostosowywać sposób komunikowania do typu odbiorcy. Szczególnie ważne jest rozpoznanie, czy mamy do czynienia ze wzrokowcami, słuchowcami czy kinestetykami. Pozwala to na dobranie odpowiedniego stylu wypowiedzi do preferowanego przez odbiorcę kanału przekazywania informacji.

Zapewniam, że zastosowanie odpowiedniego stylu wypowiedzi pozwoli dojść do consensusu z najtrudniejszym interesariuszem.

W trakcie spotkań analitycznych w celu identyfikacji "ukrytych" wymagań warto stosować następujące techniki:

- a) efektywne zarządzanie słownikiem pojęć ("Używaj języka odbiorcy aby osiągnąć lepsze porozumienie"),
- b) tworzenie diagramów wysokiego poziomu abstrakcji umożliwiającego realizację analizy metodą "top-down",
- c) korzystanie ze spotkań typu "burza mózgów",
- d) dokumentowanie spotkań, tele- i videokonferencji oraz wymiana dokumentów analitycznych (między dostawcą i odbiorcą) w trybie zmian.

### Organizacja prac oraz dokumentowanie wymagań

Zarządzanie wymaganiami to zarówno systematyczne podejście do uzyskiwania, organizowania oraz dokumentowania wymagań systemu informatycznego, jak i proces, który ustala i zachowuje umowę między klientem a zespołem realizującym przedsięwzięcie w zależności od zmieniających się wymagań systemu.

W praktyce uzyskałam potwierdzenie zasady, że

organizacja procesu zarządzania wymaganiami oraz sposób dokumentowania zależy od wielkości projektu (im większy projekt, tym bardziej rozbudowana organizacja i dokumentacja) oraz interesariuszy.

W trakcie procesu analizy wymagań zgłaszane są potrzeby różnych użytkowników budowanego systemu (np. pracownicy w biurze obsługi klienta, pracownicy działów księgowych, klienci przedsiębiorstwa, którzy korzystają z udostępnionych dla nich usług systemu, administratorzy systemu). W potrzebach tych widoczne są różne aspekty użytkowania budowanego systemu i korzystania z jego usług.

Niezależnie od techniki definiowania i dokumentowania wymagań (np. VORD, Use Case, User Story) ważne jest, aby zebrane wymagania odpowiednio klasyfikować (wg wybranych kategorii), usuwać sprzeczności występujące między nimi oraz nadawać im priorytety zarówno z punktu widzenia przyszłych użytkowników (wymagania biznesowe), jak i z punktu widzenia twórców oprogramowania (wymagania systemowe). Działania te wpłyną bardzo pozytywnie na efektywność testów oraz jakość wyprodukowanego oprogramowania.

## Wpływ braku decyzji odbiorcy na efektywność analizy biznesowej

Co zrobić, gdy wśród członków zespołu po stronie odbiorcy nie ma osoby odpowiedzialnej za podejmowanie decyzji merytorycznych?

Moim zdaniem efektywność analizy wymagań zmniejszają takie zachowania odbiorcy jak:

- wydłużanie czasu odpowiedzi merytorycznych,
- przekazywanie sprzecznych uwag do dokumentu specyfikacji wymagań,
- nagłe wprowadzenie zasady „time-to-market” (czas między wizją produktu a jego dostępnością na rynku).

W sytuacji braku decyzji odbiorcy można przeciwdziałać spadkowi efektywności przez:

- dokładne dokumentowanie zmian założeń biznesowych (dokument specyfikacji wymagań, rejestr zmian),
- organizowanie spotkań z ustaloną wcześniej agendą, listą uczestników oraz sprecyzowanym celem spotkania,
- raportowanie ryzyka związanego z brakiem decyzji do szefa/sponsora projektu.

## Czy zmiana wymagań w trakcie realizacji projektu może być zaletą?

Czy ktoś z czytelników tego artykułu miał do czynienia z sytuacją, w której pojawia się zmiana wymagań w projekcie? Przypuszczam, że wszyscy odpowiedzieli: tak.

Zainteresuje Was z pewnością opisany poniżej przypadek.

Polska firma usługowa z branży motoryzacyjnej poprosiła o przygotowanie systemu obsługi procesu fakturowania sprzedaży swoich usług z uwzględnieniem możliwości fakturowania części zamiennych. Został przeprowadzony proces analizy i udokumentowania wymagań uwzględniający algorytm naliczania podatku VAT. Został przygotowany prototyp aplikacji. W trakcie jego prezentacji jeden z potencjalnych użytkowników aplikacji poprosił o przedstawienie sposobu wystawiania faktur dla elementów wysyłanych do kontrahenta zagranicznego z uwzględnieniem opłat celnych.

Okazało się, że część działalności firmy to sprzedaż części dla kontrahentów zagranicznych (bez usług ich wymiany). Ta informacja pojawiała się po raz pierwszy w trakcie prezentacji prototypu. Mimo, że obsługa opłat celnych wykracała poza zakres projektu zmiany (za dodatkową opłatą) zostały opisane i zaimplementowane, co pozwoliło na wdrożenie obsługi procesu fakturowania 100% sprzedaży firmy.

„

**Nie zależy od techniki definowania i dokumentowania wymagań ważne jest, aby zebrane wymagania odpowiednio klasyfikować, usuwać sprzeczności oraz nadawać im priorytety.**

Art. 48. 1. Konsument ma prawo w każdym czasie do spłaty całości lub części kredytu przed terminem określonym w umowie.

2. Kredytodawca nie może uzależnić wcześniejszej spłaty kredytu od jego poinformowania przez konsumenta.

W tym przypadku główny problem analityczny polegał na rozpoznawaniu „intencji” konsumenta. Czy środki przekazane przez niego dotyczą spłaty

najbliższej raty, czy też powinny zostać potraktowane jako spłata przedterminowa?

Mimo, że rozwiązanie wymagało dodatkowej pracy analitycznej klient był zadowolony z szybkiej reakcji na zmiany w projekcie, a analityk zmierzył się z ciekawym tematem.

Do głównych przyczyn zmian wymagań biznesowych w projektach realizowanych dla przedsiębiorstw i banków zaliczam:

- zasadę „time-to-market”,
- błędy w określeniu potrzeb biznesowych przez odbiorcę systemu,
- zmiany wynikające z przyczyn leżących poza dostawcą lub odbiorcą (np. zmiany prawne, zmiany w otoczeniu technologicznym lub gospodarczym),
- błędy w analizie wymagań popełnione przez dostawcę systemu.

Uważam, że nie należy ignorować żadnej zmiany wymagań zgłoszonej przez interesariuszy. Zachęcam do tego, aby na każdą zmianę spojrzeć przez pryzmat korzyści dla obu stron. Według mnie główne zalety zmian wymagań w projektach, to:

- możliwość rozwoju systemu informatycznego,
- dodatkowe przychody dla dostawcy,
- zadowolenie klienta,
- możliwość rozwiązywania ciekawych problemów analitycznych.

Skupiamy się na jasnym precyzowaniu celu biznesowego projektu i wizji rozwiązania w całym okresie realizacji projektu. Jeśli wszyscy interesariusze będą mieli do niej dostęp pozwoli to wspólnie wybrać do realizacji te zmiany, które są rzeczywiście istotne.

**Bożena Rumak**, absolwentka AGH w Krakowie oraz Uniwersytetu Warszawskiego.

Pracowała na stanowiskach analityka biznesowego i systemowego w sektorze przedsiębiorstw (handlowo-usługowych, produkcyjnych) oraz banków. Pełniła rolę Głównego Analityka w kilkudziesięciu projektach. W zakresie analizy biznesowej specjalizuje się w obszarze finansów i rachunkowości.

W trakcie wielu szkoleń zdobywała wiedzę zarówno z zakresu inżynierii oprogramowania, jak i rozwoju umiejętności miękkich przydatnych w pracy analitycznej oraz obszarze zarządzania. Uzyskała certyfikat w zakresie zarządzania projektami „Certified Project Management Practitioner (IPMA LEVEL D)” oraz ukończyła studia podyplomowe z zarządzania projektami Unii Europejskiej.

Obecnie pełni rolę Dyrektora Działu Analiz w Departamencie Planowania i Rozwoju Produktów, w Pionie Banków Komercyjnych w Asseco Poland SA.

Obszarem jej szczególnego zainteresowania zawodowego jest rozwój technik analizy biznesowej i dosłownie ich do tempa zmian potrzeb klientów oraz coaching w pracy menedżera.

## Zakończenie

W mojej opinii trudny interesariusz w procesie analizy wymagań, to interesariusz:

- Ciekawy
- Umożliwiający rozwój kompetencji analityka
- Inicjujący rozwój systemu informatycznego
- Przynoszący dodatkowe dochody dostawcy

Życzę wszystkim analitykom satysfakcji i radości ze współpracy z interesariuszami!

Jeśli ktoś chciałby porozmawiać o sprawach, które zasygnalizowałem w artykule, to serdecznie zapraszam (rumak.bozena@wp.pl).

„**Nie należy ignorować żadnej zmiany wymaganej zgłoszonej przez interesariuszy.**



Hanna  
Wesołowska

# Inżynieria Wymagań

## Wiedza czy doświadczenie, czyli na czym warto budować karierę

### Jestem praktykiem

**O**d jakiegoś czasu odnoszę wrażenie, że żyjemy w czasach kultu doświadczenia. Chętnie idziemy na studia, jednak samo ich ukończenie często postrzegamy jako зло konieczne.. Zdaniem wielu, uczymy się rzeczy odcinanych od rzeczywistości, a prawdziwie przydatna wiedza przyjdzie dopiero we właściwej pracy. Osoby, które mają już jakieś doświadczenie, na pytanie o wiedzę czy metodyczne podejście, nierzadko pogardliwe prychają: „Ja jestem praktykiem.”.

Przekornie dla hołdu doświadczenia przychodzi mi na myśl pytanie: „Czy to 10 lat doświadczenia, czy też rok powtórzony 10 razy?”. Czy poszukiwanie działających metod pracy możemy zakończyć, gdy poczujemy, że mamy ich wystarczająco, aby przetrwać kolejne projekty i w takim stanie trwać wygodnie do czasu, gdy życie nie naruci wyzwań przekraczających nasze możliwości, tak aby w napięciu szybko załatać dziury prowizorycznym rozwiązaniem?

Drugą kwestią jest paradox efektywności. Kiedy szybko zabieramy się do rozwiązywania problemu, obserwujemy wysiłek, działanie, które powoduje wewnętrzne zadowolenie (często również zewnętrzne, ponieważ szef cieszy się, że sprawy idą do



przodu). Jednak patrząc z perspektywy, na którą w pogoni za wynikami nie znajdujemy czasu, zauważylibyśmy, że nasza droga do rozwiązania wydaje się popłatańska jak spaghetti, a my tracąc mnóstwo energii, wymyślamy koło na nowo.

„Im większe mam doświadczenie, tym więcej wiedzy”. Idąc tym tropem, w imię zbierania dodatkowych doświadczeń, za każdym razem możemy chcieć wziąć na barki więcej, niż sami możemy udźwignąć. Praca po 12 godzin dziennie? Będę taki doświadczony... A może zbyt zmęczony, żeby nadal trzeźwo myśleć i wyciągać wnioski? Pytanie czy jedno idzie z drugim w parze? Doświadczenia to sytuacje, w których się znaleźliśmy. Pech chce, że projekty z definicji są przedsięwzięciami niepowtarzalnymi, nie ma także ludzi niepowtarzalnych. Czy to, że mamy na koncie X doświadczeń, jest równoznaczne z tym, że zrozumieliśmy wszystko?

stkie leżące u ich źródła przyczyny? Czy jest to równoznaczne z tym, że wyciągnęliśmy właściwe wnioski? Czy pewnikiem jest, że wnioskami tymi będziemy kierować się w przyszłości?

### Mam certyfikaty/Mam tytuły

Po drugiej stronie barykady praktyków stoją miłośnicy certyfikatów. Pojawia się kolejne, wiecznie żywe, pytanie „Czy certyfikat o czymś świadczy”? Ilu znamy takich, którzy byli w stanie spełnić wymagania testów, zaliczeń, po czym okazuje się, że nie rozumieją zaliczanej treści i nie są w stanie zastosować jej w praktyce?

Często spotykane jest wzdryganie ramionami na dźwięk słów „teoria”, „model”, „metoda” jako oznaki przerostu formy nad treścią i nieżyciowego podejścia. Mamy w głowach stereotyp naukowca zaszywającego się na całe życie na uniwersytetach głoszącego prawdy, które sprawdzają się 40 lat temu, a które ni opuścili nigdy murów akademickich.

Teoretycy nie poznali ograniczeń. Nie widzieli sytuacji, w których ich modele nie mają prawa działać. Tworzą swoje teorie w świecie idealnym, problemem jest to, że w większości działają one tylko na papierze. Może, gdyby klient, rynek, sytuacja była inna...

Teoria jest powolna, nie nadąża za szybko zmieniającymi się realiami. Pisane niekiedy latami książkami, później zalegają w księgozbiorach tak długo, że data wydania coraz bardziej przywołuje na myśl dawne epoki. Czy mogą być zatem odpowiedzią na problemy, które pojawiają się właśnie tu i teraz i zmieniają w szalonym tempie dynamicznego rozwoju wolnego rynku?

Czy bez doświadczenia jesteśmy w stanie pojąć, o czym mówimy? Stajemy się niczym kawaler opowiadający o najlepszych sposobach rozwiązywania małżeńskich kryzysów. Czy uczeni oderwani od zastosowania teorii są w stanie przygotować je tak, żeby rzeczywiście były użyteczne?

„Żadna lektura i inteligentne rozumowanie nie zastąpi bezpośredniego doświadczenia.” (John Fowles).

### Wiedza czy doświadczenie

Powstaje pytanie, co wybrać? Na czym się skupić w poszukiwaniu najlepszej drogi rozwoju swojego życia zawodowego? Postawić na zdobywanie wiedzy akademickiej, czy rzucić w kąt książki, uczelnie i poznać prawdziwe wyzwania, zdobyć praktyczne umiejętności w pracy?

Zamiast chwiać się w nierównowadze, najlepiej stać się pewnie na dwóch nogach. Zbieranie doświadczenia bez wiedzy to okrężna droga. Wiedza bez doświadczenia to sucha łodyga bez deszczu i słońca.

### Wiedza to zapis doświadczeń

„Cóż to takiego  
Czy to edza?  
Nic innego jak  
zapisane doświadczenia.”

„Cóż to takiego wiedza? Nic innego jak zapisane doświadczenie.” (Thomas Carlyle). Wiele teorii, standardów, dobrych praktyk w naszej branży, i pewnie nie tylko, ma swoją wspólną historię. Istniał problem, pewna trudność, z której borykało się wiele osób. Znalazł się człowiek na tyle przedsiębiorczy, zdeterminowany lub z większą łatwością potrafiący znajdować rozwiązania, by podzielić się nimi z innymi. Wokół tych inicjatyw zbierali się kolejni, by je jeszcze bardziej rozwijać, udoskonalać. Czym, jak nie tym, jest między innymi: BABOK, PMBOK, CMMI, Agile, Scrum czy inne. To, co dziś widzimy w formie pisanej, kiedyś miało swój początek w praktyce projektów i zostało potwierdzone przez doświadczenia wielu, którzy znali świetnie swoją profesję i branżę. „Wiedza to nic innego jak unaukowiony i usystematyzowany zdrowy rozsądek.” (Thomas Henry Huxley).

### Wiedza to umiejętność wykorzystania

„Żadna lektura  
i inteligentne  
rozumowanie  
nie zastąpi  
bezpośredniego doświadczenia.”

„Wiedza to ogół wiarygodnych informacji o rzeczywistości wraz z umiejętnością ich wykorzystywania” (Nowa Encyklopedia Powszechna). Jeśli przyjrzeć się definicji wiedzy, to znika nam problem oderwanego od rzeczywistości teoretyzowania. Znasz PRINCE2, ale uważasz, że jest zbyt obszerny, by wykorzystać go w projekcie i jednocześnie zachować jego rentowność? Posiadziesz wiedzę dopiero wtedy, gdy będziesz wiedział, jak przystosować go do swoich potrzeb.

## Wiedza to wykorzystanie wymyślonego koła

W ilu projektach musimy popełnić ten sam błąd, żeby nauczyć się tego, co moglibyśmy przeczytać na jednej stronie mądrzej książki i wziąć sobie tę radę do serca? Po co uczyć się na swoich błędach, skoro mamy opisane błędy popełniane wielokrotnie przez innych? Wystarczy się rozejrzeć. Znajdziemy nie tylko błędy, ale też historie sukcesu! Nic w tym motywującego, gdy wymyślamy rozwiązania, które zweryfikowały lata doświadczeń i tysiące projektów.

## Wiedza to skrócenie drogi

„Bardzo wielu, a może większość ludzi, aby coś znaleźć, musi najpierw wiedzieć, że to istnieje” (Georg Christoph Lichtenberg). Jeśli poświęcamy czas na poszerzanie horyzontów, w odpowiednim momencie, gdy pojawi się potrzeba, będziemy wiedzieli, że rozwiązanie już istnieje. Tworząc skrzynkę narzędziową, wkładając do niej coraz to nowe narzędzia, w razie potrzeby będziemy mogli wybrać to, które najlepiej przyda się w danym zadaniu. Inaczej całe życie w obliczu różnorodnych wyzwań możemy pozostawać w ręku jedynie ze śrubokrętem i taśmą klejącą. „Doświadczenie uczy nas, że dzięki długiemu błądzeniu odkrywamy krótszą drogę.” (Hardy Thomas). Po co błędzić?

## Jak rozwijać swoją karierę?

Znajdzmy czas na rzeczy ważne i pilne oraz te mniej pilne a bardzo ważne. Równoważmy pracę, doświadczenie oraz projekty ze zdobywaniem wiedzy, nowych narzędzi, usprawnianiem tego, jak działamy.

## Pasjonuj się

Ani wiedza, ani doświadczenie nie działają wiele, jeśli nie czujemy w sobie radości/żaru na myśl o wybranej profesji. Jeśli wykorzystuje ona nasze zainteresowania, talenty, droga do doskonałości będzie przyjemniejsza i bardziej ekscytująca.

## Wyznacz cel

Tak jak projekty zaczynamy od określenia celu, bez którego nie wiemy dokąd zmierzamy i czy droga ta ma w ogóle sens, tak samo życie zawodowe powinno mieć swoją wizję, strategię, mierzalne kryteria sukcesu. Dla odmiany przeprowadźmy sobie analizę AS-IS i TO-BE, analizę luk kompetencyjnych, narzędziowych, jak również miękkich. Znajdzmy sposoby na ich zapewnienie. Przygo-



tujmy harmonogram realizacji.

Określanie miejsca, w którym znajdujemy się aktualnie z naszymi kompetencjami, nie jest oczywiście czymś nowym, więc możemy posłużyć się latami doświadczeń innych, np. IIBA (International Institute of Business Analysis). Członkowie organizacji mają dostęp do modeli kompetencji (Business Analysis Competency Model) a nawet możliwości oceny swojego aktualnego stanu i określenia planów rozwoju (Business Analysis Competency Assessment).

w przyszłości nierazko kosztować w formie szkoleń grube tysiące.

Jako osoby z bogatym już doświadczeniem, pamiętajmy, że nasza ścieżka wcale nie musi się kończyć jako ślepa uliczka. Próbujmy ciągle podnosić poprzeczkę i gdy poczujemy wolne moce przerobowe wziąć na siebie nowe wyzwanie. Robimy coś, co jest nieco ponad nasze możliwości, wtedy będziemy się ciągle rozwijać, a w naszej profesji, wyzwań i możliwości rozwoju absolutnie nie brakuje.

## Gdzie znaleźć doświadczenie

Nic nie zastąpi doświadczenia na polu boju. Dlatego wcześniej znajdzmy dla siebie miejsce w firmie, w prawdziwych projektach. Nie czekajmy na skończenie studiów. W zderzeniu z doświadczeniami, do których możemy się odwoływać, zajęcia na uczelni nabiorą nowego, pełniejszego znaczenia. Zaczniemy zauważać zastosowania wykładowanych teorii. Nie rzucajmy też jednak studiów dla pracy. W pędzie projektowego życia możemy nie znaleźć tyle czasu, by usiąść i zastanowić się, czy to, co robimy, jest optymalne i czy być może ktoś już od dawna zna rozwiązanie. To, co możemy dla siebie wynieść ze studiów za darmo będzie nas

## Gdzie znaleźć wiedzę

Dróg do wiedzy jest wiele. Studia (analiza biznesowa, modelowanie systemów, analiza procesów biznesowych), książki (podejścia, notacje, umiejętności miękkie, doświadczenia – polecam klasykę, nowinki z Amazona i ogromnej biblioteki IIBA dostępnej dla członków organizacji), standardy (BABOK), organizacje (SIW, IIBA), strony (Modern Analyst, Business Analyst Times, IIBA, AION), blogi (Jarosław Żeliński, Monika Perendyk, Adrian Reed, James Archer, Bridging the Gap), spotkania branżowe (seminaria IIBA, PAM Summit). Najważniejsze to zacząć szukać. Kiedy to zrobimy, zauważymy, jak coraz więcej informacji związanych z tą dziedziną sama znajduje do nas drogę. A jeśli czegoś nam brakuje, być może jest to właśnie miejsce, jakie przygotowała dla nas historia?

Dobrze, że mamy praktyków, bo rzucają wyzwania i weryfikują nasze teorie. Dobrze, że mamy teoretyków, bo badają i odkrywają to, do czego dochodzilibyśmy znacznie dłużej metodą prób i błędów. Rozwijajmy w sobie oba charaktery.



Ewa  
Brzeska

## Inżynieria Oprogramowania

# Na styku biznesu i IT

Od potrzeby biznesowej do gotowego produktu – rozważania w kontekście inżynierii oprogramowania

### Wstęp

**B**iznes i IT to dwa z pozoru odległe światy. Pierwszy z nich zainteresowany jest przede wszystkim stopami wzrostu, wskaźnikami wydajnościowymi, sprzedażą, zyskiem, etc.. Drugi fascynują nowoczesne technologie, języki programowania, platformy sprzętowe i systemowe, metodyki realizacji projektów informatycznych oraz tworzenia kodu, itp. Jednocześnie oba te światy wzajemnie się uzupełniają, przenikają i de facto nie mogą bez siebie istnieć.

Biznes potrzebuje IT, bo który dzisiaj może sobie wyobrazić zarządzanie czymkolwiek bez pomocy wspomagających ten proces programów czy systemów komputerowych. Z drugiej strony nawet najnowocześniejszy i najlepszy program czy aplikacja nie zapewnia jego autorom środków utrzymania i nie wygenerują zysku, jeśli pozostaną w miejscu powstania i nie zostaną sprzedane, gdyż nie będą spełniać wymogów ani realizować potrzeb biznesu.

„*Na pewno biznes i IT powinny dobrze rozumieć nawzajem swoje potrzeby i wychodzić naprzeciw swoim oczekiwaniom.*

A zatem oba te światy są na siebie skazane. Czy naprawdę „skazane”? To nie jest dobre słowo na określenie wzajemnych relacji. Na pewno biznes i IT powinny dobrze rozumieć nawzajem swoje potrzeby i wychodzić naprzeciw swoim oczekiwaniom. Prześledźmy zatem drogę od idei i potrzeby biznesowej do gotowego produktu informatycznego, który zespół IT wytwarza w odpowiedzi na potrzebę klienta. Będziemy patrzyć od strony biznesu i od strony IT, a szczególnie będzie nas interesował punkt styku tych dwóch światów.

### Inżynieria oprogramowania

Oprogramowanie komputerowe jest specyficzny produktem, wytwarzanym w procesie wytwarzania oprogramowania. Efektem końcowym realizacji tego procesu jest kod źródłowy, który tworzy funkcje i moduły, z których składa się gotowy produkt – oprogramowanie. Dziedzina, która zajmuje się całokształtem działań związanych z produkcją oprogramowania, czyli: analizą wymagań, projektowaniem, implementacją (pisaniem kodu wg projektu i specyfikacji w konkretnym języku i określonym środowisku programistycznym), testowaniem oraz wdrożeniem gotowego rozwiązania, a także jego utrzymaniem i dalszym rozwojem, to inżynieria oprogramowania.

Od strony IT wszystkie rozważania dotyczące produkcji oprogramowania odbywają w kontekście inżynierii oprogramowania. IT jest mocno nastawione na wytworzenie produktu: aplikacji, programu czy systemu informatycznego. W zespołach informatycznych dywaguje się często: może wykorzystać jedną z najnowszych technologii, a może wytwarzać w metodyce, która jest obecnie na topie, może napisać program na tę czy inną platformę systemową. Niestety, podczas tych rozważań niepokojąco często zespołom IT gniezie z oczu cel główny pisania oprogramowania: zaspokojenie potrzeby biznesowej, od-

Biznes ma pomysł i środki, a IT może dostarczyć produkt

W pewnym momencie decydent ze świata biznesu dochodzi do wniosku, że potrzebuje oprogramowania komputerowego. Ma środki na jego wykonanie i mniej lub bardziej sprecyzowany pomysł jak to oprogramowanie ma wyglądać i co robić. Rozpoczyna się proces poszukiwania wykonawcy oprogramowania, a po szczęśliwym finale poszukiwań, biznes staje się klientem IT. Relacja ta oznacza, że zleceniodawca oczekuje od zespołu IT wytworzenia oprogramowania o określonych przez siebie cechach, w ramach przeznaczonego na jego wytworzenie budżetu i w uzgodnionym czasie.

Na pierwszy rzut oka mamy w tym momencie do czynienia niemal z sytuacją idealną: zleceniodawca biznesowy jest głęboko przekonany, że za pewną umówioną kwotę, w zaplanowanym terminie, zespół IT wykona oprogramowanie, które będzie dokładnie takim produktem, jaki zleceniodawca sobie wyobraża i jakiego chce. Natomiast zespół IT zwykle z werwą przystępuje do realizacji rozwiązania, mając przekonanie graniczące z pew-

nością, że jest w stanie wytworzyć oprogramowanie satysfakcjonujące klienta. Skoro na pierwszy rzut oka może być tak dobrze, to dlaczego czasami bywa źle?

### Po co powstaje dane oprogramowanie, czyli cel przedsięwzięcia

Biznes zamawiając oprogramowanie chce zwykle osiągnąć konkretny cel: usprawnić proces, spowodować wzrost wydajności, znaleźć antidotum na szczególnie uciążliwą bolączkę itp. Na etapie pierwotnej idei, pomysłu na oprogramowanie, nie jest jeszcze zwykle określona jego pełna funkcjonalność. Często nie jest nawet znany zakres systemu czy programu. Jednak najczęściej już w tym momencie jest ścisłe określony cel biznesowy, który ma być osiągnięty po wdrożeniu tego oprogramowania. Na przykład: wdrożenie systemu monitorowania i zarządzania produkcją ma umożliwić śledzenie produkcji w toku oraz spowodować wzrost wydajności produkcji. Wdrożenie aplikacji monitorującej online flotę samochodową ma na celu obniżenie kosztów zużycia paliwa.



Cel biznesowy jest niezmiernie istotny, gdyż odpowiada na pytanie po co powstaje oprogramowanie. Cały proces wytwarzania oprogramowania powinien być podporządkowany temu, aby zapewnić realizację tego celu. Szczególna odpowiedzialność w tym zakresie spoczywa przede wszystkim na analitykach wymagań, którzy identyfikują wymagania odnośnie oprogramowania w kontekście jego celu.

## Nazwa programu i jego zakres

Nazwę przyszłego oprogramowania często propo-  
nuje ten, kto zleca jego wykonanie, czyli zlecenio-  
dawca (biznes). Warto jednak na samym wstępnie,  
czyli przy pierwszym styku IT z biznesem, przeana-  
lizować adekwatność tej nazwy w stosunku do pla-  
nowanego zakresu oprogramowania w dziedzinie  
biznesowej. Te dwa elementy powinny ściśle ze-  
sobą współpracować. Nazwa zbyt szeroko nadana, rod-  
zi pokusę rozszerzania zakresu już w fazie wytwarz-  
ania oprogramowania. Z kolei nazwa zbyt wąska  
w stosunku do przyszłego zakresu, może być pod-  
stawą do zarezerwowania budżetu zbyt małego  
na wykonanie oprogramowania o planowanym  
zakresie.

Jako przykładem posłużę się autentyczną historią ze swojej praktyki zawodowej. Zdarzyło mi się kiedyś pracować jako zewnętrzny, niezależny konsultant w ramach projektu, którego celem było wytworzenie oprogramowania pod nazwą „System zarządzania produkcją”. Ponieważ dołączylem do zespołu projektowego już w trakcie realizacji tego projektu, nie miałam wpływu ani na określanie nazwy, ani na precyzyjne ustalanie zakresu systemu. W momencie, kiedy zaproszono mnie do współpracy, projekt był mocno zagrożony. Klient nieustannie rozszerzał funkcjonalność rozwiązania, które było już w zaawansowanej fazie implementacji. Budżet, ustalony i zatwierdzony na początku projektu, kurczył się w zastraszającym tempie, a czas realizacji oprogramowania niebezpiecznie wydłużał, przy czym do końca prac nad projektem nie było jeszcze blisko. Moje zadanie polegało m.in. na tym, aby doprowadzić do konsensusu na linii zleceniodawca – wykonawca, aby projekt mógł być dalej realizowany, a następnie zakończony sukcesem. Dokładanie kolejnych funkcji do oprogramowania, czyli ciągłe powiększanie jego zakresu, klient motywował w tym wypadku stwierdzeniem, że nie wyobraża sobie sytuacji, żeby „System zarządzania produkcją”, nie realizował tej funkcji, o której właśnie mówi, ponieważ jest ona bardzo istotna dla procesu zarządzania produkcją. W opisywanym przypadku nazwa oprogramowania była ewidentnie zbyt szeroka. Jednocześnie zakres rozwiązania niestety nie został precyzyjnie określony na etapie zbierania wymagań, co pozwoliło klientowi na generowanie coraz to nowych żądań w stosunku do wytwarzanego oprogramowania. Na szczęście zleceniodawca dał się przekonać do podziału prac nad systemem na



etapy o wyraźnie zdefiniowanych zakresach, harmonogramach realizacji i budżetach. Dzięki temu projekt został zrealizowany pomyślnie.

## Użytkownicy oprogramowania

Ważną czynnością początkowej fazy prac nad realizacją oprogramowania jest poprawna i kompletna identyfikacja wszystkich przyszłych użytkowników tego oprogramowania.

Zamawiający oprogramowanie, czyli ktoś ze świata biznesu, może, lecz nie musi, być jednocześnie jego przyszłym użytkownikiem. Grono użytkowników może być dość szerokie, a w jego skład może wchodzić wiele grup o różnych potrzebach i interesach. Czasami interesy te mogą być wzajemnie sprzeczne.

Jako przykład posłuży nam aplikacja monitorująca on-line flotę samochodową. Oprogramowanie takie dostarcza najczęściej informacji o przebiegu tras poszczególnych samochodów, zatrzymaniach, tankowaniu paliwa, kierowcach pojazdów itp. Użytkownikami tego typu oprogramowania są zazwyczaj m.in.: dyrektor przedsiębiorstwa, kierownik floty oraz każdy z kierowców. Kierownik floty, jako osoba bezpośrednio nadzorująca pracę kierowców jest zainteresowany dokładnymi raportami tras, tankowań, zużycia paliwa itp. Dyrektor

przedsiębiorstwa, który najczęściej jest zamawiającym tego typu aplikację (biznes) potrzebuje syntetycznych raportów, dzięki którym będzie mógł określić zbiorcze koszty zużycia paliwa w jednostce czasu. Natomiast grupa kierowców nie będzie zupełnie zainteresowana wdrożeniem takiego oprogramowania (konflikt interesów), lecz będzie zobowiązana do korzystania z niego w zakresie swoich kompetencji, np. wprowadzając dane dotyczące przyczyn postoju lub ceny jednostkowej zatankowanego paliwa.

Przed inżynierami wymagań, analitykami biznesowymi, stoi niełatwne zadanie: powinni oni jednoznacznie i poprawnie zidentyfikować wszystkie grupy użytkowników, a także ich potrzeby oraz wymagania odnośnie oprogramowania. Jeśli wymagania są niespójne lub sprzeczne, zadaniem analityków jest doprowadzenie do ich spójności i niesprzeczności. Jednak, aby cały proces odkrywania potrzeb i pozyskiwania wymagań zakończył się sukcesem, konieczna jest dobra współpraca pomiędzy IT a biznesem, w ramach całego zespołu projektowego.

## Potrzeby a wymagania

Potrzeby użytkowników często są artykułowane w bardzo ogólny sposób: „program ma działać szybko”, „ma być prosty w obsłudze”, „ma robić

raporty miesięczne”, itd. Proces zbierania wymagań polega na określeniu na podstawie potrzeb użytkowników, kompletnych wymagań funkcjonalnych odnośnie oprogramowania (czyli co program ma robić). Dodatkowo określa się również tzw. wymagania niefunkcjonalne, w skład których wchodzą np. kryteria wydajnościowe i jakościowe, jakie ma spełniać oprogramowanie oraz ograniczenia, przy jakich oprogramowanie musi pracować.

W odróżnieniu od swobodnie artykułowanych potrzeb, wymagania zebrane przez analityków muszą być m.in.:

- kompletne,
  - spójne,
  - jednoznaczne,
  - niesprzeczne z innymi wymaganiami,
  - weryfikowalne.

Rola analityka wymagań jest nie tylko pogodzenie sprzecznych interesów, a co za tym idzie często wymagań, różnych grup użytkowników. Wszystkie ogólne potrzeby typu „program ma działać szybko” muszą zostać zamienione w konkretne wymagania czasowe, np. „Oczekiwany czas odpowiedzi systemu dla raportu X – poniżej 3 sekund”. Przy czym podczas całego procesu zbierania i analizy wymagań należy pamiętać o tym, po co oprogramowanie ma zostać wytworzony, a następnie wdrożone. Wszystkie wymagania powinny być pozyskiwane oraz analizowane w kontekście celów określonych przez biznes.

## Właściwa komunikacja podstawą tworzenia dobrego oprogramowania

Członkowie zespołu projektowego, którego zadaniem jest wytworzenie oprogramowania, komunikują się ze sobą nieustannie. Analitycy biznesowi podczas rozmów (wywiadów analitycznych) z użytkownikami zbierają wymagania. Na podstawie specyfikacji wymagań architekci i projektanci systemów projektują rozwiązanie, osadzając go w konkretnej technologii informatycznej. Często konsultują oni projekt z analitykami z jednej strony, a z programistami z drugiej. Następnie programiści implementują zaprojektowane rozwiązanie, przekazując go testerom. Testerzy testują zgodność oprogramowania z wymaganiami, a w przypadkach wątpliwych zwracają się o wyjaśnienia do analityków. Na koniec wdrożeniowcy wdrażają gotowe oprogramowanie, szkoląc użytkowników ze świata biznesu. Niezmiernie istotnym jest, aby wszyscy interesariusze projektu mówili tym samym językiem. Tymczasem dialogi często przebiegają w taki mniej wiecej sposób:

A. „Chciałbym, aby dostęp do poszczególnych funkcji systemu był autoryzowany za pomocą

nazwy użytkownika i hasła. System może pamiętać hasła użytkowników, ale tylko tych, który nie mają dostępu do newralgicznych funkcji systemu, które za chwilę określę”

B. „Dobrze, zrobimy zatem ekran logowania z dwoma entry fieldami: Login i Password. Na ekranie będzie Checkbox do ustawienia, czy hasło ma być zapamiętane. Ale Checkbox będziemy wyświetlać tylko dla tych User'ów, którzy będą mieli poziom uprawnień większy niż 3”

Obie osoby mówią o tej samej funkcji systemu, lecz język ich jest zupełnie inny. Pierwsza kwestia wypowiadana jest językiem ogólnym, druga zaś to typowy żargon informatyczny. Przy tak różnych językach trudno o dokładne i wzajemne zrozumienie wypowiadanych kwestii, szczególnie jeśli dotyczyłyby one skomplikowanych zagadnień z dziedziny biznesowej. Dlatego w pierwszej fazie prac nad tworzeniem oprogramowania, w fazie zbierania wymagań, powinien powstać słownik pojęć, zawierający kompletny zbiór pojęć z dziedziny oprogramowania, którym będą posługiwać się wszystkie zainteresowane strony.

## Projektowanie i implementacja, czyli niech technologia i metodyka będą dopasowane do wymagań i nie przysłonią celu biznesowego

Czy w ramach projektu zrealizować klientowi aplikację internetową, czy rozwiązanie mobilne? Czy duży system, monitorujący produkcję ciągłą w elektrowni 24 godziny na dobę przez 7 dni w tygodniu, mający bardzo wysokie wymagania odnośnie wydajności oraz niezawodności, projektować i implementować na platformę Windows, czy AS/400? Pisać w C++, czy w Javie? Czy to oprogramowanie realizować za pomocą prototypu, czy może jednak odpowiedniejsza byłaby metoda Agile, a może pozostać przy metodyce klasycznej?

Na takie i podobne pytania odpowiadają zwykle projektanci oprogramowania i kierownicy projektów. Podejmują oni decyzje, które mają wpływ zarówno na ostateczny kształt oprogramowania, jak i na łatwość jego wdrożenia oraz późniejszego utrzymania. Decyzje te jednak przede wszystkim są istotne dla realizacji celu biznesowego przedsięwzięcia. To przede wszystkim ten aspekt powi-



nien być brany pod uwagę podczas odpowiedzi na powyższe pytania.

Tymczasem niepokojąco często zdarza się, że głównym, jeśli nie jedynym czynnikiem, mającym wpływ na decyzje projektowe jest to, jakiego typu specjalistów mamy aktualnie w zespole. Czyli, jeśli w zespole IT mamy programistów PHP i C#, to sprzedajmy klientowi aplikację webową. Za specjalistów od Javy, czy Objective-C, którzy mogliby zaimplementować aplikację mobilną, musielibyśmy dodatkowo zapłacić, co obniży nasz zysk. Jeśli nasz zespół implementuje „od zawsze” w środowisku Windows, to nawet nie rozpatrzymy, że w tym szczególnym przypadku powinniśmy wziąć pod uwagę inną platformę sprzętową (AS/400), choć tak naprawdę to ona lepiej spełnia kryteria opisane w wymaganiach.

Reasumując: dobór technologii, w której zostanie zrealizowane oprogramowanie oraz metodyki realizacji projektu nie może być ani przypadkowy, ani też głównie zgodny z aktualnym trendem, modą. Dobór ten zawsze powinien być przeprowadzony w taki sposób, aby przede wszystkim zapewnić realizację celu projektu narzuconego przez biznes oraz realizację wszystkich wymagań, które zostały wyspecyfikowane w stosunku do tworzonego oprogramowania.

## Podsumowanie

Co jest istotne, aby przedsięwzięcie polegające na wytworzeniu oprogramowania, będącego odpowiedzią na potrzeby biznesu i spełniającego określone kryteria, zakończyło się sukcesem?

Poniżej najważniejsze, choć nie jedyne czynniki i elementy, z jakimi mamy do czynienia na styku biznesu i IT, na które należy zwrócić baczną uwagę w procesie twórczym oprogramowania:

1. Jasno określone cele, jakie biznes chce osiągnąć wdrożeniem oprogramowania (cel biznesowy przedsięwzięcia, czyli po co oprogramowanie ma być zrealizowane i wdrożone).
2. Nazwa programu adekwatna do zakresu oprogramowania w dziedzinie biznesowej.
3. Jednoznacznie określony zakres programu. Stanowi on podstawę rozliczeń pomiędzy wykonawcą oprogramowania i zleceniodawcą, czyli pomiędzy IT i biznesem. Jasno i precyzyjnie określony zakres pokazuje, które funkcje należą do programu, a które są funkcjonalnością dodatkową, pozostającą poza jego zakresem. W przypadkach konfliktowych ułatwia rozstrzygnięcie kto płaci za rozbudowę oprogramowania: wykonawca (IT), ponieważ funkcja jest we wcześniej ustalonym zakresie, czy zleceniodawca (biznes), ponieważ funkcja wykracza poza ramy uzgodnionego zakresu.
4. Poprawna identyfikacja wszystkich użytkowników oprogramowania i ich wymagań.
5. Właściwa komunikacja w całym zespole projektowym i wspólny język, którym mówią wszystkie zainteresowane strony, zarówno po stronie zamawiającego, jak i wykonawcy oprogramowania.
6. Najważniejszy jest cel biznesowy przedsięwzięcia. Zatem technologia, w jakiej oprogramowanie zostanie wytworzone oraz metodyka realizacji projektu powinny być podporządkowane realizacji tego celu, a nie być celem samym w sobie.

**Ewa Brzeska**, Współzałożycielka i Prokurent Ebitech Sp. z o.o.

Od 25 lat związana czynnie z branżą IT. W tym czasie uczestniczyła w realizacji wielu projektów informatycznych, pełniąc rolę analityka, projektanta, architekta, programisty oraz kierownika projektów. Wśród zrealizowanych przez nią systemów znajdują się rozwiązania wspomagające zarządzanie różnymi sferami biznesu i produkcji, dedykowane na różne platformy systemowe i sprzętowe. Ma ponad 15 lat doświadczenia w tworzeniu systemów klasy MES (ang. Manufacturing Execution Systems) dla energetyki i przemysłu. Zarządzała i zarządza projektami i produktami informatycznymi oraz produkcją oprogramowania. Zajmuje się również konsultingiem i doradztwem IT, wykorzystując przy tym bogate doświadczenie praktyczne oraz wiedzę teoretyczną, zdobytą między innymi w Szkole Głównej Handlowej w Warszawie.

W 2009 roku wraz ze wspólnikiem założyła firmę informatyczną Ebitech, która zajmuje się głównie tworzeniem oprogramowania biznesowego na platformy systemowe Apple, ze szczególnym uwzględnieniem urządzeń mobilnych.

Współzałożycielka oraz Członek Zarządu Stowarzyszenia Inżynierii Wymagań.



Krystian  
Kaczor

# Zarządzanie Projektami

## Wymagania w Agile

### Czas na Agile

**N**o i stało się. Nadszedł ten dzień, kiedy szef poinformował Cię, że nadszedł czas na zmianę sposobu pracy na Agile. Jeśli miałeś wprowadzenie Agile robione metodą skoku na głęboką wodę, to możesz poczuć się jak szeregowiec Cage, główny bohater filmu z Edge of Tomorrow. Nie ma czasu na wyjaśnienia, weź nowe narzędzia i biegij. Różnica jest taka, że Ty nie będziesz miał/ miała kolejnego podejścia w przypadku porażki projektu. Dla całego zespołu zmieni się część rzeczywistości dotyczącej pracy, a może nawet zmieni się sam zespół. Zmiana będzie dotyczyła wielu aspektów pracy, na potrzeby tego artykułu musimy zawieźć zakres omawianych tematów. Proponuję, żebyśmy tym razem skupili się na temacie wymagań. Pracując z metodami Agile przez ostatnie 8 lat powiem Ci, że możemy również bezpiecznie założyć, że podejściem, które będzie Ciebie dotyczyło to framework Scrum. Zgodnie z koncepcją nauki sztuk walki, Shu-Ha-Ri najpierw potrzebujesz poznać podstawy.

Żeby dobrze omówić pracę z wymaganiami w Agile potrzeba przynajmniej jednego dnia szkoleniowego, wypełnionego ćwiczeniami praktycznymi. Ci, którzy byli na moich szkoleniach, wiedzą, o czym mówię. Poruszę tutaj najważniejsze zagadnienia i rozwięję kontrowersje, jakie narosły wokół tematu. Są narzędzia i koncepcje, które niewątpliwie w cyklu życia oprogramowania pomogą zespołowi utrzymać kierunek działań i sprawdzać słuszność podjętych decyzji. Jakie dokładnie? Dowiesz się w dalszej części artykułu.

### Co zmienia Agile?

Z punktu widzenia wymagań Agile wprowadza kilka znaczących zmian. Przyjrzyjmy się im po kolei.

bardziej odległych Sprintów nie ma sensu i byłaby stratą czasu. Praca nad wymaganiami odbywa się tutaj zgodnie z zasadą „Just In Time” zaczerpniętą z Lean i następuje w etapach. W związku z tym Rejestr Produktu będzie przypominał swoim kształtem górną lodową. Na czubku góry będziemy mieli małe elementy (np. User Story) opracowane wystarczająco i gotowe do implementacji w kolejnych 1-2 Sprintach. Elementy zaplanowane na kolejne iteracje będą tym większe i bardziej ogólne, im ich implementacja jest oddalona w czasie. Rejestr Produktu ciągle się zmienia i niektóre elementy mogą nigdy nie zostać zaimplementowane, a inne zupełnie się zmienią.

Skąd wiadomo, czy User Story jest opracowana wystarczająco, żeby ją zaimplementować w Sprincie? Właściciel Produktu i Zespół Developerski wspólnie uzgadniają ten standard i określają go jako Definicje Gotowości. Powinny się znaleźć tutaj takie elementy jak: zgodność z szablonem, Warunki Satysfakcji, kroki demonstracji, makietka ekranu, szkic przepływu danych itp. Jednak w żadnym wypadku „User Story” nie powinna narzucać rozwiązania i zawierać szczegółów technicznych.

Zdecydowanie mniej negatywnych myśli przywodzi na myśl fakt **zbliżenia klienta lub przedstawiciela biznesu do zespołu**. Stały i bezpośredni kontakt z biznesem to elementy, które na pewno wpływają korzystnie na jakość wymagań. Klient, czy użytkownik końcowy to źródło informacji zwrotnej, dzięki której zespół buduje to co jest potrzebne i ma wartość.

Jak już wcześniej wspomniałem, **nie ma czasu ani potrzeby na dogłębną analizę wymagań i projektowanie z góry**. Opracowujemy tyle ile potrzeba i kiedy potrzeba. Pojawiają się zatem pytania o to, skąd wiemy jaki poziom szczegółowości jest potrzebny i jak planować dalszą przyszłość. Odpowiedem na te pytania w dalszej części artykułu.

### Nowa forma wymagań

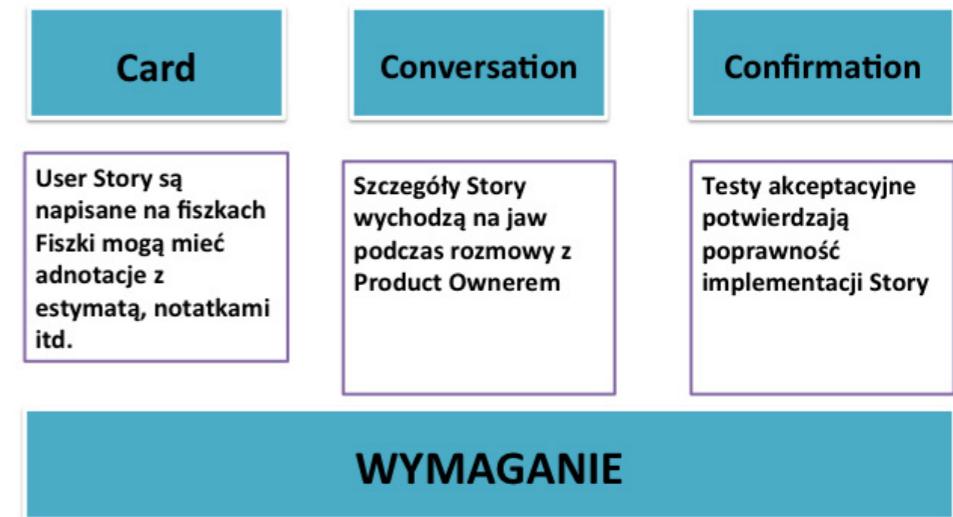
Scrum Guide nie mówi nam za dużo na temat wymagań. Właściwie mówi bardzo niewiele. Rejestr Produktu (ang. Product Backlog) jest jedynym źródłem wymagań i zawiera uporządkowaną listę wszystkiego, co jest potrzebne w produkcie oraz wszystkich zmian wymaganych w tym produkcie. Lista nigdy nie jest kompletna, i będzie się zmieniać. Rejestr Produktu jest złożony z wszystkich funkcjonalności, zadań i poprawek jakie trzeba

wykonać dla Produktu, który buduje zespół. Każda z tych rzeczy nosi nazwę: Elementu Rejestru Produktu (ang. Product Backlog Item) i posiada przynajmniej minimum takie właściwości jak: wartość, oszacowanie, opis. Koniec i kropka.

Przyjęło się, że w Scrum korzystamy z narzędzia wprowadzonego w Programowaniu Ekstremalnym, którym jest User Story. User Story ma na celu skupienie uwagi na realnym użytkowniku systemu i rozpoczęcie dyskusji na temat, co jest potrzebne i co można zrobić, że zaspokoić potrzebę tego użytkownika lub rozwiązać jego problem. Tylko tyle i aż tyle.

Ciekawostką samą w sobie jest to, że tak naprawdę to nie sama User Story tworzy wymaganie. User Story jest pretekstem do rozpoczęcia rozmowy na temat wymagania i doprecyzowania wymagania poprzez testy akceptacyjne, które potwierdzą właściwą implementację. Ten koncepcja została sformułowana przez Roniego Jeffrie jako 3C.

### 3C – Ron Jeffrie



Rysunek 1. Koncept 3C autorstwa Ron'a Jeffrie

Słyszałem o przypadkach, gdzie przejście z tradycyjnych wymagań na User Story polegało na komendzie wyszukaj „System powinien...” i zamień na „Jako użytkownik, chcę...”. Język polski ze swoją gramatyką nie ułatwia takiej migracji i dobrze, bo nie o to chodzi. Scrum Guide wyraźnie mówi, że Rejestr Produktu jest jedynym źródłem wymagań, więc niedopuszczalne jest umieszczenie linków do wymagań w Sharepoint, czy innym systemie. Nikt nie będzie skakał w tą i z powrotem, żeby upewnić się, że rozumie wymaganie i niczego nie pominął. To zwykła strata, która powinna być wyeliminowana zgodnie z filozofią Lean.

Są przynajmniej dwie popularne metody pisania User Story. Jedna z nich jest bardziej znana - jest to forma często określana, jako Podstawowa struk-

tura User Story, przedstawiona w ramce.

### Podstawowa struktura User Story

**Jako** <konkretny użytkownik, Persona, rola w systemie>, **chcę** <potrzeba>, **żeby** <problem do rozwiązania, cel do osiągnięcia>.

### Warunki Satysfakcji

Równie ciekawą i mniej znaną alternatywą jest stosowanie struktury pochodzącej z **Five Ws**<sup>1</sup>, czyli Who, When, What, Where, Why, która jest stosowna na przykład w dziennikarstwie. Po polsku będzie to Kto, Co, Kiedy, Gdzie, Dlaczego. Zobacz w ramce jak wygląda struktura takiej User Story.

### Struktura User Story z Five Ws

**Jako** <kto> <kiedy> <gdzie>, **chcę** <co>, **ponieważ** <dlaczego>.

### Warunki Satysfakcji

W dalszej części artykułu będziemy skupiali się na pierwszej, bardziej popularnej formie. Zauważ, że nieodzownym elementami User Story są odpowiedzi na pytania: Kto potrzebuje funkcjonalności?, Co to za funkcjonalność? i Dlaczego jest potrzebna?

Przyjrzyjmy się, co dokładnie wpisujemy w poszczególne elementy

Jako

- Persona;
- Konkretny użytkownik;
- Ewentualnie rola w systemie;

Chcę

- Potrzeba;
- Problem;
- Otwarte pytanie;
- Wyraź znaczenie dla klienta;
- Nie może to być rozwiązanie np.: chcę pole tekstowe;

Ponieważ/Żeby

- Powód;
- Wartość dla klienta;
- Często reprezentuje prawdziwy problem;

Porównajmy teraz wymagania w formie IEEE 830 i User Story, żeby zobaczyć efekt i korzyści płynące z używania tej struktury.

Zapraszam Cię do ćwiczenia, które polega na określeniu, jaki produkt Zespół ma zbudować

w kolejnej iteracji na podstawie wymagań. Po przeczytaniu każdego zdania, zastanów się, co to jest.

- Produkt powinien mieć silnik benzynowy dużej mocy.
- Produkt powinien mieć 4 koła.
- Produkt powinien mieć kierowcę.
- Produkt powinien mieć stalową karoserię.
- Produkt powinien być czerwony.

Niech zgadnę, to może być Ferrari, furgonetka dostarczająca mleko, samochód opancerzony do przewozu kosztowności, quad i kilka innych rzeczy.

Sprawdźmy jak szybko określisz, co Zespół ma zbudować czytając poniższe User Story.

*Jako właściciel działki rekreacyjnej, chciałbym móc kosić trawę szybko i łatwo, żeby nie tracić czasu na pracę podczas wypoczynku.*

*Jako właściciel działki rekreacyjnej, chciałbym siedzieć w wygodnej pozycji kosząc trawę, żeby nie mieć bólu pleców wieczorem.*

Szybko domyślisz się, że to jest kosiarka do trawy, prawda? I to jest jedna z zalet User Story. Od razu wiadomo, kto potrzebuje tej funkcjonalności, jaki ma problem lub potrzebę i jak sobie wyobraża rozwiązanie. Może się okazać, że jest wiele sposobów rozwiązania tego problemu lub, że istnieje już sposób zaspokojenia tej potrzeby. Ten sposób może nie być idealny, ale wystarczający na tą fazę rozwoju produktu.

Co się stanie, jeśli pominiemy niektóre elementy? Sprawdźmy.

*Jako użytkownik chcę ponownie zarezerwować lot.*

Czego tutaj brakuje? Brakuje problemu do rozwiązania lub celu do osiągnięcia i konkretnego użytkownika. Może to być użytkownik, który lata często na tej samej trasie i chce, żeby system mu podpowiadał ostatnio wybrane opcje, może to być użytkownik, który ma kilka ulubionych miejsc docelowych i chciałby wybierać je z listy, może to być użytkownik korporacyjny, który chce dodać kolejne osoby do podróży służbowej. Zauważ jak będą różniły się implementacje techniczne ze względu na faktyczną potrzebę użytkownika.

*Jako Babcia Zosia\* chcę ponownie zarezerwować ostatnio wybrany lot, żeby uniknąć wypełniania całego formularza wyszukiwania i przechodzenia przez wszystkie kroki rezerwacji.*

\*Babcia Zosia to osoba latająca zawsze na tej samej trasie, odwiedzająca wnuki w Anglii.

*Jako kierowca ciężarówki chcę widzieć zaplanowaną trasę.*

Czego tutaj brakuje? Mamy wprawdzie kierowcę ciężarówki, ale nie wiemy, po co mu ta mapa trasy, czyli brakuje potrzeby lub problemu. I znowu możemy się domyślać, że kierowca chce zobaczyć czas przejazdu pomiędzy punktami trasy. A może chce zobaczyć natężenie ruchu na trasie? Może chce zobaczyć samą listę punktów, gdzie dostarcza towar? A może chce zobaczyć spodziewany czas przejazdu i aktualny, żeby wiedzieć czy wyrabia plan? A może chce ...? Jednak każde takie domyślanie się to założenie, które może się zemścić na Zespole.

*Jako kierowca ciężarówki chcę widzieć planowany czas dojazdu do kolejnego punktu na trasie i numer telefonu kolejnego klienta, żeby móc skontaktować się z klientem zanim dojadę.*

Z kilku względów dobrze jest pisać i sortować User Story na tak zwanych fiszkach (ang. index cards). Rozmiar fiszki ogranicza ilość informacji, jaką da się zapisać, a zatem ogranicza rozmiar User Story. Fiszki mogą być przekazywane z ręki do ręki i każdy może dopisać kolejne Warunki Satysfakcji. To bardziej angażuje grupę niż obserwowanie jednej osoby wpisującej tekst, który jest wyświetlany przez projektor. Wszelkie sortowanie i grupowanie takich fiszek jest też dużo łatwiejsze i bardziej interaktywne. Na koniec można wprowadzić zmiany do systemu przechowywania naszego Product Backlog, na przykład Atlassian Jira. Spójrz na przykładową User Story.

order 22

8pt

### USER STORY 51

*Jako klient księgarni internetowej, chcę mieć możliwość uiszczenia płatności prosto z mojego konta, żeby móc zapłacić za zakupy nie posiadając karty kredytowej.*

Rysunek 2. User Story na fiszce

Jakie elementy są wyróżnione? Mamy tutaj wszystkie elementy, które wymaga od nas Scrum, żeby taką Story uznać za pełnoprawne Product Backlog Item, czyli Opis, Porządek na liście (Order 22) i Oszacowany rozmiar (5 punktów) i dodatkowo podane jest Id User Story używane w systemie przechowywania Backlog'u (USER-STORY-51).

W formie User Story można także przedstawić wymagania zupełnie techniczne takie jak konieczność zainstalowania oprogramowania, aktualizacji wersji oprogramowania, czy implementację interfejsu.

Jako system weryfikacji płatności chcę otrzymywać wszystkie transakcje w XML, żeby móc przekazać środki z karty klienta na konto właściciela sklepu.

Jako tester, chcę API dla bazy danych, żeby mogłem łatwo usuwać całe transakcje, kiedy testuję system.

Jako użytkownik systemu chcę aktualizacji biblioteki jQuery do najnowszej wersji, żeby móc bez przeszkód korzystać z systemu w nowej wersji przeglądarki Firefox.

Teraz przejdźmy do drugiej strony naszej fiszki, czyli spójrzmy na Warunki Satysfakcji (ang. Acceptance Criteria), lub jak kto woli Warunki Akceptacji. Tutaj zaczyna się precyzowanie ustaleń i określanie działania systemu. Testy Akceptacyjne zapisane jako Warunki Satysfakcji mają za zadanie potwierdzenie, że Story została zaimplementowana tak, jak spodziewał się tego klient. Kiedy zacznie się implementacja, programiści zaczną pisać kod spełniający te warunki, a testerzy zaczną pisać Przypadki Testowe rozszerzające te warunki oraz wynikające z metod testowania, np. warunki brzegowe, tabele decyzyjne i tak dalej. Warunki Satysfakcji są wysokopoziomowymi testami akceptacyjnymi i nie zastępują pełnego testowania funkcjonalności. Jak dobierać metody i priorytety testowania? Pełna odpowiedź na to pytanie wykracza poza ramy omawianych tutaj zagadnień. Teraz powiem Ci krótko: „To zależy od ryzyka”. Acceptance Criteria mogą być zapisane w postaci pytań takich, jak „Czy mogę zabrać kartę z bankomatu po trzech nieudanych próbach?”, notkach o tym co przetestować „Przetestuj niepoprawny numer CCV karty”, lub w postaci odpowiedzi na zadane pytania „Karta jest blokowana po trzech nieudanych próbach i zostaje w bankomacie”. Zobacz poniżej jak to może wyglądać dla naszej przykładowej User Story.

## Nowe role

Żeby promować samoorganizację i stworzyć wspólne poczucie odpowiedzialności za wyniki pracy, w Scrumie mamy trzy role: Scrum Master, Właściciel Produktu (ang. Product Owner) i Developer. Jak to wpływa na wymagania i kto jest za nie odpowiedzialny?

**Scrum Master** jest odpowiedzialny za procesy związane ze Scrum i przestrzeganie zasad framework. **Developer** jest odpowiedzialny za współpracę w Zespole, organizację swojej pracy i dostarczenie działającego oprogramowania na koniec Sprintu. **Właściciel Produktu** jest odpowiedzialny za dostarczanie wymagań i zapewnienia ich dobrego zrozumienia w zespole. Jako przedstawiciel biznesu powinien również powiedzieć Zespołowi Deweloperskiemu jakie są priory-

<sup>1</sup> [http://en.wikipedia.org/wiki/Five\\_Ws](http://en.wikipedia.org/wiki/Five_Ws)

tety kolejnych User Story z punktu widzenia ich wartości dla biznesu i realizowania celów organizacji. Z kolei Zespół jest odpowiedzialny za szacowanie wielkości User Story. Dlaczego? Ponieważ jako osoby, które będą wykonują pracę wiedzą ile ona zajmie i które elementy są skomplikowane do dostarczenia, a które proste.

Hej, ale nie widzę tutaj roli Analityka Biznesowego, co się z nim stało? Deweloper to każda osoba, która wykonuje pracę w celu dostarczenia potencjalnie gotowego do wydania kawałka wartościowego oprogramowania. Zatem możliwe jest, że Analityk to jeden z Developerów. Tak samo Tester będzie określany jako Developer, ale będzie wykonywał pracę w swojej dziedzinie specjalizacji. Analityk z uwagi na obszar specjalizacji może zostać także Właścicielem Produktu. Takie podejście ma sens, jeżeli to on lub ona faktycznie podejmie decyzje co do określenia wymagań. Jeżeli nie jest to osoba z wystarczającą władzą w organizacji i musi pytać o zdanie i decyzje inną osobę, to taki Właściciel Produktu jest jedynie proxy pomiędzy prawdziwym PO, a Zespołem Deweloperskim. Czy jest to efektywne rozwiązanie? Najczęściej nie, bo tworzy tylko zbędny bufor. Analityk jako proxy sprawdzi się dobrze jako Właściciel Produktu, jeżeli Zespół Scrum pracuje z zewnętrznym kli-

- można zapłacić przez mTransfer
- można zapłacić przez iPKO
- można zapłacić przez PayPal
- klient może zrezygnować z płatności
- płatność jest anulowana po 3 nieudanych próbach
- po anulowaniu płatności klient wraca widoku zamówienia
- w przypadku anulowania wyświetlony jest odpowiedni komunikat błędu

Rysunek 3. Warunki Satysfakcji dla User Story na fiszce

entem.

No dobrze, skoro już otworzyliśmy ten temat, to odpowiedzmy sobie na pytanie, jakie cechy posiada dobry Właściciel Produktu.

Szukamy osoby, która spełnia następujące kryteria:

- zna domenę biznesu,
- ma jasną Wizję Produktu,
- potrafi zarządzać zaangażowaniem interesariuszy,
  - ma kontakt z końcowymi użytkownikami systemu,
  - nie boi się podejmowania decyzji,
  - ma określone zdanie i nie zmienia go w trakcie Sprintu,
  - ma autorytet do samodzielnego podejmo-

wania decyzji związanych z Produktem, Pojawia się też pytanie, kto pisze User Story? Jest to odpowiedzialność Właściciela Produktu, ale Zespołu może go w tym wspierać, a Scrum Master uczyć jak wypełniać obowiązki. Omawianie, szacowanie i dzielenie User Story wspiera negocjację wymagań.

## Kiedy opracowywać wymagania i do jakiego poziomu?

Wymagania w świecie metod zwinnych mają pewną hierarchię. Na samej górze znajduje się Wizja Produktu, która powinna być zakomunikowana i jasna dla Zespołów tworzących Produkt. Na tej postawie będą określone tematy do zrealizowania i cele kolejnych wydań. Następnie tematy będą podzielone na Epiki lub od razu na User Story. Każdy Sprint powinien realizować Cel Sprintu zgodny z Celem Wydania i zawierać User Story, których dostarczenie jest niezbędne.

Tematy i Epiki to elementy grupujące Rejestru Produktu, które pomagają określić priorytety i wartość biznesową funkcjonalności.

Epik to bardzo duża Story, która wymaga dopracowania i trudno jest prawidłowo oszacować jej wielkość. Prawdopodobnie w trakcie Pielęgnacji Rejestru zostanie rozbity na kilka User Story. Epik zwykle reprezentuje funkcjonalność, którą trzeba zaimplementować.

Temat to grupa User Story podobna pod pewnym względem. Możemy grupować User Story ze względu na użytkownika, którego dotyczą, obszar aplikacji itp.

W Scrum mamy trzy oficjalne momenty, kiedy pracujemy nad Rejestrem Produktu, ale to nie oznacza, że nie ma pracy wykonywanej pomiędzy. Tak więc mamy Pielęgnację Rejestru Produktu, która zajmuje do 10% czasu Zespołu przeznaczonego na pracę w Sprincie. Jest to czynność, która musi zajść i może być dowolnie zaplanowana w trakcie Sprintu. Celem pielęgnacji jest opracowywanie kolejnych elementów Rejestru Produktu tak, żeby mieć gotowe User Story w ilości od 1 do 2 Sprintów. Taki proces jest potrzebny przede wszystkim, żeby Zespół nie musiał czekać na wsad do nowych Sprintów oraz, żeby nie próbować robić tego na Planowaniu Sprintu, bo takie postępowanie niepotrzebnie wydłuży spotkanie. W trakcie pielęgnacji Zespół Developerski i Właściciel Produktu szacują, doprecyzowują i dzielą User Story.

Planowanie Sprintu, to moment kiedy Zespół Developerski i Właściciel Produktu wybierają i ne-

gocują zakres User Story, które zobowiązują się dostarczyć w Sprincie. Bierzemy pod uwagę przedkość Zespołu określana jako Velocity oraz dostępność członków zespołu w tym konkretnym Sprincie. Dla jasności dodam, że Zespół Developerski jest w 100% zaangażowany w pracę nad jednym Produktem lub ewentualnie projektem, a mówiąc o dostępności mam na myśli święta państwowe, zwolnienia lekarskie i urlopy.

W trakcie Sprintu może nadal dochodzić do renegocjacji elementów w wypadku, gdy Zespół Developerski trafił na przeszkody lub dowiedział się nowych faktów o wykonywanej pracy i wie, że nie zdąży dostarczyć wszystkich User Story czy wręcz odwrotnie, ma pojemność na dostarczenie większej ilości, niż zaplanował. W tym drugim przypadku pomaga nam to, że wcześniej przygotowaliśmy pulę User Story w ilości 1-2 Sprintów i teraz z łatwością możemy dać Zespółowi kolejną z góry Rejestru.

Ostatnim momentem, gdzie formalnie zajmujemy się Rejestrem Produktu jest Przegląd Sprintu. Przejrzenie tego, co Zespół dostarczył, zmiany w organizacji i demo rozwiązania mogą wpływać na zmianę priorytetów lub zmianę w elementach rejestru.

Równolegle, przez cały czas Właściciel Produktu powinien pracować z interesariuszami i nieustannie opracowywać Rejestr Produktu, żeby zawsze prezentował on aktualny stan oraz reprezentował realizację celów organizacji.

## Podsumowanie

Z punktu widzenia osoby pracującej w sztywnej strukturze korporacyjnej i przyzwyczajonej do pierwotnej roboty, praca w metodach miękkich może wydawać się mało poważna i ryzykowna. Jednak przy bliższym poznaniu Agile zyskuje ze względu na łatwość reakcji na zmiany, utrzymywane tempo pracy i bliski kontakt z klientem. Jak wi-

**Krystian Kaczor**, to doświadczony coach, trener i konsultant, który na co dzień łączy ze sobą dwa światy, twardych umiejętności i technologii oraz miękkich umiejętności i psychologii. Jego wieloletnie doświadczenie na międzynarodowych projektach (m. in. Szwecja, Holandia i Iran) pozwoliło na zdobycie wyjątkowych i wszechstronnych umiejętności z obu tych obszarów. W swojej pracy kieruje się maksymą, że jedną miarą postępu są mierzalne rezultaty.

W trakcie 10 lat pracy w branży IT, Krystian zdobył wszechstronne doświadczenie w całym cyklu wytwarzania oprogramowania. Pracował jako programista, wdrożeniowiec, tester, wsparcie klienta, Scrum Master, Test Manager i Project Manager, dzięki czemu patrzy na oprogramowanie oraz proces jego wytwarzania z kilku perspektyw i znajduje wspólny język zarówno z biznesem, jak i IT.

Jest jednym z niewielu coachów i trenerów nadal biorących aktywny udział w projektach. Współpracuje z firmami, których logo jest łatwo rozpoznawalne w wielu krajach usprawniając Zespoły, procesy, projekty i całe programy. Jako Agile Coach buduje i prowadzi Zespoły Agile, przeprowadza transformacje organizacji i ulepsza wdrożone procesy. Posiada doświadczenie w pracy z Zespołami w zakresie od małych, lokalnych zespołów do geograficznie rozproszonych korporacji, pracujących w modelach skalowanego i rozproszonego Agile. Prowadzi szkolenia z obszaru Agile, Scrum, testowania oraz umiejętności miękkich i rozwoju osobistego. Szkolił i konsultował Zespoły z Polski, Holandii, Niemiec i Rosji. Autor książki "Scrum i nie tylko. Teoria i praktyka w metodach Agile" wydanej przez PWN, publikacji po polsku i po angielsku w czasopismach oraz na stronach internetowych. Współzałożyciel czasopisma c0re. Prelegent na polskich i zagranicznych konferencjach z obszarów Agile i Testowania. Certified Scrum Professional, CSM, PSM I, PMI-ACP, ISTQB Advanced Level - Test Manager, Master Practitioner NLP, ICF Associate Certified Coach i Erickson Certified Professional Coach.

dać nadal potrzebne będą procesy i jasne zasady wykonywania pracy. W nowej rzeczywistości, w której obecnie znajdują się organizacje, w których pracujemy, potrzebne są metody, które wykorzystują potencjał profesjonalnych pracowników i pozwalają na łatwą reakcję na zmiany. Pisanie szczegółowych wymagań przez pojedyncze osoby i próba zgadywania jak ma wyglądać efekt końcowy zawsze przegra w konkurencji z samoorganizującymi się zespołami i dostosowywaniem się do informacji zwrotnej po zaprezentowaniu kolejnej wersji działającego oprogramowania.

Osoby zainteresowane dokładnym poznaniem metod pracy z wymaganiami odsyłam do mojej książki „Scrum i nie tylko. Teoria i praktyka w metodach Agile”.

## Bibliografia

Krystian Kaczor, *Scrum i nie tylko. Teoria i praktyka w metodach Agile*, PWN, 2014

Dean Leffingwell, *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise (Agile Software Development Series)*, Upper Saddle River, NJ, Addison-Wesley 2011

Scrum Guide wersja Lipiec 2013, Ken Schwaber, Jeff Sutherland, <http://www.scrumguides.org>

M. Cohn, *User Stories Applied: For Agile Software Development*, Upper Saddle River, NJ, Addison-Wesley 2004.



Firma szkoleniowo-konsultingowa założona przez Krystiana Kaczora działa w Polsce i zagranicą zdobywając uznanie i zaufanie klientów od 2010 roku. QAgile specjalizuje się w obszarach metod zwinnych i zapewnienia jakości oprogramowania.

- ✓ Szkolenia z metod Agile
- ✓ Szkolenia REQB, IBBQA, ISTQB
- ✓ Przygotowanie do egzaminów PSM I i PMI-ACP
- ✓ Autorskie szkolenie Scrum i nie tylko - teoria i praktyka metod Agile
- ✓ Autorskie szkolenie Agile Testing
- ✓ Doradztwo w zakresie wprowadzania metod Agile w organizacji
- ✓ Egzaminy ISTQB, REQB, IQBBA, IBUQ
- ✓ Agile Coaching
- ✓ Prowadzenie zespołów Scrum, Kanban i XP
- ✓ Testowanie oprogramowania



Wejdź na stronę firmy, wybierz szkolenie dla siebie, zapytaj o ofertę dopasowaną do potrzeb!

[www.qagile.pl](http://www.qagile.pl)



# I Ogólnopolski Turniej Inżynierii Wymagań

Warszawa, 17-18 marca 2015

[re-challenge.pl](http://re-challenge.pl)



[facebook.com/turniejrechallenge](https://facebook.com/turniejrechallenge)



[twitter.com/rechalance](https://twitter.com/rechalance)

Pięć konkursów, ciekawe nagrody, udział zespołowy lub indywidualny.

„Inżynieria wymagań nie tylko decyduje o powodzeniu lub niepowodzeniu projektów, ale jest też fascynującą przygodą intelektualną, emocjonalną i duchową”

# Relacja z wydarzenia: II Kongres Profesjonalistów IT



2 października 2014 roku, w Rzeszowie, odbył się II Kongres Profesjonalistów IT, przebiegający w tym roku pod hasłem „Praktyczne zastosowanie nowych technologii”. W wydarzeniu tym wzięło udział blisko 150 osób, w tym praktycy biznesu i eksperci branży informatycznej oraz komunikacyjnej z całej Polski. Miło nam poinformować, że w gronie partnerów Kongresu znalazło się Stowarzyszenie Inżynierii Wymagań.

Program Kongresu Profesjonalistów IT tworzyło szereg wystąpień plenarnych, dotyczących między innymi roli narzędzi informatycznych we wspieraniu zmian w firmach oraz budowaniu przewagi konkurencyjnej w biznesie, nowych metod i technologii, które stosowane są na coraz szerszą skalę w samym IT oraz mają wpływ zarówno na sferę zawodową, jak i osobistą każdego z nas. Poniżej wymieniamy niektóre z nich.

Już na samym wstępnie, Anna Sieńko - Dyrektor ds. Projektów Strategicznych IBM Europa Centralna i Wschodnia, przedstawiła wyniki badań, wg których aż 81% potencjalnych klientów, sprawdza w internecie opinie o produkcie, którego zakupu chce dokonać oraz o firmie, która ten produkt sprzedaje. Właściwy wizerunek firmy w internecie przesądza zatem często o jej wynikach sprzedaży. Jak budować taki wizerunek mówił w swojej prelekcji „Budowa i promocja biznesu w sieci - dobre praktyki oparte na doświadczeniach zespołu Brand24” Założyciel i Prezes, Brand24 S.A. - Michał Sadowski.

Anna Sieńko wspomniała także o tym, że suma wyników poszukiwań każdego z nas w internecie stanowi jednocześnie informacje o naszych upodobaniach i preferencjach. Dzięki tak zebranym

danym, trafia do każdego z nas coraz bardziej precyzyjna, personalizowana reklama. O internetowym śledzeniu i o tym jak wykorzystać go w biznesie i co na to Kowalski, mówił szerzej Maciej Zagórowski, Współzałożyciel, Dyrektor Interaktywny FEB Sp. z o.o.

Podczas Kongresu dyskutowano także o dobrych praktykach zwiększących efektywność realizacji projektów informatycznych. „Jak zwiększyć efektywność wdrożenia projektów IT?”, „Jak rozmawiać z klientami w różnych sytuacjach?”, „Jak poradzić sobie w złożonej rzeczywistości projektowej i osiągnąć z tego korzyści?“ - to tylko niektóre pytania, na które odpowiadali w czasie swoich prelekcji eksperci, a jednocześnie praktycy biznesu: Rafał Roszak - Dyrektor Marketingu e24cloud, Beyond Sp. z o.o., Wojciech Murzyn - Erdo-pracownia dobrej komunikacji oraz Bogdan Michałek - Założyciel i Prezes Zarządu, BMM Sp. z o.o.

Krystian Kaczor, ICF Associate Certified Coach, Qagile, poruszył temat „Agile w dużej organizacji“, omawiając przykłady ze swojej bogatej praktyki zawodowej. Jarosław Sokolnicki - Business Development Manager, Microsoft Polska w swojej prelekcji „Cloud Computing - Projekt informatyczny czy biznesowy?“ mówił o korzyściach płynących z przeniesienia swoich produktów i usług w chmurę publiczne, prywatne i hybrydowe. Michał Bartyzel - Trener, konsultant, BNS IT Sp. J. w prelekcji „Najważniejsza rzecz, którą każdy profesjonalista IT musi wiedzieć o współpracy z klientami“, przekazywał tajniki skutecznej komunikacji z klientami w fazie zbierania wymagań.

Z ramienia Stowarzyszenia Inżynierii Wymagań, z prelekcją pt. „Inżynieria wymagań - modna nowinka, fanaberia dostawcy, czy żywotny interes klienta“ wystąpiły Ewa Brzeska i Małgorzata Wawryń. Przekonywały one zgromadzonych słuchaczy jak ważne i istotne z punktu widzenia skutecznej i zakończonej sukcesem realizacji projektów IT, jest sprawne pozyskanie od klienta kompletnych wymagań dotyczących rozwiązania. Mamy nadzieję, że to wystąpienie przekonało uczestników Kongresu, że inżynieria wymagań jest niezbędną składową procesu tworzenia oprogramowania, a właściwie wyspecyfikowanie wymagań leży w żywotnym interesie zarówno dostawcy, jak i odbiorcy oprogramowania.



1 Fotografie – Organizatorzy Kongresu Profesjonalistów IT



Najbardziej przewrotne i dające wiele do myślenia było chyba ostatnie wystąpienie w bloku prelekcji. Michał Korba - Menedżer ds.IT w Vena Art, CEO i CO-Founder w iWisher.pl pytał publicznie „Czy wszyscy będziemy cyborgami, czyli dlaczego kiedyś każdy wszczępi sobie chipa?“. Konkluzja z tej prelekcji była jednak dość nieoczywista: otóż właściwie nie potrzebujemy chipa... Wielu z nas ma w kieszeni smartfon, który wyposażony w odpowiednie oprogramowanie, zbiera nieustannie, w czasie rzeczywistym, tysiące i miliony bajtów danych o jego posiadaczu. Które miejsca i kiedy odwiedzamy, z kim wymieniamy maile i co w nich piszemy, do kogo dzwonimy, komu robimy zdjęcia, a więc z kim przebywamy, jaka jest siatka naszych znajomych z portali społecznościowych, co i kiedy kupujemy, jakie mamy hobby – to tylko niektóre dane, jakie może zbierać o nas i zbiera producent urządzenia. Czy wszyscy jesteśmy świadomi jak dużo informacji dostarczamy o sobie?

Kongres Profesjonalistów IT podsumowała debata panelowa z udziałem publiczności. W gronie pięciu zaproszonych do debaty ekspertów-panelistów znalazła się Ewa Brzeska, reprezentująca nasze Stowarzyszenie.

Pytania, z którymi przyszło się zmierzyć podczas dyskusji jej uczestnikom, dotyczyły przede wszystkim roli i znaczenia nowych technologii w życiu każdego z nas. Czy jeszcze to my zarządzamy własnym czasem i preferencjami, czy też nowe technologie kreują nasze potrzeby oraz sterują



naszymi zachowaniami? Młodsze pokolenia korzystają z wszelkich nowinek technologicznych wyjątkowo intensywnie - jakie to ma skutki teraz, a jakie może mieć w przyszłości?

Blisko godzinna burzliwa dyskusja nie przyniosła jednoznacznych odpowiedzi na powyższe pytania. Z jednej strony wszyscy zgodzili się z tym, że nowe technologie to nowe możliwości: docieranie do klientów z precyzyjną, personalizowaną reklamą oraz możliwość korzystania ze szczególnie ciekawych ofert, dopasowanych dokładnie do preferencji każdego z nas. Dzięki temu, że duża część społeczeństwa jest obecnie ciągle online, można wykorzystywać we wzajemnej komunikacji wiele kanałów, w tym media społecznościowe. Sieci kontaktów są obecnie nieporównywalnie bardziej rozległe niż kiedyś. To niewątpliwe plusy. A jakie są minusy? Młodsze pokolenia coraz częściej wolą spędzać czas w świecie wirtualnym, zamiast w realnym. Siedząc samotnie przed komputerem młodzi ludzie liczą znajomych na portalach społecznościowych, lecz ilość kontaktów nie jest równoznaczna z ich jakością. Jeszcze kilkanaś-



cie lat temu naturalnym środowiskiem, w którym dzieci i młodzież spędzały czas wolny była przestrzeń publiczna. Obecnie kilka, a nawet kilkanaście godzin dziennie ta grupa wiekowa spędza przed komputerem lub ze smartfonem czy tabletem w ręku. Taki tryb życia może mieć w przyszłości nie najlepsze skutki zdrowotne. A personalizowana reklama? Z punktu widzenia konsumenta jest cenna, jeśli dociera wówczas, kiedy jest rzeczywiście potrzebna. Jeśli jednak zaczyna kreować potrzeby, wpływać mocno na decyzje zakupowe, to zaczyna sterować naszym życiem.

Na koniec dyskusji wszyscy zgodzili się z tym, że nowych technologii należy używać zarówno w sferze biznesu, jak i w życiu osobistym – są one potrzebne oraz stwarzają nowe, cenne możliwości. Nie należy jednak pozwalać, aby nowe technologie, modne nowinki, czy trendy, zdominowały obie sfery naszej aktywności. Niech technologia będzie jedynie narzędziem, z którego będziemy świadomie korzystać.

# REQ MAGAZYN

