Modelowanie ekosystemów informacyjnych dla innowacyjnych społeczności programistycznych

Sebastian Grabowski
Orange Polska S.A.
Mieczysław Muraszkiewicz
Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Abstrakt

Cel/Teza: Celem artykułu jest przedstawienie generycznego modelu ekosystemu informacyjnego wspierającego innowacyjne społeczności programistyczne, zwłaszcza takie, które tworzą się i działają w środowiskach miejskich.
Koncepcja/Metody badań: Metodę badań oparto na analizie wybranych, zagranicznych sieciowych ekosystemów wspierających społeczności programistyczne (m.in. Deutsche Telekom Development Garden i AT&T Developers Portal) oraz współtworzonych przez Sebastiana Grabowskiego społeczności Open Middleware 2.0 Community i BIHAPI. Zanalizowano strukturę tych ekosystemów i ich *modus operandi*. Wnioski wyciągnięte z analizy, z naciskiem na czynniki wspomagające procesy innowacyjne, posłużyły do zbudowania generycznego modelu ekosystemu informacyjnego, wspierającego innowacyjne społeczności programistyczne
Wyniki i wnioski: Przeprowadzone badania ekosystemów oraz doświadczenia własne zebrane przez autorów w wyniku współpracy ze społecznościami programistycznymi w ramach różnych projektów i programów, w tym programów Open Middleware 2.0 Community i BIHAPI oraz licznych hackathonów, potwierdziły istotną rolę, jaką już odgrywają i mają do odegrania w przyszłości innowacyjne społeczności programistyczne. Okazało się, że szczególnie efektywnym i produktywnym modelem organizowania się i działania takich społeczności jest korzystanie z otwartych platform, a w szczególności z otwartego oprogramowania, otwartych danych oraz otwartych innowacji.
Zastosowanie praktyczne: Przedstawiony model generyczny ekosystemu informacyjnego wspierającego innowacyjne społeczności programistyczne może posłużyć jako punkt wyjścia do dalszych prac nad konstruowaniem, architekturą, funkcjami i zarządzaniem społecznościami programistycznymi, których znaczenie w tworzeniu innowacji technicznych i wspierania nowatorskich rozwiązań gospodarczych i społecznych, nabiera szczególnego znaczenia w kontekście wysiłków wyrwania się Polski z pułapki średniego rozwoju.
Oryginalność/Wartość poznawcza: Zaproponowany model ma charakter generyczny, co oznacza, że może być adaptowany do środowisk programistycznych o różnym profilu i celach. Jest on oparty na otwartych platformach dotyczących danych i oprogramowania oraz kładzie nacisk na funkcjonowanie w trybie charakterystycznym dla otwartych innowacji, sam wykazując nowatorskie podejście do funkcjonowania programistów. Model ma charakter otwarty, co tworzy solidną podstawę do dalszych prac nad modelowaniem i realizacją ekosystemów informacyjnych wspierających środowiska programistyczne. Wartość modelu została potwierdzona w praktyce, m.in. w czasie trwania programów Open Middleware 2.0 Community oraz BIHAPI.

Słowa kluczowe

API. Ekosystem informacyjny. Interfejs programowania aplikacji. Otwarte dane. Otwarte innowacje. Otwarte oprogramowanie. Społeczność programistyczna.

Otrzymany: 23 października 2017. Zrecenzowany: 24 listopada 2017. Zatwierdzony: 27 grudnia 2017.

1. Wstęp

Informatyka i jej zastosowania to dziedzina, która horyzontalnie przenika niemal wszystkie obszary współczesnego życia, niosąc ze sobą jakościowe zmiany w gospodarce, administracji, obronności, edukacji, służbie zdrowia, kulturze, rozrywce i generalnie w stylu życia. Rosnące moce obliczeniowe i transmisyjne sprzętu teleinformatycznego na niewiele jednak by się zdały, gdyby nie nieprzerwany i znaczący rozwój oprogramowania narzędziowego i aplikacyjnego oraz wręcz lawinowy wzrost liczby różnorakich aplikacji. Tylko w Stanach Zjednoczonych Ameryki Północnej w 2016 r. pracowało 3.97 mln twórców oprogramowania (Global Developer, 2017); szacuje się, że w Polsce jest ich obecnie około 250 tys. Programiści, podobnie jak inne społeczności, wytworzyli swą własną kulturę, z własnym stylem pracy, sposobem bycia i komunikowania się oraz z własnym slangiem. Świat programistów, ze względu na stałe kontakty z wysokimi technologiami i z uwagi na szybkie zmiany tych technologii oraz różnorodność tematyki, wymaga kreatywności, gotowości do ustawicznego uczenia się, zachęcając jednocześnie do innowacyjnego myślenia i działania.

Cechami tego świata, które są szczególnie istotne dla programistów i ich rozwoju, są otwartość i sieciowość. Programiści, nawet jeśli pracują w różnych zespołach czy organizacjach współpracują ze sobą, wymieniając się informacjami, doświadczeniami i elementami swojej pracy, zarówno jeśli chodzi o metody, jak i konkretne produkty (programy komputerowe, zbiory danych), tworząc w ten sposób sieci, które w wielu przypadkach stają się częściami większych ekosystemów. Jednym z powodów tej współpracy jest stały „głód" informacji, który towarzyszy indywidualnym programistom i zespołom programistycznym. Każda zatem inicjatywa, każdy system informacyjny, które mogą im pomóc w uzyskaniu relewantnych informacji i wiedzy wywołuje w środowiskach programistycznych zainteresowanie i chęć wykorzystania w bieżącej pracy. Niniejszy artykuł pokazuje, jak zaspokoić ten „głód" informacji i jednocześnie wesprzeć społeczności programistyczne w tworzeniu innowacyjnych rozwiązań. Można to osiągnąć tworząc ekosystemy informacyjne skierowane do programistów. Generyczny model takiego ekosystemu został przedstawiony w tym artykule. Jego prezentację poprzedzono charakterystyką procesów innowacyjnych oraz krótkim przeglądem istniejących ekosystemów, które zawierają komponenty przeznaczone dla programistów.

2. Procesy innowacyjne

Innowacje i innowacyjność są przedmiotem licznych debat, seminariów, publikacji, konferencji i badań naukowych. Obecność tematów poświęconych innowacyjności w dyskursie publicznym i branżowym wynika z tego, że innowacje w czasach postępującej globalizacji, rozwoju środków komunikacji, opartych na Internecie stacjonarnym i mobilnym,

narastającej konkurencyjności, przemożnego znaczenia rynków finansowych i ogromnych postępów nauki i techniki stały się jednym z głównych czynników pozwalających na uzyskanie przewagi porównawczej i/lub konkurencyjnej, a niekiedy wręcz czynnikiem decydującym o utrzymaniu się na rynku. W szerszym planie uważa się, że innowacyjność jest motorem nowoczesnej gospodarki, co prowadzi do konkluzji, że państwo powinno nie tylko tworzyć ramy prawne, finansowe, organizacyjne, edukacyjne i promocyjne do wspierania procesów innowacyjnych, ale powinno też uczynić zagadnienia innowacyjności stałym elementem swojej strategii i polityki gospodarczej i społecznej. Ten drugi wymiar, czyli wymiar społeczny, nie pojawia się tu przypadkowo – innowacyjność bowiem nie dotyczy wyłącznie spraw, urządzeń i systemów technicznych; jest dla niej także miejsce w opracowywaniu, realizacji i ulepszaniu procesów i instytucji społecznych i politycznych w skali makro, oraz w organizowaniu i wielowymiarowym wspieraniu społeczności i inicjatyw lokalnych.

Istnieje szeroka gama definicji terminu „innowacje" i wielość podejść do zagadnień innowacyjności. Za szczególnie trafne przyjmuje się w tej pracy krótkie określenie, które powiada, że innowacją jest nowy produkt, usługa, lub zastosowanie, które powstały w wyniku nowatorskiego połączenia elementów już istniejących, lub też nowe zastosowanie dostępnych urządzeń czy rozwiązań. Przykładem innowacji jest smartfon, który skonstruowano z wcześniej znanych i istniejących podzespołów, ale nie jest innowacją na przykład tranzystor, który słusznie ma status wynalazku, a więc czegoś zupełnie nowego w momencie swego pojawienia się w domenie elektroniki. Warto zatem odróżniać innowacje od wynalazków i odkryć naukowych. Te ostatnie są w swej naturze różne i od innowacji, i od wynalazków – odkrycia naukowe bowiem dotyczą zjawisk i procesów oraz ich opisu w postaci praw i zasad. Mogą one, rzecz prosta, być pomocne w pracy nad innowacjami, czy też stanowić inspiracje dla wynalazców, ale ich etiologia jest zupełnie inna – są one w gruncie rzeczy motywowane ciekawością poznawczą, a nie powodami utylitarnymi. Jeśli zaś chodzi o granicę pomiędzy innowacjami i wynalazkami, to bywa ona rozmyta i może być przedmiotem sporu. O ile jednak wynalazczość i odkrycia naukowe są w zasadzie procesami spontanicznymi, trudno poddającymi się kontroli i rygorystycznemu zarządzaniu (często są wręcz dziełem przypadku), to procesy innowacyjne można traktować jako działania „wstecznego" projektowania, które rozpoczyna się „od końca", to znaczy od określenia potrzeby na nowy przedmiot (produkt, usługę lub zastosowanie), czyli w gruncie rzeczy od specyfikacji finalnego przedmiotu, i następnie krok po kroku, drogą dekonstrukcji rozłożenie go na składniki elementarne, którymi są dostępne komponenty, procesy i technologie uwzględniając przy tym czynniki ergonomiczne, finansowe, kulturowe i inne. W świecie korporacyjnym procesy innowacyjne są zazwyczaj przedmiotem starannego planowania, zarządzania, kontroli, oceny i marketingu, czego znakomite przykłady dostarczają m.in. firmy Apple i Facebook.

W literaturze dotyczącej innowacyjności istnieje mnogość modeli procesów innowacyjnych. Ta mnogość *notabene* świadczy o tym, że nie istnieje jeden, uniwersalny model i jedno podejście do innowacyjności przydatne we wszystkich okolicznościach i dla wszystkich organizacji, co zresztą nie powinno dziwić, gdyż innowacyjność, to proces o dużej różnorodności i złożoności. Jedną z lekcji badań nad innowacjami jest to, że każdy projekt innowacyjny jest z natury jednostkowy, specyficzny i wymaga przystosowania ogólnych zasad i dobrych praktyk dotyczących innowacyjności do lokalnych uwarunkowań oraz do kultury organizacji, w której ma miejsce, uwzględniając przy tym trendy i ograniczenia narzucone przez otoczenie, którym w przypadku innowacji przeznaczonych do komercjalizacji jest

rynek. Jest jednak jedna rzecz wspólna dla udanych projektów innowacyjnych prowadzonych w świecie korporacyjnym i w organizacjach publicznych, a mianowicie dokładny plan działania i rygorystyczne kontrolowanie jego wykonania. Praktyka pokazuje, że wprowadzanie innowacyjnych rozwiązań często wywołuje opór środowiska, głównie z powodu konieczności zmiany przyzwyczajeń, nawyków, nauczenia się nowych procedur, czy wręcz na skutek naruszenia lokalnych relacji i interesów. Z tego powodu istotne jest, aby zarówno prace nad innowacjami, jak i wdrażanie innowacji miały wsparcie i mocną autoryzację grupy kierowniczej i zarządczej organizacji.

B. Jacobfeuerborn i M. Muraszkiewicz (2012) zaproponowali liniowy model procesu informacyjnego ze sprzężeniem zwrotnym. Jest to dobry punkt wyjścia do rozszerzenia procesu modelowania – oto jego przebieg: pracę nad innowacją (wszystko jedno czy będzie to nowy produkt, usługa, czy zastosowanie) rozpoczyna się od zdefiniowania „zadania innowacyjnego", które określone jest przez „potrzebę innowacyjną" zamawiającego. W zależności od natury i złożoności zadania jego język opisu jest bardziej lub mniej sformalizowany. Do wykonania zadania zostaje powołany zespół (w szczególności jednoosobowy), wyposażony w kompetencje dotyczące samej innowacji, jak i środowiska, do którego ma zostać wprowadzona. Zespół dokonuje morfologii innowacji, rozkładając jej przedmiot na czynniki prostsze oraz, stosując techniki generowania pomysłów i ich oceny (np. drogą burzy mózgów i filtrowania pomysłów), proponuje zbiór alternatywnych rozwiązań zadania innowacyjnego. Następnie przeprowadzana jest kwerenda badawcza w dostępnych zasobach informacyjnych, naturalnie z uwzględnieniem zasobów Internetu, w celu możliwie wszechstronnego pozyskania informacji o składnikach proponowanych rozwiązań (cena, patenty, licencje itp.) oraz – w przypadku innowacji przeznaczonej do komercjalizacji – informacji o sytuacji na rynku, do którego jest ona adresowana. Kolejnym etapem jest zbudowanie prototypu i jego ewaluacja. Po pozytywnym zakończeniu tego etapu innowacja gotowa jest do wdrożenia, w szczególności do komercjalizacji, jeśli ma charakter rynkowy. Trzeba zauważyć, że we wszystkich fazach procesu (z wyjątkiem fazy definiowania i komercjalizacji) możliwe jest cofnięcie się do fazy wcześniejszej, jeśli stwierdzono przeszkody lub błędy w pracy nad innowacją. W praktyce zaproponowany wyżej model może być przedmiotem różnego rodzaju rozszerzeń, uszczegółowień i mutacji w zależności od przeznaczenia. Jednak już w obecnej postaci dobrze modeluje na poziomie ogólnym trzy rodzaje procesów innowacyjnych, a mianowicie:

(1) Ambitne, znaczące innowacje, które powstają jako wynik dużych projektów naukowo-badawczych, prowadzonych w profesjonalnych laboratoriach i w ośrodkach badawczych finansowanych przez państwo lub korporacje. Są to przedsięwzięcia, w których mają miejsce badania naukowe, często angażujące ośrodki uniwersyteckie. Ponieważ badania naukowe są kosztowne i z definicji obciążone relatywnie dużym ryzykiem niepowodzenia, proces innowacyjny oparty na tych badaniach wymaga znacznych nakładów finansowych i jest przedsięwzięciem ryzykownym, narażonym na porażkę.
(2) Innowacje usprawniające istniejące produkty, usługi lub zastosowania. W środowisku korporacyjnym są to rutynowe działania, których celem są lokalne ulepszenia pozwalające na utrzymanie przewagi porównawczej na rynku. Poziom ryzyka i nakłady finansowe na takie działania są zazwyczaj niewielkie.
(3) Innowacje z przysłowiowego „garażu", realizowane przez pojedyncze osoby, startupy i małe/średnie przedsiębiorstwa. Są to niewielkie inicjatywy (choć, jak pokazuje

historia, mogące przynieść przełomowe produkty, czego sztandarowymi przykładami są komputer osobisty i wyszukiwarka Google), niewymagające dużych nakładów finansowych i złożonych form i procedur organizacyjnych.

Produkcja innowacji w społecznościach programistycznych opiera się na wariancie (c) wyżej przedstawionego modelu. Pojedynczy programiści lub niewielkie zespoły programistów, często tworzących startup, działają właśnie według modelu „innowacje z garażu". Jednakże istotne i wychodzące daleko poza klasyczny model innowacji z garażu jest to, iż uczestnicy społeczności programistycznej należą do ekosystemu, w którym szczególną rolę odgrywają przepływy informacji. Podmioty ekosystemu, w zależności od okoliczności, współpracują lub konkurują ze sobą, tworząc w ten sposób własną kulturę, której cechy przypominają opisy znane z amerykańskiej Doliny Krzemowej. Model ekosystemu informacyjnego społeczności programistycznych, z uwzględnieniem czynnika innowacyjnego, zostanie przedstawiony w dalszej części tego artykułu.

W odniesieniu do społeczności programistycznych szczególne znaczenie ma koncepcja *otwartych innowacji*. Otwarte innowacje, o czym przekonują autorzy artykułu (Hartmann & Trott, 2009), wprowadzono do dyskusji już w latach 70. XX w. (Griffiths & Pearson, 1973; Pearson et al., 1979), a potem do teorii i praktyki innowacyjności wprowadził je H. Chesbrough w pierwszej dekadzie XXI w. (Chesbrough, 2003; Chesbrough et al., 2006). W odróżnieniu od podejścia „ogrodu za murem" (ang. *walled garden*), praktykowanego powszechnie w ubiegłym stuleciu, według którego prace nad innowacjami prowadzone są w zamkniętym obszarze firmy czy w organizacji i mają status poufnych, głównie z powodu obawy przed konkurencją, podejście otwartych innowacji zakłada, że w procesie prac nad nowymi rozwiązaniami biorą udział podmioty zaproszone spoza środowiska inicjatora zadania innowacyjnego. Mogą to być laboratoria i ośrodki badawcze, organizacje pozarządowe, agencje regulujące rynek powołane przez państwo czy reprezentanci przyszłych użytkowników innowacji. Na tak rozumianej otwartej innowacyjności B. Jacobfeuerborn oparł swój model ekosystemu otwartej innowacyjności (Jacobfeuerborn, 2012). W modelu tym, oprócz wymienionych już uczestników, są jeszcze dwa elementy, a mianowicie „wcześni użytkownicy" oraz „fabryka innowacji". Ten pierwszy termin wprowadził do literatury E. M. Rogers (1962). Są to osoby i/lub organizacje, które jako pierwsze uzyskują – kupując na rynku lub otrzymując od innowatora na podstawie specjalnych relacji (na przykład do testowania) – produkt lub usługę innowacyjną. Rola wczesnych użytkowników jest szczególnie istotna, oni bowiem ukształtują opinie o produkcie, co wpłynie na sukces lub jego brak na rynku. Wczesnymi użytkownikami są niekiedy prosumenci, którzy współpracowali z innowatorem w czasie definiowania problemu i generowania pomysłów. Tu warto dodać, że choć znalezienie się w grupie wczesnych użytkowników może środowiskowo być przedmiotem swego rodzaju prestiżu (czego wymownym przykładem jest wczesne pozyskanie produktów firmy Apple), czy uzyskaniem przewagi porównawczej, to jednak wiąże się to z możliwością pewnego ryzyka finansowego i zawodu „moralnego" związanego z tym, że produkt nie spełnia oczekiwań, nierzadko bowiem bywa, że firmy pod naciskiem konkurencji wprowadzają na rynek produkty w wersji beta, a więc nie do końca przetestowanej, a nawet niekompletnej, co do funkcjonalności.

Drugi element, czyli „fabryka innowacji" odnosi się do specjalistów, środków technicznych, finansowych i organizacyjnych, które organizacja podejmująca zadanie innowacyjne stawia do dyspozycji zespołu innowacyjnego. B. Jacobfeuerborn wymienia tu m.in.: własne

zasoby informacyjne organizacji w postaci różnorakich baz danych oraz wiedzy i doświadczenia zawodowego pracowników organizacji, środki i techniki wywiadu gospodarczego, w tym zaawansowane heurystyki przeznaczone do wyszukiwania informacji w Internecie, metody analizy patentowej, a także narzędzia do prototypowania oraz systemy komunikacji z klientami: CRM (Customer Relationship Management System) oraz CEM (Customer Experience Management System), jeśli organizacja takowe posiada (Jacobfeuerborn, 2012).

Powyższa uwaga, dotycząca ryzyka związanego z innowacjami, zachęca do poczynienia szerszego komentarza. Otóż za sprawą swoistej „propagandy" na rzecz innowacyjności uprawianej przez środki masowego przekazu, polityków i działaczy gospodarczych i samorządowych terminy „innowacje" i „innowacyjność" zostały w powszechnym odczuciu nacechowane pozytywnie, co zresztą należy uznać za efekt ogólnie korzystny. Umknęło jednak uwadze to, że innowacyjność oprócz korzyści może być źródłem kłopotów, a niekiedy nawet poważnych zagrożeń. Więcej na ten temat, a w szczególności na temat zagrożeń można znaleźć w artykule Jacobfeuerborna i Muraszkiewicza (2013). Tutaj wśród zagrożeń warto wymienić to, że innowacje, zwłaszcza w zakresie robotyki i sztucznej inteligencji mogą doprowadzić do zlikwidowania wielu miejsc pracy, a nawet całych zawodów, czego współczesnym przykładem są pojazdy autonomiczne (bez kierowcy), które prawdopodobnie spowodują utratę pracy milionów kierowców samochodów dostawczych i autobusów komunikacji miejskiej na całym świecie. Innym niekorzystnym efektem jest to, że nawet niewielkie innowacje, polegające na przykład na zmianie kształtu czy elementów interfejsu użytkownika (dobrym przykładem są tu smartfony) powodują, że wciąż użyteczne dotychczasowe produkty są zastępowane przez ich nowe wersje. Efekt ten już dawno dostrzegł Joseph Schumpeter, jeden z pierwszych badaczy innowacyjności i jej roli w kapitalizmie (Schumpeter, 1989). Odkładając na bok dyskusję na temat tego, że nieustająca i rosnąca konsumpcja jest motorem współczesnego kapitalizmu, nie ma żadnych wątpliwości, że ten w gruncie rzeczy pozorny rodzaj innowacji, prowadzi do nieuzasadnionego wyczerpywania zasobów naturalnych planety.

3. Ekosystemy społeczności programistycznych

W jednym z wcześniejszych artykułów S. Grabowskiego (2016) znajduje się opis i analiza wybranych sieciowych ekosystemów działających w obszarze technik i usług teleinformatycznych oraz opis społeczności programistycznych, które wytworzyły się wokół tych ekosystemów. Tutaj dokonamy obszerniejszego przeglądu wybranych ekosystemów.

3.1. DT Developer Garden

DT Developer Garden działał w latach 2009–2015 i w swym zamierzeniu był projektem niemieckiego operatora telekomunikacyjnego Deutsche Telekom, który udostępnił firmom i programistom otwarte API[1] i inne narzędzia programistyczne. Pierwotnie DT Developer

[1] API (ang. *Application Programming Interface*), czyli interfejs programowania aplikacji. Jest to sposób (zrealizowany zwyczaj w formie oprogramowania) zapewniający komunikację pomiędzy aplikacją korzystającą i/lub współdziałającą z systemem, którego funkcjonalnym i/lub ergonomicznym rozszerzeniem jest ta aplikacja.

Garden działał jako punkt kontaktowy dla kreatywnych programistów, udostępniający interfejsy API oferowane przez Deutsche Telekom w celu realizacji usług dla stron internetowych i aplikacji, takich jak mashupy, widżety i wtyczki. Potem przekształcił się w program, którego celem było przyspieszenie cyklu tworzenia oprogramowania ułatwiającego tworzenie nowych aplikacji. Oprócz tradycyjnych API „telekomunikacyjnych" obsługujących wiadomości SMS i MMS oraz połączenia głosowe, DT Developer Garden udostępniał narzędzia do transkrypcji mowy na tekst i usługi geolokalizacyjne. Uzupełnieniem zestawu usług oddanego do dyspozycji społecznościom programistów były narzędzia do zarządzania i rozwoju aplikacji oraz aplikacje do testowania wydajności i analizy kodu. Partnerami DT Developer Garden były takie społeczności i grupy zorganizowane lub wspierane m.in. przez duże firmy, jak Microsoft Developer Network (MSDN), Intel Developers Services, IBM Mobile Foundation, czy M2M Developer Community.

3.2. Ribbit British Telecom i Twilio

Założona w 2006 r. firma Ribbit podeszła do problemu dostarczania usług komunikacyjnych inaczej niż jej ówcześni konkurenci. Celem firmy było udostępnienie programistom tworzącym aplikacje platformy pozwalającej na interakcję z sieciami komórkowymi operatorów telekomunikacyjnych. Z jej pomocą programiści mogli włączać do swoich stron internetowych na przykład usługi komunikacji głosowej i wysyłania wiadomości tekstowych. Użytkownicy mogli również realizować połączenia telefoniczne przez Internet do wybranego numeru za pośrednictwem prostego widżetu. Ribbit umożliwiał implementację za pomocą otwartych API następujących usług: obsługę wiadomości SMS i MMS, obsługę poczty głosowej oraz usługi głosowe. Programiści mogli korzystać z następujących technik narzędziowych: Adobe Flash, Java, JavaScript, PHP, .NET, Silverlight, RESTful API. Rabbit zamknął swoją platformę w 2011 r., rekomendując jej użytkownikom przeniesienie aktywności do platformy firmy Twilio, która jest firmą specjalizująca się w komunikacji opartej na chmurze obliczeniowej. Udostępniane przez Twilio API pozwala programistom tworzyć aplikacje umożliwiające wysyłanie i odbieranie wiadomości tekstowych, a także wykonywanie i odbieranie połączeń głosowych. Platforma Twilio działa w modelu PaaS (Platform-as-a-Service), a opłata za korzystanie z jej usług jest naliczana w zależności od czasu użycia i wielkości wykorzystanych zasobów. Z rozwiązań udostępnianych przez Twilio może korzystać każdy z zastrzeżeniem, że nie wykorzysta ich do rozsyłania spamu.

3.3. AT&T Developers portal

AT&T jest jednym z największych przedsiębiorstw telekomunikacyjnych na świecie. Z jego laboratoriów wyszły takie urządzenia i rozwiązania, jak: tranzystory, system operacyjny Unix, czy języki C, C++. AT&T było jedną z pierwszych firm, które już od 1996 r. udostępniały przez Internet narzędzia programistyczne, pakiety SDK (Software Development Kit) oraz podzbiory kodu źródłowego. Od 2012 r. program współpracy z programistami został rozszerzony przez dodanie języka HTML5 oraz oprogramowania ułatwiającego tworzenie systemów i aplikacji mobilnych. AT&T Developers Program, równolegle z oferowaniem technicznego wsparcia programistom, co polega m.in. na udostępnianiu API oraz narzędzi do sprawniejszego, wygodniejszego czy bardziej zaawansowanego tworzenia aplikacji oraz

na udostępnianiu różnego rodzaju opracowań i dokumentów, organizuje grupy zrzeszające osoby zainteresowane i amatorsko praktykujące programowanie i tworzenie aplikacji.

3.4. Open Middleware 2.0 Community

Społeczność Open Middleware 2.0 Community powstała w 2011 r. w ówczesnym Centrum Badawczo-Rozwojowym Orange Labs jako inicjatywa S. Grabowskiego i jego współpracowników. Celem tej inicjatywy było stworzenie mechanizmu budowania społeczności programistów i umożliwienie im bezpłatnego korzystania z zaawansowanych funkcji sieci Orange, dostępnych w formie otwartego API. W ramach Open Middleware 2.0 Community młodzi programiści, głównie studenci, tworzą własne aplikacje z obszaru telekomunikacji i informatyki. W ciągu kilku lat działalności opracowano kilkadziesiąt innowacyjnych rozwiązań, m.in. aplikację dla osób wymagających opieki, społecznościową grę miejską, usługę dla pasażerów komunikacji publicznej, system parkingowy oraz system do obsługi imprez publicznych. We współpracy ze szkołami wyższymi powstało wiele prac inżynierskich i magisterskich oraz kilkadziesiąt publikacji naukowych. Obecnie Open Middleware 2.0 Community współpracuje z następującymi uczelniami: Politechniką Warszawską, Politechniką Łódzką, Uniwersytetem Warmińsko-Mazurskim, Uniwersytetem Ekonomicznym w Poznaniu, Uniwersytetem Mikołaja Kopernika, Politechniką Gdańską, Politechniką Krakowską Politechnika Śląską, Uniwersytetem Jagiellońskim oraz Polsko-Japońską Akademią Technik Komputerowych.

3.5. Ekosystem informacyjny BIHAPI (Business Intelligence Hackathon Application Programming Interfaces)

BIHAPI jest rozszerzeniem programu Open Middleware 2.0 Community. Jest to ekosystem, którego uczestnikami oprócz programistów, inwestorów, przedstawicieli środowiska akademickiego oraz firm z obszaru technik teleinformatycznych, udział biorą przedstawiciele władz miejskich (m.in. Warszawy i Krakowa) oraz aktywiści miejscy. Ekosystem BIHAPI jest pomyślany i działa jako cykliczny konkurs skierowany głównie na tworzenie aplikacji wspierających transformację miast w stronę tzw. *smart cities*. Pierwsza edycja konkursu została zorganizowana w 2013 r. we współpracy z: firmą Oracle Communications jako partnerem technologicznym, IQPartners jako partnerem inwestycyjnym oraz z Urzędem Miasta Warszawy i Politechniką Warszawską. Głównymi celami konkursu były: (i) popularyzacja otwartych API jako narzędzia do tworzenia aplikacji mobilnych i serwerowych (Grabowski, 2014), (ii) promocja usług telekomunikacyjnych dostępnych przez API, (iii) wykorzystanie oraz udostępnianie otwartych danych znajdujących się w posiadaniu polskich miast, (iv) budowa środowiska społecznego skupionego wokół otwartych API oraz otwartych danych.

4. Generyczny model ekosystemu informacyjnego społeczności programistycznych

Społeczności programistyczne różnią się od wielu innych tworzących się oddolnie społeczności. Różnica nie polega jednak na tym, że ich etiologia, struktura czy sposób

funkcjonowania są diametralnie różne od odpowiadających im komponentów w innych społecznościach. Polega ona na specyficznej kompozycji, konfiguracji, wewnętrznych relacjach i *modus operandi* społeczności programistycznych. Współcześnie powstają spontanicznie lub inspirowane przez instytucje państwowe, publiczne i organizacje gospodarcze, w tym *par excellence* biznesowe, heterogeniczne społeczności programistyczne oparte na sieci, głównie na Internecie stacjonarnym i mobilnym. W skład tych społeczności wchodzą programiści indywidualni, nieformalne zespoły programistów, a także firmy komercyjne, głównie startupy i małe przedsiębiorstwa. W dalszej części artykułu właśnie te elementy zostaną wykorzystane do stworzenia generycznego modelu ekosystemu informacyjnego społeczności programistycznych tworzących się i działających głównie w dużych środowiskach miejskich.

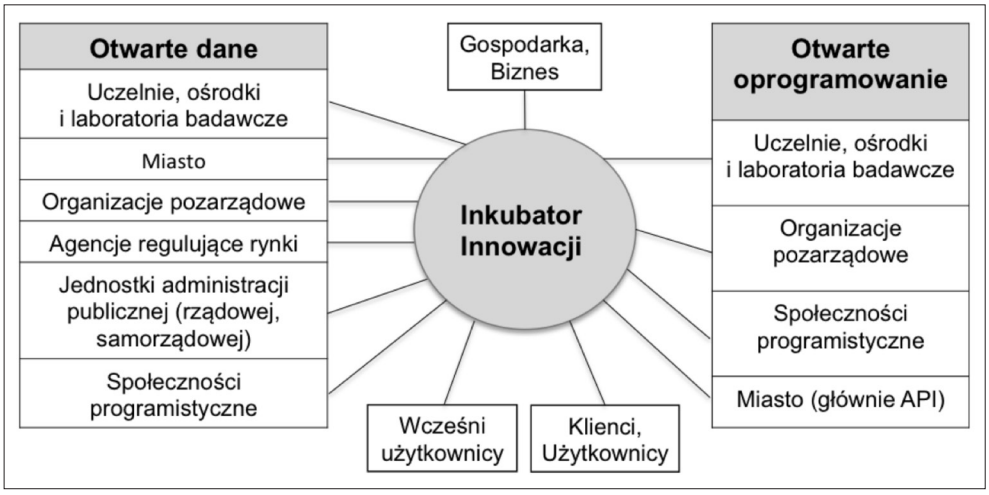
4.1. Podstawowe założenia modelu generycznego

Programowanie, zwłaszcza większych systemów, jest czynnością, a właściwie procesem o dużej złożoności, wymagającym specjalistycznej wiedzy, której składnikami są: (i) zbiór tzw. kompetencji twardych, którymi głównie są: umiejętność rozumienia algorytmów, praktyczna znajomość języków programowania i związanych z nimi bibliotek komponentów programistycznych oraz dobrych praktyk, kodowania algorytmów w wybranym języku i testowania utworzonych programów, oraz (ii) tzw. kompetencje miękkie, którymi są m.in.: cierpliwość, odporność na stres i zdolność do współpracy zarówno z innymi programistami, jak i z innymi, nieinformatycznymi podmiotami. Utrzymanie na wysokim poziomie i rozwijanie kompetencji twardych wymaga silnego i wszechstronnego wsparcia informacyjnego w zakresie technik informatycznych i inspirujących studiów przypadków, zarówno tych pozytywnych, jak i negatywnych. Co do kompetencji miękkich, to wsparcie informacyjne przyda się z pewnością w zakresie technik kreatywnej pracy zespołowej nakierowanej na rozwiązywanie problemów oraz budowania, utrzymywania i rozszerzania siatki współpracy z otoczeniem. Na marginesie warto odnotować zastanawiający fakt: programiści, to często absolwenci wyższych studiów technicznych, którzy uczęszczali na zajęcia z programowania, w tym na zajęcia z zakresu inżynierii oprogramowania, poświęcone m.in. metodom i technikom tworzenia systemów informatycznych i informacyjnych, a jednocześnie – jak pokazuje praktyka – rzadko korzystają z tej wiedzy w trakcie prowadzonych przez siebie projektów, zdając się na intuicyjne sposoby organizowania pracy. Prowadzi to do nieefektywnego wykorzystania zasobów i zmniejsza szanse na osiąganie konkurencyjnych wyników. Słabość tę w dużym stopniu może zniwelować dołączenie programistów do istniejących sieci programistów i wykorzystanie przez nich dostępnych zasobów informacyjnych i kompetencyjnych sieci.

Jak wspomniano, społeczności programistyczne i ich związki tworzą się raczej na obszarach metropolitalnych – ich naturalnym środowiskiem są miasta, zwłaszcza miasta przekształcające swą tkankę miejską w miasto inteligentne ze wszystkimi wynikającymi z tego korzyściami, wśród których szczególną rolę odgrywają: dobrze rozwinięta infrastruktura teleinformatyczna, efektywne, wykorzystujące tę infrastrukturę i stosujące nowe metody zarządzania, władze municypalne oraz duża liczba organizacji i inicjatyw pozarządowych. Władze miejskie we współpracy z tymi organizacjami działają na rzecz tworzenia, utrzymania i rozwoju dóbr wspólnych, z których specjalne znaczenie dla

środowisk programistycznych mają ogromne zasoby otwartych danych dotyczące zjawisk, procesów, przedsięwzięć, wydarzeń, historii, miejsc i ludzi – przykładem takich danych były 53 zbiory danych (m.in. dotyczące komunikacji miejskiej, rozkładów jazdy, parkowania, sportu, edukacji, jednostek użyteczności publicznej), z których korzystali uczestnicy projektu „Dane po Warszawsku"[2]. Jednocześnie przedsięwzięcia podejmowane przez różne prywatne, publiczne i państwowe podmioty w modelu otwartych innowacji, nawet jeśli nie są to inicjatywy *par excellence* informatyczne, potrzebują rozwiązań i/lub wsparcia informatycznego i programistycznego, najczęściej jako specjalistycznych aplikacji. I tu właśnie jest miejsce i rola do odegrania, a właściwie już odgrywana, przez społeczności programistyczne.

Powyższe stwierdzenia upoważniają do rozszerzenia przedstawionego wyżej Jacobfeuerborna modelu ekosystemu otwartej innowacyjności do postaci diagramu pokazanego na rysunku 1.



Rys. 1. Model ekosystemu otwartej innowacyjności

W modelu tym pojawiły się nowe elementy, którymi są miasto i społeczności programistyczne oraz otwarte oprogramowanie i otwarte dane, zaś miejsce fabryki innowacji zajął inkubator innowacji. O otwartym oprogramowaniu i otwartych danych traktuje bardzo duża liczba publikacji. Tutaj warto zwrócić uwagę na to, że otwarte dane, zarówno jako dane faktograficzne, jak i dane tekstowe oraz multimedialne są generowane przez uczelnie, ośrodki badawcze, laboratoria, organizacje pozarządowe, agencje (rządowe) regulujące rynki oraz przez administrację miejską, ale także przez społeczności programistyczne, które korzystając z danych pozyskanych z tych źródeł tworzą nowe zasoby danych, którym w wielu przypadkach nadają status danych otwartych. Jeśli chodzi o otwarte oprogramowanie, to jest ono tworzone głównie w uczelniach i ośrodkach badawczych, organizacjach pozarządowych oraz naturalnie w środowiskach programistycznych, niekiedy związanych z służbami informatycznymi miast.

[2] http://www.danepowarszawsku.pl/

Społeczności programistyczne jako samodzielny podmiot występujący w strukturach innowacyjnych, zwłaszcza tych związanych z funkcjonowaniem miast, są relatywnie nowym zjawiskiem. Jaskółką, która zwiastowała ich pojawienie się były hackathony, które można traktować jako prefiguracje współczesnych społeczności i sieci programistycznych. Na marginesie warto wspomnieć, że za pierwszy hackathon uważa się wydarzenie OpenBSD, które miało miejsce w Calgary w 1999 r. Wikipedia pod hasłem *hackhaton* podaje, że jest to

> wydarzenie skierowane do programistów, podczas którego informatycy i inne osoby związane z rozwojem oprogramowania, takie jak projektanci grafiki, twórcy interfejsów i menedżerowie projektów, stają przed zadaniem rozwiązania określonego problemu związanego z projektowaniem. Hackathony odbywają się w krótkim czasie, zazwyczaj na przestrzeni dnia lub weekendu. Zadanie do wykonania ogłaszane jest w dniu rozpoczęcia konkursu. Podczas oceniania pod uwagę brana jest wyłącznie praca wykonana podczas trwania wydarzenia (Hackathon, 2016).

Hackathony były częścią ekosystemów, opisanych w sekcji trzeciej: Deutsche Telekom Developer Garden, Twilio, Open Middleware 2.0 Community oraz BIHAPI. Z pewnością rola hackathonów w tworzeniu innowacyjnych rozwiązań będzie rosła i choćby z tego tytułu będą one przedmiotem dalszych badań.

Biorąc pod uwagę podane wyżej fakty, obserwacje i wnioski, można pokusić się o stworzenie generycznego modelu ekosystemu informacyjnego dla innowacyjnych społeczności programistycznych. Trzy aspekty tych społeczności są szczególnie istotne, a mianowicie: sieciowość, otwartość i „informacyjność". Model ten przedstawiono poniżej w dwóch aspektach: informacyjnym (źródła informacji) i procesualnym.

4.2. Informacyjny aspekt modelu generycznego

Głównymi (a więc nie jedynymi) źródłami informacji i wiedzy, z których korzystają społeczności programistyczne są: otwarte dane, otwarte oprogramowanie oraz crowdsourcing, to bowiem te źródła są dużą i ważną częścią tego, co można nazwać „informacyjnym dobrem wspólnym". Otwarte dane pochodzą z różnorakich miejsc, wśród których najsilniejszymi generatorami danych są procesy zachodzące i rejestrowane informacyjnie w miastach, w różnorakich placówkach rządowych i samorządowych, państwowych uczelniach, instytutach naukowych i laboratoriach badawczych oraz w całej rzeszy organizacji pozarządowych. Dane te tworzone i przechowywane są w różnych formatach oraz dostępne za pomocą różnych narzędzi i procedur informatycznych, w tym procedur ochrony danych. Jest to zatem niezwykle heterogenna przestrzeń informacyjna, co bardzo utrudnia dostęp i korzystanie z niej, na co nierzadko nakłada się niechęć dysponentów danych do ich udostępnienia, pomimo istnienia formalnego obowiązku w tym względzie. Dodatkowe problemy związane z otwartymi danymi wiążą się z tym, że – po pierwsze – zazwyczaj nie są one uporządkowane i ustrukturyzowane (zbiory danych zapisane są w postaci tekstowej, np. w formacie DOCX lub PDF, czy tabelarycznej w formacie XLSX, a nie na przykład według otwartego standardu *OpenDocument*, czy otwartej relacyjnej bazy danych w rodzaju SQLite, MySQL czy PostgreSQL, czy serwera i wyszukiwarki tekstowej Sphinx) oraz – po drugie – brakiem metadanych opisujących zawartość zbiorów danych, ich pochodzenie, tryb aktualizacji, kontakt do administratora itp. Metadane nie tylko ułatwiają korzystanie ze zbiorów danych, ale bywa, że ich brak uniemożliwia korzystanie z samych danych. Trzeba podkreślić, że społeczności programistyczne korzystając z otwartych danych często same

tworzą nowe zbiory otwartych danych, które są wynikiem przetworzenia, na przykład zagregowania danych źródłowych albo wyselekcjonowania z nich podzbiorów relewantnych do określonej potrzeby informacyjnej.

Abstrahując od dyskusji na temat różnic pomiędzy otwartym oprogramowaniem, a wolnym oprogramowaniem, najważniejsze jest to, że użytkownik ma dostęp do kodu źródłowego i może dokonywać w nim modyfikacji w zależności od swoich potrzeb. Więcej, respektując licencję (zwykle dość liberalną) nałożoną na ten kod, może upowszechniać swoją wersję. W ten sposób społeczności programistyczne, udostępniając opracowane przez siebie programy, kolektywnie pracują nad rozwojem oprogramowania. Warto odnotować, że ten tryb pracy i dyfuzji oprogramowania nie wyklucza jego pewnych zastosowań komercyjnych.

Trzecim elementem informacyjnego dobra wspólnego, wchodzącym w skład modelu ekosystemu informacyjnego społeczności programistycznych, jest crowdsourcing. Ta forma pozyskiwania wiedzy z otoczenia, które na tzw. otwarte zapytanie (ang. *open call*) reaguje w postaci zbioru propozycji rozwiązań zadania sformułowanego w zapytaniu jest wykorzystywana przez społeczności programistyczne w sytuacji, gdy nie posiadają one kompetencji do rozwiązania problemu, którym się zajmują. I odwrotnie, to dany zespół programistyczny, jeśli posiada potrzebne kompetencje albo tylko dobrze uzasadnione i realistyczne pomysły, może proponować rozwiązania problemów sformułowanych przez inne podmioty.

Oto inne, ważne źródła informacji i wiedzy wykorzystywane przez społeczności programistyczne w ekosystemie informacyjnym. Niewątpliwie społeczności te dysponują własnymi zasobami informacyjnymi, swoim *know-how* i regułami zarządczymi. Elementy te są wynikiem doświadczenia i wraz z rozwojem społeczności nabierają coraz większego znaczenia w kolejnych projektach przez nie realizowanych. W przypadku większych projektów pojawia się nierzadko moment, w którym trzeba skorzystać z technik wywiadu gospodarczego w celu zidentyfikowania komponentów potrzebnych do realizacji tych projektów. Zwykle społeczność robi to siłami własnymi, ale bywa też, że korzysta z kompetencji współpracujących z nią innych podmiotów, na przykład tych, które należą do ekosystemu otwartych innowacji albo sięga po ekspertów zewnętrznych (którymi często są osoby z innych społeczności programistycznych) lub po prostu zamawia taką usługę w wyspecjalizowanej placówce. Wynikiem tych działań najczęściej są dane rynkowe, dane gospodarcze, informacje o patentach, licencjach i innych kwestiach z zakresu praw własności intelektualnej, dane o technologiach, producentach, użytkownikach itp. Ważnymi źródłami informacji są prosumenci, czyli osoby, grupy osób czy nawet firmy, które współpracują ze społecznością programistyczną już w fazie testowania, a niekiedy nawet projektowania produktu. To oni właśnie są głównymi użytkownikami wersji beta wytworzonego oprogramowania czy aplikacji. Niebagatelnym źródłem informacji, stale obecnym w pracy programistów są media społecznościowe, będące zarazem wehikułem komunikacyjnym i instrumentem promocyjnym i/lub marketingowym.

4.3. Procesualny aspekt modelu generycznego

Jak wspomniano, model ekosystemu informacyjnego dla społeczności programistycznych obejmuje także aspekt procesowy, czyli sposób ich tworzenia się i funkcjonowania. Społeczności takie powstają najczęściej jako grupy startupów w parkach technologicznych czy pod auspicjami uczelni wyższych. Tworzą się one także *ad hoc* na użytek realizacji dużych

projektów finansowanych z grantów przyznawanych przez agencje rządowe lub samorządowe; w Polsce są to m.in.: Narodowe Centrum Badań i Rozwoju, Polska Agencja Rozwoju Przedsiębiorczości i Urzędy Marszałkowskie. W takich przypadkach lider projektu zaprasza do udziału w nim różne zespoły programistyczne, biorąc na siebie obowiązek zdefiniowania i przydzielenia im zadań, koordynację prac oraz integrację wyników cząstkowych w produkt końcowy. Inicjatywę utworzenia społeczności programistycznych podejmują także korporacje we współpracy z uczelniami, czego przykładem jest społeczność Open Middleware 2.0 Community i ekosystem informacyjny BIHAPI, wspomniane w sekcji trzeciej tego artykułu.

Ekosystem BIHAPI jest o tyle interesujący, że jest dobrym przykładem środowiska do akceleracji innowacyjnych produktów informatycznych. W ramach akceleracji zwykle powstaje prototyp o statusie *proof-of-concept*, czyli rozwiązanie, które ma przekonać inwestora(ów) do zainwestowania środków finansowych w opracowanie produktu finalnego i jego komercjalizację. Najczęściej są to różnorakie aplikacje przeznaczone dla urządzeń mobilnych takich, jak smartfony czy tablety, choć naturalnie akceleracja może dotyczyć projektowania i realizacji większych przedsięwzięć opartych na platformach cyfrowych.

Modelowo akcelerację można przedstawić jako proces składający się z następujących etapów:

(1) Zdefiniowanie zadania innowacyjnego, czyli opracowanie specyfikacji produktu finalnego. Realizację tego zadania bierze na siebie firma lub organizacja zainteresowana utworzeniem nowego produktu.
(2) Utworzenie ekosystemu innowacji. Jeśli zadanie innowacyjne będzie realizowane według modelu otwartych innowacji, to w składzie ekosystemu znajdą się inwestor(rzy), placówki badawcze i inne wyżej wymienione podmioty.
(3) Ogłoszenie konkursu (może on być otwarty albo zamknięty, czyli ograniczony do wybranej grupy zespołów programistycznych) na realizację zadania. Konkurs ogłasza podmiot sprawujący rolę koordynatora ekosystemu. Konkursy mogą przybierać popularną obecnie formę hackathonu.
(4) Opracowanie produktów (najczęściej w postaci prototypów). Zespoły programistyczne, które przystąpią do konkursu tworząc produkty (aplikacje) spełniające warunki konkursu, stają się automatycznie częścią ekosystemu i uzyskują dostęp do tych jego zasobów, które udostępniają uczestnicy ekosystemu na czas realizacji zadania. Zespoły pracują niezależnie od siebie.
(5) Ocena wyników pracy zespołów wedle kryteriów wcześniej ustalonych przez inwestorów.
(6) Selekcja przedłożonych rozwiązań. Inwestorzy wybierają te rozwiązania (prototypy), które zamierzają przekształcić w produkty spełniające warunki komercjalizacji.
(7) Opracowanie wersji finalnej (komercyjnej) produktów.
(8) Komercjalizacja, czyli wprowadzenie produktów na rynek.

5. Zakończenie

Zaproponowany model może posłużyć jako punkt wyjścia do dalszych prac nad konstruowaniem, architekturą, funkcjami i zarządzaniem społecznościami programistycznymi, których znaczenie w tworzeniu innowacji technicznych i wspierania nowatorskich rozwiązań gospodarczych i społecznych, nabiera szczególnego znaczenia w kontekście wysiłków wyrwania się Polski z pułapki średniego rozwoju.

Warto ponownie podkreślić, że przedstawiony generyczny model ekosystemu informacyjnego innowacyjnych społeczności programistycznych, łączy w sobie trzy niezwykle ważne zjawiska współczesnej gospodarki i życia społecznego. Pierwsze z nich to sieciowy, horyzontalny charakter powiązań i interakcji w ramach zachodzących procesów w obrębie samych społeczności i procesów łączących je z otoczeniem. Drugie zjawisko, w coraz większym stopniu uświadamiane i wykorzystywane przez programistów, to rosnąca rola informacji i wiedzy w tych procesach, z naciskiem na zapewnienie otwartego i szerokiego dostępu do informacji z zachowaniem, w możliwie jak największym stopniu ochrony tożsamości użytkowników. Trzeci fenomen, który w tak dużej skali, jaki obecnie pojawił się w historii informatyki i jej zastosowań po raz pierwszy, to wola tworzenia innowacyjnych produktów i rozwiązań problemów społecznych, co wobec coraz większej presji na wykorzystanie zasobów naturalnych planety i nasilające się problemy demograficzne jest szczególnie godne odnotowania.

Bibliografia

Chesbrough, H. (2003). *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Boston: Harvard Business School Press.

Chesbrough H., Vanhaverbereke W., West J. (2006). *Open Innovation. Researching a New paradigm*. Oxford University Press.

Global Developer (2017). *Global Developer Population and Demographic Study 2016 V2*. Evans Data Corporation.

Grabowski, S. (2014). What is Open Data and How to Benefit from It? *Zagadnienia Informacji Naukowej*, 52(1), 43–51.

Grabowski, S. (2016). Społeczności programistyczne w obszarze otwartych interfejsów programistycznych. Open API: przykłady i wnioski. W: B. Sosińska-Kalata et al. (red.), *Nauka o informacji w okresie zmian. Informatologia i humanistyka cyfrowa*. Miscellanea Informatologica Varsoviensia. vol. VIII (121–134). Warszawa: Wydaw. SBP.

Griffiths, D., Pearson, A. W. (1973). The Organization of Applied Research and Development with Particular Reference to the Customer – Contractor Principle. *R&D Management* 3(3), 121–124.

Hackathon (2016). Wikipedia. Wolna encyclopedia [online] [23.10.2017], https://pl.wikipedia.org/wiki/Hackathon

Hartmann, D., Trott, P. (2009). Why 'Open Innovation' is Old Wine in New Bottles. *International Journal of Innovation Management* 13(4), 715–736.

Jacobfeuerborn, B. (2012). A Knowledge Concocter to Sustain Innovation Propensity. W: Z. E. Zieliński (red.), *Rola informatyki w naukach ekonomicznych i społecznych. Innowacje i implikacje interdyscyplinarne* (40–50). Kielce: Wydaw. Wyższej Szkoły Handlowej.

Jacobfeuerborn, B., Muraszkiewicz, M. (2012). ICT Based Information Facilities to Boost and Sustain Innovativeness. *Studia Informatica* 33(2B), 485–496.

Jacobfeuerborn, B., Muraszkiewicz, M. (2013). A Sketchy Critique of Innovation by Advocati Dei and Diaboli. W: B. Jacobfeuerborn (ed.), *Innovating Innovation. Essays on the Intersection of Information Science and Innovation* (11–20). Warszawa: MOST Press, IINiSB UW, 11–20.

Pearson, A. W., Green, T., Ball, D.F. (1979). A Model for Studying Organizational Effects of an Increase in the Size of R&D Projects. *IEEE Transactions on Engineering Management*, EM-26(1), 14–21

Rogers, E. M. (1962). *Diffusion of Innovations*. Free Press of Glencoe, Macmillan Company.

Schumpeter, J. A. (1989). *Essays: On Entrepreneurs, Innovations, Business Cycles and the Evolution of Capitalism*. New Brunswick, N.J.: Transaction Publisher.

Modeling of Information Ecosystems for Innovative Programming Communities

Abstract

Purpose/Thesis: The purpose of the paper is to propose a generic model of an information system supporting innovative communities of ICT programmers, in particular those who work in urban environments.

Approach/Methods: The method adopted in the study consisted in analyzing a selected set of foreign network ecosystems that boost programming communities (among others: Deutsche Telekom Development Garden and AT&T Developers Portal) and the communities co-established by Sebastian Grabowski, that is, Open Middleware 2.0 Community and BIHAPI. The analysis covered structures of these ecosystems and their *modus operandi*. Conclusions, in particular those related to the factors supporting innovation processes, helped to define a generic model of an information ecosystem for supporting innovative programming communities.

Results and conclusions: The examination of the analyzed ecosystems along with the authors' own experience acquired as a result of cooperation with programming communities on different projects and programs, including Open Middleware 2.0 Community and BIHAPI and numerous hackathons, proved an important role currently played and to be played in the future by the programming communities. It was discovered that particularly efficient and productive way for these communities to organize and work was the use of open platforms, with focus on open-source software, open data and open innovation.

Application: The proposed model of a generic information ecosystem supporting innovative programming communities provides the background for further work on the construction, architecture, functionality and management of programming communities whose role in producing technical innovations and boosting innovative economic and social solutions becomes of tremendous importance in the context of Poland's ambition to escape the middle-income trap.

Originality/Value: The proposed model is generic which makes it adaptable to programming environments of varying profiles and objectives. It is based on open platforms as regards data and software and puts emphasis on the work mode specific for open innovations, with an innovative approach to the programmers' work. Furthermore, the model is open which offers reliable background for further research on modelling and developing information ecosystems supporting programming communities. The value of the model discussed was confirmed in practice during Open Middleware 2.0 Community and BIHAPI programs.

Keywords

API. Application Programming Interface. Information ecosystem. Open data. Open innovation. Open-source software. Programming community.

SEBASTIAN GRABOWSKI jest dyrektorem Centrum Badawczo-Rozwojowego Orange Polska S.A. oraz Prezesem Fundacji „ArchitectsPL" zajmującej się m.in. wpływem otwartych danych na społeczeństwo. Jest współtwórcą programu Open Middleware Community zrzeszającego różne środowiska deweloperów, naukowców, dostawców technologii oraz firm IT oraz twórcą ogólnopolskiej inicjatywy poświęconej tematyce otwartych danych – BIHAPI. Jego badania dotyczą otwartych danych, interakcji pomiędzy użytkownikami sieci telekomunikacyjnych i Internetu oraz zagadnień związanych z otwartym rządem. Wśród jego ostatnich publikacji znajdują się: S. Grabowski (2014). What is Open Data and How to Benefit from It? ZIN, 52(1), 43–51; K. Litwiniuk, T. Czarnecki, S. Grabowski, J. Legierski (2012). BusStop — Telco 2.0 application supporting public transport in agglomerations. Federated Conference on Computer Science and In-formation Systems (FedCSIS), 9(12), 649–653; S. Grabowski, J. Legierski (2011). Interfejsy programistyczne w systemach Otwartego Rządu, PITWIN, Kielce.

Kontakt z autorem:
sebastian.grabowski@orange.com
Orange Labs, Centrum Badawczo-Rozwojowe
Obrzeżna 7
02–691 Warszawa

Prof. dr hab. MIECZYSŁAW MURASZKIEWICZ jest profesorem zwyczajnym w Instytucie Informatyki Politechniki Warszawskiej. Jego zainteresowania naukowe dotyczą inteligentnych systemów informacyjnych, metod reprezentacji wiedzy, informacji dla nauki oraz relacji wiążących technikę i kulturę, a także zagadnień innowacyjności. Wśród jego ostatnich publikacji znajdują się: M. Muraszkiewicz (2016). Techniki komunikacyjne i informacyjne dla inteligentnych miast. W: D. Gotlieb, R. Olszewski (eds.), Smart City. Informacja przestrzenna w zarządzaniu inteligentnym miastem (53–68). Warszawa: PWN; B. Jacobfeuerborn, M. Muraszkiewicz (2014). Some Challenges and Trends in Information Science. In: R. Bembenik et. al. (eds.), Intelligent Tools for Building a Scientific Information Platform: From Research to Implementation (3–14). Springer [Studies in Computational Intelligence No. 541]; B. Jacobfeuerborn, M. Muraszkiewicz (2013). Media, Information Overload, and Information Science. In: R. Bembenik et. al. (eds.), Intelligent Tools for Building Scientific Information Platform. Advanced Architectures and Solutions (3–13). Springer [Studies in Computational Intelligence No. 467].

Kontakt z autorem:
m.muraszkiewicz@ii.pw.edu.pl
Instytut Informatyki
Wydział Elektroniki i Technologii Informacyjnych
Politechnika Warszawska
Nowowiejska 15/19
00–665 Warszawa