FORMY PROSUMPCJI W SYSTEMACH INFORMATYCZNYCH

Wiesław WOLNY

Streszczenie: Tradycyjny model tworzenia i użytkowania systemów informatycznych polegał na tym, że oprogramowanie wytwarzane było przez producenta, a korzystający z niego użytkownicy byli tylko klientami. Ruch Open Source spowodował rozwinięcie się nowego modelu, w którym osoby korzystające z oprogramowania mogli je współtworzyć, modyfikować i wpływać na jego rozwój. Zjawisko to wpisuje się w szersze ramy zjawisk prosumpcyjnych obserwowanych w gospodarce w ostatnich latach. Artykuł przedstawia różne formy zjawisk prosumpcyjnych, które można zaobserwować w dziedzinie tworzenia i korzystania z systemów informatycznych ze szczególnym uwzględnieniem prosumpcji w systemach informatycznych zarządzania.

Słowa kluczowe: Prosumpcja, Open Source, Crowdsourcing, Systemy Informatyczne Zarządzania.

1. Prosumpcja

Termin prosumpcja powstał z połączenia słów: produkcja (pro-) oraz konsumpcja (-sumpcja). Po raz pierwszy użył go w swojej książce "Trzecia fala" amerykański socjolog i futurolog - Alvin Toffler [1]. Autor poświęcił temu pojęciu cały rozdział swego dzieła: "Prosument - klient nowego typu". Pod pojęciem prosumpcja rozumiał on przesuwanie procesu produkcji ze sfery gospodarki oficjalnie uznanej przez ekonomistów do sfery gospodarki lekceważonej. Prosumpcję widział on zarówno w organizacjach, które skupiają osoby z podobnymi problemami, jak i w luźnych powiązaniach między ludźmi, którzy wymieniają porady, obserwacje i doświadczenia.

Toffler rozumiał prosumpcję, jako nową formę interakcji między producentem a konsumentem, przede wszystkim, jako przesunięcie pewnych zadań na konsumenta – zgodnie z koncepcją zrób to sam. Zwiększające się zaangażowanie konsumenta polega na wykonywaniu przez niego czynności, które wcześniej wykonywał dla niego ktoś inny. Wiele osób bardzo sobie ceni pracę rąk własnych, jest to dla nich powód do dumy.

Toffler zaobserwował, że produkcja standardowych wyrobów już nasyciła rynek, podstawowe potrzeby zostały zaspokojone. Aby kontynuować dalszy wzrost, przedsiębiorstwa muszą zainicjować proces kastomizacji, w postaci masowej produkcji bardzo spersonalizowanych produktów. Jednakże, w celu osiągnięcia wysokiego poziomu dostosowania produktów, celowe jest wciągnięcie konsumentów w proces produkcji, zwłaszcza na etapie określania wymagań projektowych. W pewnym sensie jest to tylko przedłużenie lub rozszerzenie rodzaju relacji, jaki mają od wielu dziesięcioleci klienci z np. architektami. W niedawno opublikowanych pracach Toffler rozszerzył tą i wiele innych idei do warunków 21 wieku. Można w nich znaleźć koncepcję prosumpcji rozwiniętą już do skali globalnej [2].

W 1995 roku D. Tapscott ponownie wprowadził ten termin określając prosumpcję, jako pragnienie (lub życzenie) posiadania dóbr, które są zgodne z wyobrażeniem klienta. Autor

uważa, że poprzez swój indywidualny wybór i działanie nabywcy stają się współtwórcami konkretnego produktu [3].

Inaczej nieco prosumpcję widział Bill Quain [4]. Rozumiał on ją, jako pewnego rodzaju program lojalnościowy. W ramach niezależnej partnerskiej współpracy producent zapewnia lojalnym konsumentom atrakcyjny system rabatów, a także udział w zyskach z obrotu za pomoc w promowaniu jego produktów. Dzięki takiej formule biznesowej, środki wydane przez nich na niezbędne produkty wracają do z powrotem i stają się źródłem dochodu.

Prawie dziesięć lat po publikacji Tofflera, Tapscott i Williams w swojej pracy [5] spopularyzowali prosumpcję nadając jej status podstawowej działalności w nowej gospodarce, gdzie dominuje współpraca i relacje peer-to-peer pomiędzy jej uczestnikami.

Wśród działań charakterystycznych określających prosumpcję wymieniają [5]:

- Dopasowanie do potrzeb użytkowników: dopasowanie produktów do specyficznych zastosowań, ale także głębsze zaangażowanie w proces projektowania,
- wyzbywanie się kontroli: produkty, jako platforma własnych innowacji, bez względu na pozwolenie lub jego brak,
- narzędzia dla konsumentów i angażowanie kontekstu: produkty są traktowane jako podstawa do eksperymentowania,
- partnerstwo: użytkownicy są traktowani jako partnerzy, a nie – klienci,
- dzielenie się owocami: prawa prosumentów do tego, co stworzyli i by na tym dodatkowo zarabiali, co zwiększa tempo konsumenckiego współtworzenia.

Pomimo kilku dekad istnienia termin prosumpcja od niedawana jest ponownie opracowywany teoretycznie. G. Ritzer i N. Jurgenson w swoim artykule [6], twierdzą, że prosumpcja stała się istotną cechą Web 2.0.

Wielu autorów rozpatrując prosumpcję głównie z punktu widzenia konsumenta, zauważa, że prosument robi coś więcej niż dopasowanie, czy personalizowanie towarów. Tworzą społeczności prosumenckie, w których dzielą się informacjami o produktach, wspólnie pracują nad projektami, wymieniają się wskazówkami dotyczącymi produktu, przydatnymi narzędziami itp. Z kolei, z punktu widzenia producenta, prosumpcja jest nowym sposobem na osiągnięcie ich podstawowego celu: zwiększenie zysków.

W dziedzinie oprogramowania prosumpcja jest rozumiana, jako idea, zgodnie z którą konsument dobra, czyli użytkownik oprogramowania jest jednocześnie jego producentem lub współproducentem, czyli ma możliwość jego rozwijania, modyfikacji, testowania itp.

2. Prosumpcja w systemach informatycznych

Można przyjąć, że prosumpcja w systemach informatycznych ma swoje źródło w kilku ideach. Pierwszą z nich jest koncepcja oprogramowania Open Source. Innymi, badanymi współcześnie formami prosumpcji, mogą być: Crowdsourcing, Consumer Involvement, hacking produktów, czy też Collective Intelligence.

2.1. Open Source

Termin ten najczęściej określa oprogramowanie komputerowe o powszechnie dostępnym kodzie źródłowym. To właśnie odróżnia je od komercyjnego modelu oprogramowania, tzw. closed source – kod źródłowy, umożliwiający dokonywanie zmian w programie, nie jest chroniony przez licencję czy inne prawa własności przed nieuprawnionym dostępem. Koncepcja otwartego oprogramowania stosowana jest w branży IT już od ponad trzydziestu lat, stąd kojarzona jest głównie z programami

komputerowymi i pracą programistów. Stanowi ona sposób prowadzenia współpracy w zakresie badań i rozwoju produktu końcowego dający możliwość użytkownikom oprogramowania w dowolny sposób je modyfikować i rozwijać. Powstanie oprogramowania open source było wyrazem sprzeciwu przeciwko bardzo dużym sumom, jakie korporacje żądały za licencje do aplikacji oraz przeciwko uniemożliwianiu wprowadzania w nich zmian.

Wolne oprogramowanie według manifestu GNU [7] to kwestia wolności użytkowników do uruchamiania, kopiowania, rozpowszechniania, analizowania, zmian i ulepszania programów. Dokładniej mówiąc, oznacza to, że użytkownikom programu przysługują cztery podstawowe wolności:

1. Wolność do uruchamiania programu, w dowolnym celu (wolność 0).
2. Wolność do analizowania, jak działa program, i zmieniania go, aby robił, co i jak potrzebujecie (wolność 1). Warunkiem koniecznym jest dostęp do kodu źródłowego.
3. Wolność do rozpowszechniania kopii, byście mogli pomóc innym ludziom (wolność 2).
4. Wolność do udoskonalania programu i publicznego rozpowszechniania własnych ulepszeń, dzięki czemu może z nich skorzystać cała społeczność (wolność 3). Warunkiem koniecznym jest tu dostęp do kodu źródłowego.

Oprogramowanie jest wolnym, jeśli jego użytkownicy mogą skorzystać z wszystkich powyższych wolności. Zatem mają swobodę rozpowszechniania kopii, zmodyfikowanych bądź oryginalnych, za darmo bądź pobierając opłatę za dystrybucję. Dzięki zagwarantowaniu takich wolności, setki, tysiące, a obecnie nawet miliony osób pracują nad udoskonaleniem systemu oraz aplikacji wspomagających, tworząc ogromny potencjał intelektualny. Eksperci wskazują, że rosnąca popularność modelu open source wynika z jego przewagi nad tradycyjnym, zamkniętym sposobem produkcji [8].

E. S. Raymond [9] sformułował zasady postępowania wyjaśniające rozwój programów Open Source, które bardzo dobrze, z punktu widzenia prosumpcji, interpretuje Gajewski [10]:

1. „Dobrej jakości praca nad programem rozpoczyna się od tego, że programista drapie się tam, gdzie go swędzi" – tłumacząc dosłownie metaforę Reymonda, każda praca powinna rozpocząć się wtedy, kiedy istnieje konkretny problem (potrzeba).
2. „Dobrzy programiści wiedzą, co pisać. Wielcy wiedzą, co przepisać (i ponownie wykorzystać)" – tą zasadą Reymond daje do zrozumienia, że warto korzystać z pomysłów innych osób, ponieważ żadna jednostka nie jest w stanie wymyślić wszystkiego sama.
3. „Pomyśl o tym, by się go pozbyć; i tak to zrobisz" – mówiąc inaczej, często dzieje się tak, że nie rozumie się problemu do momentu, kiedy podejmuje się próbę jego rozwiązania, natomiast za drugim razem rozumie się go na tyle, by zrobić to dobrze. Warto więc być przynajmniej raz przygotowanym na rozpoczęcie prac od samego początku.
4. „Jeśli masz odpowiednie nastawienie, spotkają cię interesujące problemy" – tylko osoby z odpowiednią motywacją będą zaangażowane w pracę.
5. „Kiedy tracisz zainteresowanie programem, twoim ostatnim obowiązkiem jest przekazać go kompetentnemu następcy" – Reymond w tej zasadzie nawołuje do dzielenia się wiedzą, pomysłami, jak i efektami swojej pracy i przeciwstawia się ochronie praw własności intelektualnej.

6. „Najkrótszą i najłatwiejszą drogą w kierunku szybkiego rozwoju kodu i poprawienia jego błędów jest traktowanie użytkowników jako osób uczestniczących w projekcie" – zasada ta odnosi się do współpracy i kooperacji jako najefektywniejszej metody doskonalenia oraz podkreśla znaczenie użytkowników jako źródła wiedzy.
7. „Udostępniaj nowe wersje wcześnie i często. I słuchaj swoich klientów" – dzięki takiemu sposobowi działania można szybko reagować na zmiany potrzeb klientów i odnaleźć powstające błędy. Tylko klient wie, gdzie tkwi problem, ponieważ to jego potrzeba ma być najlepiej zaspokojona.
8. „Jeśli tylko środowisko użytkowników i testerów wersji Beta oprogramowania będzie odpowiednio duże, niemal każdy problem zostanie szybko zdefiniowany i znajdzie się jego rozwiązanie" – ta zasada pokazuje, że większa liczba podmiotów pracujących przy danym projekcie dysponuje większą wiedzą (wiedza kolektywna) i jest w stanie zrobić więcej niż jednostka. Zasada ta została nazwana Prawem Linusa.
9. „Zręcznie zaprojektowane struktury danych i bezładny kod działają znacznie lepiej niż odwrotna kombinacja" – planowanie według Reymonda powinno zostać zastąpione spontanicznością i swobodą działania.
10. „Jeśli traktujesz swoich testerów wersji beta oprogramowania jak swój największy skarb, staną się twoim największym skarbem" – jeśli w tym przypadku uznamy beta-testerów za klientów, to ta zasada staje się w XXI w. niemal truizmem, ponieważ to oni będą w przyszłości posługiwać się danym programem i to oni decydują tym, czy dany wynalazek staje się innowacją.
11. „Najlepsze, co może nas spotkać oprócz własnych dobrych pomysłów, to słuchanie dobrych pomysłów użytkowników. Czasem to drugie jest lepsze" – ta zasada w praktyce oznacza, że prócz własnych pomysłów trzeba również korzystać z rozwiązań innych i dlatego też jest to mocno powiązane z zasadą nr 2.
12. „Często najbardziej uderzające i nowatorskie rozwiązania mają u swych źródeł zrozumienie, że pierwotne postrzeganie natury problemu było błędne" – to zasada informuje nas o tym, że w momencie, kiedy natrafimy na opór, który jest ciężko pokonać, może lepiej zdefiniować jeszcze raz problem, a nie na siłę implementować pierwotne rozwiązanie.
13. „Doskonałość projektu osiąga się nie wtedy, kiedy nie ma co dodać, lecz raczej kiedy nie ma co odjąć" – ta zasada pokazuje, że innowacja nie musi być czymś bardzo skomplikowanym i nadmiernie rozbudowanym, a często jest bardzo prostym
    i trywialnym rozwiązaniem danego problemu czy zaspokojeniem potrzeby.
14. „Każde narzędzie powinno być użyteczne w przewidywalny sposób, jednak naprawdę wielkie narzędzie znajduje nieoczekiwane zastosowania" – Reymond zachęca do eksperymentowania, w dodatku w sposób niestereotypowy i bardzo spontaniczny.
15. „Pisząc jakiekolwiek oprogramowanie pełniące rolę pośrednika, należy podjąć wysiłki, by nie naruszać strumienia danych i nigdy nie odrzucać żadnej informacji, chyba, że program zostanie do tego zmuszony przez odbiorcę".
16. „Jeśli kompletność języka nie zbliża się do poziomu Turinga, dodatki składniowe mogą bardzo pomóc".
17. „System bezpieczeństwa jest tak silny, jak silna jest jego tajemnica. Należy się wystrzegać pseudotajemnic" – Reymond nawiązuje do otwartości, ponieważ

tajemnice i nadmierna ochrona stają na przeszkodzie innowacyjności.

18. „Aby rozwiązać interesujący problem, należy zacząć od znalezienia problemu, który nas osobiście interesuje" – dzięki tej zasadzie Reymond pokazuje, jak w przypadku realizacji innowacyjnego projektu odpowiednie osoby samodzielnie przypisują się do odpowiedniego zadania.
19. „Zakładając, że koordynator projektu dysponuje medium przynajmniej tak dobrym jak Internet i potrafi zarządzać projektem bez stosowania przymusu, wiele głów ma z pewnością większą wartość niż jedna" – Reymond w tej zasadzie nawiązuje do idei wiedzy rozproszonej F.A. Hayeka, która neguje planistyczne podejście na rzecz podejścia opartego na działaniu indywidualnych jednostek.

Powyższe zasady decydują o tym, że cały proces ma charakter otwarty, a jego najważniejszym elementem są jednostki kooperujące. Reymond twierdził, że każdy projekt, aby mógł być tworzony i doskonalony, musi opierać się na wiedzy samych użytkowników, czyli osób, które będą korzystały z danego rozwiązania. Ta zasada jest zbieżna z ideą von Hippela, który twierdził, że użytkownik tylko wtedy przyłączy się do danego projektu i będzie wprowadzał innowacje, gdy widzi w tym własny interes. Nie interesuje go, czy pozostali uczestnicy mają te same czy inne potrzeby, i kieruje się tylko własną korzyścią, maksymalizując w ten sposób korzyści dla całej grupy [11].

Do najbardziej znanych przykładów prosumpcji w dziedzinie, jaką jest tworzenie oprogramowania, należy system operacyjny Linux, którego kod źródłowy został stworzony przez Linusa Torvaldsa studenta Uniwersytetu w Helsinkach. W 1991 roku Torvalds zaprezentował pierwszą wersję Linuxa, a informację o tym systemie operacyjnym umieścił na jednym z uniwersyteckich serwerów, zachęcając innych programistów do jego wspólnego rozwijania. Innowacja Torvaldsa nie polegała na stworzeniu jądra całego systemu, a bardziej na stworzeniu modelu postępowania przy jego rozwoju. Jego inicjatywa w tym przypadku zaowocowała powstaniem społeczności skupionej wokół dużego projektu, mającego na celu stworzenie systemu operacyjnego z kodem źródłowym, dowolnie wykorzystywanym, modyfikowanym i rozpowszechnianym. Z czasem do projektu włączyły się największe światowe firmy informatyczne, tj. IBM, HP czy Intel. Przedsiębiorstwa te zapewniają zarówno wynagrodzenie dla programistów pracujących nad systemem, jak i niezbędne narzędzia technologiczne czy wsparcie marketingowe. Powstał również zarząd, którego zadaniem jest podejmowanie decyzji o kierunkach dalszego rozwoju Linuxa, jednak mimo wszystko Linux Inc. nie stał się tradycyjną firmą – nie ma własnej siedziby, nie publikuje raportów rocznych itp. Projekt nadal jest wspólnym przedsięwzięciem realizowanym przez pracowników kilkunastu firm i tysiące indywidualnych programistów. [8]

Przykładem darmowego oprogramowania jest nie tylko Linux, ale także m. in. znany pakiet biurowy OpenOffice, czy Firefox – jedna z najpopularniejszych przeglądarek internetowych na świecie. Podobna sytuacja ma miejsce przy tworzeniu i rozwoju innego wolnego oprogramowania jak np. systemów zarządzania treścią (Drupal, PHP-Nuke, Plone), Wikie (MediaWiki), baz danych (Firebird, MySQL, PostgreSQL), środowisk graficznych (GNOME, GNUstep), pakietów biurowych (OpenOffice, KOffice), składu tekstu (TeX, LyX, LaTeX), wspomagania nauczania (Moodle), przeglądarek internetowych (Mozilla) i wielu innych.

2.2. Crowdsourcing

Za kolejny nurt prosumpcji w oprogramowaniu może być uważany crowdsourcing Jeffa Howe [12]. Pod tym pojęciem Autor rozumiał wykorzystywanie zewnętrznego źródła

wiedzy organizacji, jakim są społeczności, do rozwiązywania problemów [13]. Głównym założeniem tej koncepcji jest fakt, że grupa posiada szerszą wiedzę niż jednostki. Wykorzystuje się tu tzw. efekt synergiczny, a więc fakt, że współdziałanie różnych jednostek przynosi lepsze rezultaty niż suma ich pojedynczych działań.

Crowdsourcing oznacza czerpanie wiedzy, pomysłów i inspiracji „z tłumu", W odróżnieniu od działów R&D w firmach, to konsumenci często najlepiej wiedzą, czego tak naprawdę potrzebują. Zaoferowanie im możliwości wypowiedzenia się na temat produktu, którego używają, lub też takiego, którego chcieliby używać, daje gwarancję świeżego spojrzenia na zagadnienie, a tym samym możliwość powstania wielu udoskonaleń trafionych w potrzeby grupy docelowej.

W ciągając konsumentów w procesy produkcyjne można osiągnąć wiele korzyści. Najbardziej oczywiste to:

- problemy mogą być rozwiązane przy względnie niskich kosztach, na ogół bardzo szybko,
- płaci się w zależności od rezultatu, czasem wręcz nagradzanie się pomija,
- organizacja dociera do szerszego grona talentów, niż tylko tego, które jest obecne w samej organizacji,
- poprzez słuchanie społeczności, organizacje zyskują wgląd z pierwszej ręki do potrzeb i pragnień klientów,
- społeczność może poczuć, że współtworzy markę, co skutkuje poczuciem posiadania poprzez przyłożenie się do współpracy.

Crowdsourcing jest szczególnie popularny w dziedzinie wytwarzania oprogramowania. Systemy informatyczne rozwijane z pomocą tego modelu to np.:

1. Każde oprogramowanie rozwijane poprzez testowanie open beta – wytwórcy oprogramowania często udostępniają wszystkim zainteresowanym testerom wersje beta swoich systemów.
2. cTuning.org – infrastruktura i repozytorium dotyczące analizy kodów programów i dostrojenia kompilatorów w celu lepszej optymalizacji programów.
3. Mindpixel – online projekt sztucznej inteligencji mający na celu budowę bazy wiedzy stwierdzeń true/false.
4. Navfree – pierwsza darmowa aplikacja nawigacyjna na iPhone. Ma ponad 10 milionów użytkowników, którzy tworzą, usuwają błędy i usprawniają nawigację.
5. OpenStreetMap – wolna edytowalna mapa świata mająca ponad 100 000 współtwórców. Tworzenie, utrzymanie geoprzestrzennych danych jest bardzo pracochłonne i drogie w tradycyjnym modelu. W tym projekcie jest to robione przez crowdsourcing.
6. Waze – darmowa nawigacja na telefony komórkowe, która wykorzystuje crowdsourcing do wyznaczania tras i informacji o ruchu drogowym.

Do zarządzania wytwarzaniem oprogramowania w tym modelu powstała platforma ComCrowd, ułatwiająca współpracę pomiędzy firmami a wytwórcami niezatrudnionymi, działającymi na zasadzie freelance.

2.3. Consumer Involvement

Consumer Involvment, tłumaczony jako zaangażowanie konsumenta jest koncepcja marketingową zbliżoną do crowdsourcingu. Polega na identyfikacji i rozwoju możliwości zaangażowania klientów w proces produkcyjny produktu, najczęściej w fazach projektowania, marketingu, sprzedaży, obsługi klienta itp. Stopień zaangażowania może być tak daleki, że klient staje się częścią procesów dostarczania produktów na rynek.

2.4. Hacking produktów

Hacking w nowym wydaniu to modyfikowanie zakupionych produktów, za czym stoi chęć ich ulepszenia, dostosowania do swoich potrzeb. Tak jak hackowanie w informatyce – hackowanie produktów odbywa się bez wiedzy i zgody marki. Przynajmniej na początku. Potem zależy, co marka z danym pomysłem, a często bywa, że udogodnieniem, zrobi.

Najbardziej znane przykłady hackingu produktów to modyfikacje urządzeń firmy Apple i Sony Playstation. Użytkownicy produktów firmy Apple proces doskonalenia swoich produktów posunęli do tego stopnia, że sprawa wymknęła się aż z pod kontroli macierzystej firmy. Użytkownicy stworzyli oprogramowanie umożliwiające instalowanie aplikacji bez użycia oficjalnego serwisu iTunes. Szereg aplikacji niezgodnych z polityką firmy dostępnych jest w na alternatywnej platformie dystrybucyjnej niezależnej od firmy Apple. Podobnie konsumenckie przeróbki konsoli Playstation firmy Sony pozwoliły na uzyskanie funkcjonalności niezaplanowanej i niezaprojektowanej przez producenta. Pozwoliły one na instalowanie na urządzeniu innych systemów operacyjnych, wykorzystanie wydajnego procesora Cell do wykonywania dowolnych obliczeń, łączenia kilku konsoli w celu utworzenia maszyny wieloprocesorowej, czy też kopiowania zawartości płyt SACD, które jest niemożliwe do wykonania w żaden inny sposób.

2.5. Collective Intelligence

W dosłownym tłumaczeniu mowa o zbiorowej inteligencji. Tym razem koncepcja podobna do crowdsourcingu przybiera zupełnie nowy wymiar. Serwisy typu Collective Intelligence gromadzą jednocześnie problemy wielu firm. Dzięki internetowi możliwa jest komunikacja globalna, a co za tym idzie znalezienie i pozyskanie pomysłów od fachowców z całego świata. Rozwiązanie konkretnego problemu staje się łatwiejsze z uwagi na zaangażowanie niespotykanej dotąd liczby osób. W tym wypadku możemy mówić wręcz o „rynku pomysłów”.

Przedmiotem obrotu na takim rynku są dobra niematerialne [14]:

- wiedza ekspertów zewnętrznych,
- wiedza innych przedsiębiorstw,
- wiedza uczestników rynku (np.: klientów),
- produkty związane z wiedzą.

Dzięki otwarciu się na otoczenie organizacje mogą uzyskać szansę zaangażowania osób o wyjątkowych kwalifikacjach, czy na podjęcie współpracy w celu znalezienia innowacyjnych rozwiązań.

3. Prosumpcja w systemach informatycznych zarządzania

Systemy informatyczne zarządzania (SIZ) w dominującej liczbie przypadków dostarczane są, jako gotowe rozwiązania, gdzie użytkownik dostaje z góry przygotowany zbiór funkcjonalności. Oczywiście w procesie projektowania wytwarzania lub wdrożenia oprogramowania, użytkownicy uczestniczą w projekcie wdrożeniowym i mają możliwość zgłaszania swoich potrzeb. Jednak po zakończeniu wdrożenia system jest zamrażany i dalsza eksploatacja odbywa się już tylko według wcześniej przyjętych scenariuszy.

Powszechne slogany zasad projektowania głoszą, że "kluczowym czynnikiem wdrożenia system informatycznego zarządzania jest zaangażowanie użytkowników w projektowanie takiego systemu". Nie gwarantuje to jednak działania systemu zgodnie z oczekiwaniami odbiorców. Przy takim podejściu tworzy się obszerne tomy specyfikacji, następnie następują konsultacje, później wdrożenie, a na końcu walka o zgodność ze

specyfikacją. Nawet w przypadku poprawnego przeprowadzenia wszystkich procesów, specyfikacja, i tym samym system, odzwierciedla wizję systemu z przed wdrożenia.

Modyfikacje systemu po wdrożeniu, które praktycznie zawsze okazują się konieczne, wymagają ponownego zaangażowania wytwórcy oprogramowania lub firmy wdrażającej. Klient, który zakupił oprogramowanie musi zwrócić się ponownie do dostawcy oprogramowania i zlecić mu wykonanie przeróbek, najczęściej w postaci dodatkowych raportów. Gotowe systemy informatyczne zarządzania przetwarzają dane zwykle w bardzo poprawny sposób, gromadzą przetwarzają, zapewniają spójność itp., ale w większości, jeżeli nie we wszystkich, tych wdrożeń zachodzi w praktyce taka sytuacja, że użytkownikowi nie wystarczają gotowe raporty, gdyż są po prostu sztywne. Chciałby mieć możliwość ich przerabiania, obrabiania, aby otrzymać wyniki w pożądanym układzie.

W tradycyjnym podejściu do wytwarzania oprogramowania użytkownik nie miał takich możliwości. Wytwórcy oprogramowania obwarowali możliwości zmian w trosce o spójność danych, „jednolity obraz firmy", „system ma wiedzę, którą dostarczyli mu konsultanci" itp. Nie bez znaczenia był również fakt osiągania korzyści z każdej modyfikacji systemu.

Można jednak zauważyć, że dostawcy SIZ zmieniają jednak stanowisko i zamiast walczyć ze zjawiskiem idą na współpracę dostarczając narzędzi do samodzielnego przetwarzania danych. W tej sytuacji pojawia się pole dla rozwoju działań prosumenckich w SIZ. Prosument – użytkownik SIZ powinien mieć możliwość samodzielnego tworzenia systemu przynajmniej poprzez modyfikację raportów wyjściowych. Na tym poziomie dostępna jest i najczęściej kończy się możliwość ingerencji w system informatyczny zarządzania. Doświadczeni prosumenci żądają jednak więcej. Chcą mieć dostęp do modelu systemu, i tym samym poprzez narzędzia prototypowania móc go zmieniać. Na tak głęboką ingerencję w system jednak niewielu wytwórców zezwala. Pojawiają się już rozwiązania informatyczne dające użytkownikom – prosumentom głębszą możliwość wpływania na działanie systemu, bądź nawet możliwość tworzenia systemu poprzez składanie go z potrzebnych modułów-funkcji. W tej dziedzinie prowadzone są już prace naukowo-wdrożeniowe i osiągane pierwsze efekty.

Korzyści wynikające z rozwiązań prosumenckich w SIZ są wielorakie. Pierwszą, najbardziej oczywistą jest fakt, że użytkownik np. sam tworzy raporty. Ponieważ on wie najlepiej, co ma w nim być i może go dowolnie modyfikować wraz ze zmianami potrzeb informacyjnych. System informatyczny jest żywy, ma odzwierciedlać zmieniające się warunki funkcjonowania firmy i otoczenia, poprawki, więc muszą być wykonywane na bieżąco. Robi je użytkownik. W standardowym modelu wdrożenia SIZ, zatrudnia się specjalistów projektantów, programistów, bądź zleca modyfikację systemu. W nowym podejściu dostawca systemu o takiej poprawce nawet się nie dowiaduje. Rodzi to głębsze zmiany w sposobie funkcjonowania przedsiębiorstwa. O zmianach nie dowiaduje się nie tylko dostawca oprogramowania, najczęściej nie znajduje ona odzwierciedlenia w dokumentacji systemu. Użytkownik otrzymuje, więc nie tylko prawo do zmian, przejmuje również odpowiedzialność za formuły raportu.

Warunkiem rozwoju prosumpcji w SIZ są narzędzia pozwalające tworzyć systemy bez programowania na niskim poziomie. Większość współczesnych narzędzi zaczyna spełniać te wymogi.

5. Podsumowanie

Koncepcja prosumpcji okazuje się doskonałym uzupełnieniem tradycyjnego podejścia do wytwarzania i korzystania z systemów informatycznych, jakim są zhierarchizowane

wewnętrzne działy badawczo-rozwojowe lub korzystanie z usług firm wdrożeniowych. Dzięki jej zastosowaniu można nie tylko minimalizować koszty związane z tworzeniem i funkcjonowaniem systemów, ale również przyspieszyć cały proces, a przez to zwiększyć tempo rozwoju technologicznego. Dodatkowym argumentem przemawiającym za zastosowaniem tej koncepcji jest fakt, że przy danym projekcie może zostać zaangażowanych znacznie więcej osób niż w tradycyjnym podejściu, a dzięki temu zwiększy się również ilość zaangażowanej wiedzy.

Literatura

1. Toffler A.: Trzecia fala. Państwowy Instytut Wydawniczy, Warszawa, 1997.
2. Toffler A.: Revolutionary Wealth. Knopf Doubleday Publishing Group, New York, 2006.
3. Gach D.: Pozyskiwanie i wykorzystywanie wiedzy klientów. E-Mentor nr 23, Warszawa, 2008, s. 57-60.
4. Quain B.: Era pro-sumenta. InterNet Services Corporation of Poland Sp. z o.o., Warszawa, 2002.
5. Tapscott D., Williams A.: Wikinomia. O globalnej współpracy, która zmienia wszystko. WAiP, Warszawa, 2008.
6. Ritzer G., Jurgenson N.: Production, Consumption, Prosumption. The nature of capitalism in the age of the digital 'prosumer'. Journal of Consumer Culture 10:1;13-26, 2012-12-24.
7. Filozofia Projektu GNU. www.gnu.org, 2012-01.
8. Siwińska J.: Open Source – zastosowanie otwartego podejścia w procesach innowacyjnych. http://www.pi.gov.pl/PARP/chapter_86197.asp?soid=D37B2ADA8D354866860DCA0A5C36156A, 2012-12-21.
9. Raymond E.S.: The Cathedral and the Bazaar. O'Reilly Media, 1999.
10. Gajewski Ł.: Zastosowanie koncepcji Otwartej Wynalazczości w procesach innowacyjnych.http://www.bibliotekacyfrowa.pl/Content/37111/03_Lukasz_Gajewski.pdf, 2012-12-21.
11. von Hippel E.: The sources of innovation. Oxford University Press, New York, 1988.
12. Howe J.: The Rise of Crowdsourcing. „Wired". czerwiec 2006.
13. Howe J.: Crowdsourcing. Why the power of the crowd is driving the future of business. Three Rivers Press, New York, 2008.
14. Gilbert P., Steffen R., Kai R.: Zarządzanie wiedzą w organizacji. Oficyna Ekonomiczna, Kraków, 2004, s. 118.

Dr Wiesław Wolny
Katedra Informatyki
Uniwersytet Ekonomiczny w Katowicach

40-287 Katowice, ul. 1 Maja 50
tel./fax: (0-32) 257 72 89
e-mail: wieslaw.wolny@ue.katowice.pl

Publikacja finansowana z Grantu nr 4100/B/H02/2011/40 projekt badawczy własny Nr NN115 410040, Prosumpcja produktów informatycznych wspomagających zarządzanie w organizacjach gospodarczych.