Sebastian Dawid Kotuła

Wstęp do Open Source

NAUKA – DYDAKTYKA – PRAKTYKA

WYDAWNICTWO SBP

Polish Librarians Association
SCIENCE-DIDACTICS-PRACTICE

Sebastian Dawid Kotuła

INTRODUCTION TO OPEN SOURCE

WYDAWNICTWO SBP

Warsaw 2014

Stowarzyszenie Bibliotekarzy Polskich
NAUKA-DYDAKTYKA-PRAKTYKA

Sebastian Dawid Kotuła

WSTĘP DO OPEN SOURCE

WYDAWNICTWO SBP

Warszawa 2014

Druk publikacji został współfinansowany przez Uniwersytet Marii Curie-Skłodowskiej w Lublinie w ramach grantu Wydziału Humanistycznego

Recenzent
Seweryn DOBRZELEWSKI

Projekt okładki
Funky Worky

Redaktor prowadzący
Marta LACH

Redakcja techniczna i korekta
Magdalena ORCZYKOWSKA

ISBN 978-83-64203-33-6

CIP – Biblioteka Narodowa
Kotuła, Sebastian.
Wstęp do open source / Sebastian Kotuła ;
Stowarzyszenie Bibliotekarzy Polskich. - Warszawa :
Wydawnictwo Stowarzyszenia Bibliotekarzy Polskich,
2014. - (Nauka, Dydaktyka, Praktyka ; 156)

Wydawnictwo Stowarzyszenia Bibliotekarzy Polskich
00-335 Warszawa, ul. Konopczyńskiego 5/7 tel. (22) 827-52-96
www.sbp.pl; wydawnictwo@sbp.pl, biuro@sbp.pl
Warszawa 2014. Wyd. I. Ark. wyd. 9,5. Ark. druk. 10,5
Łamanie: Robert LIS
Druk i oprawa: Mazowieckie Centrum Poligrafii
ul. Piłsudskiego 2A, 05-270 Marki, www.c-p.com.pl
e-mail: biuro@c-p.com.pl, tel. (22) 497-66-55

SPIS TREŚCI

TABLE OF CONTENTS

WSTĘP

W codziennej ludzkiej rzeczywistości funkcjonują rozmaite technologie informacyjne. *Technologie informacyjne* będę rozumiał następująco: „to termin łączący narzędzia i metody używane do komunikacji i operowania informacją. Narzędzia to sprzęt informatyczny i telekomunikacyjny, programy komputerowe oraz środki służące do transmisji informacji w postaci cyfrowej"[1]. Wprowadzanie do społecznego obiegu technologii informacyjnych wymagało odpowiedniego pośrednika. W tę rolę wszedł Internet i zwłaszcza najpopularniejsza jego usługa, tj. World Wide Web[2]. Web jest jednocześnie technologią informacyjną, systemem, usługą internetową oraz, przede wszystkim, przestrzenią informacyjną i komunikacyjną[3].

Rozproszony system hipertekstowy World Wide Web *per se* powstał na gruncie stosownych technologii informacyjnych. Do stworzenia WWW konieczne były: odpowiednie środowisko, tj. komputery (zwłaszcza osobiste – tzw. pecety), których nieustanną ekspansję i rozwój obserwuje się od lat 70. XX wieku; idea sieci, przy czym badacze infrastruktury informacyjnej prapoczątków sieci dopatrują się w powstaniu sieci transportu (infrastruktura drogowa), usług pocztowych oraz infrastruktury teleko-

---

[1] W. Gogołek, *Komunikacja sieciowa. Uwarunkowania, kategorie i paradoksy*, Warszawa 2010, s. 17.

[2] Wymiennie będę stosował terminy *World Wide Web, WWW, W3, Web*.

[3] *Przestrzeń informacyjna* to „wielowymiarowy, dynamiczny, otwarty zbiór treści (danych i informacji), ich nośników oraz użytkowników". M. Kisilowska, *Przestrzeń informacyjna jako termin informatologiczny*, „Zagadnienia Informacji Naukowej" 2011, nr 2, s. 35-52. W stosunku do World Wide Web używa się też niekiedy określenia środowisko cyfrowe.

munikacyjnej[4]; hipertekst, który zdaniem komunikologów powstał na długo przed stworzeniem samego terminu *hipertekst* – podaje się bowiem, że kultura piśmienna u swych początków wytworzyła wiele tekstów, które odznaczają się właściwościami podobnymi do hipertekstu, a zatem każdy tekst, który można odczytywać nielinearnie, traktuje się jako hipertekst. Na przykład, gdy czytelnik przechodzi do innego rozdziału, fragmentu, a nawet do innego tekstu (który mógł być wskazany we fragmencie tekstu czytanym wcześniej lub po prostu dlatego, że ten tekst już go nie interesuje), a pomija przy tym inne teksty pośrednie (ułożone w kolejności linearnej), recypuje *de facto* tekst w sposób hipertekstualny[5].

W kontekście W3 oznaczało to każdorazowo opracowanie odpowiednich technologii informacyjnych. W przypadku komputerów były to odpowiednie urządzenia (hardware). Poza samymi komputerami, należało stworzyć odpowiednie karty sieciowe pozwalające połączyć komputery ze sobą zarówno przewodowo, jak i bezprzewodowo. Ponadto konieczne było stworzenie odpowiedniego oprogramowania (software). Ważnym krokiem w rozwinięciu sektora komputerowego było przejście na kod binarny, co znacznie usprawniło programowanie. Wśród programów należy wskazać systemy operacyjne, które nie tylko sterują pracą komputera, ale również pozwalają na korzystanie z zasobów WWW. W przypadku sieci należało opracować odpowiednie protokoły komunikacyjne, które zapewniły swobodną komunikację i wymianę informacji pomiędzy różnymi urządzeniami komputerowymi. Zapewnienie kompatybilności komunikacji komputerów tworzonych przez różne firmy stało się kamieniem milowym w tworzeniu sieci i, jak pokazuje historia, wymagało dziesięciu lat pracy. Uściślając, protokoły komunikacyjne pozwalające na wymianę informacji *via* sieć internetowa są również odpowiednimi technologiami informacyjnymi, które zarządzają transmisją danych w sieciach. W przypadku hipertekstu potrzeba było natomiast ponad dwudziestu lat pracy, poczynając od pomysłów Theodora Holm Nelsona, a kończąc na technologii W3 Tima Bernersa Lee[6], nie wspominając o po-

---

[4] Por. D. McQuail, *Mass communication theory*, ed. 6., London 2010, s. 16; T. Goban-Klas, *Cywilizacja medialna. Geneza, ewolucja, eksplozja*, Warszawa 2005, s. 148.

[5] Por. A. Kumar, *Internet and information technology*, New Delhi 2002, s. 202. Do zaistnienia WWW potrzebne były komputery, odpowiednie środowisko sieciowe oraz koncepcja system hipertekstowego. Por. J. Gilles, R. Cailliau, *How the Web was born*, New York 2000, s. 2-47.

[6] Por. *The Internet. A historical encyclopedia*, red. H. W. Poole, L. Lambert, C. Woodford, C.J. P. Moschovitis, Santa Barbara 2005, s. 186-188.

mysłach Vannevara Busha, Douglasa Engelbarta, Ivana Sutherlnada czy Josepha C. R. Licklidera[7].

Z jednej strony WWW jest wytworem technologii informacyjnych, z drugiej zaś – stymuluje i przyspiesza ich rozwój. Dziś każda niemalże sfera ludzkiej aktywności została wypełniona przez odpowiednie technologie informacyjne, które wspomagają pracę i zapewniają rozrywkę. Internet staje się repozytorium uniwersalnym, w którym znaleźć można rozmaite treści (w tym wiedzę – w sensie potocznym), w społecznej świadomości wyobraża się go jako podstawowe źródło wszystkich niezbędnych informacji[8]. Zasoby Internetu, na co też niekiedy zwraca się uwagę, „stanowią jeden ogromny komunikat o niespotykanych wcześniej parametrach. Jest gigantyczny i nieuporządkowany – to chaos, smog komunikacyjny, ogrom nieuporządkowanych treści, które w dodatku mogą zaraz przestać istnieć. Jest zarazem różnorodny: oprócz treści ważnych są w nim też informacyjne i nieinformacyjne śmieci, no i efemeryczny, zmienny. To stwarza nowe wyzwania, tak w zakresie porządkowania, opracowania i przygotowania informacji o tych zasobach, jak też w zakresie ich archiwizacji"[9]. Niezależnie od tego, ile i jakich informacji faktycznie jest w Internecie, nie ulega wątpliwości, że jest to pokaźna baza, z której można czerpać wartościowe treści. Skuteczność pozyskiwania pożądanych informacji zależy m.in. od możliwości ekonomicznych i kompetencyjnych internautów[10]. Obraz Internetu jako ogromnej bazy informacji skutkuje tym, że powstaje coraz więcej rozmaitych technologii informacyjnych pozwalających na swobodne korzystanie z jego zasobów. Tworzy się coraz poręczniejsze urządzenia przenośne (telefony komórkowe, tablety), które równocześnie wyposażane są w odpowiednie

---

[7] Por. także M. Góralska, *Piśmienność i rewolucja cyfrowa*, Wrocław 2012, s. 82-93.

[8] Popularność WWW doprowadziła do utożsamiania Webu z Internetem, choć są to zupełnie różne technologie. Internet jest siecią łączy przewodowych i bezprzewodowych zapewniających komunikację pomiędzy różnymi urządzeniami, natomiast WWW jest jedną z usług Internetu, jednym ze sposobów, w jaki można wykorzystywać tę sieć. Jak się zdaje, WWW jest w Internecie usługą najpopularniejszą. Por. J. Gilles, R. Cailliau, op. cit., s. 48. Większość internautów z zasobów Internetu korzysta za pośrednictwem W3.

[9] Por. J. Wojciechowski, *Biblioteczna wartość naddana,* Kraków 2006, s. 98.

[10] Konieczne w tym miejscu jest zasygnalizowanie problemu wykluczenia cyfrowego, które odnosi się „do grup społecznych, pozbawionych dostępu do nowoczesnej infrastruktury informatycznej (w tym szczególnie – dostępu do sieci teleinformatycznych) i tym samym wykluczonych z udziału w rozwoju cywilizacyjnym". P. Gawrysiak, *Cyfrowe wykluczenie treści*, [w:] *Informacja w sieci. Problemy, metody, technologie*, red. B. Sosińska-Kalata, E. Chuchro, W. Daszewski, Warszawa 2006, s. 117.

oprogramowanie zapewniające coraz skuteczniejsze nawigowanie w sieci internetowej. Programuje się także aplikacje na komputery stacjonarne, które sprzyjają dostarczaniu informacji potrzebnych użytkownikowi. Zwłaszcza w dobie Internetu drugiej generacji (Web 2.0) stworzono wiele technologii informacyjnych umożliwiających swobodną interakcję między internautami, a także dwukierunkowo między siecią i internautami. Web więc wymusza tworzenie odpowiednich technologii, które zapewniają do niego dostęp, co wpływa na rozszerzanie zasięgu oddziaływania WWW, a tym samym sprzyja powstawaniu nowych technologii. W rezultacie pojawia się coś w rodzaju samonapędzającego się środowiska.

W krajobrazie technologii informacyjnych pojawiły się również technologie określane terminem *open source*[11]. Początkowo przyjmowane zarówno entuzjastycznie, jak i nieufnie, dziś zyskują coraz większą popularność i są coraz szerzej wykorzystywane. Powszechność Webu i webowych technologii informacyjnych zaowocowała tym, że technologie open source stały się prężnie rozwijanym sektorem technologii.

Niniejsza praca poświęcona jest technologiom open source. Na polskim rynku wydawniczym, w literaturze z zakresu bibliologii i informatologii, wciąż brakuje syntetycznego opracowania w przejrzysty i jasny sposób przedstawiającego zagadnienie open source. Celem niniejszej książki jest więc wprowadzenie w tę problematykę, zwłaszcza na potrzeby informatologii, na gruncie której temat open source pozostaje dotychczas nieopracowany. Publikacja ma też na celu wskazanie przykładowych aplikacji open source z różnych grup, tj. wybranych reprezentantów popularnych kategorii programów oraz przekonanie do używania technologii informacyjnych open source. W przestrzeni zmieniającego się prawa aplikacje open source, nieobciążone ograniczeniami wynikającymi z mechanizmu copyright, zdają się ciekawym rozwiązaniem i fundamentalną koncepcją pokazującą specyficzną i możliwą drogę rozwoju (głównie software'u). Praca opiera się na dorobku bibliologii i informatologii oraz socjologii komunikacji społecznej, w tym medioznawstwa i komunikacji internetowej, a także informatyki.

Książka składa się ze wstępu, zakończenia, bibliografii i pięciu zasadniczych części, wydzielonych ze względu na poruszane zagadnienia, w któ-

[11] Wymiennie będę stosował terminy: *open source*, *open source software*, *oprogramowanie open source* oraz akronimy OS, OSS. Zdaniem Manuela Castellsa to m.in. technologie open source doprowadziły do ekspansji Internetu i do jego upowszechnienia we wszystkich sferach aktywności. Por. M. Castells, *Społeczeństwo sieci*, Warszawa 2010, s. 15-16.

rych kolejno przybliżyłem historię ruchu open source (m.in. działania Richarda Matthew Stallmana i Free Software Foundation oraz Erica Stevena Raymonda i Open Source Initiative) i kwestie prawne, tj. podstawowe licencje dotyczące oprogramowania open source. Następnie omówiłem korespondujące z zagadnieniami poruszanymi w pracy społeczne podstawy rozwoju ruchu open source; podjąłem także próbę odpowiedzi na pytanie, skąd wziął się w ogóle pomysł na open source. Wykazałem intensywny rozwój zjawiska open source. Następnie wyjaśniłem termin *open source*. Wreszcie porównałem oprogramowanie open source z oprogramowaniem komercyjnym, zwracając uwagę na ich wady i zalety. Omówiłem również mity krążące na temat open source software.

W kolejnej części książki uwagę zwróciłem na problem dotarcia do pożądanych programów z grupy open source, w rezultacie czego poświęciłem odrębny rozdział bazom tego oprogramowania. Ostatnią częścią pracy jest przybliżenie wybranych technologii open source dla następujących obszarów: systemy operacyjne, oprogramowanie biurowe, aplikacje do zarządzania informacją, programy przydatne w edukacji, aplikacje internetowe, programy multimedialne, oprogramowanie rozrywkowe oraz aplikacje przydatne w pracy bibliotecznej.

1. KRÓTKA HISTORIA RUCHU OPEN SOURCE

Nie wdając się zbytnio w rozważania dotyczące historii rozwoju komputerów, przypomnę jedynie, iż pierwsze komputery z lat 40. XX wieku korzystały z oprogramowania wczytywanego z zewnątrz. Było to możliwe, ponieważ komputery były wykorzystywane do rozwiązywania naukowych problemów, zwłaszcza do prowadzenia obliczeń matematycznych, a ich użytkownikami byli naukowcy (inżynierowie) posiadający dobrze ugruntowaną wiedzę matematyczną. Instrukcje dla komputerów były więc opracowywane przez naukowców i przeznaczone dla naukowców. Jednym z pierwszych komputerów wykorzystywanych przez kręgi naukowo-badawcze był na przykład ENIAC (1945 rok)[12]. Wczytywanie programów z zewnątrz jest mało efektywne, dlatego rozpoczęto prace nad rozwijaniem projektów pozwalających wykorzystywać pamięć wewnętrzną. W tym zakresie ważne okazały się pomysły Johna von Neumanna, który w 1945 roku zaproponował, aby zacząć projektować komputery, które będą wykorzystywać pamięć

[12] Por. J. Feller, B. Fitzgerald, *Understanding open source software development*, London 2002, s. 26. „Pierwsze komputery z początku lat 40. pod względem technicznym i fizycznym w niewielkim stopniu przypominały dzisiejsze urządzenia i służyły przede wszystkim do wykonywania skomplikowanych obliczeń użytecznych w doskonaleniu technik wojennych, ale wykorzystywano je również w celach naukowych i inżynierskich niezwiązanych z kontekstem militarnym. Na przełomie lat 40. i 50. nastąpiła zasadnicza zmiana w podejściu do komputerów. Wizja cyfrowych kalkulatorów, błyskawicznie wykonujących skomplikowane obliczenia na potrzeby techniczne i badawcze, została zastąpiona koncepcją urządzeń uniwersalnych, przetwarzających ogromne ilości danych obecnych nie tylko w nauce, ale także przemyśle czy handlu". M. Góralska, *Perspektywy e-booków w kontekście rozwoju komputerów jako urządzeń uniwersalnych i specjalistycznych*, [w:] *Biblioteka, książka, informacja, Internet 2010*, red. Z. Osiński, Lublin 2010, s. 78.

wewnętrzną nie tylko do magazynowania danych, lecz także do przechowywania w niej programów obsługujących pracę całego komputera. Koncepcja Johna von Neumanna przeszła do historii pod nazwą *The Stored-Program Concept* i jest do dziś wykorzystywana przy konstruowaniu komputerów. Implementację tego pomysłu w konkretnym urządzeniu nazywa się *architekturą von Neumanna*[13]. Pierwszymi komputerami budowanymi w oparciu o pomysły von Neumanna były EDSAC (1949 rok) i UNIVAC (1952 rok)[14]. Nieco upraszczając można przyjąć, że do lat 50. XX wieku rozwijano jedynie fizycznie istniejący sprzęt (hardware), natomiast wraz ze stworzeniem jednego z pierwszych języków programowania, tj. języka ALGOL (1958 rok), rozpoczęła się historia oprogramowania (software'u)[15].

Termin *software* (pol. *oprogramowanie*) oznacza zbiór instrukcji, które kierują pracą komputera w celu wykonania określonej czynności[16]. Software jest czymś odmiennym od hardware'u (pol. *sprzęt komputerowy*)[17]. Software pracuje z hardwarem, czy też steruje jego pracą po to, aby rozwiązać określony problem czy wykonać określone zadanie[18]. Historia oprogramowania, co zrozumiałe, nierozerwalnie związana jest z historią komputerów. Oprogramowanie bowiem dostosowywane jest do pracy z odpowiednimi urządzeniami komputerowymi. Równocześnie, w zależności od posiadanego sprzętu komputerowego, programiści mogą opracowywać odpowiednie aplikacje, tj. takie, które wykorzystają potencjał tkwiący w aktualnie funkcjonujących komputerach.

Począwszy od lat 50. poprzez lata 60. programy komputerowe dostarczano w pakiecie wraz ze sprzętem, zapewniając ich użytkownikom nowe funkcjonalności, co automatycznie zwiększało zainteresowanie samymi urządzeniami[19]. W latach 60. oraz 70. pisanie programów komputerowych

---

[13] Por. N. Dale, J. Lewis, *Computer science illuminated*, ed. 4., Sudbury 2011, s. 123; J. S. Warford, *Computer systems*, ed. 4., Sudbury 2010, s. 168.

[14] Por. P. Gawrysiak, *Cyfrowa rewolucja. Rozwój cywilizacji informacyjnej*, Warszawa 2008, s. 134-166.

[15] Por. U. Hashagen, R. Keil-Slawik, A. Norberg, *History of computing. Software issues*, Berlin 2002, s. 11-22.

[16] W niniejszej pracy będę stosował zamiennie terminy: *program, program komputerowy, oprogramowanie, aplikacja, software, produkt.*

[17] Wymiennie będę też używał określeń *hardware, komputer, sprzęt komputerowy, urządzenie komputerowe, maszyna* itp.

[18] Por. P. E. Ceruzzi, *A history of modern computing*, ed. 2., London 2003, s. 80.

[19] Por. P. Kanwal, A. Gupta, R. K. Singla, *Open source software development. Exploring research perspectives*, [in:] *Emerging trends in computing, informatics, systems sciences and engineering*, ed. T. Sobh, K. Elleithy, New York 2013, s. 609, 607-618.

```
         */
        lunid = le32_to_cpu(*((__le32 *) lunaddrbytes));
        *bus = 0;
        *target = 0;
        *lun = (lunid & 0x3fff) + 1;
        *bus = 1;
        *target = (lunid >> 16) & 0x3fff;
        *lun = lunid & 0x00ff;
} else {
        /* not p1210m... */
        lunid = le32_to_cpu(*((__le32 *) lunaddrbytes));
        if (is_msa2xxx(h, device)) {
                /* msa2xxx way, put logicals on bus 1
                 * and match target/lun numbers box
                 * reports.
                 */
                *bus = 1;
                *target = (lunid >> 16) & 0x3fff;
                *lun = lunid & 0x00ff;
        if (likely(is_scsi_rev_5(h))) {
                /* All current smart arrays (circa 2011) */
                *bus = 0;
                *target = 0;
                *lun = (lunid & 0x3fff) + 1;
        } else {
                /* Traditional smart array way. */
                /* Traditional old smart array way. */
                *bus = 0;
                *lun = 0;
                *target = lunid & 0x3fff;
                *lun = 0;
```

Fot. 1. Fragment kodu źródłowego jądra systemu operacyjnego Linux[20].

było domeną małej grupy naukowców. Oprogramowanie było tworzone i rozwijane na uniwersytetach przy ścisłej współpracy z laboratoriami naukowymi kilku dużych firm. Wytwarzane programy były następnie swobodnie rozpowszechnianie i udostępniane w kręgach zainteresowanych osób[21]. Można przyjąć, iż oprogramowanie traktowano podobnie, jak wiedzę naukową, tj. jako wspólne publiczne dobro, stąd zakładano, że każdy powinien mieć do niego wolny dostęp. W rozpowszechnianiu aplikacji znaczącą rolę odgrywała ówczesna sieć ARPANET. Dzielenie się programami i ich wymiana wraz z kodem źródłowym były gwarancją rozwoju tego sektora. Ponadto zapewniało to szybkie tworzenie

[20] Źródło: *The Linux kernel archives* [online], [dostęp: 7.11.2012]. Dostępny w WWW: http://www.kernel.org/diff/diffview.cgi?file=%2Fpub%2Flinux%2Fkernel%2Fv3.0%2Fpatch-3.2.33.bz2;z=all.

[21] Por. J. A. Lee, *Nonprofit organizations and the intellectual commons*, Cheltenham, Northampton 2012, s. 33.

oprogramowania, stałe poprawianie różnych niedociągnięć i jeszcze szybsze wprowadzanie do obiegu nowych wersji. Nietrudno się domyślić, że przyczyniało się to istotnie do poprawy jakości software'u[22]. Poza tym, do połowy lat 60. XX wieku oprogramowanie było dostarczane wraz z komputerami jako darmowy dodatek, a kod źródłowy (fot. 1) tych programów był ogólnie dostępny dla każdego. W tym okresie firmy komputerowe zyski czerpały przede wszystkim ze sprzedaży komputerów[23].

Kod źródłowy (ang. *source code*) są to szczegółowe instrukcje, które wykonuje określony program, napisane i przedstawione w odpowiednim języku programowania[24]. Można podać następującą definicję: „*kod źródłowy* (również *źródło* lub *źródła*) – program komputerowy w postaci takiej, jaką tworzy ją człowiek w pewnym języku programowania, zazwyczaj jako tekst, ale też jako dane dla translatora, przeznaczony do analizowania i modyfikacji przez człowieka. Kod źródłowy jest przetwarzany przez translator na kod maszynowy zrozumiały dla maszyny (procesora) lub jest analizowany i wykonywany przez specjalny program zwany interpreterem, może być też przetworzony na kod pośredni"[25].

W tym samym czasie w branży komputerowej zaczęto zmieniać politykę marketingową i już w 1965 roku niektóre firmy zaprzestały dostarczania kodu źródłowego wraz z oprogramowaniem oraz dostarczania oprogramowania z komputerami, co miało zapewnić tym firmom wzrost zysków poprzez rozdzielną sprzedaż tych komponentów[26]. W latach 60. rozpoczęto więc rejestrowanie programów komputerowych w urzędzie ochrony praw autorskich. Od lat 70. zaczęto zaś coraz powszechniej oprogramowanie sprzedawać, tym samym kod

---

[22] Por. M. Muffatto, *Open source. A mutlidisciplinary approach,* London 2006, s. 4.

[23] Por. G. R. Madey, *Open source software*, [in:] *Berkshire encyclopedia of human-computer interaction*, ed. W. S. Bainbridge, Great Barrington 2004, s. 523.

[24] Por. N. Hirst, M. Brocklebank, M. Ryder, *Containment systems. A designed guide,* Rugby 2002, s. 182.

[25] *Helionica. Sieciowa encyklopedia informatyki* [online], [dostęp: 28.01.2012]. Dostępny w WWW: http://encyklopedia.helion.pl/index.php/Kod_źródłowy.

[26] Por. C. J. Moschovitis, H. V. Poole et. al., *The Internet. [Volume 3], chronology. A historical encyclopedia*. Santa Barbara 2005, s. 198; H. T. Tavani, *Ethics and technology : controversies, questions, and strategies for ethical computing*, Hoboken 2011, s. 232; C. J. Woodard, J. West, *Strategic responses to standardization. Embrace, extend or extinguish*, [in:] *Project-based organizing and strategic management*, ed. G. Cattati et al., Bingley 2011, s. 269.

źródłowy zaczęto traktować jako pilnie strzeżoną tajemnicę[27]. W latach 70. narodziło się zatem oprogramowanie komercyjne[28].

Dodatkowo, w 1976 roku wraz z ustawą o prawie autorskim (The Copyright Act) w USA, twórcy (zleceniodawcy konkretnych aplikacji – w rzeczywistości po prostu przedsiębiorcy) m.in. oprogramowania uzyskali szereg kluczowych praw. Przede wszystkim możliwość kopiowania, dystrybuowania i tworzenia kolejnych wersji programów. Tylko w gestii twórcy programu pozostawało, drogą stosownych licencji, przekazywanie niektórych lub wszystkich z tych praw innym użytkownikom[29]. Jednak w latach 70. oprogramowanie nie było jeszcze tak bardzo obciążane wartością komercyjną, w przeciwieństwie do urządzeń (hardware'u). Aplikacje były traktowane raczej jako narzędzia zapewniające prawidłową pracę urządzeń. Swobodne rozpowszechnianie oprogramowania wpływało bowiem na zwiększone zainteresowanie komputerami, a więc wpływało na rozpowszechnianie samych komputerów[30]. Rezultatem tego była niewielka, w porównaniu z XXI wiekiem, ilość programów zgłoszonych do tamtejszego urzędu ochrony praw autorskich. W latach 1964-1978 zgłoszono około tysiąca dwustu tego typu programów[31].

Dopiero na początku lat 80. prywatni inwestorzy zdali sobie w pełni sprawę z tego, że oprogramowanie, na tworzenie którego łożyli środki, stanowi wartościową własność intelektualną, którą trzeba chronić, a nie dzielić się nią za darmo. Firmy zaczęły więc ograniczać prawem autorskim wszystko to, co sponsorowały, ograniczając zwłaszcza możliwość wykorzystywania kodu źródłowego programów, jak również opracowując aplikacje tak, aby ich kod źródłowy nie był widoczny lub bardzo trudny do odczytania dla innych[32]. Jedną z metod jest tzw. zaciemnianie kodu (ang. *obfuscation*), polegające na jego przekształceniu tak, aby innym trudno było odtworzyć pierwotną strukturę składniową aplikacji

---

[27] Por. S. Richter, *Critique for the open source development model*, München 2007, s. 3-4.

[28] Por. P. Kanwal, A. Gupta, R. K. Singla, op. cit., s. 609.

[29] Por. M. Overly, *The open source handbook*, Silver Spring 2003, s. 1.

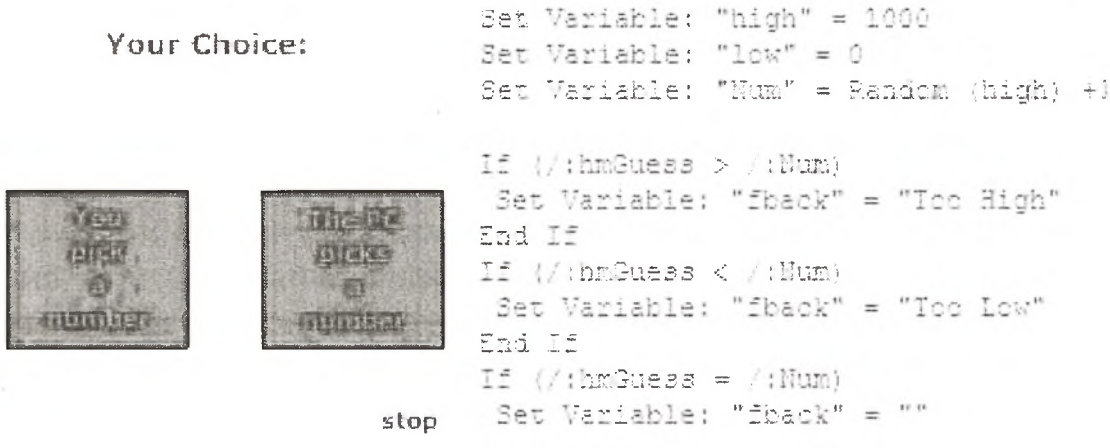
[30] Por. M. Muffatto, op. cit., s. 6.

[31] Por. P. Samuelson, *A case study on computer programs*, [in:] *Global dimensions of intellectual property rights in science and technology*, ed. M. B. Wallerstein, M. E. Mogee, R. A. Schoen, Washington 1993, s. 286.

[32] Por. C. Lowe, *A brief history of open source. Working to make knowledge free* [online], [dostęp: 22.01.2012]. Dostępny w WWW: http://english.ttu.edu/kairos/6.2/binder.html?news/opensource.htm.

i dokonywać na niej tzw. inżynierii wstecznej[33]. Celem inżynierii wstecznej jest m.in. wykorzystywanie fragmentów kodu do własnych celów.

Zatem aby stworzyć aplikację należy wykorzystać do tego celu odpowiedni język programowania, np. linearny (Fortran), funkcjonalny (LISP), proceduralny (Pascal) czy obiektowy (Java), języki te bowiem potrzebne są *de facto* do wyrażania (utrwalania) określonych algorytmów[34]. Algorytm jest ciągiem ściśle określonych czynności służących do wykonania pewnego zadania. „Algorytm ma przeprowadzić system z pewnego stanu początkowego do pożądanego stanu końcowego"[35]. Następnie z algorytmu wyrażonego w określonym języku i utrwalonego w określonym pliku tworzy się program wykonujący ten algorytm, a ten z kolei zapisywany jest w pliku o innym rozszerzeniu (tzw. pliku wykonywalnym) i wreszcie taki program przeznaczony jest do działania w określonym środowisku systemowym, tj. środowisku systemu operacyjnego, lub innym specyficznych środowisku uruchomieniowym[36]. W tym przypadku ma



Fot. 2. Przykład gry (*Simple guessing game with actionscript*) wraz z fragmentem kodu źródłowego[37].

---

[33] Por. M. Jacob, D. Boneh, E. Felten, *Attacking by obfuscated cipher by injecting faults*, [in:] *Digital rights management. ACM CCS-9 workshop, DRM 2002, Washington, DC, USA, November 18, 2002. Revised papers*, ed. J. Feigenbaum, Berlin 2003, s. 17.

[34] Por. J. Jełowicki, *Zajęcia z podstaw informatyki* [online], [dostęp: 7.11.2012]. Dostępny w WWW: http://karnet.up.wroc.pl/~jasj/cwiczenia/kwpp4.html.

[35] *Wikipedia. Wolna encyklopedia* [online], [dostęp: 7.11.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Algorytm.

[36] Por. J. Jełowicki, op. cit.

[37] Źródło fotografii: *28 Exciting Flash Game Tutorials With FLA Files* [online], [dostęp: 28.07.2013]. Dostępny w WWW: http://www.designsmag.com/wp-content/uploads/2010/10/guess-flash-game.jpg.

się już więc do czynienia z kodem źródłowym programu. Jego postać jest taka, jaką nadał mu programista. Natomiast użytkownik końcowy programu widzi na ekranie komputera lub innego urządzenia wynik pracy programisty i realizację tej pracy, tj. wykonanie przez maszynę polecenia wydanego przez ten program. Rezultatem pracy programu jest więc określona funkcjonalność tego programu. Postaram się zobrazować to poniższym przykładem (fot. 2).

Fotografia przedstawia prostą grę komputerową. Po lewej stronie znajduje się interfejs tej gry widoczny dla użytkownika, natomiast po prawej stronie widać fragment kodu źródłowego tej samej gry.

Dzięki dostępowi do kodu źródłowego programista bez trudu może odnaleźć stosowne fragmenty kodu odpowiedzialne za konkretne funkcje programu, jak również może je zmienić, poprawić, usunąć. Z punktu widzenia użytkownika końcowego *prima facie* nie ma znaczenia, czy kod będzie dostępny czy ukryty. Gdy jednak dodam, że jawny kod źródłowy oznacza w rezultacie szybsze skorygowanie błędów i bardziej niezawodną pracę programu, to może się okazać, że ma to już jednak znaczenie. To odwrócenie perspektywy pozwala zatem lepiej zrozumieć istotę całego problemu.

Odnosząc to do codziennej rzeczywistości można podać przykład odzieży. Kupując np. spodnie, można je skrócić lub zwęzić tak, aby idealnie pasowały w danej chwili. Może się jednak okazać, że za rok trzeba będzie je jeszcze bardziej dopasować. Nikogo to nie dziwi i jest normalne, gdy mowa o ubraniach. Inaczej jednak sprawa ta jawiłaby się, gdyby prawo do przeróbek krawieckich posiadał tylko ich producent.

W historii ruchu open source wydziela się trzy zasadnicze okresy. Pierwszy przypada na lata 60. i 70., kiedy to wspólne prace nad oprogramowaniem dopiero się rozwijały. Drugi – na lata 80., kiedy doszło do ekspansji oprogramowania zamkniętego. I wreszcie trzeci, od początku lat 90., kiedy rozwinął się Internet, co doprowadziło do upowszechnienia sieciowej współpracy[38].

[38] Okresy te wydzielił A. Walczak, *Open source projects – how are they organized and financed* [online], [dostęp: 26.03.2014]. Dostępny w WWW: www.slideshare.net/walec51/open-source-projects-how-are-they-organized-and-financed.

1.1. RICHARD MATTHEW STALLMAN I FREE SOFTWARE FOUNDATION

Kolejna dekada doprowadziła zatem do tego, że programy zaczęto tworzyć w zupełnej izolacji, która zapewniała programistom pełną kontrolę nad opracowywaną technologią. Dawna atmosfera swobodnej współpracy i dzielenia się własnymi produktami pomiędzy ówczesnymi programistami („hakerami") zorganizowanymi w dwa główne środowiska, tj. zawodowe, pracujące na uniwersytetach oraz hobbystyczne, zrzeszone w Homebrew Computer Club, ustąpiła kulturze korporacyjnej[39]. Postępująca komercjalizacja rynku IT dotknęła wszystkich, włączając Stallmana (od 1980 roku pracował w prywatnej firmie informatycznej), dla którego własne poglądy etyczne okazały się nie do pogodzenia z obowiązującymi zasadami i związanymi z tą pracą wymaganiami (m.in. koniecznością tworzenia oprogramowania zamkniętego – ang. *closed source* – dostępnego w postaci kodu wykonywalnego)[40]. Na rozwój poglądów Stallmana wpłynęła przede wszystkim codzienna rzeczywistość pracy z urządzeniami elektronicznymi, np. drukarkami. Sprzęt drukujący i dostarczane wraz z nim oprogramowanie było licencjonowane tak, że w zasadzie nie można było go przeprogramować, dostosowując do własnych potrzeb. „Konsument, nabywając jakiś produkt – taki jak drukarka laserowa – powinien mieć możliwość wykorzystania go w dowolny sposób, zaś ograniczenie dostępności kodu źródłowego owe możliwości wykorzystywania znacząco ogranicza. Drugim powodem była z kolei natura ograniczeń prawnych, jakie w tym przypadku spowodowały niemożliwość otrzymania kodu źródłowego"[41]. Choć Stallman skontaktował się z programistami, którzy w tym przypadku napisali program, to, z powodu zawartych umów (nieujawnianie tajemnicy handlowej), nie mogli oni jednak kodu źródłowego nikomu udostępnić. W konsekwencji Stallman odczytywał to jako chowanie części ludzkiej wiedzy przed światłem dziennym[42].

---

[39] Por. P. Gawrysiak, op. cit., s. 286.

[40] Por. Ibidem. Stallman pracował wówczas w MIT (Massachusetts Institute of Technology, pol. Instytut Technologii w Massachusetts), a dokładniej w AI Laboratory MIT oraz w firmie Lisp Machines. Por. Ibidem.

[41] Ibidem, s. 289.

[42] Por. Ibidem.

Rozwijająca się sytuacja wpłynęła więc na jego stanowisko i już w 1983 roku ogłosił plan stworzenia systemu operacyjnego GNU. Ten sam rok przyjmuje się za datę zainicjowania ruchu wolnego oprogramowania (ang. *free software movement*)[43]. Stallman obawiał się bowiem, że tworzony od końca lat 60. i rozwijany zwłaszcza w latach 70., kiedy to programiści mogli swobodnie wymieniać się kodem lub jego fragmentami, system UNIX przestanie być tak prężnie rozwijany i utraci swój potencjał, skoro pojawiły się plany, aby dostęp do kodu źródłowego tego systemu ograniczyć i zacząć prawnie chronić[44].

Moreno Muffatto przypomniał, że pierwsza wersja systemu operacyjnego UNIX była stworzona w latach 1969-1974 przez Kena Thompsona i grupę naukowców z Bell Labs Computer Reserach Department. UNIX był systemem operacyjnym, który mógł być używany na różnych komputerach. Innymi słowy UNIX był pierwszym kompatybilnym z różnymi urządzeniami komputerowymi systemem operacyjnym. Kod źródłowy tego systemu był również swobodnie dostępny. Zapewniało to stałe ulepszanie sytemu, nad którym poza pracownikami Bell Labs mogli pracować także inni programiści[45]. Obowiązujące wówczas firmę regulacje prawne uniemożliwiały wykorzystywanie UNIX-a komercyjnie, stąd jego kod źródłowy udostępniany był darmowo, a przy tym firma nie zapewniała żadnego wsparcia[46]. Użytkownikami UNIX-a byli w większości naukowcy, którzy w konsekwencji tego zaczęli się zrzeszać w nieformalne społeczności, aby wzajemnie sobie pomagać w pracy z tym systemem. Co zrozumiałe, współpraca ta polegała w dużej mierze na przekazywaniu sobie fragmentów kodu systemu[47].

Rok 1984 stał się przełomowym, ponieważ wtedy właśnie właściciel Bell Labs firma AT&T zdecydowała się na to, aby zacząć sprzedawać system UNIX, tym samym UNIX zaczął być chroniony prawem własności, co oznaczało, że od tej pory przestał być dostępny za darmo, a co za tym idzie, zablokowano możliwość rozwijania go przez kogokolwiek

---

[43] Por. C. DiBona, S. Ockman, M. Stone, *Open sources. Voices from the open source revolution*, Beijing, Sebastopol 1999, s. 2-3.

[44] Por. K. S. Amant, B. Still, *Open source software. Technological, economic, and social perspectives*, Hershey 2007, s. 24.

[45] Por. M. Muffatto, op. cit., s. 5.

[46] Por. M. Sojer, *Reusing open source code. Value creation and value appropriation perspectives on knowledge reuse*, Wiesbaden 2011, s. 37.

[47] Por. Ibidem.

bez zgody firmy AT&T[48]. Z roku na rok opłaty za licencje na używanie UNIX-a stale rosły. Choć drogą licencji uzyskiwało się dostęp do kodu systemu, to licencjobiorcy mogli w wersji binarnej rozpowszechniać tylko swoje własne wersje systemu, co ostatecznie doprowadziło do tego, że do obiegu wprowadzono szereg wersji binarnych UNIX-a, które były przy tym niekompatybilne z innymi wersjami[49].

Trzeba pokrótce wyjaśnić, że „program komputerowy składa się najczęściej z dwóch części: z kodu programu napisanego w języku wyższego poziomu (bliższego j. naturalnemu) oraz z binariów, czyli kodu maszynowego stworzonego dla konkretnego komputera. Zależność między kodem programu, a jego binarną wersją jest stosunkowo prosta – binarna wersja programu powstaje po kompilacji, czyli przetworzeniu przez kompilator danego języka programowania, kodu programu. [...] Z punktu widzenia programisty binarna wersja programu to czarna skrzynka, której poznanie, wykorzystanie w innym programie czy też modyfikacja graniczy z niemożliwością. W przypadku posiadania kodu programu możliwe jest jego poznanie jeszcze przed uruchomieniem jego skompilowanej wersji. Natomiast w momencie obcowania z binarną jego wersją o wiele prościej jest uruchomić program, niż analizować kod maszynowy w nim zawarty"[50].

W 1984 roku Stallman odszedł z MIT i rozpoczął tworzenie zapowiadanego przed rokiem systemu[51]. Rok 1984 jest więc czasem, kiedy Stallman rozpoczął działania zmierzające do stworzenia warunków pozwalających wszystkim programistom na swobodną i nieskrępowaną ograniczeniami komercyjnymi działalność. Zdaniem Stallmana otwarta współpraca w zakresie tworzenia oprogramowania przyczynia się do poprawy jakości oprogramowania w ogóle[52]. Cele, które mu przyświecały, przedstawił w 1985 roku w *Manifeście GNU* (*GNU manifesto*)[53]. Czytamy w nim m.in., że manifest został napisany „w celu uzyskania współpracowników i poparcia [...]. Uważam, że złota zasada wymaga, żebym programem, który mi się podoba, podzielił się z innymi, którym też się spodobał. Sprzedawcy opro-

---

[48] Por. M. Muffatto, op. cit., s. 6.

[49] Por. M. Sojer, op. cit., s. 37.

[50] K. Zalewski, *Czym jest free oraz open source software?* [online], [dostęp: 28.07.2013]. Dostępny w WWW: http://blog.4zal.net/2009/08/06/czym-jest-free-oraz-open-source-software/.

[51] Por. C. DiBona, S. Ockman, M. Stone, op. cit., s. 2-3.

[52] Por. S. Richter, op. cit., s. 4.

[53] Polskojęzyczna wersja *Manifestu* znajduje się pod adresem: *Manifest GNU* [online], [dostęp: 23.01.2012]. Dostępny w WWW: http://www.gnu.org/gnu/manifesto.pl.html.

gramowania chcą podzielić użytkowników i nad nimi zapanować poprzez zmuszanie ich, by zgodzili się nie dzielić zakupionym oprogramowaniem. [...] Duża ilość programistów jest niezadowolona z faktu komercjalizacji oprogramowania systemowego. Wprawdzie pozwala to im zarabiać więcej pieniędzy, ale zarazem powoduje, że czują się poróżnieni z innymi, zamiast czuć się jak towarzysze. Podstawowym aktem przyjaźni pomiędzy programistami jest dzielenie się programami; typowe współczesne układy marketingowe zasadniczo nie pozwalają programistom traktować innych jak przyjaciół. Nabywca oprogramowania musi wybierać pomiędzy przyjaźnią a przestrzeganiem prawa. Oczywiście wielu decyduje, że przyjaźń jest ważniejsza. Ale ci, którzy wierzą w prawo, często nie są usatysfakcjonowani żadną z tych alternatyw. Stają się cyniczni i myślą, że programowanie to tylko sposób zarabiania pieniędzy. [...] Gdy GNU już zostanie napisane, wszyscy będą mogli zyskać dobry system bez żadnych ograniczeń, niczym powietrze. Oznacza to znacznie więcej niż tylko zaoszczędzenie wydatku na licencję uniksową. Oznacza to, że nie będzie się już marnować czasu i wysiłku na duplikowanie tych samych partii kodu systemowego. Ten czas i wysiłek będzie można zamiast tego przeznaczyć na postęp w dziedzinie programowania. [...] Kompletne źródła systemu będą dostępne dla każdego. [...] Szkoły będą mogły zaoferować o wiele bardziej pouczające zajęcia poprzez zachęcanie studentów do badania i ulepszania kodu systemowego. [...] Umowy zmuszające ludzi do płacenia za program, włączając licencjonowanie kopii, zawsze ściągają na społeczeństwo ogromne koszty poprzez kłopotliwe mechanizmy potrzebne do określenia ile (tzn. za które programy) trzeba zapłacić. [...] W dalszej perspektywie rozwój wolnych programów jest krokiem ku światu dostatku, w którym nikt nie będzie musiał ciężko harować tylko po to, by przeżyć. Ludzie będą mogli oddawać się czynnościom, które sprawiają radość, takim jak programowanie"[54].

W 1985 roku Stallman założył Free Software Foundation (pol. Fundacja Wolnego Oprogramowania) i zaczął opracowywać i upubliczniać kod systemu operacyjnego GNU. Akronim GNU należy odczytywać jako *GNU's Not UNIX*[55]. Na polski tłumaczy się jako 'działający podobnie do systemu UNIX' („podobny" do systemu UNIX[56]). Słowo *gnu* bezpośrednio wskazuje na gatunek zwierzęcia, tj. antylopę gnu, co uwidacznia stosowane na potrzeby systemu GNU logo (fot. 3).

---

[54] Ibidem.

[55] Por. K. S. Amant, B. Still, op. cit., s. 24.

[56] Por. J. Feller, B. Fitzgerald, op. cit., s. 32; C. DiBona, S. Ockman, M. Stone, op. cit., s. 2-3.

Fot. 3. Logo GNU[57].

W tym samym czasie Stallman wprowadził również do publicznego dyskursu termin *free software* (pol. *wolne oprogramowanie*). W jego założeniu oprogramowanie powinno być wolne w takim stopniu, aby każdy mógł je modyfikować, dostosowując je do własnych potrzeb[58]. Na stronie internetowej inicjatywy Stallmana można przeczytać, że „wolne oprogramowanie daje użytkownikowi cztery prawa: Wolność do uruchamiania programu w dowolnym celu (wolność 0). Wolność analizowania, jak program działa i dostosowywania go do swoich potrzeb (wolność 1). Warunkiem koniecznym jest tu dostęp do kodu źródłowego. Wolność do rozpowszechniania kopii, byście mogli pomóc innym ludziom (wolność 2). Wolność 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"[59]. „Traktowane jako całość powyższe prawa są *de facto* przeniesieniem praw naturalnych, dotyczących przedmiotów kultury materialnej, na obiekty wirtualne, jakimi są programy komputerowe. W przypadku przedmiotów materialnych wszystkie owe wolności są niejako <<wbu-

[57] Logo pobrane ze strony: *Wikipedia. Wolna encyklopedia* [online], [dostęp: 23.01.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Projekt_GNU. (licencja CC BY-SA 2.0)

[58] Por. P. Kavanagh, *Open source software. Implementation and management*, Amsterdam 2004, s. 2. „Wolne oprogramowanie to kwestia wolności użytkownika aby uruchamiał, powielał, rozprowadzał, badał, zmieniał oraz ulepszał oprogramowanie". *Czym jest wolne oprogramowanie?* [online], [dostęp: 22.01.2012]. Dostępny w WWW: http://www.gnu.org/.

[59] Ibidem.

dowane>> w cechy samych przedmiotów i uważane są – a przynajmniej ma to miejsce w kulturze europejskiej – za naturalne prawa posiadacza przedmiotu"[60]. Innymi słowy, jak wyjaśnił Piotr Gawrysiak, Stallman chciał uzyskać dla użytkowników programów możliwość wykorzystania programu uniwersalnie, a nie tylko tak, jak chce tego producent. Poza tym chciał zapewnić możliwość zastosowania tzw. inżynierii wstecznej (ang. *reverse engineering*), tj. umożliwić poznawanie zasad działania programu oraz umożliwić niczym nieograniczone powielanie i dystrybuowanie programu. Wreszcie przewidywał możliwość modyfikowania programu, ulepszania i rozbudowywania go, co miałoby ochronić aplikację przed staniem się tzw. *abandonware*, czyli programem, którego już dalej rozwijać nie sposób, np. z powodu zniknięcia z rynku firmy, która miała wyłączne prawa do modyfikowania go[61].

Wyjściowe założenia Stallmana zasiliły następnie działania legislacyjne. Innymi słowy idea dotycząca własności została przekuta na zadania polegające na zabezpieczaniu dystrybucji i dostępu w miejsce ograniczania tego dostępu. Idea ta musiała zostać unormowana, aby faktycznie sprawnie oddziaływać. Na mocy stosownych licencji określone sytuacje stały się klarowne. Jeżeli ktoś korzysta z oprogramowania wolnego, modyfikuje je ale odrzuca możliwość późniejszej dystrybucji swoich modyfikacji, tym samym zaprzeczając idei leżącej u podstaw całej inicjatywy, musi ponieść konsekwencje. Jedną z nich jest brak wsparcia osób skupionych wokół tej społecznej inicjatywy, co jednakże wydaje się zbyt błahe. Drugą jednak, która jest o wiele skuteczniejszym środkiem zaradczym przeciwdziałającym takim praktykom, jest publikowanie oprogramowania na stosownych licencjach. Licencjonowanie programów powoduje, że tego typu praktyki stają się po prostu nielegalne, a co za tym idzie, osoba, która dopuszcza się takich aktów, łamie prawo[62].

W przestrzeni pojawiającego się i funkcjonującego już prawa chroniącego m.in. kod źródłowy programów komputerowych, tworzenie oprogramowania wolnego od tych obciążeń wymagało opracowania stosownych licencji, mocą których klarownie i precyzyjnie przedstawiono by prawa obejmujące wolne oprogramowanie. Tak się też stało. Dla potrzeb Projektu GNU Richard M. Stallman zaproponował licencję GNU General Public

---

[60] P. Gawrysiak, op. cit., s. 291.

[61] Por. Ibidem, s. 291-295.

[62] Por. T. Jordan, *Hacking. Digital media and technological determinism*, Cambridge 2008, s. 107.

License (w skrócie GPL, pol. Powszechna Licencja Publiczna). Pierwsza wersja licencji GPL została opublikowana w 1989 roku[63]. Był to dość znaczący krok w historii rozwoju wolnego oprogramowania. Dysponentem licencji GPL została Fundacja Wolnego Oprogramowania, która umożliwiła innym wykorzystywanie licencji do dystrybuowania oprogramowania[64]. Sama licencja GPL nie została jednak udostępniona na zasadach GPL, gdyż prawo do jej modyfikowania zostało ograniczone. W późniejszym okresie przyjęto możliwość, aby inni modyfikowali licencję GPL i tworzyli swoje własne jej wersje. Zabroniono jednak używania nazwy GNU w nowej licencji, a także wprowadzono dodatkowe formalne ograniczenia. Chcąc tworzyć nową wersję licencji opartej o model GPL najlepiej skontaktować się z Fundacją Wolnego Oprogramowania[65].

W dyskursie hakerskim krąży stwierdzenie, że licencja GPL była największym włamaniem (ang. *greatest hack*) Stallmana. Była ona bowiem znaczącym ulepszeniem poprzedniej wersji licencji stosowanej przez niego wcześniej, tj. od 1985 roku na potrzeby tworzonej aplikacji Emacs stosował notę licencyjną copyright[66]. Sam Williams w książce *W obronie wolności : krucjata hakera na rzecz wolnego oprogramowania* wyjaśnił, że początkowo Stallman stosował bowiem notę copyright jako elastyczną postać licencji, wiedząc, że istniała możliwość zrzeczenia się części swych praw pod warunkiem, że użytkownik zastosuje się do określonych zasad. Można było na przykład zezwolić na kopiowanie programu z zastrzeżeniem, że program ten nie będzie wykorzystywany w celach komercyjnych. Użytkownicy wersji 15.0 programu Emacs mogli tworzyć kopie programu i je swobodnie rozpowszechniać, jak również tworzyć modyfikacje, z zastrzeżeniem, że nie będą przypisywali sobie wyłącznego autorstwa. Obowiązująca dla aplikacji Emacs licencja była jednak zbyt nieformalna, co wykluczało możliwość wykorzystywania jej w projekcie GNU. Stallman rozpoczął więc konsultacje

---

[63] Por. D. M. German, J. M. González-Barahona, *An empirical study of the reuse of software licensed* under the GNU General Public License, [in:] Open source ecosystems. Diverse communities interacting. 5th IFIP WG 2.13 International Conference on Open Source Systems, OSS 2009, Skövde, Sweden, June 3-6, 2009. Proceedings, ed. C. Boldyreff et.al., Berlin, New York 2009, s. 188.

[64] Por. S. Colford, *Free and open source software*, [in:] *More technology for the rest of us. A second primer on computing for the non-IT librarian*, ed. N. Courtney, Santa Barbara 2010, s. 116.

[65] Por. *Wikipedia. The free encyclopedia* [online], [dostęp: 23.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/GNU_GPL.

[66] Por. H. W. Poole, L. Lambert, C. Woodford, C.J. P. Moschovitis, *The Internet. A historical encyclopedia*, Santa Barbara 2005, s. 216.

z prawnikami oraz innymi pracownikami Fundacji dotyczące stworzenia odpowiedniej licencji. Miał jedynie dwa warunki: licencja miała powodować, że oprogramowanie będzie maksymalnie otwarte oraz zachęcać innych do stosowania tej samej licencji. Ponadto ogłoszono, iż każdy może modyfikować swobodnie GNU Emacs pod warunkiem, że będzie publikował wprowadzone przez siebie zmiany. Każda nowa wersja programu miała być opatrywana tą samą licencją GNU. Można przyjąć, że taka forma licencji była jakby prostym kontraktem, który zawierał określoną z góry cenę programu, z tym, że użytkowników proszono, aby zamiast uiszczać opłaty, udostępnili innym użytkownikom wszelkie wprowadzane przez siebie zmiany. Licencja GNU Emacs wprowadzona została w 1985 roku[67]. Na styku prawa autorskiego i inicjatywy wolnego oprogramowania powstała koncepcja copyleft (L) (fot. 4), zaproponowana przez Stallmana i Fundację Wolnego Oprogramowania, której celem stało się prowadzenie działań zmierzających do tworzenia wolnych programów (ang. *free software*) i zapewnianie, aby nowo powstające programy (lub kolejne ich wersje) były udostępniane na tych samych wolnych zasadach[68]. Licencja GPL zapewnia możliwość kopiowania programów, tworzenia utworów zależnych oraz dystrybuowania programów pierwotnych oraz pochodnych[69].

Fot. 4. Logo (symbol) licencji copyleft[70].

[67] Por. S. Williams, *W obronie wolności. Krucjata hakera na rzecz wolnego oprogramowania* [online], [dostęp: 23.01.2012]. Dostępny w WWW: http://stallman.helion.pl/.

[68] Por. M. Overly, op. cit., s. 1.

[69] Por. Ibidem, s. 1. „Copyleft jest to rodzaj systemu licencjonowania praw autorskich, zezwalający na modyfikację i dowolną redystrybucję pracy". *Wikipedia. Wolna encyklopedia* [online], [dostęp: 24.01.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Copyleft.

[70] Logo pobrane ze strony: *Wikipedia. The free encyclopedia* [online], [dostęp: 27.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Copyleft.

Warto podkreślić, że termin *copyleft*, jak również logotyp (symbol) licencji typu copyleft, wprost nawiązują do licencji typu copyright. Na potrzeby ruchu Creative Commons (CC) w miejsce powszechnie obowiązującej zasady 'wszystkie prawa zastrzeżone' (ang. *all rights reserved*) wprowadzona została zasada 'niektóre prawa zastrzeżone' (ang. *some rights reserved*)[71]. Proweniencja znaku copyleft i jego znaczenie w ruchu open source są jednak nieznacznie inne. Otóż Stallman wyjaśnił, że „w 1984 lub 1985 roku Don Hopkins (znajomy obdarzony wielką wyobraźnią) wysłał mi list. Na jego kopercie wypisał szereg zabawnych sentencji, między innymi następującą: <<Copyleft – all rights reversed>> [<<Copyleft – wszystkie prawa odwrócone>>]. Użyłem słowa *copyleft*, żeby nazwać koncepcję dystrybucyjną, jaką rozwijałem w owym czasie"[72]. Tym samym nawiązano do koncepcji copyright, na dwa sposoby ją niejako „odwracając": w sensie dosłownym, poprzez stworzenie symbolu (logotypu), który stanowi lustrzane odbicie logotypu copyright oraz w sensie metaforyczno-anagramowym. Z jednej bowiem strony niektóre prawa, które zastrzegane są na mocy copyright, na mocy copyleft mogą być łagodzone (są „zdejmowane"). Z drugiej zaś strony wyrazy angielskie re**serv**ed i re**vers**ed mają zbliżone brzmienie, gdyż w wyrazach tych przestawieniu podlegają niektóre tylko litery, a prefiksy i sufiksy pozostają niezmienione. Nietrudno więc dostrzec podobieństwa, ich zbieżność jest oczywista.

Należy wprowadzić jeszcze i rozróżnić takie określenia, jak: *oprogramowanie copyleft*, *oprogramowanie open source*, *licencja open source* oraz *licencja copyleft*. *Oprogramowanie copyleft* to 'wszelkie oprogramowanie podlegające licencji copyleft'. *Oprogramowanie open source* to 'wszelkie oprogramowanie podlegające licencji open source'. *Licencja open source* to 'każda licencja spełniająca postulaty Definicji Open Source'[73]. Wreszcie *licencja copyleft* to 'każda licencja, która narzuca na używanie, modyfikowanie i rozpowszechnianie oprogramowania nią licencjonowanego (tj. oprogramowania copyleft) konieczność udostępniania oraz rozpowszechniania tego oprogramowania lub jego modyfikacji wraz z kodem źródłowym tego programu, a także po drugie, zakłada możliwość two-

---

[71] Por. *W świetle GNU* [online], [dostęp: 3.03.2012]. Dostępny w WWW: http://pl.wikibooks.org/wiki/W_%C5%9Bwietle_GNU.

[72] R. M. Stallman, *Projekt GNU* [online], [dostęp: 3.03.2012]. Dostępny w WWW: http://www.gnu.org/gnu/thegnuproject.html.

[73] Więcej na temat definiowania oprogramowania open source znajduje się w rozdziale 2.

rzenia dalszych modyfikacji tego programu oraz jego rozpowszechniania bez żadnych dodatkowych opłat'[74].

Zanim przejdę do omówienia popularnych modeli licencyjnych chwilę uwagi poświęcę działaniom Erica Stevena Raymonda i Open Source Initiative.

1.2. ERIC STEVEN RAYMOND I OPEN SOURCE INITIATIVE

Jak już wspomniałem, początkowo używano określenia *wolne oprogramowanie*. Dopiero w 1997 roku utworzony został i wprowadzony do dyskursu publicznego termin *open source*. Twórcami tej koncepcji była grupa osób, m.in. Eric S. Raymond, Tim O'Reilly i Bruce Perens[75]. Przełomowym na tej drodze okazał się esej Raymonda pt. *The cathedra and the bazaar* (pol. *Katedra i bazar*). W pracy tej Raymond, przez lata związany z Fundacją Wolnego Oprogramowania, porównał model tworzenia oprogramowania w Fundacji z modelem, jaki wprowadził Linus Torvalds wraz z projektem systemu operacyjnego Linux (od 1991 roku)[76]. Dotychczas stosowany styl tworzenia wolnego oprogramowania Raymond porównał do budowania katedry, kiedy to programista sam lub w grupie przy ścisłej współpracy, aczkolwiek w ciszy, skupieniu i odosobnieniu, a więc nieco w odizolowaniu od innych, tworzy określoną aplikację. Następnie udostępnia ją innym, którzy również w ten sam sposób program ten modyfikują lub korzystają z jego kodu źródłowego, w celu stworzenia własnej wersji programu, zgodnej z indywidualnymi oczekiwaniami. Natomiast styl rozwoju oprogramowania wprowadzony przez Torvaldsa charakteryzował się większą otwartością już na etapie wstępnym, tj. tworzenia samego (pierwotnego) kodu. W miejsce odosobnienia i ciszy społeczność linuksowa przyjmowała formę „hałaśliwego bazaru", gdzie wszyscy od samego początku projektu wprowadzali modyfikacje, proponowali własne rozwiązania, wypowiadali się na temat konkretnych elementów programu – każdy bowiem miał jakiś pomysł

[74] Przywołane określenia oraz ich eksplikacje pochodzą z pracy H. J. Meeker, *The open source alternative. Understanding risks and leveraging opportunities*, Hoboken 2008, s. 240.

[75] Por. P. Kavanagh, op. cit., s. 2-3.

[76] Historię powstania Linuxa omawiam w rozdziale 5.1.

i określony pogląd na to, jak ów program miał w przyszłości wyglądać. Choć początkowo Raymond nie widział możliwości, aby w takim trybie pracy powstał spójny i stabilny system operacyjny, to stało się inaczej. Model bazarowy okazał się o wiele skuteczniejszym modelem współpracy[77]. Linux to nie tylko zestaw oprogramowania, ale przede wszystkim metodologia rozwoju tego oprogramowania przy udziale zespołu zaangażowanych programistów rozsianych globalnie, ale połączonych *via* Internet, którzy tworzą poszczególne elementy składające się na całość, jaką jest system operacyjny (jego dystrybucja)[78].

Chciałem jeszcze zwrócić uwagę na fakt, że podstawowym elementem charakteryzującym ruch open source jest wolontariat. Chcący wziąć udział w procesie open source sam wybiera w czym chce partycypować i w jakim zakresie. Może zaproponować nowy projekt oprogramowania, zasilić już działający, wesprzeć powstający lub wykorzystać dostępne aplikacje. Generalnie jednak jest tak, że zainteresowany wedle własnego uznania wybiera sposób swojego uczestnictwa w tym społecznym przedsięwzięciu. Te dość samodzielne wybory dają ostatecznie rozbudowane i owocne rezultaty, w postaci dobrze funkcjonujących programów. Istnieją więc dwa główne modele tworzenia oprogramowania open source, tj. model katedrowy i bazarowy[79].

Warto jeszcze dodać, że znaczącym krokiem w upowszechnianiu oprogramowania open source był pomysł Netscape Communications Corporation, firmy produkującej popularną wówczas przeglądarkę internetową Netscape, aby ten właśnie sztandarowy produkt zacząć upowszechniać na zasadach open source. Z początkiem 1998 roku pomysł został wdrożony. Zachęceni również tą inicjatywą informatycy Eric Raymond i Bruce Perens powołali organizację non profit – Open Source Initiative (skrót OSI). W tym samym czasie stworzyli też podstawy definicji open source[80].

Pierwszą inicjatywą podjętą przez nowo utworzoną organizację OSI była w 1998 roku propozycja stosowania w miejsce terminu *wolne oprogramowanie* terminu *oprogramowanie open source* (ang. *open source software*, skrót OSS). OSI narzuciła również konieczność włączania do licencji uży-

---

[77] Polskie tłumaczenie *Katedry i bazaru* znajduje się pod adresem: *Katedra i bazar* [online], [dostęp: 25.01.2012]. Dostęp w WWW: http://www.linux-community.pl/node/4.

[78] Por. G. Moody, *Rebel code. The inside story of Linux and the open source revolution*, Cambridge 2001, s. 7.

[79] Akapit opracowałem na podstawie pracy: T. Jordan, op. cit., s. 47.

[80] Informacje zawarte w akapicie powstały w oparciu o pracę: R. Dixon, *Open source software law*, Norwood 2004, s. 7-8.

wanych dla oprogramowania open source takich zapisów, jak: żadne opłaty nie mogą być nałożone na wszelkie redystrybucje oprogramowania; kod źródłowy musi być dostępny; musi być zagwarantowane tworzenie programów na bazie programu pierwotnego, jak również tworzenie modyfikacji tego pierwotnego programu; licencja może narzucić konieczność dystrybuowania pierwotnego programu wraz z jego modyfikacjami, a nie tylko samych modyfikacji; licencjodawca nie może komukolwiek ograniczać dostępu do programu; prawa ciążące na mocy licencji na programie pierwotnym muszą zostać przeniesione na utwór zależny; licencja dotyczy nie tylko samego (całego) oprogramowania, lecz także wszystkich jego komponentów; licencja nie może ograniczać innego oprogramowania, które jest dostarczane wraz z oprogramowaniem licencjonowanym[81]. Tym samym zaproponowane postulaty stały się równocześnie elementami składowymi definicji *oprogramowania open source*[82].

1.3. POPULARNE LICENCJE OPEN SOURCE

Integralną częścią każdego programu komputerowego jest licencja wiążąca ten program i określająca jego użytkowanie. Licencja oprogramowania to nic innego, jak umowa na korzystanie z tego programu zawierana pomiędzy podmiotem, któremu przysługują majątkowe prawa autorskie do programu, a podmiotem, który zamierza z tego programu korzystać[83]. Licencja chroni zatem autora przed nieautoryzowanym kopiowaniem i rozpowszechnianiem programu. Zasadniczo licencja jest wypadkową zasady copyright stosowanej w większości krajów na świecie. Nawet jeśli autor programu nie umieścił przy swoim produkcie stosownego zapisu, prawo i tak jego produkt chroni[84]. Victor Reijswoud i Arjan Jager wyjaśniają, że w przypadku programów OS stosowane są dwie kategorie licencji, tj. te, które nie narzucają żadnych ograniczeń na rozpowszech-

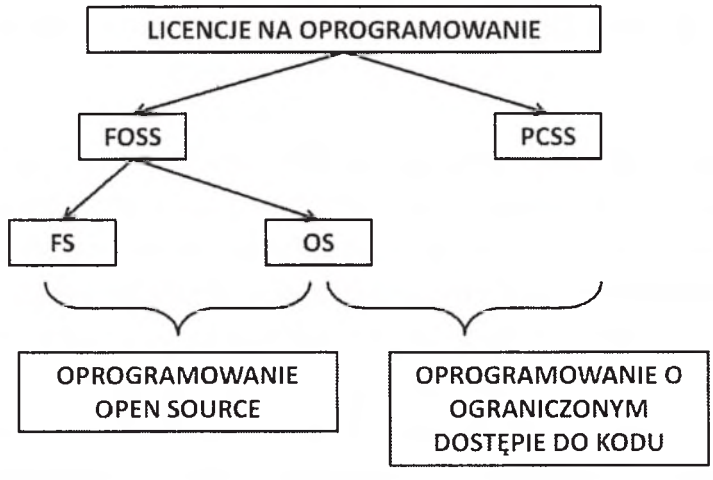
[81] Akapit powstał w oparciu o pracę: M. Overly, op. cit., s. 5-6.

[82] Definicję terminu *open source* omawiam w rozdziale 2.

[83] Por. *Wikipedia. Wolna encyklopedia* [online], [dostęp: 1.08.2013]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Licencja_oprogramowania.

[84] Por. V. V. Reijswoud, A. D. Jager, *Free and open source software for development. Exploring expectations, achievements and the future*, Monza 2008, s. 48.

nianie programu oraz te, które ograniczenia takie narzucają. W rezultacie wyłoniły się więc dwa modele licencyjne: free and open source software (FOSS) oraz prioprietary and closed source software (PCSS). Choć obydwa modele chronią przede wszystkim prawa autorskie do programu, to jednak w kilku kwestiach różnią się zasadniczo. Przede wszystkim celem licencji FOSS jest zdejmowanie większości praw narzucanych etykietą copyright, podczas gdy licencje PCSS utrzymują w mocy te ograniczenia. Jednakże w obszarze FOSS wskazuje się dalej na dwa podstawowe trendy w licencjonowaniu, tj. OS i FS. Licencje FS są bardziej restrykcyjne od licencji OS. Drogą licencji FS postuluje się, jak sama nazwa wskazuje, większą wolność oprogramowania, co ma się wyrażać w nieskrępowanym dostępie do kodu źródłowego programu oraz dowolnym wykorzystaniu programu. Sztandarowym przykładem tego typu praktyk licencyjnych są licencje z grupy GPL. Natomiast licencje OS pozwalają również ograniczać dostęp do kodu źródłowego oraz tworzyć na jego bazie programy zamknięte[85]. Obrazuje to poniższy schemat (rys. 1).



Rys. 1. Główne typy licencji na oprogramowanie [opracowanie własne].

Jak widać na załączonym schemacie (rys. 1), licencje FOSS umożliwiają tworzenie zarówno programów open source'owskich, jak i komercyjnych, tj. o ograniczonym dostępie do kodu źródłowego, podczas gdy licencje PCSS mogą być używane jedynie do licencjonowania tych drugich.

Dla uporządkowania wywodu należy przypomnieć, że w 1989 roku powstała pierwsza wersja licencji, GPLv1, w 1991 roku – druga wersja, GPLv2, a w 2007 – trzecia, GPLv3. Choć druga wersja również jest obowią-

[85] Por. Ibidem, s. 48-50.

zującą i uznawaną (wiele programów udostępnionych jest na tej wersji), to jednak rekomenduje się stosowanie licencji w wersji trzeciej (GPLv3). Żadna z licencji GPL nie jest też kompatybilna z pozostałymi wersjami licencji GPL[86]. Aktualnie zalecaną jest więc licencja GPLv3 (fot. 5), która jest rodzajem licencji typu copyleft na oprogramowanie i inne rodzaje utworów. W preambule Powszechnej Licencji Publicznej GNU (Wersja 3, 29 czerwca 2007 r.) można przeczytać, że „licencje na większość programów komputerowych i na inne utwory zostały stworzone po to, aby ograniczyć swobodę dzielenia się nimi i dokonywania w nich zmian. Natomiast celem Powszechnej Licencji Publicznej GNU jest zagwarantowanie swobody udostępniania i zmieniania wszystkich wersji programu – sprawienie, by oprogramowanie pozostało wolnym dla wszystkich użytkowników. [...] Kiedy mówimy o wolnym oprogramowaniu, chodzi nam o swobodę, nie o cenę. Nasze Powszechne Licencje Publiczne zostały zaprojektowane tak, aby zapewnić ci swobodę rozpowszechniania kopii wolnego oprogramowania (a jeżeli chcesz – swobodę pobierania wynagrodzenia z tego tytułu), zagwarantować, że otrzymasz kod źródłowy lub uzyskasz do niego dostęp, że będziesz mógł zmieniać oprogramowanie lub użyć jego fragmentów w nowych wolnych programach, a także że będziesz miał świadomość, że masz do tego wszystkiego prawo. Aby chronić twoje prawa, musimy zagwarantować, że nikt inny nie będzie mógł zakwestionować twoich praw ani domagać się, żebyś z nich zrezygnował. W związku z tym, jeżeli rozpowszechniasz kopie takiego oprogramowania lub modyfikujesz je, spoczywa na tobie odpowiedzialność za poszanowanie wolności innych. Na przykład, jeżeli rozpowszechniasz kopie takich programów, czy to odpłatnie, czy też nieodpłatnie, musisz przekazać odbiorcom dokładnie te same prawa, jakie sam otrzymałeś. Musisz zagwarantować, że oni także otrzymają kod źródłowy lub uzyskają do niego dostęp. Musisz również pokazać im niniejsze warunki, tak aby mogli poznać swoje prawa. Twórcy oprogramowania korzystający z Powszechnej Licencji Publicznej GNU chronią swoje prawa dwuetapowo: (1) zastrzegają prawa autorskie do oprogramowania oraz (2) oferują ci niniejszą Licencję, udzielając ci prawnego zezwolenia na kopiowanie, rozpowszechnianie i/lub modyfikację tego oprogramowania. Powszechna

[86] Por. D. M. German, J. M. González-Barahona, op. cit., s. 188-189. Na marginesie warto przypomnieć, że w ciągu lat następujących po wprowadzeniu licencji GPLv1 powstawały też inne licencje, jak np. GNU Lesser General Public Licence (LGPL) w latach 1991, 1999 oraz 2007; GNU Affero Public Licence (AGPL) w 2007 roku. Por. Ibidem, s. 188.

Licencja Publiczna chroni programistów i twórców, wyraźnie stwierdzając, że na wolne oprogramowanie nie jest udzielana żadna gwarancja"[87]. Licencje open source'owskie przede wszystkim chronią prawa autora.

Fot. 5. Logo licencji GNU GPLv3[88].

Licencja GPLv3 narzuca wszystkim korzystającym z opartego na niej programu konieczność opatrywania wytworów swojej pracy również tą licencją. Stąd uznaje się ją za wirusową i proponuje mniej restrykcyjne rozwiązania, takie jak np. Licencję BSD[89]. Tę więc grupę licencji (GPL) można uznać za sztandarowy przykład licencji o charakterze *stricte* open source'owskim.

Zanim przejdę do omówienia licencji BSD, zasygnalizuję w tym miejscu główną różnicę pomiędzy nurtami wolnego i otwartego oprogramowania. Każdy z wymienionych nurtów „kojarzony jest z odmiennym podejściem, inną filozofią, innymi wartościami, a nawet z odmiennym kryterium, według którego akceptowane są licencje. Oba ruchy, Wolne Oprogramowanie i Open Source, są obecnie osobnymi ruchami [...]. Fundamentalna różnica między tymi dwoma ruchami leży w uznawanych przez nie wartościach, sposobach patrzenia na świat. Dla Open Source kwestia, czy oprogramowanie powinno mieć dostępne otwarte źródła to problem praktyczny, nie etyczny. Jak to ktoś ujął: <<Open Source to metodyka konstruowania, wolne oprogramowanie to ruch społeczny>>. Dla

[87] M. Maruta, *Tekst licencji gpl v3 po polsku plus prośba o pomoc* [online], [dostęp: 24.01.2012]. Dostępny w WWW: http://itlaw.computerworld.pl/index.php/2008/03/10/tekst-licencji-gpl-v3-po-polsku-plus-prosba-o-pomoc/.

[88] Logo pobrane ze strony: *Wikipedia. The free encyclopedia* [online], [dostęp: 23.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/GNU_General_Public_License.

[89] Por. R. W. Hahn, *Government policy toward open source software. An overview* [online], [dostęp: 28.01.2012]. Dostępny w WWW: http://www.brookings.edu/press/books/chapter_1/governmentpolicytowardopensourcesoftware.pdf; *Wikipedia. Wolna encyklopedia* [online], [dostęp: 24.01.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/GNU_General_Public_License.

ruchu Open Source oprogramowanie, które nie jest wolne to rozwiązanie gorsze niż optymalne. Dla ruchu Wolnego Oprogramowania programy, które nie są wolne to problem społeczny, którego rozwiązaniem jest wolne oprogramowanie"[90]. Sam Stallman wyjaśnił, że obydwa ruchy są jak dwa polityczne obozy działające w ramach zbliżonych ideałów i choć ich wyjściowe założenia są niejednolite, to w kwestiach praktycznych zaleceń dochodzą do mniejszego lub większego konsensusu. Wyraźnie trzeba podkreślić, że mimo dzielących ich różnic stale ze sobą współpracują na polu walki, w której ich przeciwnikiem jest oprogramowanie własnościowe (komercyjne)[91].

Licencja BSD (Berkeley Software Distribution License) została stworzona zaraz po licencji GNU GPLv1, a jej główną cechą jest to, że nie narzuca konieczności udostępniania kodu źródłowego[92]. Licencja BSD, umożliwiając tworzenie zamkniętego oprogramowania, zezwala *de facto*, aby można je było potem sprzedawać[93]. Ponadto nie zawiera mechanizmu copyleft, stąd zapewne o wiele bardziej popularne są licencje GPL[94]. Do produktu licencjonowanego BSD należy jednak zawsze załączać informacje o autorach oryginalnego kodu oraz informacje o treści licencji[95].

Licencje z grupy GPL oraz BSD mają dużo więcej podobieństw niż różnic. Niemniej jednak w jednym aspekcie różnią się zasadniczo. Różnica ta przekłada się na późniejszą możliwość dalszego rozwijania lub nierozwijania określonego oprogramowania. Można bowiem wyróżnić dwa główne typy licencji. Pierwszym są licencje typu GPL, które zapewniają, iż kod źródłowy programu ma pozostać wolny. Drugim natomiast są licencje typu BSD, które dają użytkownikom większą swobodę w udostępnianiu lub nieudostępnianiu kodu źródłowego. Innymi słowy użytkownik licencjonujący aplikację na licencji BSD może ograniczyć dostęp

---

[90] R. M. Stallman, *Dlaczego termin „free software" jest lepszy niż „open source"* [online], [dostęp: 1.08.2013]. Dostępny w WWW: http://www.gnu.org/philosophy/free-software-for-freedom.pl.html.

[91] Por. C. M. Kelt, *Two bits. The cultural significance of free software*, Druham 2008, s. 321-322.

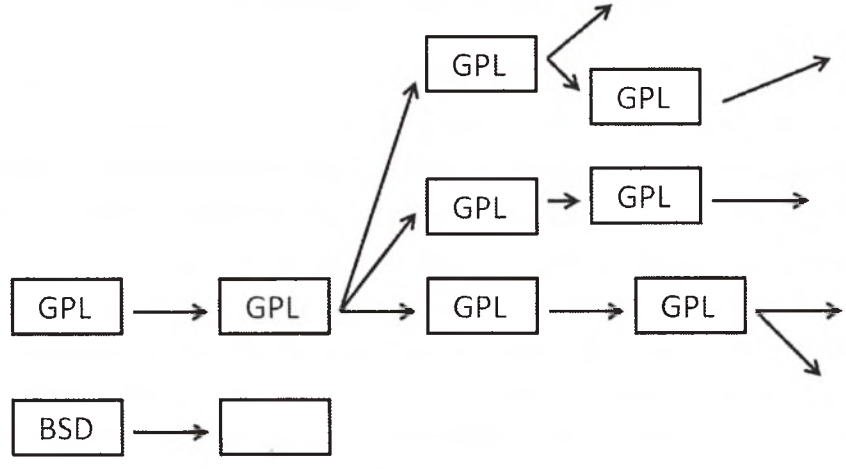
[92] Por. R. Williams, *Real-time systems development*, Oxford, Burlington 2006, s. 220.

[93] Por. A. K. Singh, *Science and technology for civil service*, New Delhi 2008, s. 182.

[94] Por. F. Questier, W. Schreus, *Open courseware and open scientific publication*, [in:] *How open is the future? Economic, social & cultural scenarios inspired by free & open-source software*, ed. M. Wynants, J. Cornelis, Brussels 2005, s. 120-121.

[95] Por. *Wikipedia. Wolna encyklopedia* [online], [dostęp: 24.01.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Licencja_BSD.

do kodu źródłowego lub może z tego zrezygnować, zapewniając do niego dostęp[96]. „Podstawową zaletą licencji tego typu [...] jest prostota. Licencja ta nie daje jednak właściwie żadnej ochrony samemu oprogramowaniu, traktowanemu jako dobro wspólne, zapewniając jedynie ochronę cywilnoprawną jego twórcy, czemu służy końcowa klauzula dotyczącą wyłączenia gwarancji"[97]. Symbolicznie rozwój oprogramowania tworzonego pod tymi licencjami można zaprezentować w postaci następującego schematu (rys. 2).



Rys. 2. Schemat ilustrujący rozwijanie oprogramowania na licencji GPL oraz BSD [opracowanie własne].

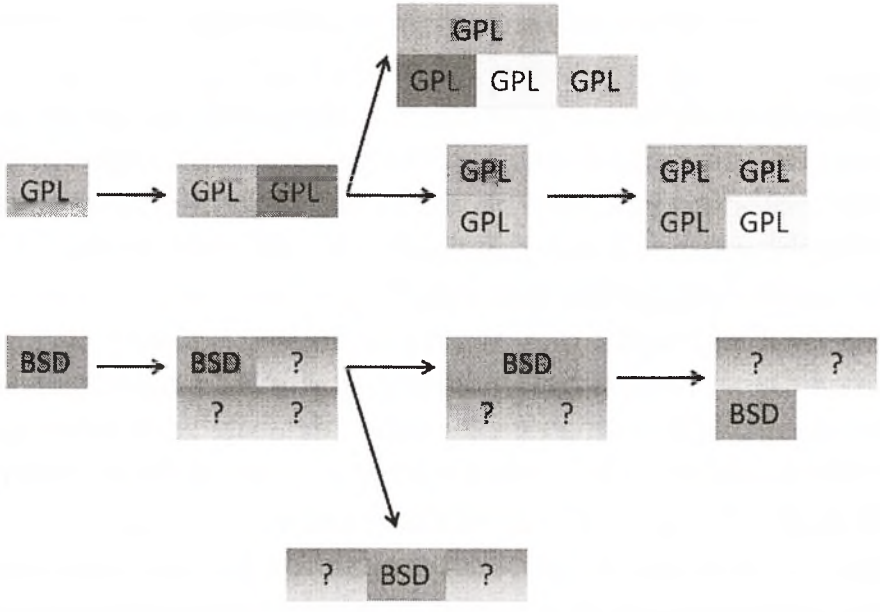
Prostokąt symbolizuje określoną aplikację. Akronimy GPL i BSD umieszczone na prostokątach oznaczają, iż dana aplikacja licencjonowana jest na wskazanej licencji. Strzałki symbolizują tworzenie lub możliwość tworzenia kolejnej wersji programu lub nowego programu opartego na kodzie źródłowym programu pierwotnego. Widać wyraźnie, że licencje typu GPL zapewniają, że oprogramowanie będzie wytwarzane i modyfikowane tylko we współpracy, co zapewnia ciągłość rozwoju oprogramowania. Aby zadanie to móc spełnić, konieczny jest dostęp do kodu źródłowego programu, jak również pozwolenie na modyfikowanie i wykorzystywanie tego kodu. W przypadku licencjonowania programów licencjami BSD rozwój programu może zostać zahamowany, licencje te zezwalają bowiem na ograniczenie dostępu do kodu źródłowego. Licen-

[96] Akapit na podstawie: V. Lindberg, *Intellectual property and open source*, Beijing 2008, s. 177.

[97] P. Gawrysiak, op. cit., s. 315.

cje GPL dodatkowo narzucają konieczność licencjonowania programów pochodnych również na tych samych licencjach, a zatem przyczyniają się do swobodniejszej wymiany informacjami.

Zaprezentowany schemat (rys. 2) należy odczytywać metaforycznie: BSD ogranicza rozwój oprogramowania i hamuje proces innowacyjności, podczas gdy GPL sprzyja jego rozwojowi. Kolejny schemat dokładniej zilustruje mechanizm stosowany w obydwu typach licencji (rys. 3).



Rys. 3. Schemat ilustrujący możliwość dalszego rozwijania oprogramowania na licencji GPL i BSD [opracowanie własne].

Prostokąty symbolizują fragmenty kodu źródłowego składające się na dany program. Prostokąty ze znakiem zapytania oznaczają, że ten fragment kodu źródłowego oprogramowania jest niedostępny. Licencja GPL narzuca konieczność stosowania tej licencji nie tylko w stosunku do wykorzystanego fragmentu kodu źródłowego udostępnionego wcześniej na licencji GPL, lecz także w stosunku do całego programu lub większej części kodu źródłowego, w których wykorzystano fragment kodu na licencji GPL. Reasumując: wszystko, co powstaje z wykorzystaniem produktu GPL, również należy udostępniać na licencji GPL. Licencja BSD z kolei nie narzuca tej konieczności. A zatem wykorzystanie fragmentu kodu lub całego kodu źródłowego udostępnionego na licencji BSD oznacza, że tylko ta część kodu oraz modyfikacje tej części kodu mogą być dalej udostępniane na licencji BSD. Ujmując to prościej: wszystko, co powstaje

z wykorzystaniem produktu na licencji BSD, można udostępniać w sposób zależny od własnej woli, tj. to, co było udostępnione na licencji BSD, na niej pozostaje. Inne elementy programu bazujące również na tym, co było udostępnione na licencji BSD, mogą pozostać zamknięte (dostarczane bez kodu źródłowego). Licencja nie narzuca konieczności udostępniania kodu źródłowego pozostałych części programu.

Podsumowując, zgodnie z popularną licencją BSD z 1999 roku, określaną także jako: *BSD-new*, *New BSD*, *revised BSD*, *BSD-3*, *3-clause BSD*, *3-klauzulowa licencja BSD*, użytkownik, wykorzystując oprogramowanie udostępniane na tej licencji, może „używać i kopiować w niekomercyjnych projektach, modyfikować, łączyć z innym kodem, dystrybuować pod swoją nazwą, zmieniać licencję, sprzedawać skrypt, implementować do własnego niekomercyjnego projektu"[98], nie może natomiast „usuwać praw autorskich, modyfikować oryginalnych praw autorskich, używać znaków firmowych w jakimkolwiek celu"[99].

„GPL daje wolność oprogramowania, a BSD – wolność wyboru"[100]. W interesie użytkowników programów jest, aby wiele z nich było dostępnych na licencjach GPL. Z kolei *prima facie* zdaje się, że w interesie programistów powinno być odwrotnie. Skoro ktoś poświęcił dużo pracy, aby stworzyć program, to niekoniecznie zależy mu na udostępnieniu swojej pracy innym. To jednakże zależy od celu, jaki przyświeca programiście. Jeżeli tworzy program po to, aby na nim zarobić, to faktycznie tak jest. Jeżeli natomiast sam nie jest w stanie stworzyć aplikacji, która zapewni mu pożądaną funkcjonalność, to wtedy może okazać się konieczne pracowanie w grupie, tj. udostępnienie kodu i pozwolenie na jego modyfikowanie. W rezultacie może to dać taki efekt, że inny programista zrealizuje wyjściowe założenie, a kod programu i cały program będą dostępne bez opłat.

---

[98] *Licencja BSD k/3 – proste wytłumaczenie, limitacje oraz przywileje* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://funkcje.net/view/5/12225/index.html.

[99] Ibidem. Warto może, jako przykład zastosowania licencji BSD, wskazać, że wykorzystuje się ją do licencjonowania niektórych systemów operacyjnych (z grupy programów własnościowych), które składają się z całego szeregu komponentów open source udostępnianych właśnie na licencjach BSD. Por. M. W. Lucas, Absolute OpenBSD. UNIX for the practical paranoid, San Francisco 2003, s. 5; Wikipedia. The free encyclopedia [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Mac_OS"

[100] Wpis z dnia 13 lutego 2010 na forum Webhosting użytkownika Saladin, dotyczący wątku: *Licencja BSD & licencja FNU GPL* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://forum.webhosting.pl/licencja-BSD-and-lhcenbja-F-t2474.html&mode=threaded&pid=9586.

W wyniku wprowadzenia do obrotu licencji BSD, Fundacja Wolnego Oprogramowania zaproponowała nową licencję, tj. LGPL (Lesser General Public License, pol. Mniejsza Powszechna Licencja Publiczna, wcześniej Library General Public License). Licencję LGPL można uznać za próbę wypracowania kompromisu pomiędzy *de facto* dość restrykcyjnymi, w pozytywnym tego słowa znaczeniu, licencjami GPL a liberalnymi licencjami BSD. Treść licencji wraz ze Stallmanem przygotował Eben Moglen[101]. LGPL nakłada pewne ograniczenia na pliki źródłowe, ale nie na całe programy. Licencja przeznaczona jest do publikowania kodu tzw. bibliotek (zbiór klas, funkcji), z których korzystają inne programy. LGPL stosuje się do bibliotek, ale nie do programów korzystających z tych bibliotek. Program korzystający z biblioteki na licencji LGPL może pozostać programem zamkniętym (dostarczanym bez kodu źródłowego). Z tego też względu uznaje się LGPL za mniej wirusową licencję od licencji GPL[102].

Warto zestawić i porównać (tab. 1) powyżej przedstawione typy licencji, aby się przekonać o ograniczeniach wynikających z ich stosowania. W tabeli zawarto wyniki porównania w zakresie możliwości łączenia oprogramowania licencjonowanego z innymi programami, tworzenia na programach pierwotnych oprogramowania własnościowego[103] oraz ponownego licencjonowania programu na nowych licencjach. Dodatkowo w tabeli umieszczono domenę publiczną jako swoisty typ „licencji" najbardziej liberalnej, jednak w praktyce stosowanej do specyficznej sfery twórczości[104].

---

[101] Eben Moglen jest profesorem prawa i doradcą prawnym Fundacji Wolnego Oprogramowania. Przygotowywał on większość licencji dla Fundacji. Por. *Wikipedia. Wolna encyklopedia* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Eben_Moglen.

[102] Wyjaśnienie dot. LGPL powstało w oparciu o prace: G. K. Landy, A. J. Mastrobattista, *The IT/digital legal companion. A comprehensive business guide to software, Internet, and IP law,* Burlington 2008, s. 251-252; *Wikipedia. Wolna encyklopedia* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/GNU_Lesser_General_Public_License.

[103] Zamiennie będę stosował określenia: *oprogramowanie komercyjne, oprogramowanie własnościowe* oraz skrót PS, który pochodzi od angielskiego terminu *proprietary software* (pol. *oprogramowanie własnościowe*).

[104] *Domena publiczna* „(ang. *public domain*) – w najwęższym znaczeniu jest to twórczość, z której można korzystać bez ograniczeń, wynikających z uprawnień jakie mają posiadacze autorskich praw majątkowych, gdyż prawa te wygasły lub twórczość ta nigdy nie była lub nie jest przedmiotem prawa autorskiego". *Wikipedia. Wolna encyklopedia* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Domena_publiczna.

Tabela 1

Licencja	Czy może być łączone z innym oprogramowaniem niż OSS?	Czy na bazie tego oprogramowania można tworzyć oprogramowanie własnościowe?	Czy oprogramowanie może być ponownie relicencjonowane?
GPL	Nie	Nie	Nie
LGPL	Tak	Nie	Nie
BSD	Tak	Tak	Nie
Domena publiczna	Tak	Tak	Tak

Porównanie praktyk licencyjnych popularnych licencji open source'owskich oraz domeny publicznej[105].

Aby przekonać się o popularności wymienionych licencji warto przywołać statystki ich stosowania w największej jak dotąd funkcjonującej bazie projektów oprogramowania open source, tj. SourceForge. W marcu 2004 roku 79% licencji, którymi sygnowało się projekty, stanowiły licencje GPL i LGPL[106]. W październiku 2004 roku projekty programów open source były udostępniane na następujących licencjach: 69,5% wszystkich projektów licencjonowano na licencjach GPL, 11% programów licencjonowano na licencjach LGPL, 7% programów licencjonowano na licencjach BSD, a pozostałą grupę – 12,5% programów – na innych dostępnych licencjach open source[107]. Natomiast pod koniec stycznia 2012 roku stosunek ten wyniósł: GPL – 66,53%, LGPL – 11,4%, BSD – 8,2% oraz inne – 13,87% (wyk. 1). Komunikolodzy podają, że ok. 85% wszystkich przedsięwzięć open source sygnowanych jest licencją GPL[108].

---

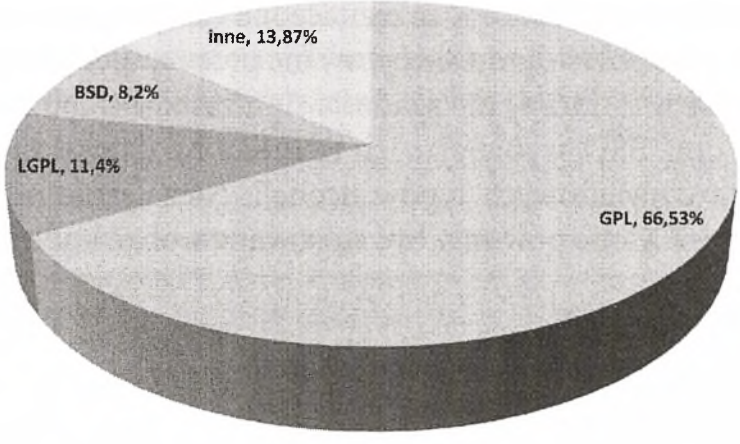
[105] Tabelę opracowałem na podstawie dwóch prac: Por. T. M. Egyedi, R. van Wendel de Joode, *Standardization and other coordination mechanisms in open source software*, [in:] *Information technology standards and standardization research*, ed. K. Jakobs, Hershey 2005, s. 79; B. Perens, *The open source definition* [online], [dostęp: 2.02.2012]. Dostępny w WWW: http://oreilly.com/openbook/opensources/book/perens.html.

[106] Por. R. Goldman, R. P. Gabriel, *Innovation happens elsewhere. Open source as business strategy*, Amsterdam, Boston 2005, s. 116.

[107] Por. K. Szafranek, *Licencjonowanie otwartego oprogramowania* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://szafranek.net/works/articles/opensourcelicenses/.

[108] Por. D. Woods, G. Guliani, *Open source for enterprise. Managing risks, reaping rewards*, Beijing 2005, s. 119.

Jak widać, licencje GNU są bardzo popularne, a ich udział procentowy w stosunku do wszystkich programów udostępnianych na licencjach zgodnych z zasadami przedstawionymi przez OSI wynosi ok. 80%[109]. Przedstawiony powyżej procentowy stosunek kilku najpowszechniej stosowanych licencji na przestrzeni lat nie uległ zbyt dużym zmianom i utrzymuje się na podobnym poziomie. Od wielu też lat licencja GPL jest najpopularniejszą i najchętniej używaną wśród licencji OSS[110], choć należy podkreślić, że jest ona jednocześnie jedną z bardziej restrykcyjnych[111]. GPL swoją wielką popularność może zawdzięczać m.in. temu, że jest to podstawowa i pierwsza z licencji tego rodzaju[112].



Wyk. 1. Udział procentowy licencji open source stosowanych w projektach zarejestrowanych w bazie Sourceforge.net [opracowanie własne][113].

Zaproponuję jeszcze jedno zestawienie (wyk. 2), w którym wykażę udział procentowy dla następujących grup licencji: wszystkie licencje potwierdzające, że dane oprogramowanie jest open source, projekty licencjonowane na licencjach Creative Commons (skrót CC), projekty należące do domeny publicznej oraz pozostałe (inne).

---

[109] Do zasad licencjonowania OSS powrócę w rozdziale 2.

[110] Por. J. D. Campbell, T. Kreidl, D. Pace, *Why central IT must embrace open source*, [in:] *Open-source solution in education. Theory and practice*, ed. J. Burton Browning, Santa Rosa 2010, s. 9; M. Fink, *The business and economics of Linux and open source,* Upper Saddle River 2003, s. 43; H. J. Meeker, op. cit., s. 29.

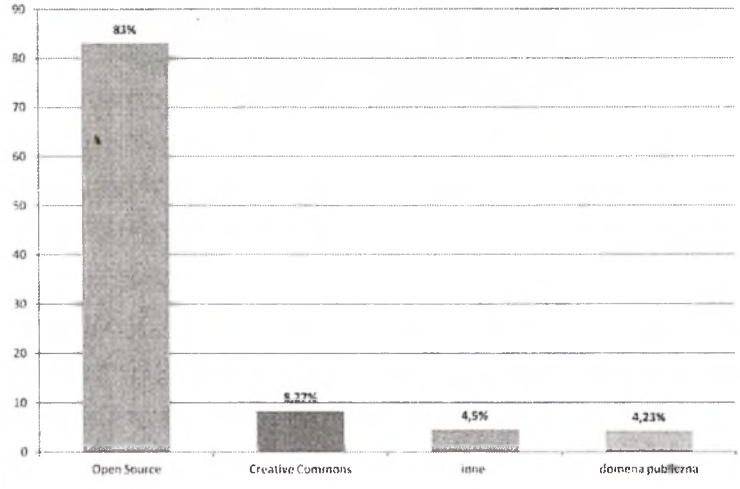
[111] Por. R. Dixon, op. cit., s. 43.

[112] Por. J. Locke, *Open source solution for small business problems*, Hingham 2004, s. 481.

[113] Dane ekscerpowano z serwisu Sourceforge w dniu 29 lipca 2013. Por. *Sourceforge* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://sourceforge.net/directory/?q=licenses.

W tym kontekście należy się słowo wyjaśnienia dotyczące CC. Popularność ruchu CC doprowadziła do tego, że wielu internautów zaczęło licencjonować licencjami CC programy komputerowe. Powodem mogło być to, że tzw. *sharelike* licencje CC zawierają się w grupie licencji copyleft[114]. Licencje CC nie są jednak stworzone do tego celu, toteż Creative Commons w celu licencjonowania oprogramowania zaleca stosowanie licencji stworzonych przez Fundację Wolnego Oprogramowania lub Open Source Initiative[115]. Pełna lista licencji dostępnych dla oprogramowania open source jest pokaźna. Zbiór licencji figuruje na stronach Fundacji Wolnego Oprogramowania (http://www.gnu.org/licenses/license-list.html) oraz na stronie organizacji Open Source Initiative (http://www.opensource.org/licenses/alphabetical)[116].

Mimo że licencje CC nie są przeznaczone dla programów komputerowych, to są w pewnym stopniu akceptowane przez społeczność open source. Stosują je wszyscy ci, którzy nie zgadzają się z zasadami narzucanymi drogą licencji GPL oraz BSD, wybierając ich zdaniem bardziej elastyczną alternatywę dla tych copyleftowych typów licencji. Niezależnie od tego, copyrightowe licencje CC nie powinny być stosowane wobec kodu źródłowego[117].



Wyk. 2. Udział statystyczny praktyk licencyjnych projektów zarejestrowanych w bazie Sourceforge.net [opracowanie własne][118].

[114] Por. H. J. Meeker, op. cit., s. 240.

[115] Por. *Creative Commons Polska* [online], [dostęp: 25.01.2012]. Dostępny w WWW: http://creativecommons.pl/faq/#15.

[116] Por. *Open Source Initiative uznają licencje GPLv2 oraz GPLv3i*; D. M. German, J. M. GonzálezBarahona, op. cit., s. 189.

[117] Dane w akapicie zaczerpnąłem z pracy: Por. H. J. Meeker, op. cit., s. 50.

[118] Dane ekscerpowano z serwisu Sourceforge w dniu 30 stycznia 2012. Por. *Sourceforge* [online], [dostęp: 29.07.2013]. Dostępny w WWW: http://sourceforge.net/directory/?q=licenses.

Wykresy (wyk. 1 i wyk. 2) demonstrujące dane uzyskane na przestrzeni półtora roku obrazują pewną tendencję. Na fali entuzjazmu dla inicjatyw open source powołuje się do życia coraz to nowe projekty. Jednakże wraz z tym nie idzie w parze znajomość przepisów prawa. Zdawałoby się, że w największej publicznej bazie tego typu programów powinny znajdować się same aplikacje open source[119]. Jak się okazuje, tak do końca nie jest. Aby bowiem zaliczyć program do grupy programów open source, musiałby on być licencjonowany na licencji open source (zgodnie z literą prawa w tym zakresie), a taką nie jest już choćby Creative Commons. Dowodem jest to, że w połowie 2013 roku tylko ok. 83% (wyk. 2) aplikacji zarejestrowanych w tej bazie można było uznać za OSS.

Podsumowując i nieco porządkując zebrane dotychczas dane oraz prezentowany wywód, posłużę się tabelą, w której w układzie chronologicznym przedstawię kalendarium ważniejszych wydarzeń związanych z ideą i ruchem open source (tab. 2).[120, 121]

Tabela 2

Rok	Wydarzenie	Pomysłodawca
1969	Powstanie pierwszej wersji UNIX-a	AT&T Bell Labs[120]
1979	Wydanie siódmej wersji UNIX-a, nazywanej dziadkiem wszystkich istniejących systemów uniksowych	AT&T Bell Labs[121]
1984	Wprowadzenie terminu *free software*	Richard M. Stallman
1985	Powołanie Free Software Foundation	Richard M. Stallman
1985	Wprowadzenie terminu *copyleft*	Richard M. Stallman
1989	Publikacja licencji General Public License v1	Richard M. Stallman
1989	Publikacja licencji BSD	Uniwersytet w Kalifornii
1991	Rozpoczęcie projektu Linux	Linus Torvalds

[119] Bazy oprogramowania open source omówiłem w rozdziale 4.

[120] Por. S. Coleman, *Open-source as an alternate to commercial software. Final report 583*, Tempe 2009, s. 8.

[121] Ibidem.

1996	Pomysł koncepcji open source	Eric S. Raymond
1994	Oficjalne wydanie jądra Linuxa V1.0	Linus Torvalds[122]
1997	Esej *Katedra i bazar*	Eric S. Raymond
1997	Wprowadzenie terminu *open source*	Eric S. Raymond, Tim O'Reilly, Bruce Perens
1997	Wprowadzenie definicji open source (OSD)	Bruce Perens[123]
1998	Powołanie organizacji Open Source Initiative	Eric S. Raymond, Bruce Perens
1998	Propozycja stosowania terminu *open source software* w miejsce terminu *free software*	Open Source Initiative
1998	Wprowadzenie terminu *free and open source software* (skrót FOSS lub F/OSS)	Usenet[124]
2001	Wprowadzenie terminu *free libre open source software* (pol. wolne i otwarte oprogramowanie, skrót FLOSS)[125]	Rishab Aiyer Ghosh[126]
2007	Publikacja licencji General Public License v3	Free Software Foundation

Kalendarium ważniejszych wydarzeń związanych z ideą i ruchem open source [opracowanie własne].[122, 123, 124, 125, 126]

[122] Ibidem.

[123] Por. P. Kavanagh, op. cit., s. 1.

[124] Wikipedia. The free encyclopedia [online], [dostęp: 27.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Alternative_terms_for_free_software.

[125] Dziś skróty OSS, FOSS, FLOSS, WiOO stosowane są przeważnie wymiennie.

[126] Por. Wikipedia. The free encyclopedia [online], [dostęp: 27.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Alternative_terms_for_free_software.

1.4. SPOŁECZNE PODSTAWY ROZWOJU RUCHU OPEN SOURCE

Dla niniejszych rozważań istotne zdaje się udzielenie odpowiedzi na pytanie dotyczące pojawienia się pomysłu otwartości w cyfrowej przestrzeni komunikacji w tym specyficznym – programistycznym – obszarze. Skąd wziął się zatem pomysł na współdzielenie się programami komputerowymi i ich kodem źródłowym? Koncepcja open source ma korzenie społeczne. Ludzie od zamierzchłych czasów, dla własnych korzyści wymieniali się rozmaitymi informacjami, dzielili wiedzą, umiejętnościami i własnym doświadczeniem, począwszy od gestów i komunikacji niewerbalnej, poprzez wyrażanie emocji aparatem mownym, na artykulacji dźwięków skończywszy. Żaden język naturalny nie mógłby powstać w izolacji. Język jest bowiem nie tylko narzędziem komunikacji, lecz przede wszystkim sposobem scalania poszczególnych grup społecznych (funkcja socjatywna). Innymi słowy: bez komunikacji nie powstałyby złożone ludzkie społeczności. U podstaw społeczeństw leży więc wymiana informacjami. To dzięki niej ludzie zapewniają sobie lepszą egzystencję. Wraz z rozpoczęciem wymiany informacji i treści zaczęło nieustannie przybywać wiedzy, co wymusiło konieczność porządkowania tych zasobów zgodnie z ówczesnymi możliwościami. Poza przekazami niewerbalnymi, u zarania komunikacji istotną rolę odgrywało słowo mówione, tj. jego ekspresja i artykulacja, bo przecież pierwsze społeczności niepiśmienne posługiwały się w codziennych kontaktach mową. Socjolodzy wskazują, że wtedy też ludzie wytworzyli określone metody, pozwalające zapamiętywać tylko wybraną część dostępnych informacji, co pozytywnie wpłynęło na zapoczątkowanie i zorganizowanie się pierwszych społeczeństw[127]. Wśród nich znalazły się rozmaite mnemotechniki ułatwiające zapamiętywanie, przechowywanie oraz przypominanie sobie wiadomości[128]. Zdobyte doświadczenie kultur oralnych pozostało jednak w większości ulotne. Wczesne kultury posługiwały się też określonymi kodami pisanymi, które choć nie były jeszcze pismem, to jednak dawały możliwość utrwalania określonych przekazów na różnych

[127] Socjolodzy wprowadzają określenie *oral technologies*, co można by przetłumaczyć jako 'ustne technologie'. Por. C. J. Couch, *Oral technologies. A cornerstone of ancient civilization?* „The Sociological Quarterly" 1989, vol. 30, no 4, s. 587.

[128] Por. M. E. Hobart, Z. S. Schiffman, *Information ages. Literacy, numeracy, and the computer revolution*, Baltimore 2000, s. 12-20.

nośnikach[129]. Materiałem, na którym utrwalano informacje, były m.in. kości i skóry zwierząt, kamienie, skały oraz gliniane tabliczki[130]. Próby interpretacji tych pierwszych przekazów niepisemnych wypadają niepomyślnie z tego powodu, że odczytywaniu tych kodów symbolicznych towarzyszyło zawsze słowo mówione, które wyjaśniało głębszy sens prezentowanych symboli[131]. Niepiśmienne społeczeństwa miały zatem „wysoko rozwinięte i dobrze przystosowane do potrzeb metody przekazu społecznego. Pod względem objętości i zasięgu pozostają one z zasady ograniczone wskutek ścisłego powiązania z pamięcią i słowem mówionym czy też objaśnieniem ustnym"[132]. Odejście od tych metod zaczęło się wraz z pojawieniem się kolejnej „technologii" – pisma fonetycznego[133], które, jak to wyraził Wilhelm von Humboldt, utrwala język (mowę)[134]. Pismo „przedłuża" zatem ludzką pamięć. Od tej chwili możliwe było gromadzenie, a co za tym idzie – oddziaływanie tych gromadzonych treści zarówno w czasie, jak i przestrzeni, gdyż „wraz z rozwojem pisma i kultury pisanej szybko upowszechnił się zwyczaj utrwalania tą metodą wiedzy, systematycznego jej gromadzenia i przechowywania w centrach kształcenia i wiedzy – świadczy o tym wiele archiwów i bibliotek, których powstanie na Bliskim Wschodzie już od III tysiąclecia p.n.e. wielokrotnie potwierdzono archeologicznie"[135]. Pismo pozwoliło utrwalać przekazy ustne, a także tworzyć nowe przekazy pisane[136]. Geneza pisma alfabetycznego ma polityczno-ekonomiczne podłoże.

---

[129] Przedpiśmienny człowiek wypracował również odpowiednie metody liczenia i gromadzenia informacji, np. wczesnym środkiem służącym rachowaniu przedmiotów obrotu handlowego, tj. artefaktów, żywności oraz zwierząt był karbowany kij, który spełniał funkcję dzisiejszych ksiąg inwentarzowych, podobną rolę odgrywały również sznurki z węzłami (kipu), małe kamyczki i muszelki. Por. M. Kuckenburg, *Pierwsze słowo. Narodziny mowy i pisma,* Warszawa 2006, s. 106-115. Metody te wykorzystywane były w handlu, pozwalały bowiem zapanować nad szacowaniem wszelkiego towaru.

[130] Por. M. Kuckenburg, op. cit., s. 114-129.

[131] Ibidem, s. 120-124.

[132] Ibidem, s. 133.

[133] Komunikolodzy oraz socjolodzy w stosunku do pisma używają niekiedy pojemnego określenia *technologia*. Por. J. D. Bolter, *Writing space. Computers, hypertext and the remediation of Print*, ed. 2., Mahwah 2009, s. 22-23; C. J. Couch, D. R. Maines, Shing-Ling Chen, *Information technologies and social orders*, New York 1996, s. 102.

[134] Por. W. von Humboldt, *O myśli i mowie. Wybór pism z teorii poznania, filozofii dziejów i filozofii języka*, Warszawa 2002, s. 327.

[135] M. Kuckenburg, op. cit., s. 174.

[136] Odkąd zaczęły powstawać teksty, odtąd też wymagały one uprzystępnienia. Pierwszym krokiem na tej drodze stało się wyjaśnianie określonych (trudnych) wyrazów. W ten też sposób rozpoczęła się historia leksykografii. Pierwsze słowniki po-

Rządcy starożytnych cywilizacji (Egipt, Sumer) potrzebowali lepszych form przechowywania informacji niż „technologie ustne", przedpisemne kody symboliczne, kody obrazowe oraz symbole piktograficzne (pismo ideograficzne)[137]. Chcieli bowiem integrować większe grupy ludzi w systemach politycznych oraz hierarchizować je w kasty i klasy, a także potrzebowali takiego środka komunikacji, który sprostałby zarządzaniu gospodarczemu coraz większymi państwami[138].

Przez tysiąclecia ręczne pisanie tekstów było jedyną metoda utrwalania wiedzy i informacji. W czasach nowożytnych gildie kopistów oraz skryptoria utrzymywały, zwłaszcza we Francji, swój monopol na wytwarzanie manuskryptów, co przekładało się na ich wysokie ceny. Nie dziwi więc, że to poza Francją pojawiały się próby przełamania hegemonii, których ukoronowaniem była ta przeprowadzona przez Jana Gutenberga w połowie XV wieku, który stworzył w Europie nową technologię komunikacyjną (ang. *communication technology*), tj. prasę drukarską[139]. Do końca XV

---

wstawały ponad 4 tys. lat temu i przybierały formę dwu lub trzyjęzycznych list słów, spisywano je przeważnie w tym celu, aby ułatwić wybranym grupom użytkowników zrozumienie określonych tekstów, m.in. religijnych. Por. H. Bergenholtz, S. Nielsen, S. Tarp, *Introduction*, [in:] *Lexicography at a crossroads. Dictionaries and encyclopedias today, lexicographical tools tomorrow*, ed. H. Bergenholtz, S. Nielsen, S. Tarp. Bern 2009, s. 8. Przyjmuje się, że pierwszym pismem fonetycznym było pismo klinowe używane na obszarze państwa sumeryjskiego. Por. M. Kuckenburg, op. cit., s. 172. Tam też powstały pierwsze listy słów. Najstarsze zachowane słowniki pochodzą z okresu około 2200 p.n.e. i są to dwujęzyczne sumeryjsko-akadyjskie listy słów. Por. R. G. Olson, *Technology and science in ancient civilization*, Santa Barbara 2010, s. 65. Leila Avrin podaje, że w zasadzie listy te, traktując m.in. o społeczeństwie, prawie, ubiorach, roślinach i zwierzętach, minerałach, żywności, stanowią świadectwo ówczesnego życia. Por. L. Avrin, *Scribes, script and books. The book arts from antiquity to the renaissance*, Chicago 1991, s. 73. Wprowadzenie pisma pozwoliło też rozwinąć metody opracowywania informacji, takie jak „zestawianie list lub sporządzanie statystyk". M. Kuckenburg, op. cit., s. 134.

[137] Por. R. Sassoon, A. Gaur, *Signs, symbols and icons. Pre-history to the computer age*, Exeter 1997, s. 31.

[138] Por. M. Kuckenburg, op. cit., s. 211-214. Jednym z dwóch najstarszych pism świata jest egipskie pismo hieroglificzne. Ibidem, s. 175-188. Najdawniejszy zachowany zabytek tego pisma utrwalony został na tzw. palecie Namera (tabliczka z szarogłazu), której wiek nauka określa na ok. 3000 r. p.n.e. Ibidem, s. 175-180. Jako materiał pisarski Egipcjanom służył jednak przede wszystkim papirus, który okazał się rewolucyjnym materiałem, pozwolił bowiem na łatwe i szybkie powielanie większych ilości tekstów.

[139] Por. E. L. Eisenstein, *The printing revolution in early modern Europe*, ed. 2., Cambridge 2005, s. 313.

wieku wszystkie większe centra Europy posiadały już własne drukarnie[140]. Wynalazek druku zapoczątkował ogromną produkcję wydawniczą w nieznanej dotychczas skali, co automatycznie wpłynęło na znaczący przyrost informacji[141].

W ciągu stu lat od wprowadzenia drukarni do krajobrazu miejskiego Europy Zachodniej zaczęto doświadczać nowego problemu, mianowicie: nadmiaru produkcji wydawniczej. Dowodem tego zalewu książką było opracowanie bibliograficzne Konrada Gesnera *Bibliotheka universalis*, opublikowane w Zurychu w 1545 r. W zamierzeniu Gesnera jego biblioteka uniwersalna miała zawierać informacje na temat wszystkich wydrukowanych książek (do czasów jemu współczesnych), stworzonych w językach łacińskim, greckim i hebrajskim. W dziele swoim przedstawił opis około 15000 dzieł (3000 tysięcy autorów). Praca jego zapewniła mu miano ojca bibliografii powszechnej. Przedsięwzięcie Gesnera było pierwszą i ostatnią próbą stworzenia spisu, który rejestrowałby całą produkcję wydawniczą w formie jednego opracowania. Przyrost informacji był już zbyt szybki, aby można było zapanować nad jego całością. Od tej pory nowe opracowania bibliograficzne zaczęły powstawać w takim tempie, że konieczne stało się sporządzanie bibliografii bibliografii[142].

Nowy sposób duplikacji tekstów wprowadzony w XV wieku w Europie wraz z wynalazkiem Gutenberga zwiastował bardzo gwałtowne zmiany[143]. W okresie średniowiecza książki były drogie, przez co dostęp do nich był znacznie utrudniony. Profesorowie i studenci nie zawsze mogli swobodnie korzystać ze zbiorów bibliotek. Poza tym biblioteki średniowiecznej Europy zlokalizowane były głównie przy uniwersytetach i katedrach, a zbiory ich rzadko kiedy liczyły więcej niż 1000 pozycji. Druk spowodował, że produkcja książek stała się łatwiejsza i tańsza, dostęp do nich – powszechniejszy, a produkcja wydawnicza od tego momentu stale się powiększała. Drukowane teksty umożliwiły zdobywanie wiedzy indywidualnie. Do tej pory za kształcenie odpowiadały uniwersytety oraz ci, którzy byli w posiadaniu odpowiednich rękopisów. O ile teksty pisane pozwalały autorom pokonać czas i przestrzeń, o tyle tek-

---

[140] Por. H. A. Innis, *Empire and communication*, Toronto 2007, s. 164-166.

[141] Por. E. L. Eisenstein, *The printing press as an agent of change*, Cambridge 1994, s. 30.

[142] Opracowując akapit, skorzystałem z pracy: E. L. Eisenstein, *On revolution and the printed word*, [in:] *Revolution in history*, ed. R. Portera, M. Teicha, Cambridge 1986, s. 188.

[143] Por. E. L. Eisentein, op. cit., s. 318.

sty drukowane pozwalały dużo szybciej oddziaływać zawartym w nim myślom na dużo większe grupy społeczne. Druk przyniósł więc zmiany nie tylko ilościowe, lecz także jakościowe[144]. W ciągu kolejnych wieków systematycznie rozwijano drukarstwo[145].

Kolejną wielką zmianę odnotowano wraz z rozpoczęciem rewolucji informatycznej[146], gdy do życia społecznego wprowadzono komputery, które szybko zaczęły być wykorzystywane w innych sferach niż naukowa i wojskowa. W zasadzie wypełniły niemalże wszystkie obszary ludzkiej aktywności, stając się urządzeniami uniwersalnymi. Doprowadziło to do powstania różnych technologii informacyjnych, m.in. technologii pozwalających na komunikację sieciową. Tę z kolei ułatwiła upowszechniona na początku lat 90. XX wieku nowa technologia, tj. World Wide Web. W3 jest wytworem technologii informacyjnych i jednocześnie stymuluje i przyspiesza ich rozwój. Automatycznie prowadzi to do przyrostu informacji i wiedzy. Na potrzeby środowiska cyfrowego powstała jednostka, jaką jest bit, co pozwoliło wszystkie przekazy cyfrowe uznawać za informacyjne[147]. Bit bowiem można uznać za porcję informacji. W środowisku cyfrowym, policzalnym poprzez bity, wszystko jest informacją. Niezależnie od tego w świecie cyfrowym dostrzeżono remedium na informacyjne bolączki nurtujące współczesnych, jak i starożytnych. Mam na myśli chęć uprzystępnienia całego, tj. dostępnego w danym momencie historycznym, korpusu informacji (wiedzy, treści)[148]. W przestrzeni cyfrowej oznacza to powstawanie nowych technologii informacyjnych, np. technologii dostępu do informacji. Wśród nich można wskazać m.in. tworzenie coraz poręczniejszych urządzeń przenośnych, zapewniających dostęp do korpusu informacji zgromadzonych w WWW.

---

[144] Informacje w akapicie uzyskałem z pracy: K. Górniak-Kocikowa, *Revolution and the library*, „Library Trends" 2001, vol. 49, no 3, s. 454-461.

[145] Por. E. L. Eisentein, op. cit., s. 56-70 i dalsze.

[146] Rolę wynalazku Gutenberga jako czynnika wprowadzającego liczne społeczne zmiany porównuje się niekiedy do technologii informacyjnych, które również znacznie zmieniły krajobraz społeczny. Por. C. J. Couch, D. R. Maines, Shing-Ling Chen, op. cit., s. 13-14. Por. także I. Fang, op. cit. Zmiany rozpoczęły się wraz z przejściem od zwoju do kodeksu, a doszły do etapu prezentacji tekstów na ekranach. Por. E. L. Eisentein, op. cit., s. 316.

[147] Z powodu utworzenia jednostki, jaką jest „bit", wszystkie przekazy zaczęto uznawać za informacyjne, co w przypadku literatury pięknej dostępnej w postaci cyfrowej, a więc w postaci bitów, będzie bezzasadne, ponieważ literatura piękna nie jest komunikatem informacyjnym. Por. J. Wojciechowski, op. cit., s. 93.

[148] Por. S. D. Kotuła, *Od Biblioteki Aleksandryjskiej do World Wide Web*, „Biblioteka" 2012, nr 16, s. 115-137.

„Technologie" ustne oraz „technologie" przedpisemne, choć symboliczne, wymagały stosowania metod mnemotechnicznych. Te drugie – także odpowiednich materiałów pisarskich. Wraz z kolejnymi „technologiami" komunikacyjnymi, tj. pismem, drukiem oraz cyfrowymi powstały nowe nośniki informacji, formy utrwalania informacji oraz biblioteki różnego typu. Natomiast praktyka bibliotekarska wpłynęła na powstanie i wpływa na powstawanie i rozwijanie metod radzenia sobie z informacjami utrwalanymi na różnych nośnikach[149]. A zatem praktyka społeczna ułatwiła, czy raczej wymusiła, tworzenie stosownych technologii informacyjnych – rozumianych jako narzędzia i metody zarządzania informacją – ułatwiających codzienną egzystencję. Tworzone technologie powodowały z kolei przyrost informacji i wiedzy, co wpływało wreszcie na konieczność wypracowywania metod gromadzenia, opracowywania i udostępniania informacji (czy ściślej utrwalonych kwantów wiedzy)[150]. Coraz większy wachlarz różnych nośników informacji (kodów, systemów znaków), stała dywersyfikacja nośników informacji oraz rozproszenie informacji utrudniały jej sprawne odnajdywanie. Na przestrzeni dziejów symultanicznie z tymi procesami pojawiały się zatem i ewoluowały różne koncepcje gromadzenia, opracowywania, a przede wszystkim udostępniania i informowania o źródłach informacji. Wiele z nich zakładało objęcie lub obejmowało swym zakresem dostępne w określonym zbiorze, a nieraz także i całe ówczesne lub przyszłe uniwersum informacji (treści)[151].

Społeczeństwo wytwarza informacje i wiedzę. Do tego celu opracowuje odpowiednie technologie, które służą przekazywaniu, udostępnianiu i przechowywaniu informacji i wiedzy. Technologie z kolei wpływają na to, jaką informację i wiedzę się przekazuje. Na tym etapie ujawnia się również proces innowacyjności. Społeczeństwo wytwarza nie tylko nowe porcje informacji

---

[149] Z tymi ustaleniami koresponduje stwierdzenie Marty Grabowskiej, że „utrwalanie, a potem gromadzenie, opracowanie i udostępnianie informacji od wieków stymulowane było postępem technologicznym. Wyznaczał on coraz to nowe kamienie milowe w działalności bibliotek". M. Grabowska, *Biblioteka cyfrowa w środowisku wirtualnym*, [w:] *Biblioteki cyfrowe. Projekty, realizacje, technologie*, red. J. Woźniak-Kasperek, J. Franke, Warszawa 2007, s. 21.

[150] Nieco sztucznie wyróżniłem fazy „ruchów" społecznych w zakresie wytwarzania technologii informacyjnych, przyrostu informacji i opracowywania metod radzenia sobie z jej przyrostem, choć zdaję sobie sprawę, że wszystko to powstawało niemalże równocześnie. Wszystkie te fazy są elementami tego samego zjawiska. Sztuczne cezury pozwalają jedynie lepiej uchwycić istotę przemian, jak również pozwalają łatwiej wyobrazić sobie odbywający się proces.

[151] Np. S. D. Kotuła, *Prace Paula Otleta a World Wide Web*, „Biblioteka" 2013, nr 17, s. 153-167.

i wiedzy, lecz także nowe technologie, które pozwalają tworzyć nowe porcje informacji i wiedzy. Wraz z wprowadzeniem do społecznego obiegu komputerów i Internetu, a więc *de facto* technologii informacyjnych, pojawiły się takie właśnie nowe technologie (narzędzia przekazywania, sposoby prezentowania i metody utrwalania informacji i wiedzy). Jednym z nich jest nurt open source, który z kodu źródłowego aplikacji nakazuje czynić komunikat. Kod (*nomen omen*) źródłowy jest specyficznym kodem semiotycznym (choć trzeba pamiętać, że składnia kodów źródłowych jest też różna, co zależy od zastosowanego języka programowania), co oznacza, że podlega pewnym prawom. Składa się ze zbioru elementów (znaków) oraz reguł (gramatyki), które pozwalają łączyć te elementy w większe całości (wypowiedzi – programy lub ich komponenty). Porównanie kodu źródłowego do systemów znaków, jakimi są języki naturalne, jest nieprzypadkowe. Język, aby się rozwijał, musi być używany. Nieużywany język zanika. Używanie języka służy nie tylko rozwojowi tego języka, lecz także skuteczniejszemu funkcjonowaniu w określonej rzeczywistości. Na przykład język angielski, który *notabene* jest najbogatszym pod względem zasobu (liczby) jednostek językowych językiem europejskim, stał się międzynarodowym językiem handlu oraz informatyki. Fakt, że używany jest na całym świecie, wpływa na jego stale rosnące zasoby, czyli przybywanie doń coraz to nowych jednostek leksykalnych. Z drugiej strony język ten poprzez specyficzne reguły gramatyczne oraz zasób wyrazowy kształtuje handlową przestrzeń (dyskursywną, fizyczną). Zaznajamiając się z językiem, poznaje się też tzw. obraz świata, który tkwi w danym języku. Za jego sprawą, osoba ucząca się tego języka, przyjmuje od razu ten obraz, a więc zwraca uwagę na to, na co kieruje ją język. Nieustannie dochodzi do warunkowania i wpływu języka na rzeczywistość i rzeczywistości na język. Podobnie jest z kodem źródłowym: tworzy się w nim narzędzia (programy), które kształtują przestrzeń społeczną, a ona z kolei wpływa na projekty narzędzi, które w kodzie się tworzy. Tym samym wpływa też na sam kod źródłowy. Nowe projekty wymagają nowych syntagm kodu, aby sprostać kolejnym wyzwaniom. W rezultacie kod się wzbogaca, powstają nowe narzędzia, które kształtują rzeczywistość, a ta znowu staje się źródłem inspiracji dla programistów, aby tworzyć nowe ciągi kodu.

W tym kontekście można też uznać, że kod źródłowy lub jego fragment jest memem, tj. pewną porcją informacji (treści)[152]. Poprawność, czyli bezbłędność oraz skuteczność fragmentu kodu wpływa na jego popularność.

[152] Por. Y. Tanaka, *Meme media and meme market architectures. Knowledge media for editing, distributing, and managing intellectual resources*, Piscataway 2003, s. 242-245.

Poprawny kod lub jego fragment będzie dalej wykorzystywany, a zatem będzie krążył w społecznym obiegu. Nawet jeżeli będzie modyfikowany, to jego wpływ pozostanie równie silny. W przypadku kodu źródłowego, trochę inaczej niż w przypadku języka naturalnego, mem musi być jednostką dobrze realizującą swoje zadania – jednostką funkcjonalną, a nie jednostką informacji (treści), która wcale nie musi być nawet poprawna. Kod źródłowy jako mem, aby się rozpowszechniał (był używany, wykorzystywany) musi wypełniać należycie zadania, do których został stworzony. Jeżeli mamy do czynienia z kodem zapewniającym bezbłędne funkcjonowanie dużej bazy danych, to faktycznie po zbudowaniu takiej bazy powinna ona działać bezbłędnie. Natomiast językowe memy upowszechniane są z innych powodów, np. marketingowych, politycznych, kulturowych itp.

Jak pokazuje powyżej przedstawiony krótki rys historyczny, izolacja informacji i wiedzy ma na celu przede wszystkim ograniczanie użytkowników w dostępie do nich, a pośrednio także zapewnia kontrolę tym, którzy do tych informacji i wiedzy mają dostęp. Gdyby pismo pozostało domeną władzy, gdyby nie upowszechniono wynalazku druku i gdyby komputery i sieć internetowa pozostały pod kontrolą wojska i ośrodków naukowych, do dziś zapewne społeczeństwa byłyby zorganizowane w innych systemach społecznych niż te, w których żyjemy. Swoboda w wymianie informacją i wiedzą, także tą wyrażaną, w tym przypadku, kodem źródłowym oprogramowania, sprzyjać może nie tylko otwartości w komunikacji dotyczącej sfery programów komputerowych, lecz także równemu do niej dostępowi dla wszystkich i rozwojowi nie tylko tego sektora. Stwarzanie barier, zamykanie się, odgraniczanie jest procesem hamującym, który nie sprzyja konstruowaniu rzeczywistości, tylko zachowywaniu *status quo*.

Koncepcja open source wyrosła z praktyki społecznej[153]. Dzielenie się informacjami oraz wspólne tworzenie określonych artefaktów zapewniało i zapewnia bowiem systematyczny rozwój. Fundament open source zasadza się na swobodnej wymianie informacjami, a „informacja chce być wolna" (ang. *information wants to be free*)[154]. Sam termin *open source* zastosowany

[153] Projekty open source są *de facto* rodzajem sieciowych społeczności. Grupa osób pracuje przeważnie *via* Internet nad daną aplikacją open source i przez to tworzy sieciową grupę społeczną. Por. S. Dasgupta, *Encyclopedia of virtual communities and technologies*, Hershey 2006, s. 287.

[154] Jest to hasło autorstwa Stewarta Branda, który pod koniec lat 60. XX wieku stworzył *Whole Earth Catalog. Wikipedia. The free encyclopedia* [online], [dostęp: 18.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Information_wants_to_be_free. Był to katalog publikowany nieregularnie od 1968 roku do 1998 roku. W katalogu przedstawiano

pierwotnie został na potrzeby oprogramowania komputerowego. Jednakże popularność aplikacji open source w połączeniu z rozwijającą się siecią Internet (zwłaszcza wraz z usługą World Wide Web), która stymuluje permanentny rozwój nie tylko obszaru open source (OSS powstaje głównie za pośrednictwem Internetu i za pośrednictwem Internetu jest też dystrybuowane i rozpowszechniane), lecz także samego dyskursu poświęconego zagadnieniu open source, doprowadziła do tego, że zjawisko open source zaczęto rozpatrywać szerzej[155]. Można bowiem mówić o tzw. koncepcji (idei) otwartości (ang. *open concept*). Pamiętać jednak należy, że to właśnie ideolodzy związani z ruchem open source przyczynili się do tego, że informatyczny (technologiczny) *de facto* problem stał się szeroko dyskutowanym zagadnieniem społecznym w ogóle. Pojawiły się więc pomysły, aby ideę open source przenieść na inne obszary, na przykład na naukę, co zaowocowało wprowadzeniem terminu *open source science*. Można bowiem wskazać pewne paralele pomiędzy OSS i działalnością naukową, wśród nich m.in. swobodę wymiany informacjami i wiedzą, publiczne udostępnianie najnowszych osiągnięć, wyników badań, prac i idei, dostępność tych treści wraz z możliwością ich dalszego rozwijania, poprawiania itp., działania w kooperacji, tworzenie społeczności i dalsze propagowanie wspólnotowych idei oraz rozpowszechnianie informacji i wiedzy. Open source staje się więc publiczną, społeczną, a nawet polityczną sprawą[156]. Choć, jak zauważa Yochai Benkler,

---

różne towary, które wówczas można było kupić wraz z cenami. Natomiast nazwa katalogu wzięła się z pomysłu Branda, który w 1966 roku zainicjował akcję, aby NASA udostępniła posiadane przez siebie zdjęcia Ziemi zrobione z kosmosu. W ten sposób po raz pierwszy opublikowano zdjęcia przedstawiające całą Ziemię (ang. *whole Earth*). *Wikipedia. The free encyclopedia* [online], [dostęp: 18.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Whole_Earth_Catalog. Samo hasło natomiast wprowadził do publicznego dyskursu na Pierwszej Konferencji Hakerów, która odbyła się w 1984 roku. Por. *The third culture. S. Brand, K. Kelly, G. Dyson. A Edge conversation in Munich. Introduction. Adrian Kreye; moderator John Brockman,* „Edge" 2011, no 338 [online], [dostęp: 18.01.2012]. Dostępny w WWW: http://www.edge.org/documents/archive/edge338.html.

[155] Internet rozwinął się m.in. dzięki oprogramowaniu open source (np. protokoły interentowe). Por. M. Castells, op. cit., s. 15-16.

[156] Por. M. van den Boomen, M. T. Schäfer, *Will the revolution be open-sourced? How open source travels through society,* [in:] *How open is the future? Economic, social & cultural scenarios inspired by free & open-source software*, ed. M. Wynants, J. Cornelis, Brussels 2005, s. 31-38. Na jednym z wystąpień Waldemar Pawlak (ówczesny wicepremier i Minister Gospodarki RP) mówił o roli oprogramowania open source w innowacyjnej gospodarce. Jego bowiem zdaniem największym sukcesem XX wieku było właśnie wolne i otwarte oprogramowanie. Por. R. Michalski, *„Open Source" oczami Waldemara Pawlaka* [online], [dostęp: 28.01.2012]. Dostępny w WWW: http://pppit.org.pl/?a=142.

to właśnie depolityzacja open source, zwłaszcza w początkowym okresie, była kluczowym wyznacznikiem ułatwiającym nieskrępowany jego rozwój, później natomiast ruch OS został znacznie upolityczniony[157].

W kontekście aktualnej sytuacji polityczno-gospodarczo-ekonomicznej krajów europejskich rozwiązania open source zdają się rozwiązaniem niektórych problemów. Większość aplikacji OSS pozostaje wolna od opłat (w przeciwieństwie do zawyżonych kosztów oprogramowania własnościowego), co w tym sektorze, jak pokazują badania z wykorzystania wolnego i otwartego oprogramowania w rządowej administracji publicznej, przeprowadzone dla Fundacji Wolnego i Otwartego Oprogramowania, jest jednym z najważniejszych czynników (razem z funkcjonalnością) branych pod uwagę przy decydowaniu się na dane oprogramowanie, mimo że sam dostęp do kodu źródłowego nie przekłada się na zwiększone zainteresowanie modyfikowaniem aplikacji stosowanych w rządowej administracji publicznej[158]. Zdaje się więc, że w przypadku zjawiska open source doszło do swoistej egzaptacji samej koncepcji, która z obszaru informatyki przeniesiona została na relacje społeczne m.in. w zakresie prawa, ekonomii, kultury, nauki czy polityki. Sygnalizowane kwestie omówię w kolejnym podrozdziale.

1.5. IMPLEMENTACJA IDEI OPEN SOURCE

Rozwój open source można zilustrować poprzez rozmaite określenia, coraz częściej pojawiające się w publicznym dyskursie i denotujące zjawiska wyrosłe w duchu idei open source. Moim celem nie jest jednakże ukazanie pełnego wpływu OS na różne sfery życia, a jedynie sygnalizacja jego obecności w odległych od siebie obszarach. Wśród nich należy wymienić: *open hardware, open access, open archives, open publishing, open culture*[159], a także *open content*. Duch ruchu open source ujawnia się także w takich sferach, jak tzw. koncepcja inteligencji open source (OSINT), która zakłada gromadzenie informacji

[157] Por. Y. Benkler, *The wealth of networks. How social production transforms markets and freedom*, New Haven, London 2006, s. 66.

[158] Por. *Wykorzystanie wolnego i otwartego oprogramowania w rządowej administracji publicznej. Raport z badania ilościowego dla Fundacji Wolnego i Otwartego Oprogramowania, marzec 2010* [online], [dostęp: 28.01.2012]. Dostępny w WWW: http://pppit.org.pl/publikacje/badanie_pentor.pdf.

[159] Por. M. van den Boomen, M. T. Schäfer, op. cit., s. 41.

z otwartych i publicznych źródeł (blogi, wiki, World Wide Web itd.), analizowanie oraz opracowywanie ich w celu tworzenia przydatnej wiedzy. Innym obszarem jest open source journalism (citizen journalism, pol. *dziennikarstwo obywatelskie, społeczne*). Inne jeszcze obszary to, poza wymienionymi, edukacja (np. open source curriculum) oraz produkcja filmowa (open source filmmaking)[160]. Poniżej pokrótce przybliżę ewokowane zjawiska, w kolejności, w jakiej zostały wymienione i od razu zaznaczam, że będzie to raczej zasygnalizowanie tematu, niż dogłębne jego przeanalizowanie. Każde z wymienionych tu zagadnień może być bowiem przedmiotem odrębnej pracy.

Integralną częścią ruchu open source jest nurt **open hardware** (open source hardware, skrót OSH). W tym przypadku to dane urządzenie, a nie oprogramowanie, wytwarzane jest na zasadach open source. Pomysł leżący u podstaw OSH polega na tym, że informacje o danym urządzeniu są wolne i otwarte dla wszystkich. Wśród informacji, które powinny być dostępne, wymienia się: projekt (schemat) urządzenia, listę materiałów wykorzystanych do jego budowy, plan tzw. obwodów drukowanych (PCB, ang. *printed circuit board*), a także FOSS zapewniające prawidłową pracę urządzenia. Aby uznać dane urządzenie za OSH, musi ono spełniać pewne kryteria. Przede wszystkim informacje o urządzeniu powinny być ogólnie dostępne. Jeżeli ktoś określa urządzenie jako OSH, to automatycznie oznacza to, że pełna dokumentacja tego urządzenia jest lub będzie bezpłatnie dostępna wszystkim zainteresowanym. Po drugie, projekt urządzenia powinien być również ogólnie dostępny. Wzorem oprogramowania open source powinien być również zapewniony dostęp do informacji dotyczących projektu urządzenia open source po to, aby inni projektanci mogli na tej podstawie poznać budowę urządzenia oraz, co równie istotne, aby mogli dokonywać stosownych modyfikacji w kolejnych wersjach urządzenia. W praktyce ten fakt wzbudza najwięcej kontrowersji, producenci wolą bowiem chronić swoje projekty patentami, niż się nimi dzielić. Wreszcie zalecane jest również, aby udostępniać na licencjach open source oprogramowanie, które służyło do projektowania danego urządzenia OSH, co ma zapewnić możliwość szybszego i skuteczniejszego dokonywania udoskonaleń w tym urządzeniu. Najłatwiej jest bowiem poprawić projekt urządzenia, mając do dyspozycji narzędzie, na którym owo urządzenie zostało zaprojektowane[161].

---

[160] Por. S. Nambisan, M. Sawhney, *The global brain. Your roadmap for innovating faster and smarter in a networked world*, Upper Saddle River 2008, s. 33.

[161] Opracowując akapit skorzystałem z książki: M. Bahareth, *Isay. Kings of the Internet*, 2010, s. 77-79.

Jako przykład dobrej praktyki można wskazać projekt MIT i rządu Indii, którzy razem powołali organizację non profit Media Laboratory Asia (2001 rok). Jednym z zadań projektu było dostarczenie tanich komputerów zbudowanych na bazie komponentów OSH. W rezultacie wspólnych działań znacznie obniżono koszty produkcji urządzeń komputerowych, przez co stały się one szeroko dostępne tamtejszej lokalnej społeczności[162].

Przeważnie podejmuje się jednak inicjatywy na dużo mniejszą skalę niż państwowa. Są to przedsięwzięcia prywatnych osób, których wspólnym celem staje się dostarczenie na rynek rozwiązań alternatywnych dla tych czysto komercyjnych. Działania tego rodzaju realizowane są przez zespół pracujący nad projektem RepRap, czyli drukarki 3D zdolnej do samoreplikacji. Zaletą urządzenia jest nie tylko możliwość drukowania obiektów trójwymiarowych, lecz także możliwość wydrukowania kopii samej siebie. Dzięki temu posiadacz drukarki mógłby ją wydrukować i przekazać kolejnemu użytkownikowi[163]. Jest to wyjątkowo nowatorskie podejście do zagadnienia otwartej i swobodnej wymiany informacjami. Podobną próbę podjęło The Open Graphics Projects (http://wiki.opengraphics.org/). W tym przypadku skupiono się konkretnie na rozwijaniu projektów urządzeń kart graficznych oraz sterowników do tych kart. Co zrozumiałe, wszystkie tego typu projekty opatruje się wolnymi licencjami[164]. Ostatni przypadek jest także świadectwem zawiązywania się w sieci swoistych społeczności osób (w tym także anonimowych) skupionych wokół wspólnych celów. W sieci doby postdwuzerowej wiele więcej jest miejsca i sposobności na tworzenie rozmaitych społeczności.

Innym przykładem będzie Open Cores (http://opencores.org/). Stanowi bowiem *exemplum* globalnej współpracy rzesz internautów nad rozmaitymi projektami komponentów komputerowych tworzonych w duchu OSH. Wśród dostępnych prac można znaleźć plany m.in. płyt głównych, procesorów, pamięci RAM itp. Poza zrealizowanymi projektami Open Cores zapewnia też platformę do komunikacji pomiędzy zainteresowanymi taką działalnością użytkownikami. Dostęp do materiałów jest oczywiście darmowy, a wszystkie one licencjonowane są przeważnie na licencjach GPL,

---

[162] Por. D. K. Ghosh, *Governance for development. Issues and strategies*, [in:] *Good governance. Initiatives in India*, ed. E. Vayunandan, D. Mathew, New Delhi 2003, s. 37.

[163] *Wikipedia. Wolna encyklopedia* [online], [dostęp: 24.11.2013]. Dostępny w WWW: http://pl.wikipedia.org/wiki/RepRap.

[164] Por. *The Open Graphics Projects* [online], [dostęp: 18.11.2012]. Dostępny w WWW: http://wiki.opengraphics.org/tiki-index.php.

LGPL, BSD[165]. Na rynku dostępne są już tanie i małe gotowe komputery (np. BeagleBone), które są w zasadzie płytami głównymi wraz ze wszystkimi niezbędnymi komponentami, tj. procesorem, pamięcią, interfejsami HDMI, USB, Ethernet 10/100 itd.[166].

Wśród innych prac warto jeszcze wskazać projekt Dextrus (The Open Hand), czyli robotyczną protezę ręki open source, przy tworzeniu której używa się m.in. drukarek 3D i komponentów, które zapewniają kilkudziesięciukrotne (nawet stukrotne) obniżenie kosztów w stosunku do protez dostępnych na rynku[167] czy też aparatu fotograficznego w znacznej części zbudowanego z komponentów wydrukowanych na drukarce 3D (http://www.instructables.com/id/3D-Printed-Camera-OpenReflex/?ALLSTEPS).

Kolejnym znaczącym projektem jest WikiHouse (Open Source Construction Set), czyli projekt domów mieszkalnych, które mogą budować ludzie bez specjalistycznej wiedzy (http://www.wikihouse.cc).

Jednym z najbardziej udanych projektów OSH jest Arduino[168] (kontroler, platforma dla systemów wbudowanych, a więc przeznaczona do konkretnych zadań). Nie wdając się w technologiczne szczegóły, wystarczy stwierdzić, że Arduino (fot. 6) można wykorzystać na nieskończenie wiele sposobów[169]. Wśród urządzeń stworzonych przy wykorzystaniu Arduino można wskazać m.in. projekty Lightbrush, służący do tworzenia w przestrzeni trójwymiarowej obrazów świetlnych, The Turn Machine, służący do obracania przedmiotów, co uzyskuje zastosowanie w fotografii, np. pomaga dokładnie sfotografować obiekt, Burglar Alarm/Fire Alarm – alarm przeciwwłamaniowy i przeciwpożarowy, Parking Assistant pomagający w precyzyjnym parkowaniu samochodu, Mouse Glove, czyli urządzenie, *de facto* rękawica z elektronicznymi elementami, która pozwala wykonywać ruchy kursora, podobnie jak czyni to komputerowa myszka, a także zapewnia funkcjonalność przycisków akcji myszki oraz klawiatury kompu-

---

[165] Por. *Open Cores* [online], [dostęp: 18.11.2012]. Dostępny w WWW: http://opencores.org/.

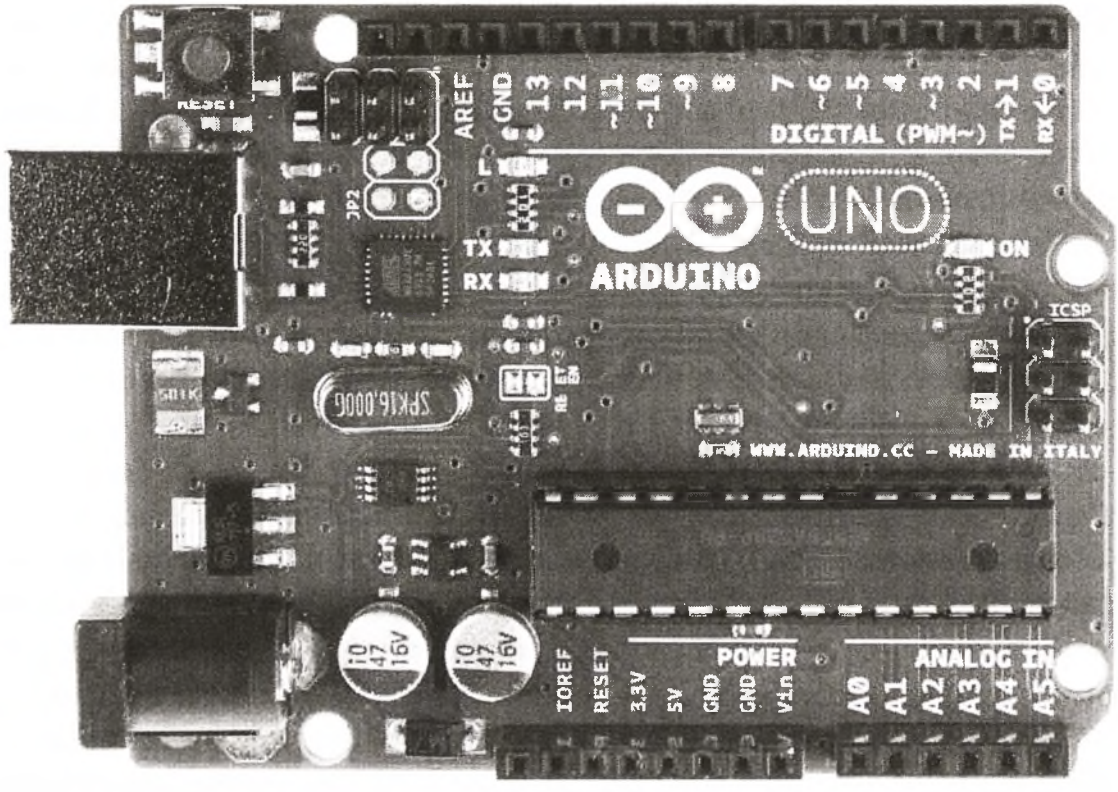
[166] Por. *Beagle board* [online], [dostęp: 27.03.2014]. Dostępny w WWW: http://beagleboard.org/.

[167] Por. *Open hand project* [online], [dostęp: 27.03.2014]. Dostępny w WWW: www.openhandproject.org.

[168] Por. J. Griffey, *Gadgets and gizmos. Personal electronics and the library*, Chicago 2010, s. 32.

[169] Szczegółowy opis rozpoczęcia pracy z Arduino przedstawił Ł. Więcek, *Od czego zacząć zabawę z Arduino?* [online], [dostęp: 18.11.2012]. Dostępny w WWW: http://majsterkowo.pl/od-czego-zaczac-zabawe-z-arduino/.

terowej, Servoelectric Guitar, czyli gitara elektroniczna, Virtual Electronic Drum, czyli elektroniczna perkusja[170]. Wykorzystując Arduino Waldemar Węgrzyn stworzył projekt Elektrobiblioteka, tj. projekt łączący w sobie zalety drukowanej publikacji książkowej oraz możliwości środowiska komputerowego, a więc coś, co na gruncie bibliologii i informatologii określa się mianem *książki konwergencyjnej*[171]. Warty wzmianki jest też projekt nanosatelity (ArduSat) zbudowanego w oparciu o m.in. Arduino (http://en.wikipedia.org/wiki/ArduSat).



Fot. 6. Mikrokontroler Arduino Uno[172].

[170] Pokaźna lista urządzeń zbudowanych w oparciu o Arduino znaduje się na stronie projektu. Por. *Arduino playground* [online], [dostęp: 31.01.2012]. Dostępny w WWW: http://www.arduino.cc/playground/Projects/ArduinoUsers.

[171] Z Elektrobiblioteką można zapoznać się na stronie internetowej. *Elektrobiblioteka* [online], [dostęp: 31.07.2013]. Dostępny w WWW: http://www.elektrobiblioteka.net/. Natomiast prezentacja realizacji całego przedsięwzięcia została udokumentowana pod innym adresem: *Elektrobiblioteka / Electrolibrary* [online], [dostęp: 31.07.2013]. Dostępny w WWW: http://vimeo.com/47656204.

[172] Źródło fotografii: *Arduino* [online], [dostęp: 18.11.2012]. Dostępny w WWW: http://arduino.cc/en/uploads/Main/ArduinoUno_R3_Front.jpg.

Przy pomocy kontrolera Arduino można tworzyć rozmaite urządzenia (zabawki, alarmy, roboty, narzędzia itp. itd.) i w zasadzie każdy – inżynier, projektant, artysta, entuzjasta komputerów i amator elektroniki – może dzięki Arduino zrealizować najróżniejsze własne przedsięwzięcia. Co najważniejsze – nie potrzeba do tego specjalistycznej wiedzy z zakresu programowania. Nie jest wymagana również wiedza z zakresu tworzenia obwodów elektronicznych. Można bowiem kupić gotowy kontroler (na rynku funkcjonuje kilka różnych modeli). W Internecie z kolei, czy nawet w źródłach drukowanych (tych cały czas przybywa), każdy bez trudu znajdzie rozmaite projekty wykorzystujące Arduino. Można również dotrzeć do podręczników wprowadzających w środowisko pracy z tym urządzeniem[173]. Z jednej więc strony można, posiłkując się zrealizowanymi urządzeniami, stworzyć dla treningu własne. Z drugiej zaś można wykorzystać zdobytą w ten sposób wiedzę po to, aby nie duplikować pracy, lecz skupić się na rozwiązaniach nowatorskich. Gotowe projekty mogą służyć jako inspiracja do własnych eksperymentów.

Bardzo ciekawe są też działania z pola *open source ecology* zainicjowane w USA przez urodzonego w Polsce doktora fizyki, wynalazcę i farmera Marcina Jakubowskiego. Podejmowane działania skupiają się na projektowaniu maszyn, urządzeń, które potrzebne są do codziennej ludzkiej egzystencji, np. traktorów, pieców do pieczenia chleba, maszyn do produkcji cegieł itp. Następnie projekty te udostępniane są za darmo wszystkim zainteresowanym. Ideą przyświecającą inicjatywie jest umożliwienie każdemu zbudowania niezbędnych urządzeń dużo taniej niż tych, które dostępne są w komercyjnej sprzedaży. Projekty mają być na tyle proste, aby każdy z zestawem podstawowych narzędzi mógł je wykonać. W tej chwili tworzony jest też projekt mikrodomu mieszkalnego (MicroHouse)[174].

Zagadnienie **open access** (pol. *otwarty dostęp*, skrót OA) jest dobrze rozpoznawane i stosunkowo mocno rozpropagowane w środowisku akademickim oraz – szczególnie – w środowisku bibliologiczno-informa-

---

[173] Por. K. Karvinen, T. Karvinen, *Make Arduino bots and gadgets. Learning by discovery*, Beijing, Sebastopol 2011; M. Margolis, *Arduino cookbook*, Beijing 2011; J. Oxer, H. Blemings, *Practical Arduino. Cool projects for open source hardware*, Berkeley 2009; M. Wierzbicki, *Arduino – Twój mały interaktywny przyjaciel* [online], [dostęp: 19.11.2012]. Dostępny w WWW: http://www.medialarts.pl/download/kadra/skrypty/interakcje/arduino.pdf.

[174] Por. M. Jakubowski, *Open-source blueprints for civilization* [online], [dostęp: 26.03.2014]. Dostępny w WWW: www.ted.com/talks/marcin_jakubowski; *Open source ecology* [online], [dostęp: 26.03.2014]. Dostępny w WWW: opensourceecology.org.

tologicznym i bibliotekarskim. Nie będę więc zagłębiał się w szczegóły, przypomnę jedynie, że określenie *open access* oznacza piśmiennictwo (naukowe, edukacyjne) w postaci cyfrowej, dostępne online, wolne od opłat (użytkownik końcowy nie ponosi dodatkowych kosztów związanych z korzystaniem z materiałów OA) oraz wolne od restrykcyjnych obciążeń prawnych (licencje z mechanizmem copyright).

OA jest nurtem działającym w zgodzie z literą prawa (prawa autorskie są zachowywane). Kwestie prawne rozwiązywane są drogą specjalnych umów. Podstawą prawną funkcjonowania OA jest uzyskanie zgody od właściciela praw autorskich na dostęp do określonych materiałów (dotyczy nowszego piśmiennictwa) lub wygaśnięcie praw autorskich (dotyczy starszego piśmiennictwa). Publikowanie artykułów w czasopismach punktowanych jest warunkiem *sine qua non* rozwoju naukowego i kariery naukowców i badaczy. Toteż nurt OA skupia się właśnie na tym, aby prace te, za które naukowcy i tak często nie otrzymują (bezpośrednio) wynagrodzenia, znalazły się w wolnym, powszechnym, trwałym i natychmiastowym dla każdego dostępie. Dodatkowym argumentem przemawiającym za tymi działaniami jest to, że z reguły badania naukowe finansowane są z budżetu państwa, a więc z pieniędzy podatników. Zrozumiałym jest, że wyniki badań prowadzonych z tych środków powinny być ogólnie dostępne. OA jest rodzajem modelu dostępu do danych, a nie modelem biznesowym. Założeniem OA jest dostarczenie większej publiczności wartościowych materiałów naukowych i edukacyjnych. Beneficjentami *in spe* są zatem wszyscy. Im więcej treści będzie dostępnych bezpłatnie, tym bardziej rozwiną się rozmaite gałęzie wiedzy i nauki. To konstruktywne założenie jest filarem całego ruchu open access. Zamierzenia OA realizuje się na dwa główne sposoby. Po pierwsze, poprzez udostępnianie całych numerów czasopism, gdzie artykuły są recenzowane. Po drugie, poprzez udostępnianie repozytoriów, w których można umieszczać oraz publikować swoje prace. W tym przypadku prace te nie muszą być recenzowane. Można przyjąć, że celem ostatecznym ruchu OA jest dojście do modelu Universal Access (pol. *dostęp uniwersalny*), a więc dostęp do wartościowych treści dla wszystkich. Na drodze ku temu stoją liczne bariery informacyjne: prawna (cenzura), językowa (nieznajomość języków, głównie angielskiego), ekonomiczna (brak środków zapewniających dostęp do Internetu), społeczna (wykluczenie cyfrowe)[175].

[175] Akpit powstał w oparciu o pracę: P. Suber, *Open access overview* [online], [dostęp: 5.02.2012]. Dostępny w WWW: http://www.earlham.edu/~peters/fos/overview.htm.

Choć historia OA jest długa, to u zarania tego nurtu znajdują się takie kamienie milowe, jak stworzenie sieci ARPANET czy Projekt Gutenberg[176]. Należy pamiętać, że zasada dzielenia się informacjami jest dużo starsza od ruchu OS. W środowisku naukowym znana jest od dawna, a instytucjonalizować zaczęła się już w XVII wieku, kiedy powstały pierwsze czasopisma naukowe[177]. W wielu wypadkach, dziś, jak i wcześniej wydawane tzw. czasopisma punktowane są własnością prywatnych wydawców, których jednym z zadań jest wypracowywanie zysków finansowych ze sprzedaży czasopism i dostępu do ich zawartości. To dlatego, w obliczu tego procederu i na fali ruchu open source, zainicjowano działania OA skupione na uwolnieniu rzetelnej i sprawdzonej wiedzy naukowej. Wyraźnie widać w tym wpływ idei ruchu open. W tym jednak wypadku dzielenie się oprogramowaniem (kodem źródłowym, aplikacją) zostało przekształcone w dzielenie się informacjami, ściślej – w swobodny dostęp do informacji naukowej[178].

Z tymi zagadnieniami związane są działania Open Archive Initiative. Celem tego ruchu jest promowanie i zachęcanie do działań zmierzających do budowy otwartych repozytoriów (***open archives***) przy instytucjach naukowych.

Kolejna kwestia to **open publishing**. Open online publishing zdaje się formą bardzo obiecującą, zwłaszcza dla nauki, gdyż zapewnia niskie koszty publikacji, lepszy dostęp do zawartości, lepsze możliwości przeszukiwania zasobów, możliwość łatwego kopiowania, a przez to i zabezpieczania danych (przez powielenie), bezpośredni dostęp do treści naukowych dla wszystkich zainteresowanych. Choć nauka w większości finansowana jest z budżetu państwa, to prace naukowe często wydawane są w prywatnych wydawnictwach, co skutkuje ograniczeniem dostępu do ich zawartości. Przeszkodą dla swobodnego udostępniania tekstów jest restrykcyjne prawo autorskie, które staje się orężem walki prywatnych przedsiębiorców o własne zyski. Dużo więc zależy od naukowców, którzy mogliby proces wydawniczy realizować za pomocą organizacji non profit. Dzięki temu znacznie obniżono by koszty dostępu do materiałów naukowych *via* cza-

[176] Por. P. Suber, *Timeline of the open access movement* [online], [dostęp: 5.02.2012]. Dostępny w WWW: http://www.earlham.edu/~peters/fos/timeline.htm.

[177] Por. *Wikipedia. The free encyclopedia* [online], [dostęp: 19.11.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Open_source.

[178] Chcąc poznać cały nurt OA, można skorzystać z darmowego kursu o open access, który *per se* opracowany został w duchu otwartości. Por. *Open access – otwarta nauka* [online], [dostęp: 6.02.2012]. Dostępny w WWW: http://otwartanauka.cel.agh.edu.pl/course/view.php?id=2.

sopisma naukowe, za dostęp do których w rezultacie płacić będzie musiała nawet macierzysta biblioteka[179].

W kontekście OA i open publishing proponuje się następujące rozwiązanie open source: koszty tworzenia projektu bazy udostępniającej za darmo wartościowe teksty można zredukować wykorzystując open source journal management software, np. Open Journal Systems (skrót OJS) z projektu Public Knowledge Project[180]. OJS jest wiodącą technologią na tym polu. Na stronie Projektu podawane są dane z grudnia 2011 roku, zgodnie z którymi na świecie wykonano ponad 11,5 tysiąca instalacji, co oznacza, że ponad 11,5 tysiąca czasopism na świecie korzystało z tego oprogramowania[181].

Zauważalne jest, że często słowo *open* (pol. *otwarty*) kojarzone jest automatycznie z ruchem wolnego i otwartego oprogramowania, a zatem każdorazowy akt tworzenia przez wspólnotę internautów jakichkolwiek programów, treści, informacji itp. łączony jest z ruchem open source[182]. Stąd zjawisko Wikipedii, którą charakteryzuje określenie ***open content*** (pol. *otwarta treść*, *wolna dokumentacja*), w potocznym dyskursie wpisuje się w ruch open source. Choć faktycznie realizuje ona pewne jego ideały, to te dwa zjawiska są zgoła różne. Warto omówić je jednak razem. Przypomnę jeszcze, że open source z otwartości kodu źródłowego rozciągnął się także na otwartość dokumentacji towarzyszącej programom o otwartym kodzie[183].

Tym samym idee open source przeniosły się na obszar szeroko pojętej kultury. Podstawą do tego stał się fakt, że open source – poza dostępem do kodu źródłowego – zaczął w pewnym momencie oznaczać dostęp do dokumentacji towarzyszącej oprogramowaniu open source. W tym kontekście będzie chodziło o dostęp do *de facto* bardziej tradycyjnych wytworów kultury, tj. tekstów, instrukcji, przepisów, opisów, projektów itp.[184].

---

[179] Akapit na podstwaie pracy: M. Wynants, J. Cornelis, *How open is the future? Economic, social & cultural scenarios inspired by free & open-source software*, Bruksela 2005, s. 128.

[180] Por. P. Suber, *Open access*, Cambridge 2012, s. 143.

[181] Por. *A sample of journals using Open Journal Systems* [online], [dostęp: 19.11.2012]. Dostępny w WWW: http://pkp.sfu.ca/ojs-journals.

[182] Por. J. M. Reagle Jr., *Good faith collaboration. The culture of Wikipedia*, Cambridge 2010, s.

[183] Por. Ibidem, s. 76.

[184] Por. A. Harrer, S. Zeini, S. Ziebarth, *Visualisation of the dynamic for longitudinal analysis of computer-mediated social networks-concept and exemplary cases*, [in:] *From sociology to computing in social networks. Theory, foundations and applications*, ed. N. Memon, R. Alhajj, Vienna, New York 2010, s. 128.

W tym czasie (1999 rok) Fundacja Wolnego Oprogramowania wprowadziła jeszcze jedną licencję, tj. GNU FDL (GNU Free Documentation License, skrót GFDL). W kolejnym milenium ten nowy model otwartości przeniósł się z obszaru dokumentacji technologicznej na obszar wszelkiej twórczości kulturowej czy – ściślej – na obszar piśmiennictwa informacyjnego i nieinformacyjnego. Jeden z pomysłodawców Wikipedii, Jimmy Walles, zainspirowany działalnością Raymonda i ruchu open source, rozpoczął pracę nad stworzeniem projektu encyklopedii internetowej, której treść generowana byłaby przez społeczność sieciową internautów w ramach szerokiej współpracy. Wikipedię i jej zasoby opatrzono od razu licencją GFDL, która, zdaniem Wallesa, wraz z połączeniem idei otwartości mogłaby się sprawdzić w tym przedsięwzięciu. Dziś, jak widać, założenie to było słuszne. Jednak sam Raymond uważa, że Wikipedia nie przystaje do modelu open source'owskiego. Powodem tego jest oczywisty fakt, że oprogramowanie można obiektywnie oceniać, tj. sprawdzać, czy dobrze wypełnia swoje zadania czy też nie. Inaczej rzecz ujmując, funkcjonują określone standardy pozwalające na ewaluację programu komputerowego, np. sprawdza się szybkość działania, wydajność, niezawodność, funkcjonalność itp., podczas gdy tych samych lub podobnych zasad nie sposób zastosować względem wiedzy. Błędny kod źródłowy programu nie może istnieć, natomiast błędna (nieprawdziwa) informacja już tak. Główna różnica pomiędzy obydwoma obszarami leży w tym, iż jeden stanowi treść funkcjonalną (OSS), a drugi – kognitywną, poznawczą i ekspresywną (open content)[185].

Dopiero wraz z propozycją Lawrenca Lessiga i jego książką *Wolna kultura* rozpropagowano ideę otwartości obszaru szeroko pojętej kultury, wychodząc już zupełnie z obszaru oprogramowania[186]. W tej perspektywie Wikipedia stanowi najpowszechniejszy przykład realizacji idei wolnej kultury. Upraszczając: w kontekście sieci wolna kultura oznacza użytkowanie wolnych mediów internetowych, poprzez które użytkownik otrzymuje dostęp do wytworów kultury. A zatem pod tym, jakże pojemnym, określeniem (*wolna kultura*) mieszczą się swobodnie dostępne, w domenie publicznej lub na licencjach cedujących wiele praw użytkownika, książki, audiobooki, artykuły oraz inne teksty (artystyczne i nieartystyczne), filmy, utwory muzyczne, zbiory fotograficzne, mapy, słowem: wszel-

[185] Dane w akapicie wprowadziłem na podstawie pracy: J. M. Reagle Jr., op. cit., s. 76-78.

[186] Cała książka dostępna jest w Internecie. L. Lessig: *Wolna kultura* [online], [dostęp: 3.02.2012]. Dostępny w WWW: http://www.futrega.org/wk/.

kie rodzaje dokumentów w sensie bibliotekarskim. Wolna kultura tworzy zatem przestrzeń, w której funkcjonują różne treści dostępne dla ogółu.

W tym miejscu można zasygnalizować jeszcze zjawisko tzw. oddolnej digitalizacji, której efektem jest nieautoryzowane dodawanie książek do zasobów cyfrowych. Digitalizacja oddolna polega na tworzeniu cyfrowych wersji dokumentów objętych ochroną prawa autorskiego. Zjawisko ma charakter autonomiczny, niezależny, a nawet odbywający się wbrew interesom różnych instytucji. W procesie udostępniania dokumentów zdigitalizowanych w sposób nieautoryzowany wykorzystuje się serwisy webdwuzerowe, np. Chomikuj (http://chomikuj.pl/), lub niezależną infrastrukturę publikowania i wymiany treści (np. dedykowane strony WWW, grupy dyskusyjne, sieci P2P, fora internetowe)[187]. „Warto zauważyć, że wyraźną część takich działań stanowi digitalizacja pomocy naukowych: skryptów, opracowań, notatek, artykułów czy książek naukowych. Realizowana jest doraźnie przez studentów i często przybiera formy studenckiego współdziałania. Trudno dostępne materiały skanowane są i udostępniane nawet przez wykładowców. W tym wypadku trudno jednoznacznie określić, czy są to działania nielegalne – czy też podejmowane w ramach dozwolonego użytku edukacyjnego [...]. Przykład ten pokazuje, że problem nielegalnej oddolnej digitalizacji dotyczy często publikacji, które z różnych względów są nieosiągalne w inny sposób dla przeciętnego użytkownika Internetu: barierą nie musi być jedynie cena. Archiwalne wydania komiksów czy nieosiągalne w bibliotekach książki czytane są na ekranach komputerów – trudno nie zauważać tutaj pewnego pozytywnego skutku związanego z ograniczaniem efektów wykluczenia spowodowanego np. mieszkaniem na wsi czy brakiem środków na wydatki kulturalne"[188].

Skrótowe, z racji głównej tematyki niniejszej pracy, omówienie zagadnienia **OSINT** (ang. *open source intelligence*, pol. *wywiad jawnoźródłowy*, *biały wywiad*) należy poprzedzić odwołaniem się do kilku innych, powiązanych terminów zaproponowanych przez Locha K. Johnsona[189]. Podstawowym jest *open source data* (używanym skrótem jest OSD), czyli 'dane

[187] Por. A. Tarkowski, J. Hofmokl, M. Wilkowski, *Digitalizacja oddolna. Partycypacyjny wymiar procesu digitalizacji dziedzictwa* [online], [dostęp: 11.11.2011]. Dostępny w WWW: http://www.nina.gov.pl/docs/kultura2.0/digitalizacja-oddolna.-partycypacyjny-wymiar-procesu-digitalizacji-dziedzictwa.pdf.

[188] Ibidem.

[189] Por. L. K. Johnson, *Handbook of intelligence studies*, London, New York 2007, s. 131-132.

open source'[190]. W tym kontekście wszystko może być danymi: tekst pisany, drukowany, wypowiedź ustna lub w inny sposób utrwalona postać jakiegoś zjawiska (nośniki informacji). A zatem dane te mogą być utrwalone na fotografiach, taśmach magnetofonowych, papierze (nośnik nośników informacji) itp. Kolejnym terminem jest *open source information* (skrót OSIF), czyli 'informacja open source'. OSIF składa się z wybranych i przefiltrowanych danych. Dane te wybiera się wedle określonych aktualną potrzebą kryteriów. Innymi słowy, z danych open source tworzy się zestawienia OSIF. Wreszcie OSINT (*Open Source Intelligence*) to informacje pochodzące z zestawień OSIF. OSINT jest już informacją przeznaczoną dla konkretnego odbiorcy, który ma konkretne potrzeby informacyjne. X zadaje pytanie A, na które otrzymuje odpowiedź XA. W tym sensie OSINT można rozpatrywać jako inteligentny. Ostatnim terminem z tego zbioru jest *OSINT-V* (*Validated OSINT*), czyli rodzaj informacji OSINT z bardzo wysokimi współczynnikami trafności i jakości. Informacje OSINT-V tworzone są przez wykwalifikowanych specjalistów, mających dostęp do wartościowych źródeł informacji. Inaczej rzecz ujmując, jest to najbardziej wartościowa informacja. O informację taką niezwykle trudno, stąd powołuje się do tego celu i utrzymuje specjalne służby, których zadaniem jest gromadzenie informacji bezpośrednio ze źródeł. W zasadzie nie jest to niczym nowym, od dawna bowiem państwa i różne organizacje znały wartość informacji bezpośrednio poznanej, tj. pochodzącej od podróżników, turystów, dziennikarzy itd. Nowością jest natomiast to, że OSINT dziś jest wypadkową pewnych procesów i trendów, z których dwa są szczególnie istotne. Pierwszym jest rozbudowywanie Internetu jako narzędzia służącego wyszukiwaniu i udostępnianiu informacji (wiedzy, treści) we wszystkich możliwych językach. Wcześniej zdobycie informacji wymagało fizycznego przemieszczenia się do odpowiedniego źródła. Dziś natomiast za sprawą rozwoju technologii informacyjnych i komunikacyjnych wszelkie dane można uzyskać drogą internetową, niekiedy z pogwałceniem prawa. Wraz ze zjawiskiem rewolucji technologii informacyjnych zrodziło się zjawisko społecznych mediów. Każdy internauta może być dostawcą informacji, a jedynie od odpowiednich narzędzi filtrujących zależy odsianie ziaren od plew. Wraz z narodzinami tego

[190] W niniejszej pracy posługuję się skrótem OSD rozwijanym do postaci Open Source Definition, toteż wskazany w aktualnym kontekście skrót OSD (Open Source Data) nie będzie w dalszej części pracy już stosowany. Przywołałem go jedynie w tym miejscu w celu omówienia zagadnienia OSINT.

społecznego fenomenu, nadprodukcji informacji pojawił się drugi czynnik. Jest nim zjawisko eksplozji informacji, którego rezultatem jest wykładniczy wzrost wiedzy[191].

Uogólniając, OSINT jest to publicznie dostępna informacja, która pojawiła się w dowolnej formie (druk, postać cyfrowa)[192]. W tym sensie OSINT musi spełniać cechy publikacji, tj. informacja ta musi zostać upubliczniona w którejkolwiek formie. Jest więc to informacja tworzona przez ogół i po przetworzeniu przeznaczona (pierwotnie dla ogółu) dla konkretnego odbiorcy. Warto w tym miejscu przypomnieć potoczne wyobrażenie Internetu jako śmietnika. W Internecie jest wiele informacji, jednak przeważają bezwartościowe lub te bardzo mało wartościowe. Są wszak i informacje, które są bardzo cenne. Wydobycie tych informacji i zestawienie w czytelnym układzie podnosi ich rangę. Tak więc OSINT nie jest tożsame z kategorią OSIF. OSINT nie należy też utożsamiać z Internetem i jego zasobami. Sieć internetowa stanowi natomiast źródło, z którego można czerpać wartościowe kwanty wiedzy.

OSINT należy uznać za analityczny proces, podczas którego tworzy się i zestawia razem kwanty informacji i wiedzy pochodzące od ludzi (ze światowego korpusu informacyjnego, tj. wytworzonego przez społeczeństwo globalne), w celu sprostania aktualnym zapotrzebowaniom. Nie jest więc zaskoczeniem, że OSINT znajduje zastosowanie głównie w obszarze wojskowym[193]. Warto też dodać, co pokazuje praktyka, że OSINT musi być systematycznie tworzone i gromadzone[194]. Jest to zrozumiałe, gdyż tylko systematycznie aktualizowane i opracowywane informacje mogą być w tym przypadku wartościowe z punktu widzenia użytkownika końcowego o konkretnych bieżących potrzebach informacyjnych. Informacje są wytwarzane permanentnie, procesy informacyjne odbywają się nieustannie, stąd potrzeba ciągłego filtrowania i opracowywania tych informacji.

Trudno jednoznacznie wskazać ideały ruchu open source w takim procesie, jak również ciężko jest obiektywnie ocenić to zjawisko. Niewątpliwie natomiast pomysłodawcy tego przedsięwzięcia doceniają jakość

---

[191] Dane i informacje zawarte w akapicie pochodzą z pracy: L. K. Johnson, op. cit., s. 131-132.

[192] Por. S. K. Das, *High-level data fusion*, Boston 2008, s. 13.

[193] Por. T. Quiggin, *Seeing the invisible. National security intelligence in an uncertain age*, Hackensack 2007, s. 157.

[194] Por. R. A. Best Jr., A. Cumming, *Open source intelligence (OSINT). Issues for Congress*, [in:] *Intelligence issues and developments*, ed. T. M. Paulson, New York 2008, s. 75.

globalnego korpusu informacyjnego, tworzonego w ramach społecznej komunikacji. W ten sposób wykorzystuje się potencjał ludzkich umysłów. Otwartość manifestuje się z jednej strony poprzez tzw. scaloną świadomość (ludzie w grupie tworzą wiedzę), z drugiej zaś poprzez traktowanie każdego z osobna jako potencjalnie cennego źródła informacji.

Kolejnym zjawiskiem wyrosłym na fali entuzjazmu wobec ruchu open source jest **open source journalism** (citizen journalism, pol. *dziennikarstwo obywatelskie, społeczne*). Termin użyty został po raz pierwszy przez Andrew Leonarda w 1999 roku[195]. Oznacza dziennikarstwo uprawiane przez dziennikarzy amatorów, więc generalnie przez wszystkich internautów. Medium internetowe wpłynęło na sposób tworzenia, upowszechniania i wartościowania informacji czy – ściślej – tzw. newsów[196]. Każdy internauta ma do dyspozycji cały szereg sieciowych technologii informacyjnych, za pomocą których może lokalnie tworzyć informacje, rozpowszechniać je globalnie oraz indywidualnie oceniać przydatność informacji pochodzących od użytkowników z całego świata. Dziennikarstwo obywatelskie (społeczne) kontrastuje się z dziennikarstwem mainstreamowym. To drugie znajduje się w nurcie chronionym mechanizmem copyright wymaga znacznych nakładów finansowych. Nie tylko wytwarzanie i rozpowszechnianie informacji, lecz także jej odbiór są w tym przypadku płatne. Na gruncie mediów społecznych zrodził się nowy fenomen, jakim są wolne społeczne media. Są one wolne od opłat i obciążeń prawnych, ograniczających *de facto* dostęp do informacji. Dziś te dwa obszary znacznie się zbliżyły, czego dowodem może być powszechne wykorzystywanie wytworów społecznego dziennikarstwa w przekazach mainstreamowych (radio, telewizja, prasa). I choć pojawiają się głosy krytyki kierowane w stronę zjawiska dziennikarstwa obywatelskiego[197], to jednak zasługuje ono na uznanie, ponieważ znacznie rozszerza obieg informacji.

Koncepcja open source znajduje również szerokie zastosowanie na polu **edukacji**. Dotyczy to zwłaszcza tych regionów świata, które rozwijają się i mają ograniczone środki finansowe, np. południowej części kontynentu afrykańskiego[198]. W kontekście edukacji open source

---

[195] Por. L. A. Lievrouw, *Alternative and activist new media,* Cambridge 2011, s. 127.

[196] Por. J. Lull, *Culture-on-demand. Communication in a crisis world*, Malden 2007, s. 45.

[197] Por. A. Keen. *Kult amatora. Jak internet niszczy culture*, Warszawa 2007, s. 61.

[198] Por. J. Mapuva, *Defining the role of online education in Today's World*, [in:] *Marketing online education programs. Frameworks for promotion and communication*, ed. E. Demiray, N. S. Server, Hershey 2011, s. 168.

oznacza bezpłatne dostarczanie do szkół oprogramowania oraz wszelkich materiałów edukacyjnych[199]. Dla przykładu można wskazać kurs zorganizowany z inicjatywy Bruce'a Perensa na uczelni Adger University College, oparty na projekcie GNU Radio (jest to zestaw narzędzi informatycznych do nauki o budowie i wdrażaniu systemów radiowych). Projekt polegał m.in. na tworzeniu lepszego interfejsu, uczeniu się przetwarzania sygnału radiowego, a także programowaniu instrumentów radiowych, protokołów komunikacyjnych i anten. GNU Radio umożliwia wszystkim poznanie zasad funkcjonowania radia oraz uczy, jak z niego korzystać, by przy pomocy narzędzi open source mieć swobodny dostęp do sygnału radiowego[200]. Założenia ruchu open source znajdują odzwierciedlenie również w obszarze twórczości filmowej. Open source filmmaking jest to metoda tworzenia i rozpowszechniania filmów przy wykorzystaniu wolnego i otwartego oprogramowania. Filmy te dostępne są na wolnych licencjach (np. zgodnych z założeniami licencji OSD), które umożliwiają wykorzystywanie materiału filmowego przy montowaniu kolejnych produkcji[201].

Wytyczne open source wykorzystywane są również do rozmaitych badań Internetu i dostępności jego zasobów, np. dla osób niepełnosprawnych. Tu można wskazać projekt The European Internet Accessibility Observatory (EIAO), który w całości opiera się o OSS. Wykorzystując robota internetowego (indeksującego) o nazwie *Harvestman*, poddaje się ewaluacji wybrane strony internetowe poszczególnych państw europejskich po to, aby sprawdzić, które z nich spełniają kryteria dostępności zawartych na nich treści dla osób z różnymi dysfunkcjami, np. wzroku[202].

Open source curriculum (skrót OSC) są to edukacyjne zasoby online (np. kursy), z których można swobodnie korzystać, jak również dowolnie

---

[199] Por. M.E. P. Deily, *The education week guide to K-12 terminology*, San Francisco 2009, s. 78.

[200] Por. W. Shrum et.al., *Past, present, and future of research in the information society*, New York 2006, s. 41-42; *Wikipedia. The free encyclopedia* [online], [dostęp: 31.01.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/GNU_Radio.

[201] Por. *Wikipedia. The free encyclopedia* [online], [dostęp: 8 lutego 2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Open_source_film.

[202] Por. W. Shrum et.al., op. cit., s. 42. Jak się okazało w badaniach z marca i kwietnia 2008 roku, wiele stron nie przeszło tego testu pozytywnie. Uogólnione wyniki analiz ukazują, że bardzo dużo stron internetowych w poszczególnych państwach europejskich nie spełnia kryteriów dostępności dla osób niepełnosprawnych. Por. *EIAO* [online], [dostęp: 31.01.2012]. Dostępny w WWW: http://www.eiao.net/.

je rozpowszechniać i modyfikować. OSC bazuje na idei open source, polegającej m.in. na udostępnieniu kodu programu komputerowego, informacji na temat surowców wykorzystanych przy produkcji określonego urządzenia czy receptury leku. W tym jednak przypadku udostępnia się określone materiały edukacyjne. Domyślić się można, że udostępnia się w ten sposób nie tylko sam kurs, ale i scenariusze, konspekty lekcji itp., słowem: wszystko, co może być przydatne do uczestnictwa w zajęciach, jak również do ich prowadzenia. Idea zasadza się na tym, aby specjaliści zaznajomieni z określoną tematyką współtworzyli materiały edukacyjne *via* sieć internetowa i w ten sposób je udostępniali. Korzyści są oczywiste: z jednej strony istnieje możliwość globalnego edukowania społeczeństw, z drugiej – możliwość pogłębiania wiedzy w określonej, specyficznej tematyce. Przykładem OSC jest projekt bliźniaczy Wikipedii, tj. Wikiversity[203].

Realizowane w ramach zajęć edukacyjnych ideały ruchu open source mają służyć zasadniczo dwóm celom. Po pierwsze – pokazaniu, że istnieje wiele narzędzi zarówno OSS, jak i OSH, których warto używać w codziennej aktywności. Dają one bowiem liczne możliwości dostosowywania ich do własnych potrzeb. Pozwalają zrozumieć, jak funkcjonują w ogóle pewne technologie, a zatem przyczyniają się do poznawania rzeczywistości. Dzieje się tak choćby dlatego, że udostępniane jest nie tylko narzędzie (OSS, OSH), lecz także pełna jego dokumentacja. W przestrzeni OSS oraz OSH wszystko jest egzoteryczne, transparentne, a więc bezpośrednio dostępne aparatowi poznawczemu ludzi. Po drugie, open source nastawia się na to, aby wszyscy ludzie mieli do informacji i wiedzy wolny dostęp. Open source „pamięta" więc o osobach niepełnosprawnych, stąd organizuje się rozmaite badania, których celem jest monitorowanie Internetu i sprawdzanie, jak to podstawowe założenie jest realizowane. Dodatkowo naukowcy związani z ruchem open source prowadzą odpowiednie szkolenia przeznaczone dla osób tworzących i zarządzających zasobami internetowymi, w celu instruowania ich, aby pamiętali, że architektura Internetu powinna być funkcjonalna, a więc dostępna wszystkim grupom osób. Open source przeciwdziała zatem wykluczeniu cyfrowemu, a przez to, poniekąd, społecznym podziałom, tj. wykluczeniu społecznemu.

---

[203] Akapit powstał w oparciu o zasoby *Wikipedii*: *Wikipedia. The free encyclopedia* [online], [dostęp: 8.02.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Open_source_curriculum.

Na zakończenie wyliczę jeszcze takie obszary zastosowania open source, jak biologia[204], kryptografia[205], robotyka[206], systemy informacji geograficznej (GIS)[207]. Niektórzy sugerują, że proces wytwarzania oprogramowania open source może stanowić wzór dla produkcji części samochodowych, a także mógłby być powielany na polu biotechnologii[208]. Ideały open source znajdują zastosowanie także w ekologii, handlu elektronicznym, medycynie, działalności wydawniczej, przemyśle odzieżowym, żywności i żywieniu itp.

Idea leżąca u podstaw open source „nakazuje" traktować ludzi nie jako biernych konsumentów, lecz przede wszystkim jako aktywnych, pomysłowych twórców, z których każdy, jeśli tylko zechce, może zasilić ten ruch własnymi pomysłami, czego najlepszym dowodem jest omówione tutaj szerokie spektrum implementacji.

1.6. ROZWÓJ RUCHU OPEN SOURCE

Podłoże ruchu open source stanowi swoboda w wymianie informacjami i, jak już wyjaśniłem, idea ta wzięła się ze społecznej praktyki. Ta z kolei wypływa z ludzkiej natury. Przynajmniej potencjalnie jest tak, że praktyka społeczna zgłasza zapotrzebowanie na takie obiekty (informacje, technologie informacyjne, narzędzia itp.), które są społeczeństwu potrzebne. Ludzi śmiało można określić jako organizmy *informacjożerne* i *wiedzożerne*, a więc takie, które do egzystencji potrzebują nie tylko pokarmu w sensie biologicznym, lecz także pokarmu w sensie metaforycznym – pokarmu dla umysłu. Tym pokarmem jest informacja i wiedza. Ludzie zdobywają ją od innych ludzi, m.in. poprzez codzienne doświadczenie,

---

[204] Np. R. H. Carlson, *Biology is technology. The promise, peril, and new business of engineering life*, Cambridge 2010.

[205] Np. A. McAndrew, *Introduction to cryptography with open-source software*, Boca Raton 2011.

[206] Np. L.A.R. W. Edwards, *Open-source robotics and process control cookbook. Designing and building robust, dependable real-time systems*, Amsterdam, Boston 2005.

[207] Np. M. Neteler, H. Mitasova, *Open source GIS. A GRASS GIS approach*, Boston 2004.

[208] Por. J. Lerner, J. Tirole, *Economic perspectives on open source*, [in:] *Perspectives on free and open source software*, ed. J. Feller et al., Cambridge 2005, s. 72.

jak również przez wyspecjalizowane instytucje, np. szkoły i uniwersytety. Wymiana informacjami nie jest niczym obcym ani nowym. Podobnie jest z wymianą wiedzą. Celem powszechnej edukacji jest przecież wzbogacanie w wiedzę i wyrównywanie poziomu wykształcenia. Z kolei celem szkolnictwa wyższego jest napędzanie i stymulowanie wszelkiego rozwoju, który zapewnia się poprzez dostęp do najnowszych osiągnięć naukowych, tj. do najnowszej informacji i wiedzy. Jeśli spojrzeć na funkcjonowanie wszelkiego szkolnictwa to okaże się, że w zakresie edukacyjnym materiały szkolne (skrypty, teksty, opracowania itp.) od zawsze krążyły wśród uczniów i studentów, przy czym każdy mógł je swobodnie modyfikować, np. wzbogacać o aktualne zagadnienia, dostosowywać do własnych potrzeb, a także poprawiać, powielać i rozpowszechniać (w tej perspektywie ideały edukacji świetnie korespondują z ideałami open source)[209]. Z drugiej zaś strony samo open source wyrosło w przestrzeni dyskursu uniwersyteckiego i uniwersyteckiej działalności badawczo-rozwojowej. Stąd wzajemny, w zasadzie, wpływ i rozwój edukacji wszelkiego szczebla i ruchu open source.

Dodatkowo ruch open source (a także dyskusja nad tym zagadnieniem) rozwijał się i rozwija dzięki stymulacji Internetem i środowiskiem hipertekstowym World Wide Web. Bez Internetu i Webu nie byłoby open source. Środowisko cyfrowe spełnia się jako miejsce dla rozwijania koncepcji i dyskusji nad zagadnieniami open source, a koncepcyjne podstawy Internetu stanowiły kolejny impuls napędzający całą inicjatywę. Oto bowiem Internet powstał jako wynik kolektywnej współpracy ogromnej rzeszy naukowców opracowujących jego podstawy. Ówczesny projekt sieci ARPANET można przecież uznać za przedsięwzięcie open source'owskie. Z jednej strony tworzono odpowiednie narzędzia począwszy od komputerów, poprzez karty sieciowe, aż po same łącza (przewody), z drugiej – odpowiednie oprogramowanie pozwalające łączyć te komputery, zapewniające standardy transmisji danych, obsługujące protokoły komunikacyjne. Nie byłoby możliwe, aby mała grupa osób udźwignęła ciężar stworzenia podłoża działającej sieci internetowej. Cały projekt, tworzony ponad dekadę, wymagał zaangażowania wielu naukowców, badaczy, inżynierów, projektantów, pomysłodawców i teoretyków werbujących się z różnych dziedzin i dyscyplin wiedzy i nauki oraz działalności ludzkiej. Na wstępie prac nikt nie mógł przewidzieć, co będzie warunkiem *sine*

[209] Por. L. Chao, *Utilizing open source tools for online teaching and learning. Applying Linux technologies*, Hershey 2009, s. 5.

qua non sieci Internet. Wszyscy ci ludzie, współpracując, działali w sposób bardzo zbliżony do tego, w jaki pracują ludzie zaangażowani w projekty open source, a więc w sposób wyartykułowany przez Henry'ego Jenkinsa: „nikt nie wie wszystkiego. Każdy wie coś. Całą wiedzę ma ludzkość"[210]. Upraszczając: powstaje projekt X, następnie zainteresowani wzbogacają go o elementy Y aż do momentu, gdy projekt zacznie prawidłowo działać. Można to zapisać następująco: X(Y1 Y2 Y3 ... Yn).

Innym znaczącym projektem rozwijanym w duchu open source, choć termin ten jeszcze nie funkcjonował, był projekt systemu operacyjnego UNIX. W latach 60. XX wieku AT&T Bell Labs, General Electric oraz MIT stworzyli razem operacyjny system Multics (Multiplexed Information and Computing Service). System nie spełniał jednak wyjściowych oczekiwań swoich pomysłodawców, przez co projekt ten zarzucono, a część organizacji skupionych wokół tego projektu zrezygnowało z dalszej współpracy. Ci, którzy zostali, kontynuowali pracę, wprowadzając liczne zmiany, począwszy od propozycji nowej nazwy systemu, który został określony jako UNICS (UNIX). W rezultacie prac UNIX stał się w latach 70. i 80. wiodącym systemem obsługującym ówczesne serwery i komputery osobiste. W tym czasie powstało wiele różnych jego wersji. Generalnie jednak programiści zostali skupieni w dwóch obozach. Pierwszy realizował koncepcję oprogramowania własnościowego, a drugi – otwartego i wolnego (OSS). W 1984 roku nastąpił wyraźny rozłam. Firma AT&T zdecydowała się skoncentrować na pracach nad UNIX-em jako programem własnościowym. Później inne firmy dołączyły do tej grupy, co ostatecznie przełożyło się na wytworzenie różnych wersji systemu UNIX jako oprogramowania własnościowego. Dodać można, że wersje te nie były kompatybilne, co poskutkowało tym, że działania komercyjne zastopowały innowacyjność i nie zyskały społecznego uznania. Druga grupa zorganizowana była przez programistów z Berkeley Software Distribution (BSD) przy Uniwersytecie Berkeleya w Kalifornii. Stworzyli oni open source'owskie wersje systemu UNIX, m.in. FreeBSD, NetBSD, OpenBSD. Inicjatywa z Berkeley zaowocowała tym, że później powstało jeszcze wiele innych i innowacyjnych rozwiązań unixowych, tj. różnych wersji systemu z rodziny BSD, np. Net BSD 3.0.1, AIX Version 6.1[211].

---

[210] H. Jenkins, *Kultura konwergencji. Zderzenie starych i nowych mediów,* Warszawa 2007, s. 31.

[211] Dane w akapicie przywołałem z pracy: L. Chao, op. cit, s. 5-7.

Warto dodać, że naukowcy z Berkeley byli sponsorowani przez ARPA, a ich znaczącym wkładem w rozwój systemu UNIX było jego dostosowanie do protokołu TCP/IP[212]. Nietrudno się domyślić, że był to warunek konieczny do tego, aby komputery wyposażone w ten system mogły komunikować się za pośrednictwem sieci internetowej. W ten sposób, finansowany z pieniędzy publicznych, UNIX udostępniany był po kosztach dystrybucji i jednocześnie sprzyjał rozwojowi sieci internetowej, dzięki niemu bowiem sieci lokalne i regionalne można było tworzyć wszędzie tam, gdzie były komputery i linie telefoniczne[213].

Kolejnym przełomowym wynalazkiem promującym sieć internetową był World Wide Web, którego powstanie również zbliżyło się do ideałów realizowanych później w ruchu open source. Mam na myśli m.in. to, że na WWW nie ciążył mechanizm copyright. System W3 został udostępniony wszystkim użytkownikom Internetu za darmo[214]. Gdyby nie to swobodne rozpowszechnienie wśród szerokich rzesz internautów, Web nie zyskałby zapewne tak dużego uznania i tak jak, na przykład, aplikacja HyperCard straciłby swoją szansę i przepadł bezpowrotnie[215]. Założenie Tima Bernersa Lee, aby stworzyć uniwersalne medium służące wymianie wszelkich informacji (ang. *universal medium for sharing information*) dla wszystkich użytkowników wpisało się w wielki społeczny fenomen[216]. Polega on na tym, że wytwarza się wiedzę i informacje, a następnie udostępnia je innym. Kolejnym krokiem jest tworzenie narzędzi, które zapewniają dostęp do tych zasobów. Jeżeli jeszcze towarzyszy temu uwalnianie informacji, wiedzy, jak również samych narzędzi, to już w zasadzie wchodzi się w obszar wyznaczany zasadami open source. Tim Berners Lee chciał usprawnić współpracę z kolegami, wykorzystując technologie informacyjne, stąd początkowo WWW miała być aplikacją pomocną w wyszukiwaniu numerów telefonicznych[217]. Później przeznaczeniem powstającego WWW miało być „poprawienie efektywności dostępu do literatury

---

[212] Por. M. Castells, op. cit., s. 84.

[213] Ibidem.

[214] Por. T. Berners-Lee, *Weaving the Web. The original design and ultimate destiny of the World Wide Web*, New York 2000, s. 74.

[215] Por. M. Levene, *An introduction to search engines and web navigation*, ed. 2., Hoboken 2010, s. 224.

[216] Por. T. Berners-Lee, op. cit., s. 84.

[217] Por. I. Tuomi, *Networks of innovation. Change and meaning in the age of the Internet*, Oxford 2002, s. 48.

z fizyki jądrowej"[218], a także ułatwienie wymiany informacji pomiędzy fizykami pracującymi w różnych instytucjach naukowych na całym świecie. System określano również jako pozwalający organizować i dystrybuować wewnętrzne informacje krążące po CERN-ie[219]. W konsekwencji Berners Lee wraz ze współpracownikami z CERN stworzyli Hyper Text Transfer Protocol (HTTP), który ustandaryzował komunikację na osi klient – serwer oraz opracowali język HTML (Hyper Text Markup Language), który pozwolił na łączenie hiperłączami różnych obiektów zgromadzonych na odległych komputerach[220]. Pierwotnie HTTP wykorzystywano do usprawnienia pracy pomiędzy komputerami znajdującymi się w CERN, później jednak został wykorzystany na potrzeby WWW. Patrząc na hipertekstową konstrukcję WWW, można zauważyć, że każda strona WWW ma lub może mieć (dziś HTML wypierany jest przez inne języki) otwarty i swobodnie dostępny kod źródłowy HTML, co stoi w zgodzie z założeniami ruchu open source.

Nawiążę jeszcze do ustaleń Tima O'Reilly, który w XX wieku posłużył się terminem *infoware* na oznaczenie specyficznego rodzaju oprogramowania. Jego ustalenia wpisują się w dyskutowane w tym miejscu problemy. Infoware charakteryzuje się tym, że składa się z małej pojemnościowo aplikacji (mało software'u), a z dużej ilości informacji. Odwrotnie było dawniej, kiedy tradycyjne oprogramowanie składało się z dużej ilości software i małej ilości informacji. Jeżeli natomiast spojrzeć na programy takie, jak np. serwisy aukcyjne, księgarskie serwisy e-commerce czy gigantyczne sklepy internetowe, jak np. Amazon, to okaże się, że na ich zasoby składa się skąpy pod względem ilościowym program – jest to jakaś strona internetowa wraz z podstronami, będąca *de facto* strukturą – oraz ogrom informacji o dostępnych tam obiektach, np. produktach. Infoware nie jest bynajmniej czymś prostym w obsłudze, gdyż zarządzanie ogromną ilością danych wymaga dużych nakładów pracy, jednak z punktu widzenia software'u nie jest to rozbudowany program. Część softwareowa aplikacji typu infoware ogranicza się do dynamicznego i atrakcyjnego interfejsu oraz przycisków akcji, np. wejdź, kup, zaznacz itp. W rezultacie podjętych działań użytkownik otrzy-

[218] D. Kołcio, J. Sobecki, *Wykorzystanie WWW w bibliotekach i ośrodkach informacji naukowotechnicznej*, „Praktyka i Teoria Informacji Naukowej i Technicznej" 1995, nr 4, s. 8.

[219] Por. *The encyclopedia of new media*, ed. S. Jones, New York 2003, s. 25-26.

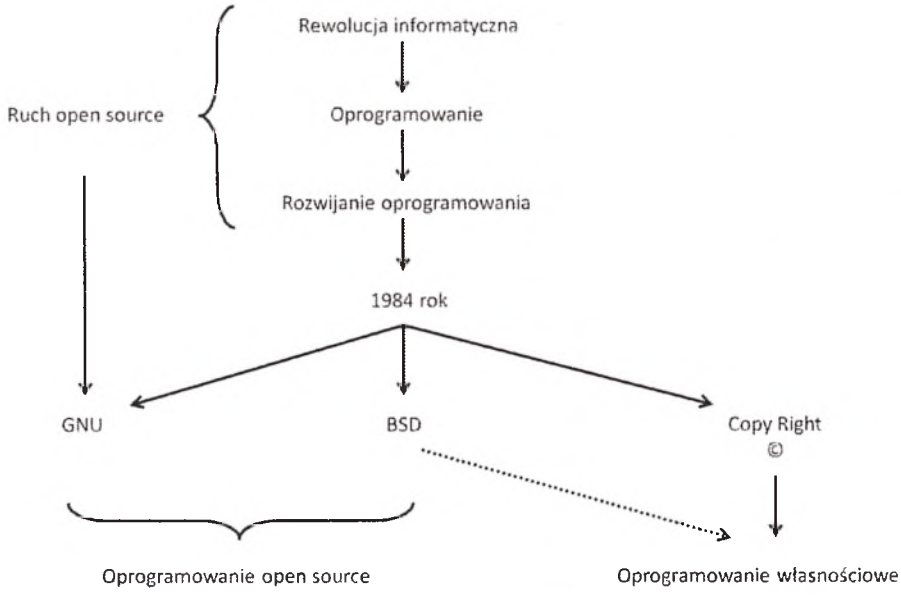
[220] Jego zadaniem, jak sam to wyraził, miało być jedynie „ożenienie" hipertekstu z Internetem. Por. T. Berners-Lee, op. cit., s. 6. Hipertekst istniał już w postaci różnych projektów, programów, Internet natomiast – jako funkcjonująca i popularna sieć komunikacyjna.

muje informację zwrotną, zapoznaje się z informacją o możliwościach nabycia lub o nabyciu towaru (zawartej umowie) i sposobach płatności. Część informacyjna nie ogranicza się tylko do prostych informacji o produkcie przedstawionych przez producenta, lecz wzbogacona jest o informacje pochodzące od internautów (opinie, komentarze) oraz rozmaite zestawienia i rankingi popularności oferowanych towarów. Oczywistą koneksją, łączącą infoware z inicjatywami typu open source, jest zaangażowanie rzesz internautów w funkcjonowanie określonej aplikacji. Nie muszą oni tworzyć programu w ukryciu, tj. poprawiać kodu i programować nowych aplikacji, lecz mogą na powierzchni generować ogromne ilości treści. Współudział w tworzeniu zasobów aplikacji oraz w funkcjonowaniu aplikacji jest realizacją ideałów ruchu open source. Zjawisko infoware dostrzeżono już ponad dziesięć lat temu, mniej więcej w okresie inicjowania działalności OSI. Jak się okazuje, ruch open source nie musiał, i zapewne nadal nie musi, rywalizować z inicjatywami firm komercyjnych, tworzących oprogramowanie własnościowe. Posługując się metaforą, zmienił on bowiem reguły gry. OSS tworzone jest oddolnie przez ludzi i dla ludzi, powstaje więc po to, aby rozwiązywać ich problemy. Już dawno temu cała branża informatyczna przeszła z modelu urządzeniocentrycznego na programocentryczny. Ruch open source jeszcze bardziej ten paradygmat wzmocnił i przy okazji rozciągnął na nowe pole. Nie będzie więc przesadą uznać, że infoware realizuje założenie skomputeryzowania obszaru, którego wcześniej nie udało się skomputeryzować. Interakcje na osi człowiek – komputer i odwrotnie realizuje się w obszarze infoware, a nie samego software. Przewidywania O'Reilly'ego co do tego, że open source, prężnie rozwijany na przełomie XX i XXI wieku, wytyczy nowe cele dla branży informatycznej, sprawdziły się. Nie tylko OSS nadal jest rozwijane, lecz także idee współpracy i współtworzenia wiedzy i informacji znalazły swe odzwierciedlenie w nowych sieciowych inicjatywach[221].

Aktualnie obowiązującym modelem współczesnego Internetu jest bowiem model Web 2.0, którego fundament stanowi kolektywna praca rzesz internautów[222]. W podsumowaniu należy zauważyć, że ruch open source

[221] Podane w akapicie informacje wprowadziłem na podstawie: T. O'Reilly, *Hardware, software, and infoware* [online], [dostęp: 2.01.2012]. Dostępny w WWW: http://oreilly.com/openbook/opensources/book/tim.html.

[222] Model Web 2.0 omówiłem w innej pracy: S. D. Kotuła, *WEB 2.0 – współczesny paradygmat Internetu*, [w:] *Oblicza Internetu. Architektura komunikacyjna Sieci*, red. M. Sokołowski, Elbląg 2007, s. 181-188.



Rys. 4. Historia rozwoju oprogramowania open source i oprogramowania własnościowego [opracowanie własne].

faktycznie wpłynął na ukształtowanie całej przestrzeni Internetu, całego środowiska cyfrowego, wytyczając jemu właściwe (nowe) priorytety.

Na zakończenie posłużę się jeszcze jednym schematem ilustrującym rozwój idei open source (rys. 4). Schemat, mam nadzieję, przyczyni się do uwypuklenia zasadniczego faktu, że ruch open source nie został zainicjowany w latach 80. XX wieku, już od dawna bowiem funkcjonował.

Zdaję sobie sprawę, że można by cofnąć się znacznie dalej w czasie i przypomnieć historię rozwoju matematyki, wynalazek cyfr, zwłaszcza cyfry 0, przypomnieć nazwiska wielkich naukowców, filozofów, matematyków, uczonych, bez wkładu których nie rozwinęłaby się ta gałąź wiedzy. Trzeba by wymienić rozmaite wynalazki, projekty, tworzone lub tylko projektowane na papierze oraz nazwiska konstruktorów rozlicznych urządzeń, liczne prace naukowe oraz doświadczalno-eksperymentalne, projekty pierwszych maszyn liczących, komputerów generacji zerowej i następnych itd. Wszystko to miałoby cel taki, aby dotrzeć do tego momentu w rozwoju sektora, jakim jest oprogramowanie komputerowe. Wszystko to, o czym tu wspomniałem, a czego szczegółowo nie rozwijam, objąłem wspólnym określeniem *rewolucja informatyczna*. Według mnie, sformułowanie to bardziej niż określenie *rewolucja komputerowa* przystaje do ogromnego procesu, który dokonał się w ostatnich tysiącleciach i bez którego nie byłoby dziś informatyki, komputerów i oprogramowania, słowem: technologii informacyjnych. Pamiętać przy

tym należy, że cały ten rozwój przebiegał w duchu otwartości, swobodnej wymiany i modyfikowania pomysłów poprzedników. Można więc uznać, że zachodził w duchu open source. Co więcej, początkowo samo oprogramowanie również rozwijano w pełnej współpracy i otwartości. Choć pierwsze programy komputerowe zaczęto patentować już w latach 70. XX wieku, to jeszcze nie uzyskiwały one wówczas takiej popularności. Rynek informatyczny był wówczas bardzo mały. Aby cała branża informatyczna przeszła na nowy model, asumpt do tego musiała dać duża firma i musiało to dotyczyć czegoś bardzo popularnego. Przełomowym okazał się rok 1984, kiedy to rozpoczęto tworzenie systemu operacyjnego UNIX jako oprogramowania własnościowego (o zamkniętym kodzie), tym samym jakby wyznaczając nową zasadę: program komputerowy jest własnością tego, kto go tworzy i tylko on może z nim robić co zechce. Inni muszą dostosować się do narzuconych reguł. W 1984 roku zainicjowano zatem ruch oprogramowania komercyjnego (własnościowego). Ruch open source istniał, jak widać, od samego początku, choć nikt go tak nie nazywał. W tym czasie związani z dawnymi ideałami (open source) musieli wyraźniej zaakcentować swoją przynależność. Stallman założył Fundację Wolnego Oprogramowania, rozpropagował mechanizm copyleft i zaczął uniezależniać się od wcześniejszych wpływów i zależności. Rozpoczął prace nad stosownymi licencjami oraz w pełni wolnym systemem operacyjnym. Programiści z Uniwersytetu Berkeleya w Kalifornii zaczęli tworzyć swoją własną wersję UNIX-a i zaproponowali bardziej liberalne licencje. UNIX stał się produktem chronionym mechanizmem copyright. Nurt, który wówczas się pojawił – nurt oprogramowania własnościowego – skupiony został wokół zasad komercyjnych i biznesowych. W dalszym okresie zyskał on znacznie na popularności, co przełożyło się na dalsze tworzenie i propagowanie w pełni komercyjnych systemów operacyjnych.

Zdaję sobie sprawę z pewnych uproszczeń zarówno schematu, jak i wywodu, a także ograniczeń – mam na myśli fakt odwoływania się w tym fragmencie tylko do programów, jakimi są systemy operacyjne. Wydaje mi się jednak, że skróty te pozwoliły nakreślić główne nurty rozwoju oprogramowania w ogóle, a w kontekście systemów operacyjnych – pokazać wyraźnie, jakie tory i drogi przebywa kod, aby stać się tak zaawansowaną aplikacją, jak system operacyjny. Oto bowiem z inicjatywy Stallamana powstał nurt GNU, na falach którego wyłonił się system operacyjny Linux. Z nurtu działań BSD powstała grupa systemów operacyjnych BSD (FreeBSD, OpenBSD). Są to przykłady oprogramowania open source.

Następnie nurt BSD zasilił również rozwiązania komercyjne, tj. niektóre systemy operacyjne. Wreszcie z nurtu copyright wyłoniły się w pełni komercyjne programy. Nie będę wymieniał żadnego z nich, aby nie stwarzać wrażenia skłaniania się ku któremuś z nich. Dwa ostatnie (po części BSD oraz w pełni copyright) przykłady stanowią egzempla oprogramowania własnościowego.

Z jednej strony środowisko pracy sprzyja inicjatywom utrzymywanym w duchu open source. Z drugiej strony ideały ruchu open source wpływają na tworzenie OSS. Internet i jego technologie pierwotnie powstały jako open source'owskie. I ta open source'owska natura Internetu wpłynęła i wpływa na to, że wiele programów, tworzonych *de facto* sieciowo, tworzy się jako open source'owskie. Wszystko, co powstaje w Internecie, przesiąknięte jest przynajmniej po części ideami open source. Poza tworzeniem OSS, ruch open source odcisnął piętno na innych obszarach, takich jak OSH, OA, OSINT, TDI itd., czyli skupiał się wokół człowieka i jego najbliższego otoczenia. Każda akcja zainicjowana i rozwijana w środowisku internetowym ma szansę realizować ideały ruchu open source, a więc stać się wypadkową zjawiska open source.

* * *

Pełny obraz zjawiska open source nieco się zamazuje, a zrozumienie natury odbywających się tam procesów zdaje się umykać percepcji. Wpływa na to co najmniej kilka czynników. Są wśród nich następujące: dwie działające w kooperacji, choć niezależnie od siebie organizacje (FSF, OSI), różnice wynikające z filozofii podejścia tych organizacji do oprogramowania, dodatkowo obecny w tej przestrzeni nurt komercyjny, odznaczający się własnymi zasadami działania[223], ogromna rzesza internautów zasilająca ten gigantyczny ruch społeczny, a każdy z użytkowników ma swoją własną wizję tego, czym jest open source i zasila ten ruch po swojemu, owocem czego jest dużo różnych licencji OS oraz terminologiczne zamieszanie. Z jednej strony wprowadza się rozmaite neolo-

[223] Niektórzy komunikolodzy zauważyli, że w ostatnim czasie zjawisko open source uległo transformacji w bardziej komercyjny nurt, który można nazwać OSS 2.0. Cały proces tworzenia oprogramowania z modelu bazarowego przemienił się w ściślej planowany. Spowodowane to było m.in. zwiększonymi nakładami finansowymi inwestowanymi w tę sferę. Generalnie rzecz ujmując, wszystkie działania prowadzone z obydwu stron przybierają bardziej przemyślaną i dobrze zorganizowaną formę. Por. B. Fitzgerald et al., *Adopting open source software. A practical guide*, Cambridge 2011, s. 2.

gizmy[224], które w założeniu mają ściślej wskazywać na różne aplikacje, ale niekiedy w rezultacie są dowolnie używane, co powoduje poznawczą dezorientację u osoby postronnej lub u kogoś, kto chce się zorientować w pryncypiach całego ruchu.

---

[224] Od dłuższego czasu jedynie obowiązującymi terminami były *free software*, *free open source software* oraz *free libre open source software*, które oznaczają *de facto* to samo, czy – ściślej – używane są do denotowania tego samego rodzaju oprogramowania. Podobnie termin *open source* i denotowany przezeń byt przez wiele lat tożsame były z FS/FOSS/FLOSS. Zmieniło się to w 2007 roku. Wtedy bowiem coraz więcej firm o proweniencji komercyjnej zaczęło otwierać częściowo kod źródłowy swoich aplikacji, ograniczając jednocześnie pozostałe zasady otwartości zalecane przez ruch open source. Doprowadziło to do wyłonienia się terminu COSS, czyli *commercial open source software*. W miejsce hiperonimu *open source*, aby nadać precyzji wypowiedzi (z jakiego dokładnie rodzaju programem open source ma się do czynienia), dziś stosuje się coraz częściej bardziej szczegółowe FOSS/FLOSS lub COSS. Por. K. S. Sampathkumar, *Understanding FOSS Version 3.0i revised*, 2009, s. 2.

2. OPEN SOURCE - EKSPLIKACJA

Organizacją popularyzującą ideę open source jest Open Source Initiative. Nadzoruje ona standard open source (The Open Source Definition, OSD), który jest używany do tego, aby jednoznacznie określać, czy dany program jest programem open source. Każdy, który ma być określany tym mianem, musi zatem spełniać postulaty przedstawione w OSD. Wyjaśniając, czym jest oprogramowanie open source, należy więc w pierwszej kolejności stwierdzić, że jest to takie oprogramowanie, które zostało udostępnione na licencji zgodnej z OSD. Można zatem przyjąć, iż OSD stanowi również zbiór zasad, których należy przestrzegać, tworząc lub stosując konkretną licencję open source dla programu, który chce się uznać za OSS.

Przy tym należy pamiętać, że program open source nie jest programem przekazanym do domeny publicznej. Nikt na mocy stosownej licencji open source nie przekazuje też praw do programu osobom trzecim. Twórca zezwala na kopiowanie, modyfikowanie i dystrybuowanie programu pierwotnego oraz pochodnego, pod pewnymi jednak warunkami. Przeważnie dotyczą one tego, że w programie pochodnym należy wskazać autora programu pierwotnego oraz nakazują zastosowanie wobec programu pochodnego tej samej licencji[225].

OSI certyfikuje także licencje (OSI Certified), aby potwierdzić, że dana licencja spełnia postulaty OSD[226].

Najprościej rzecz ujmując, ideą open source jest zapewnienie każdemu użytkownikowi możliwości swobodnego kopiowania programu oraz moż-

[225] Informacje w akapicie przybliżyłem w oparciu o pracę: J. Locke, op. cit., s. 9-10.

[226] Por. A. M. S. Laurent, *Understanding open source & free software licensing*, Sebastopol 2004, s. 8.

liwości tworzenia kolejnych jego wersji (tworzenie aplikacji budowanych w oparciu o pierwowzór)[227]. We wstępie definicji OSD wskazano, że open source nie oznacza tylko dostępu do kodu źródłowego. Wprowadza ono bowiem dziesięć zasadniczych warunków, które musi spełnić program, aby był programem open source[228]. Poniżej przedstawiam te warunki. Nie jest to tłumaczenie wersji oryginalnej, tę można znaleźć bez problemu w Internecie, a raczej jest to uprzystępnienie wytycznych wraz z komentarzem do przedstawnionych tam postulatów. Dzięki temu, mam nadzieję, zalecane sposoby formułowania licencji dla OSS i warunków, jakie musi OSS spełniać, aby być programem open source, staną się jaśniejsze.

1. **Wolna redystrybucja (ang. *free redistribution*)**

   Licencja zgodna ze standardem OSD nie powinna w żaden sposób zabraniać nikomu sprzedawać lub rozdawać oprogramowania jako elementu większej całości (szersza dystrybucja zawierająca programy z różnych źródeł). Licencja nie powinna wymagać przekazywania żadnych opłat z takiej sprzedaży. Można zatem wziąć kod źródłowy lub jego fragment z oprogramowania, licencjonowanego stosowną licencją open source, i umieścić go jako część większej aplikacji. Wtedy aplikację tę można sprzedawać lub rozdawać. Natomiast licencja nie narzuca obowiązku, aby płacić komuś, kto wcześniej stworzył wykorzystany fragment kodu lub cały kod źródłowy.

2. **Kod źródłowy (ang. *source code*)**

   Program musi zawierać kod źródłowy i musi być rozpowszechniany zarówno w postaci kodu źródłowego, jak i w postaci skompilowanej (program gotowy do pracy). Preferuje się dostarczanie kodu źródłowego niezależnie od skompilowanej postaci programu w celu zmniejszenia wielkości plików i ułatwienia dystrybucji. Jeżeli program dostarczany jest bez kodu źródłowego, to musi być podany sposób uzyskania tego kodu (najlepiej bez opłat *via* Internet, ewentualnie po kosztach związanych z wykonaniem jego kopii). Kod źródłowy musi być ponadto dostępny w najprostszej postaci po to, aby łatwo można było dokonywać w nim modyfikacji. Zabronione jest celowe komplikowanie kodu źródłowego.

---

[227] Por. M. Overly, op. cit., s. 1.

[228] Pełna treść definicji OSD dostępna jest na stronie OSI. *The open source definition (annotated)* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://www.opensource.org/osd.html.

3. **Dzieła pochodne (ang. *derived works*)**

   Licencja musi zapewniać możliwość tworzenia modyfikacji programu i programów pochodnych (na bazie programu pierwotnego). Dodatkowo zarówno modyfikacje, jak i kolejne wersje programu, muszą być dostępne na zasadach przedstawionych w tej samej licencji. Paragraf ten zawiera wytyczne fundamentalne dla licencji open source. Zezwala on bowiem, choć nie nakazuje, na zastosowanie mechanizmu *copyleft* poprzez stosowną licencję. Innymi słowy, w zależności od zastosowanej licencji open source program może pozostać otwarty (licencje typu GPL) lub zamknięty (licencje typu BSD).

4. **Spójność kodu źródłowego autora (ang. *integrity of the author's source code*)**

   Licencja może zabraniać rozpowszechniania kodu źródłowego programu w zmodyfikowanej jego wersji tylko wtedy, gdy dozwolona jest dystrybucja poprawek do programu (ang. *patch*) wraz z kodem źródłowym w celu ulepszania tego programu. Licencja musi natomiast jasno zezwalać na rozpowszechnianie programu powstałego ze zmodyfikowanego kodu źródłowego. Licencja może też nakazywać, aby dzieła pochodne posiadały inną nazwę oraz inny numer wersji niż oprogramowanie oryginalne (pierwotne). OSD, jak widać, dozwala więc na uznanie za OSS takiego programu, którego kod źródłowy jest dostępny i dostępne są oddzielnie poprawki, czyli *patche*, do tego programu. Ten zapis chroni twórców, których oryginalne prace (programy) pozostaną niezależnymi produktami, tj. wersja skompilowana, jej kod źródłowy, numer i nazwa programu będą oryginalne i będą różniły się od kolejnych modyfikacji.

Można zatem opublikować OSS w wersji skompilowanej i wraz z dostępnym kodem źródłowym. Nie będzie dozwolone tworzenie wersji zmodyfikowanej tylko wtedy, gdy będzie można tworzyć poprawki do tego programu. W tym przypadku poprawki będą stanowiły niezależne jednostki (produkty): OSS1 patche (do OSS1). Natomiast udostępniając i tworząc wersję zmodyfikowaną programu wraz z kodem, należy nadać mu inną nazwę po to, aby wyraźnie różniła się od pierwowzoru. Licencja zapewnia też swobodną dystrybucję takiej zmodyfikowanej wersji programu wraz ze zmodyfikowanym kodem źródłowym, lecz pod inną nazwą niż dzieło pierwotne.

Z powodów logistycznych nie zaleca się jednak wykorzystywania tego zapisu przy licencjonowaniu oprogramowania.

5. **Bez dyskryminacji osób i grup (ang. *no discrimination against persons or groups*)**
   Licencja nie może dyskryminować jakichkolwiek osób lub grup społecznych.

6. **Bez dyskryminacji obszarów zastosowań (ang. *no discrimination against fields of endeavor)***
   Licencja nie może ograniczać kogokolwiek co do wykorzystywania oprogramowania w dowolnym obszarze. Na przykład podaje się, że licencja nie może zabraniać wykorzystywania programu w celach komercyjnych, jak również na polu badań genetycznych.

Zapisy punktu 5. i 6. wprowadzono po to, aby zdjąć z licencji możliwość ograniczania wykorzystania kodu źródłowego programu w jakimkolwiek celu i przez kogokolwiek. Innymi słowy, dzięki temu zapewnia się wsparcie potencjalnie dużo większej liczby współpracowników. Jeżeli każdy będzie mógł używać OSS we własnych celach, to jest szansa, iż sprzyjać to będzie zasilaniu ruchu open source nowymi pomysłami. A zatem możliwość dowolnego wykorzystywania kodu źródłowego dla samego kodu oznacza większą elastyczność, niezawodność oraz potencjalnie dłuższe czasowo wykorzystanie.

7. **Dystrybucja licencji (ang. *distribution of license*)**
   Prawo ciążące na programie na mocy licencji obowiązuje wszystkich, którzy otrzymują (i wykorzystują) program, bez konieczności (i możliwości) stosowania się do zapisów prawnych pochodzących z innych, dodatkowych licencji. Oznacza to, że licencja nałożona na program jest jedynym obowiązującym przepisem prawnym, który narzucony jest na to oprogramowanie. Na mocy tego zapisu następnym pokoleniom użytkowników nakazuje się egzekwowanie tego prawa. Innymi słowy dany OSS lub fragment kodu licencjonowany na licencji Y pozostaje na tej licencji i w kolejnych wersjach i modyfikacjach tego programu lub kodu należy stosować tę samą licencję Y. Przypominam w tym miejscu ostatnią kolumnę wcześniej prezentowanej tabeli (por. tab. 1).

8. **Licencja nie może być przeznaczona dla konkretnego produktu (ang. *license must not be specific to a product*).**

   Zapis postulowany w tym punkcie oznacza, że udostępniając dany program na licencji X, wszystkie jego części (wchodzące w skład tego programu jako całości) również muszą być udostępniane na prawach tej samej licencji X. Punkt ten zabezpiecza możliwość wyizolowania z danego OSS fragmentu kodu i udostępnienia go na innej licencji niż ta, która ciążyła na danym OSS.

9. **Licencja nie może ograniczać innego oprogramowania (ang. *license must not restrict other software*).**

   Licencja nie może narzucać konieczności stosowania tych samych zapisów prawnych wobec oprogramowania, które dystrybuowane jest wraz z OSS. Innymi słowy, licencja nie może nakazywać, aby, na przykład, podczas rozpowszechniania danego programu OSS na danym nośniku, wszystkie inne aplikacje umieszczone na tym samym nośniku również były udostępniane na tej samej licencji. Idzie więc o zapewnienie swobody dystrybucji programów w ogóle i o maksymalizowanie dostępności oprogramowania open source. Zapis ten pozwala bowiem wraz z oprogramowaniem własnościowych swobodnie rozpowszechniać oprogramowanie open source, a wręcz do tego zachęca.

10. **Licencja musi być neutralna technologicznie (ang. *license must be technology-neutral*).**

    Zapis ten stworzono po to, aby umożliwić przenoszenie kodu źródłowego programu na inne nośniki. Funkcjonują bowiem takie licencje, które wymagają wykonania np. kliknięcia klawiszem myszki w celu akceptacji warunków tej licencji. Takie praktyki skutecznie ograniczają możliwość transferowania kodu do innej postaci niż cyfrowa, np. do postaci papierowej[229].

Przedstawione powyżej warunki stanowią swoiste wytyczne co do tego, jak licencja, która ma obejmować OSS, powinna być skonstruowana. Na podstawie tych wytycznych można wskazać cechy, jakie program, który

[229] Objaśnienie zapisu OSD dokonałem na podstawie: A.M. S. Laurent, op. cit., s. 9-11; *Definicja oprogramowania open source* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://www.free-soft.org/mirrors/www.opensource.org/docs/osd-polish.php; *The open source definition (annotated)* op. cit.

ma być uznany za program open source, musi spełniać. Poniższa enumeracja koresponduje z powyżej omówionymi wytycznymi definicji open source. Podsumowując, oprogramowanie open source jest to takie oprogramowanie, które można scharakteryzować następującymi zdaniami:

1. Jest swobodnie redystrybuowane (może być sprzedawane lub rozdawane).
2. Jest dostarczane z kodem źródłowym, który można modyfikować.
3. Na bazie oprogramowania można tworzyć modyfikacje i programy pochodne.
4. Może stanowić integralną i niezależną autorsko całość, wyraźnie odróżnioną od kolejnych wersji, modyfikacji i wprowadzonych poprawek (tzw. patchy).
5. Może być swobodnie wykorzystywane przez kogokolwiek.
6. Może być swobodnie wykorzystywane w dowolnym celu.
7. Jest oprogramowaniem, którego nie można relicencjonować.
8. Jest to oprogramowanie, którego wszystkie komponenty są również open source'owskie. Innymi słowy, oprogramowanie jako całość oraz jako zbiór komponentów składających się na to oprogramowanie są oprogramowaniem open source.
9. Jest oprogramowaniem, które można rozpowszechniać z programami komercyjnymi i *vice versa*.
10. Jest oprogramowaniem, którego kod źródłowy można utrwalać na różnych nośnikach.

* * *

OSI została założona po to, aby uporządkować rynek OSS. Wprowadziła przejrzyste kryteria i pilnuje, aby każdy mógł uzyskać potwierdzenie, czy dany program jest programem OS. Mimo to pewien margines programów pretendujących do miana OS zupełnie nie przystaje do tej kategorii (brak licencji, brak spójności licencji z OSD). Dowodzi to konieczności prowadzenia stałej akcji informacyjnej skierowanej na wyjaśnianie, czym jest oprogramowanie open source i jakie warunki dany program musi spełniać (zachowanie i przestrzeganie standardów OSD), aby mógł zostać uznany za program open source.

3. OPROGRAMOWANIE OPEN SOURCE A OPROGRAMOWANIE WŁASNOŚCIOWE

W tej części pracy uwagę poświęcę porównaniu oprogramowania open source z oprogramowaniem własnościowym. Ponadto przybliżę funkcjonujące w społecznym dyskursie główne mity dotyczące tego oprogramowania, które automatycznie przeciwstawiane są oprogramowaniu komercyjnemu.

Zanim do tego przejdę przypomnę jeszcze, że oprogramowanie można podzielić ze względu na sposób jego upowszechniania. Dla przykładu, vaporware oznacza taki rodzaj oprogramowania, który został już zareklamowany i ustalono datę jego upublicznienia, lecz produkt wciąż nie jest gotowy, choć ustalone terminy zostały już przekroczone. Shareware i freeware są bardzo podobne do siebie. Programów typu shareware można używać przez określony czas, po którym za dalsze używanie aplikacji powinno się uiścić określoną opłatę. Freeware z kolei rozpowszechniany jest za darmo (np. tylko do użytku domowego, niekomercyjnego), jednak kod źródłowy aplikacji nie jest dostępny. A zatem terminu *freeware* nie można utożsamiać z terminem *free software*, a tym samym i z *open source software*[230]. Aplikacje typu vaporware, shareware i freeware wchodzą w zakres pojęcia oprogramowania komercyjnego (własnościowego, PS).

[230] Zaprezentowane terminy i ich znacznia na podstawie: R. Dixon, op. cit., s. 4.

3.1. PORÓWNANIE OPROGRAMOWANIA OPEN SOURCE Z OPROGRAMOWANIEM WŁASNOŚCIOWYM

Przeciwieństwem OSS jest oprogramowanie własnościowe (ang. *proprietary software*, skrót PS). To drugie, co zrozumiałe, rozwijane jest przeważnie przez konkretne firmy w celu wypracowywania zysków finansowych. Choć OSS rozwijane jest, w większości przypadków, niezależnie od wpływów komercyjnych, to może się zdarzyć, że firmy komercyjne również pracują nad OSS w celu tworzenia odpowiednich narzędzi służących społeczności internetowej. Innymi słowy, internauci tworzą takie aplikacje, które im są najbardziej potrzebne i nie są przez to ograniczani przez podaż. W przypadku rozwiązań komercyjnych, jak się zdaje, obowiązuje zasada, że podaż kształtuje popyt, a w przypadku open source jest odwrotnie – to popyt kształtuje podaż. Wśród głównych cech różniących PS i OSS można wskazać takie, że kopiowanie i instalowanie programu na komputerze bez uiszczenia opłaty jest naruszeniem prawa (PS), gdy tymczasem zastosowanie tej samej procedury dla OSS czyni program bardziej wartościowym z powodu tzw. efektu sieci (ang. *network effect*). W przypadku PS nikt poza twórcami nie wie, jak program działa. W przypadku OSS każdy może sprawdzić, jak program działa i zaproponować modyfikacje. Komunikacja marketingowa na osi twórca aplikacji – klient przybiera postać: „nie musisz wiedzieć, jak to działa, masz jedynie wiedzieć, co zapewnia program" (PS) lub „możesz się dowiedzieć, jak dane narzędzie działa" (OSS). Wykorzystując urządzenia do pracy z oprogramowaniem PS konieczne jest używanie nowoczesnych komputerów o aktualnych parametrach i standardach. Oprogramowania OSS można używać na komputerach starszych i tańszych. PS jest też przeważnie oprogramowaniem dużym, zapewniającym możliwość wykonywania wielu zadań po to, aby sprostać wymaganiom różnych użytkowników, podczas gdy OSS charakteryzuje się tym, że poszczególne komponenty aplikacji służą określonym celom. Dzięki temu internauci mogą wedle własnych potrzeb doinstalowywać konkretne komponenty. Uwaga dotycząca „rozmiarów" oprogramowania tyczy się zwłaszcza systemów operacyjnych. Dziś już coraz rzadziej obowiązuje często stawiany dawniej podstawowy zarzut wobec OSS, że oprogramowanie to jest trudne w użytkowaniu. Wyjaśniano to w ten sposób, że pisane było ono przez programistów dla innych programistów. Ponadto interfejs był ubogi, a OSS nie było wzbogacane

o szczegółowe instrukcje obsługi, co znacznie utrudniało pracę z aplikacją. Przyznać należy, że ideą open source jest to, aby oprogramowanie działało bardzo dobrze i dopiero na takim fundamencie buduje się wizerunek i tworzy odpowiednie opakowanie: najpierw funkcja (treść), potem pudełko (forma), a zupełnie na końcu reklama. Wydaje mi się, że w przypadku PS droga ta przebiega niekiedy w odwrotnej kolejności: dobra reklama, forma, logo, opakowanie, a potem treść, którą dobudowuje się do stworzonej lub faktycznej potrzeby. Natomiast etapy wprowadzania samego programu do obiegu społecznego w obydwu przypadkach (PS, OSS) są podobne. Najpierw tworzy się wersję alfa programu i rozpowszechnia ją wśród użytkowników, którzy wskazują określone błędy i wady programu. Następnie tworzy się wersję beta, również zawierającą liczne niedociągnięcia, które również się poprawia. Wreszcie pojawiają się wersje finalne programów, aż któraś z nich zostanie zaakceptowana jako ostateczna (na tym etapie odpowiednia). Zasadnicza różnica natomiast ujawnia się w interfejsie. Pierwsze wersje oprogramowania PS odznaczają się ładnym i bardziej dopracowanym interfejsem niż analogiczne wersje oprogramowania OSS, które charakteryzują się interfejsem dużo skromniejszym. Przykładanie wagi do detali formalnych może sugerować dla kogo i w jakim celu tworzy się daną aplikację[231].

Dyskutując różnice między aplikacjami OSS i PS można poruszyć jeszcze jedną kwestię: czy działalność realizowana w duchu open source jest działalnością innowacyjną czy imitacyjną? Odpowiedź wskazuje po części na obydwie formy. Z jednej strony OSS powstaje jako imitacja (w nawiązaniu) PS, kiedy programiści rozpoczynają tworzenie programu, który funkcjonalnością będzie spełniać zadania określonego programu PS. Utworzonego odpowiednika open source będzie można natomiast używać darmowo. Takie zresztą były początki OSS. Dość wspomnieć o projekcie GNU Emacs oraz o projekcie systemu operacyjnego GNU działającego jak UNIX, choć w zasadzie wcześniej już tworzono programy komputerowe jako open source'owskie. Z drugiej jednak strony w ruchu open source tkwi ogromny potencjał innowacyjności, który ujawnia się zwłaszcza w obszarze sieciowych technologii informacyjnych. Wiele technologii służących komunikacji sieciowej ma podłoże open source'owskie[232].

---

[231] Informacje zawarte w akapicie powstały w oparciu o pracę: J. Locke, op. cit., s. 7-9.

[232] Przygotowując akapit posiłkowałem się pracą: F. P. Deek, J. A. M. McHugh, *Open source technology and policy,* Cambridge 2008, s. 7.

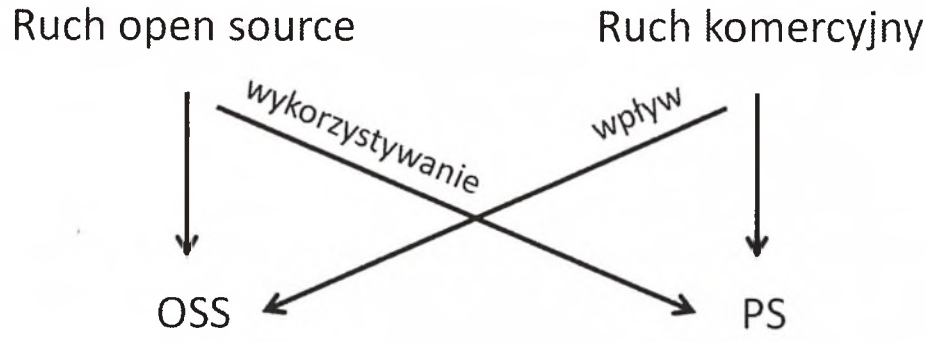
W tym miejscu wypada odnieść się do badań przeprowadzonych w lipcu 2005 roku na próbie 500 projektów z bazy Source Forge o najwyższym poziomie aktywności, czyli najpopularniejszych i dynamicznie wówczas rozwijanych. Analizy miały na celu m.in. wykazanie, czy projekty te są faktycznie innowacyjne. Okazało się, że tylko 64 projekty (12,8%) nie były bezpośrednimi imitacjami rozwiązań już istniejących. Natomiast pozostałe w różny sposób nawiązywały do rozwiązań już funkcjonujących. Wskazano również, że open source jest często automatycznie utożsamiane z innowacją, a OSS niemalże zawsze są tak etykietowane. Wzięło się to z przekonania, że metoda rozwoju oprogramowania stosowana na potrzeby open source sprzyja adaptowaniu rozwiązań innowacyjnych, podczas gdy metoda rozwoju programów zamkniętych jest z tym sprzeczna. Wyniki analiz dowiodły, że OSS wcale nie zawsze są czymś nowym, w większości przypadków naśladując już funkcjonujące narzędzia informatyczne[233]. Niezależnie od tych badań sprzed blisko dekady, dziś widać zaangażowanie w ruch open source przedsiębiorców, dawniej utożsamianych tylko z działalnością związaną z rozwijaniem programów zamkniętych. „Open source ma szczególne zastosowanie w tych gałęziach przemysłu, w których duże znaczenie ma innowacyjność"[234].

Mówiąc o potencjale można odnieść się do idei self publishingu, która zbieżna jest do pewnego stopnia z ideą open source. Twórca-amator, prosument, ktoś, kto udostępnia treści nieodpłatnie w sieci, dodatkowo pozwalając innym je modyfikować (przeważnie z zastrzeżeniem uznania autorstwa), może stworzyć coś tak oryginalnego, co nie było do tej pory obecne w głównym nurcie. Może to być nowy, ciekawy sposób pisania prozą, może to być początek nowego prądu, trendu w sztuce, może to być wreszcie program komputerowy, który będzie wykonywał zadania dotąd przez inne programy niewykonywane i cechował się oryginalną funkcjonalnością. Nie będzie przesadą stwierdzenie, że z nurtu open source pewne pomysły przenikają do obiegu komercyjnego. Mam na myśli nie tylko fakt, że niektóre programy OSS są nawet sprzedawane, uzyskują atrakcyjne opakowanie i są odpowiednio reklamowane, lecz także to, że tradycyjne, by tak rzec, programy własnościowe korzystają z pomysłów i rozwiązań, które zrodziły się w obszarze ruchu open source, a niekiedy i z samego kodu

[233] Przedstawione wyniki badań zaprezentowano w pracy: K. Klincewicz, *Innowacyjność projektów open source*, „Studia i Materiały" 2006, nr 2, s. 7-13.

[234] P. Zarzycki, *Udostępnianie oprogramowania w modelu open source w branży elektroniki użytkowej*, „Zarządzanie Zmianami. Zeszyty Naukowe" 2010, nr 4, s. 36.

open source'owskiego. W dobie społecznych mediów Internetu drugiej generacji Web 2.0, gdzie komunikacja może przebiegać wielokierunkowo, pomysły i rozwiązania krążą od OSS do PS i od PS do OSS. Dowodzić to może jedynie konieczności dalszego stymulowania procesów swobodnej wymiany informacjami i wiedzą, które przyczyniają się do kształtowania lepszego jutra. Kolejny schemat ilustruje ten proces (rys. 5).



Rys. 5. Zależność i wpływ ruchu open source i ruchu komercyjnego na powstawanie oprogramowania OSS i PS [opracowanie własne].

Celem działania ruchu open source jest przede wszystkim tworzenie oprogramowania OSS, natomiast celem działania ruchu, który określiłem jako komercyjny, jest tworzenie oprogramowania PS. Pierwszy działa na rzecz idei otwartości, drugi – dla rezultatów finansowych. Dziś można z pewnością stwierdzić, że na osi OSS – PS oraz PS – OSS ciągle dochodzi do interferencji, jednakże przybierają one, w zależności od wektora, różne formy. Wektor wskazuje zasadniczo na dwie skrajne możliwości. Często OSS tworzy się nawiązując do funkcjonalności znanej z programów PS. Innymi słowy, programiści budują też takie OSS, które odpowiada pod względem pełnionych funkcji i realizowanych zadań oprogramowaniu PS. Tworzą więc darmowe odpowiedniki programów komercyjnych. Nie mogą natomiast wykorzystywać kodu, który z jednej strony nie jest dostępny, a z drugiej chroni go prawo zabraniające dokonywania w nim jakichkolwiek modyfikacji. Z kolei PS tworzy się wykorzystując nie tylko same pomysły, idee, koncepcje programów proponowanych w przestrzeni OS, lecz także fragmenty lub całe kody źródłowe, wytworzone w ramach działalności open source. Stosowne licencje pozwalają na swobodne wykorzystywanie kodu aplikacji open source, który może zasilać kod aplikacji komercyjnej. W tym przypadku znacznie przyspiesza się efekt końcowy, skoro gotowe fragmenty kodu są dostępne i możliwe

do zaimplementowania do kodu całej aplikacji PS. Można zatem postawić retoryczne pytanie: czyżby była to forma kompromisu między obydwoma nurtami? Obydwa modele czerpią od siebie rozmaite pomysły na aplikacje (zresztą w końcu ci sami – o takim samym wykształceniu, wiedzy i doświadczeniu – informatycy tworzą oprogramowanie), a nierzadko i zasilają niezależnie obydwa obszary. Jest to więc swoista ugoda. Jedni wobec drugich spłacają w specyficzny sposób (pomysł, koncepcja, kod) swoje zobowiązania. Niezależnie od pobudek i motywów, niezależnie od ewentualnych niepisanych umów, faktem pozostaje, że OSS i PS wzajemnie się stymulują, dzięki czemu zapewniony jest permanentny rozwój oprogramowania w ogóle.

Informatycy zwracają uwagę na fakt rozróżniania dwóch postaw cechujących osoby pracujące na rzecz któregoś z nurtów tworzenia programów. Nurt OSS określa się jako altruistyczny, ponieważ programiści pracują dla idei, a nie dla wynagrodzenia (na pewno nie bezpośrednio). Natomiast nurt PS określa się jako egoistyczny, ponieważ programiści go zasilający pracują jedynie dla zysku materialnego (bezpośrednio). Patrząc na ten fakt inaczej, można stwierdzić, że OSS przyczynia się bezpośrednio do rozwoju oprogramowania, natomiast PS – pośrednio. Wracając do centralnej kwestii różnicującej oprogramowanie tych dwóch nurtów, należy zaznaczyć, że tworzący OSS uzyskują pośrednio inne korzyści. Wśród nich znajdują się takie, jak lepszy wizerunek, szersza akceptacja społeczna, co przekłada się również na ich sytuację na rynku. Programiści zaangażowani w ruch open source w późniejszym czasie lepiej zarabiają. Ich obszar działalności w zakresie OSS staje się dowodem ich talentu, umiejętności, wizytówką ich zdolności, słowem – portfolio dla przyszłego pracodawcy. OSS jest w konsekwencji nie tylko tańszym, lecz także bardziej produktywnym nurtem tworzenia oprogramowania. Produktywność OSS zawdzięcza m.in. większym potencjalnie obszarom zastosowania, programiści OSS posiadają dostęp do większej wiedzy, wreszcie nie ma konieczności powielania prac, każdy bowiem może je stale po swojemu rozwijać i nie musi nic duplikować. W inicjatywy OSS inwestuje się znacznie więcej pracy, stąd i źródeł pomysłów jest znacznie więcej, co owocuje efektywniejszymi rozwiązaniami. OSS zdecydowanie dorównują programom PS, a może być też tak, że obydwie grupy programów są równoważne. Trudno jest bowiem jednoznacznie wskazać, który program jest zdecydowanie lepszy od którego (przy założeniu, że obydwa działają poprawnie). Na takie oceny wpływają niekiedy indywidualne, subiektywne względy. Niewątpliwie

natomiast OSS są programami bardziej pożądanymi z powodu niskich kosztów, jakie musi ponieść użytkownik, by je uzyskać[235].

Kończąc ten podrozdział warto wypunktować i zestawić wady i zalety oprogramowania open source (tab. 3) oraz wymienić te właściwości oprogramowania, które warto brać pod uwagę przy wyborze OSS (tab. 4)[236].

Tabela 3

ATRYBUT	ZALETY	WADY
Cena	W większości przypadków brak opłat za oprogramowanie, licencję i konkretne wdrożenie.	Brak ceny oznacza równocześnie dewaluację oprogramowania oraz pracy programisty w myśl zasady: „jeśli coś jest za darmo, to jest gorsze". Pokutuje tu również pogląd, że na oprogramowaniu open source nie można zarobić, przez co wielu użytkowników traci motywację do jego rozwijania i chęć do angażowania się w pracę nad nim.
Konkurencyjność	Brak ceny oznacza, że OSS jest od razu bardziej pożądanym towarem niż PS.	Obawa ze strony użytkownika, że darmowy program jest gorszy.
Ilość aplikacji	Ogromny wybór OSS, różnych wersji, modyfikacji, poprawek itp.	Ogromny repertuar OSS sprawia, że trudno jest odnaleźć pożądane aplikacje (wrażenie nadmiaru programów). Może nigdy nie powstać poprawnie działająca wersja aplikacji.

[235] W akapicie wykorzystałem informacje pochodzące z pracy: S. Koch, *Free/open source software development,* Hershey 2005, s. 267-269.

[236] W sieci można znaleźć wiele różnego typu porównań odnoszących się zwłaszcza do systemów operacyjnych. Każdy zainteresowany bez trudu odnajdzie tego typu opracowania. Dla przykładu odsyłam do dwóch: L. Krakowiak, *Który system bezpieczniejszy – Linux czy Windows?* [online], [dostęp: 8.08.2013]. Dostępny w WWW: http://www.idg.pl/news/360790/ktory.system.bezpieczniejszy..linux.czy.windows.html; Tegoż, *Ubuntu zamiast Windows. Dzień 4 – nie ma Photoshopa* [online], [dostęp: 8.08.2013]. Dostępny w WWW: http://www.pcworld.pl/news/372518/Ubuntu.zamiast.Windows.Dzien.4..nie.ma.Photoshopa.html.

Licencje	Dużo różnych licencji pozwala wybrać sobie (lub stworzyć nową) idealnie dopasowaną do indywidualnych potrzeb.	Za dużo licencji, przez co użytkownik nie do końca wie, z jakim rodzajem programu ma do czynienia.
Społeczność	Wiele idei, pomysłów, koncepcji przyczynia się do rozwoju OSS.	Wielość pomysłów wprowadza chaos i zamieszanie.
Gwarancja	Duża społeczność programistów jest gwarantem poprawności działania programu.	Brak gwarancji poprawności działania danej aplikacji.
Dostępność kodu	Każdy może zmodyfikować kod dla własnych potrzeb. Większa wykrywalność błędów.	Możliwość wykorzystania kodu przeciw przyszłemu użytkownikowi.
Bezpieczeństwo	Dużo wersji OSS sprawia, że mają one *de facto* mały zasięg, przez co są mniej podatne na ataki.	Słabsze wsparcie techniczne znacznie spowalnia proces eliminowania błędów i ataków.
Interfejs	Każdy może mieć wpływ na funkcjonalność interfejsu.	Interfejs mniej dopracowany, zwłaszcza w początkowych fazach.

Zestawienie zalet i wad oprogramowania open source [opracowanie własne].

Tabela 4

KRYTERIUM	UZASADNIENIE
Reputacja	Należy zapoznać się z opiniami użytkowników danego oprogramowania i porównać je z opiniami formułowanymi dla własnościowych odpowiedników programu. Dzięki temu można dowiedzieć się, które oprogramowanie jest faktycznie lepsze.
Rozwój	Należy sprawdzić czy dany program jest rozwijany od jakiegoś czasu, czy powstają nowe wersje, poprawia się błędy, a tym samym – czy istnieje szansa, że aplikacja będzie w przyszłości rozwijana.
Normy	Należy wybierać takie oprogramowanie, które implementuje otwarte standardy oraz dobrze współpracuje z różnymi aplikacjami (OSS, PS). Ważne jest także to, aby program nie wprowadzał nowych formatów plików, a jego modyfikacja nie wymagała wcześniejszej nauki nowego języka programowania.

Wsparcie społeczności	Należy sprawdzić czy dany program ma duże wsparcie społeczności, czy istnieją listy dyskusyjne (z dostępnym archiwum), czy istnieje możliwość wymiany informacjami. Oczywiście warto pamiętać o tym, aby samemu również udzielać informacji innym.
Wersja	Należy sprawdzić, kiedy powstała ostatnia stabilna wersja, czy modyfikacje programu też są poprawiane, czy istnieje aktywne wsparcie społeczności.
Dokumentacja	Należy sprawdzić czy program posiada dokumentację rozwoju, tj. powinny być dostępne informacje o tworzeniu nowych wersji, o tworzeniu poprawek, o usuwaniu błędów itp.
Umiejętności	Należy rozważyć, czy własne umiejętności (np. programistyczne, użytkowania) są wystarczające do tego, aby tej aplikacji używać samemu lub w zespole, czy też trzeba będzie zabiegać o pomoc innych.
Model rozwoju aplikacji	OSS powinno mieć, jak każde oprogramowanie, jasno ustalony porządek rozwoju, mimo że może wyglądać to nieco chaotycznie.
Licencje	Należy zapoznać się z licencją, na której udostępniany jest program i sprawdzić, czy jej treść odpowiada potrzebom, dla których program będzie wykorzystywany[237].
Funkcjonalność	Należy sprawdzić funkcje, jakie program pełni i ustalić czy faktycznie odpowiadają one tym, które są przez użytkownika poszukiwane.
Proces wdrażania	Należy ustalić, na ile i jak bardzo implementacja programu będzie czasochłonna, pracochłonna i kosztowna.
Liczba ściągnięć	Należy sprawdzić, czy program był wielokrotnie ściągany, co może oznaczać, że jest funkcjonalny oraz dopracowany. Te cechy przekładają się na popularność programu.

Kryteria wyboru programu [opracowanie własne][237].

[237] Kryteria (poza *funkcjonalność, proces wdrażania, liczba ściągnięć*) i ich eksplikacje przywołałem *na postawie*: R. Metcalfe, *Top tips for selecting open source software* [online], [dostęp: 3.06.2012]. Dostępny w WWW: http://www.oss-watch.ac.uk/resources/tips.xml.

3.2. MITY NA TEMAT OPEN SOURCE SOFTWARE

W opracowaniach dotyczących OSS pojawiają się niekiedy wskazania funkcjonujących w tym obszarze mitów. W tym miejscu niektóre z nich wskażę i omówię.

- Jeden z głównych mitów mówi, że OSS to coś przeciwnego rozwiązaniom komercyjnym. Innymi słowy, jeżeli ktoś decyduje się na wykorzystanie OSS to musi całkowicie przejść na rozwiązania open source, nawet system operacyjny musiałby zmienić. Jest to oczywiście nieprawdą, istnieje bowiem szereg aplikacji OSS, które z powodzeniem można uruchamiać na komputerach wyposażonych w systemy operacyjne PS.
- Następny mit głosi, że oprogramowanie FLOSS rozwijane jest głównie przez wolontariuszy w ich wolnym czasie. Tym samym sugeruje się, że bez wsparcia finansowego nie sposób stworzyć dobrze funkcjonującego oprogramowania. Z jednej strony nie jest to prawdą, gdyż mogą powstawać stabilne programy bez wsparcia finansowego. Z drugiej zaś strony w ruch open source inwestuje się znaczne sumy pieniędzy, czynią to zarówno komercyjni przedsiębiorcy, jak i niezależni sponsorzy.
- Z tym wiąże się kolejny mit, tj. ten, zgodnie z którym OSS nie dorównuje PS. Już sam fakt, że znaczny odsetek pracujących na świecie serwerów działa w oparciu o systemy OS, świadczy przeciw temu mitowi.
- Duże firmy z branży IT nie interesują się oprogramowaniem FLOSS, nie inwestują w nie i go nie używają. Ten sam przykład z serwerami obala ten mit.
- Kolejny mit sugeruje, że FLOSS jest zagrożeniem dla własności intelektualnej, ponieważ w jego ramach wszystko trzeba oddać innym za darmo. Podczas omawiania popularnych modeli definicji open source wyjaśniłem, że to tak naprawdę jedynie od woli twórcy zależy, jakiej licencji użyje do tworzonej przez siebie aplikacji, a przez to to on *de facto* decyduje, jak i w jakim stopniu chronić swoje prawa. Przypominam, że każda licencja open source zabezpiecza prawa autorskie do produktu. Nikt nie może przypisywać sobie autorstwa programu OSS, którego nie zrobił.
- Funkcjonuje też mit głoszący, że po rozpoczęciu projektu OSS, tj. przekazaniu części lub całości kodu źródłowego społeczności

sieciowej, społeczność ta zacznie automatycznie pracować za darmo na rzecz przekazującego. Nie jest to prawdą, nie wystarczy bowiem rozpocząć projekt i czekać aż reszta internautów wykona całą pracę, lecz trzeba przede wszystkim zapewnić dobrą komunikację wokół programu. Po prostu wszystkie strony biorące udział w danym projekcie OSS muszą coś zyskiwać dzięki tej *nomen omen* współpracy.

- Utrwalił się też i taki mit, że OSS ma znaczenie tylko dla programistów. Natomiast nieprogramiści na niewiele się mogą przydać przy realizowaniu open source'owskich przedsięwzięć. Z kilku powodów nie jest to prawdą. Po pierwsze, użytkownik może przecież zapłacić komuś innemu za wprowadzenie stosownych poprawek i modyfikacji do programu, którego on chce używać. Po drugie, zasilać ruch OS można też m.in. poprzez dokonywanie tłumaczeń instrukcji czy podręczników programów itp. Po trzecie wreszcie, OSS może funkcjonować w ogromnej liczbie wersji, co oznacza, że wielu użytkowników będzie miało odpowiednie narzędzia i, co za tym idzie, sprzyjać to będzie redukcji kosztów aplikacji, na czym znowu skorzysta użytkownik końcowy.
- Inny popularny mit to ten, zgodnie z którym twierdzi się, że na OSS nie można zarabiać. W rozdziale dotyczącym licencji (1.3) wyjaśniłem, że wcale tak nie jest. Od rodzaju zastosowanej przez twórcę licencji zależy jedynie możliwość lub nie ewentualnej sprzedaży programu open source.
- Ruch open source nie jest trwały i jak tylko społeczność zauważy, że inni zarabiają pieniądze na ich pracy, w każdej chwili może przestać się rozwijać. Na wcześniejszych stronach książki wyjaśniłem, że ruch open source jest fenomenem społecznym, który rządzi się swoimi prawami i nastawiony jest na realizację rozlicznych celów. Poza tym cały ruch oprogramowania ma swe podstawy open source'owskie i ten choćby fakt deprecjonuje ten mit.
- Mitem jest też to, że OSS kopiuje jedynie pomysły z obszaru PS. W poprzednim rozdziale (3.1) obalono ten mit, wyraźnie wskazując, że interferencje zachodzą obustronnie[238].

[238] Przy wyliczeniu i omówieniu dotąd wskazanych mitów na temat oprogramowania open source posiłkowałem się pracą: C. Daffara, *The small/medium enterprises guide to open source software*, ed. 4., 2009 [online], [dostęp: 28.05.2012]. Dostępny w WWW: http://smeguide.conecta.it/smeguide.pdf, s. 11-18.

- Oprogramowanie open source oznacza głównie Linuxa, co jest zupełnie nieprawdą, jeśli spojrzeć na ilość aplikacji rozpowszechnianych jako open source'owskie.
- Open source jest takim samym rodzajem oprogramowania, jak oprogramowanie komercyjne. To z kolei nie jest prawdą z takich powodów, że OSS jest zupełnie inaczej tworzone, inaczej rozpowszechniane i ma zapewniony zupełnie inny rodzaj wsparcia technicznego[239].
- OSS jest „wrogiem" własności intelektualnej. Nie jest to prawdą, gdyż każdy twórca OSS może wybrać taki rodzaj licencji, jaka najbardziej mu odpowiada (i jaka jest w danym przypadku dozwolona).
- W przypadku OSS najważniejsze są licencje. Rzeczywistość natomiast pokazuje, że najważniejsze jest oprogramowanie, a licencje stanowią istotne uzupełnienie danego open source'owskiego projektu.
- Zgodnie z innym mitem, projektami OSS programiści starają się gonić ruch oprogramowania własnościowego. W praktyce jednak w obszarze OSS powstaje duża gama programów nie mających wcześniejszych odpowiedników komercyjnych[240].
- Open source jest darmowe – to kolejny mit. Oprogramowanie nie jest bowiem darmowe, tak naprawdę każdy program posiada swoją cenę (koszt, który ktoś musi ponieść). Koszt, w tym przypadku, oznacza czas i pracę na przygotowanie i poprawianie programu, a także czas i pracę po stronie użytkownika końcowego, który np. musi włożyć wysiłek w poznanie aplikacji i dostrojenie jej do własnych potrzeb.
- OSS jest czymś, co można zrobić samemu. Stwierdzenie to nie jest prawdziwe, gdy chce się wykonać średnie i duże projekty. W takich sytuacjach potrzeba zbyt dużej wiedzy (trzeba znać dużo rozmaitych alternatywnych rozwiązań, które w różnych sytuacjach mogą zadziałać) i dużej ilości kodu (wiele partii kodu już funkcjonuje na licencjach open source, więc można z nich korzystać, co w konsekwencji pozwala zaoszczędzić dużo czasu).

---

[239] Dwa wskazane tutaj mity przywołałem z pracy: B. Golden, *Succeeding with open source*, Boston 2004, s. XXIV-XXVI.

[240] Charakterystyka kolejnych mitów powstała w oparciu o pracę: T. O'Reilly, *Ten myths about open source* [online], [dostęp: 2.01.2013]. Dostępny w WWW: http://www.oreillynet.com/pub/a/oreilly/opensource/news/myths_1199.html.

- Stosowanie OSS jest bardziej ryzykowne niż oprogramowania komercyjnego. Praktyka pokazuje jednak, że aplikacje OS są masowo wykorzystywane, np. serwery Apache, które, jak pokazują statystyki, od 1996 roku są najpopularniejszymi na świecie[241]. A to sugeruje już coś przeciwnego – serwery bowiem muszą działać niemalże niezawodnie[242].

Zapewne dużo więcej tego typu mitów można by jeszcze wskazać. Te zdają się najpowszechniejsze, stąd tylko je przytoczyłem. Oprogramowanie open source posiada ogromny potencjał i, jak pokazuje rzeczywistość sieciowo-cyfrowa, sektor ten nieustannie jest rozwijany, na czym – zdaje się – korzystają internauci.

* * *

Funkcjonujące w społecznym dyskursie, podtrzymywane i powielane przez adwersarzy tego modelu przekonania o negatywnych aspektach ruchu i oprogramowania open source dowodzą, jak istotny jest jego dalszy rozwój. Nawet gdyby część z tych ujemnych not znajdowała potwierdzenie, to dla zwolenników OS mogłoby to oznaczać jedynie konieczność zmieniania (naprawiania) tego stanu rzeczy. Tak jednak do końca nie jest. Mity pozostają mitami. Pejoratywny wydźwięk towarzyszący OS stanowi raczej próbę zahamowania tego społecznoinformatycznego procesu, jakim jest woluntarne wytwarzanie programów OS (ciągów kodu). *Summa summarum* deprecjonowanie na nic się zdaje, skoro część firm komercyjnych zaczyna przyłączać się do tego nurtu.

Jak istotną umiejętnością jest dziś programowanie dowodzi fakt, że w niektórych państwach, które dostrzegły już ten potencjał, wprowadza się lekcje programowania do bloku zajęć szkolnych. A zatem kolejne dekady powinny przynieść jeszcze obfitsze plony OSS, skoro pojawią się całe pokolenia wyposażone w takie umiejętności. Co zrozumiałe znaczna część z tych osób będzie zasilać nurt OS.

---

[241] Informacja dot. wykorzystywania od 1996 roku serwerów Apache: Por. *Netcraft* [online], [dostęp: 2.01.2013]. Dostępny w WWW: http://news.netcraft.com/.

[242] Kolejne mity omówiłem na podstawie artykułu: J. D. Campbell, T. Kreidl, D. Pace, op. cit., s. 6-8.

4. BAZY OPROGRAMOWANIA OPEN SOURCE[243]

W Internecie próżno szukać, jak na razie, w miarę aktualnych baz (katalogów, list) oprogramowania open source, które rejestrowałyby wszystkie dostępne (*in corpore*) lub chociaż większość funkcjonujących aplikacji OSS. W rezultacie programistom, którzy tworzą OSS, przysparza wiele trudności umieszczenie własnego produktu w przestrzeni odpowiedniej strony po to, aby inni internauci mieli do tego programu swobodny dostęp. Brak odpowiednich serwisów bazodanowych może wynikać po części z małej wciąż popularności OSS (wśród osób postronnych, niezaangażowanych w inicjatywy OS), a także z nieznajomości definicji OSS, co utrudnia kwalifikowanie programu do grupy OSS. Inną kwestią jest to, że programy o otwartym kodzie dostępne są przeważnie za darmo, co oznacza, że w popularnych internetowych katalogach serwisów aukcyjnych, sklepów internetowych, porównywarkach cenowych programy te nie są odnotowywane. Zasadniczo nie są też wyposażane w atrakcyjne wizualnie opakowania, co dla obiektów cyfrowych ma również znaczenie (vide okładki książek elektronicznych). Natomiast aplikacje PS w wymienionych wyżej miejscach są dobrze skatalogowane, tworzy się nie tylko zestawienia, lecz także profesjonalne recenzje mające na celu zachęcenie do sięgnięcia po te programy. Innymi słowy, powszechnie się je reklamuje i promuje w rozlicznych serwisach sieciowych, podczas gdy OSS, o czym trzeba też pamiętać, jest ruchem (fenomenem) społecznym, któremu przyświecają inne ideały i cele niż sferze PS. Wśród serwisów zawiera-

[243] Wszystkie strony WWW podane w tym rozdziale były dostępne w maju 2013 roku.

jących informacje o aplikacjach OSS wiele duplikuje nazwy popularnych programów. Wiele innych aplikacji jest cały czas *in statu nascendi* lub powstałe wersje są niestabilne i nie działają prawidłowo. Niekiedy również wraz z aplikacjami OSS informuje się o aplikacjach PS. Idealną natomiast sytuacją byłoby, aby określona baza zawierała hiperłącza i informacje tylko i wyłącznie na temat OSS[244]. Wymienione powyżej oraz inne jeszcze powody wpływają na fakt, że bardzo trudno dotrzeć do kompletnych baz oprogramowania open source.

Jedną z pierwszych list programów open source stworzył w 2003 roku David A. Wheeler[245]. Po poprawkach wskazał w sumie 41 aplikacji[246]. Od tego czasu zaczęto podejmować próby opracowywania stosownych katalogów, a także metakatalogów, tj. katalogów katalogów OSS. Mimo to nierozwiązanym, jak dotąd, problemem pozostaje skuteczne poszukiwanie odpowiednich aplikacji, gdyż tak naprawdę trzeba po prostu przeszukiwać całe bazy krok po kroku. Nie ma technologii skracających ten proces. Pozostaje jedynie czasochłonne wertowanie katalogu za katalogiem, projektu za projektem. Wyszukiwanie programu open source dodatkowo utrudnia fakt, że każda baza (katalog, lista) posiada własne filtry, własne kategorie i operuje własnymi opcjami wyszukiwawczymi[247]. Nie istnieją więc ujednolicone zasady klasyfikowania i opisywania aplikacji OSS[248].

Na wstępie konieczne jest również rozróżnienie pomiędzy katalogiem (bazą, listą) aplikacji OSS a forgem, czyli kuźnią projektów OSS. Pierwszy zawiera (powinien zawierać) tylko zestawy rożnych programów open source, drugi natomiast powinien również umożliwiać współpracę nad rozwijaniem oprogramowania oraz inicjowaniem nowych przedsięwzięć. Stąd w stosunku do nich używa się określenia *forge* (pol. *kuźnia*).

---

[244] Por. *How to find FOSS (Free Software and Open Source Software)* [online], [dostęp: 27.05.2012]. Dostępny w WWW: http://how-to.wikia.com/wiki/Howto_find_Free_Software_and_Open_Source_software_(FOSS).

[245] Por. *How to find open source software* [online], [dostęp: 16.05.2012]. Dostępny w WWW: http://sosopensource.com/34.html.

[246] Por. D. A. Wheeler, *Generally recognized as mature (GRAM) OSS/FS programs* [online], [dostęp: 16.05.2012]. Dostępny w WWW: http://www.dwheeler.com/gram.html.

[247] Por. *How to find open source software...* op. cit.

[248] Warto na marginesie podać, że podejmuje się próby tworzenia (i ujednolicania) stosownych metodologii oceny, a co za tym idzie – klasyfikowania OSS. Por. *Open source software assessment methodologies* [online], [dostęp: 15.05.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Open_source_software_assessment_methodologies.

Bazy, które są dostępne, można podzielić na kilka kategorii. Pierwsze, które udostępniają nie tylko sam katalog oraz odpowiednie aplikacje OSS, lecz także przestrzeń do współpracy nad projektami programów, rozwijania projektów, dyskusji i wymiany doświadczeń między programistami oraz zainteresowanymi użytkownikami. Największą taką bazą i jednocześnie forgem (o największej, jak się zdaje, popularności) jest **SOURCE FORGE** (http://sourceforge.net/). Rejestruje ona największą liczbę projektów OSS[249]. W tej kuźni wydzielono następujące kategorie główne, które dalej są dzielone (poniżej kolejne podziały nie zostały uwidocznione)[250]: „Audio i wideo", „Biznes i przedsiębiorstwo", „Komunikacja", „Rozwój", „Dom i edukacja", „Gry", „Grafika", „Nauka i technika", „Narzędzia i bezpieczeństwo", „Zarządzanie systemem".

W tym miejscu konieczne jest wskazanie tzw. *metaforgów* (ang. *meta-forges*, pol. *meta-kuźnie*), czyli katalogów katalogów projektów OSS. Wśród nich należy odnotować OHLOH, FLOSSmole oraz Melquiades. Rolę metaforgu spełnia również SWiK.

Projekt **OHLOH** (http://www.ohloh.net/), czyli katalogprogram, który pozwala wyszukiwać gotowe bądź tworzone projekty OSS, a także osoby w nie zaangażowane. 2 sierpnia 2013 roku w bazie tej indeksowanych było ponad 590 tysięcy różnych open source'owskich przedsięwzięć. Projekty **FLOSSmole** (http://flossmole.org/) oraz **Melquiades** (http://melquiades.flossmetrics.org/) działały podobnie i rejestrowały podobne ilości projektów OSS. Podobnie działał też projekt **SWiK** (http://swik.net/), tj. na stronie głównej figurowała specjalna wyszukiwarka pozwalająca przeszukiwać zarejestrowane projekty według kilku kategorii. Można było zatem wyszukiwać po nazwie własnej, przeznaczeniu (np. game) czy osobie (użytkowniku). Poza tym możliwe było dodawanie własnych projektów, uzupełnianie innych, wymienianie się doświadczeniami itd.

Ciekawy mechanizm wyszukiwawczy zastosowano w bazie produktów OSS **Free Code** (http://freecode.com/). Otóż programy OSS można było odszukiwać poprzez chmurę tagów. Free Code był to największy indeks programów przeznaczonych dla Linuxa (przede wszystkim), Unixa oraz innych systemów. Każdy program był dokładnie opisany i poza nazwą własną, nazwą wersji, hiperłączem kierującym do miejsca, z którego program można było ściągnąć, zawierał także opis historii rozwoju. Zainteresowani programiści mogli również dodawać nowe aplikacje.

[249] Stan na 2 sierpnia 2013 roku.

[250] Stan na 2 sierpnia 2013 roku.

Dużo mniejszymi bazami prezentującymi popularne i funkcjonalne wolne i otwarte aplikacje open source były np. **ePrzegramy** (http://www.eprogramy.net/), **GARA** (http://www.gara.com/free-software/), **Open Source Living** (http://osliving.com/archive/entertainment/), **Open Source Scripts** (http://www.opensourcescripts.com/), **OSload** (http://osload.com/), **Webi** (http://www.webi.org/), **Zwodnik** (http://www.zwodnik.pl/). Bazy te zawierały katalogi indeksujące od kilkudziesięciu do kilkuset stabilnych programów open source, pogrupowanych w różnych popularnych kategoriach. Warto w tym miejscu wskazać **My Open Source** (http://www.myopensource.org/), bazę oprogramowania open source przeznaczonego konkretnie dla systemów komercyjnych.

Wśród zestawień programów OSS nie mogło zabraknąć list stworzonych w przestrzeni **Wikipedii**. Dwa adresy są szczególnie istotne: lista popularnych aplikacji (http://en.wikipedia.org/wiki/List_of_free_and_open_source_software_packages) oraz portal wolnego oprogramowania (http://en.wikipedia.org/wiki/Portal:Free_software). Podobną listę programów open source stworzono w **Sieciowej Encyklopedii Informatyki Helionica** (http://encyklopedia.helion.pl/index.php/Lista_program%C3%B3w_open-source) oraz w **Downloadpedii** (http://downloadpedia.org/Main_Page).

W zasadzie od początku funkcjonowania FSF Fundacja może poszczycić się prowadzeniem katalogu OSS, jakim jest **The Free Software Directory** (http://directory.fsf.org/wiki/Main_Page). Prezentowane aplikacje to różnego rodzaju narzędzia przydatne w codziennej komputerowej pracy. Aplikacje rekomendowane przez FSF przeznaczone są dla systemów GNU/Linux.

Powstają też bazy, których celem jest wskazywanie open source'owskich odpowiedników aplikacji komercyjnych. Jedną z nich był **OSALT** (http://www.osalt.com/). W bazie tej użytkownik mógł odnaleźć omówienia wybranych (popularnych) aplikacji własnościowych oraz wskazanie na open source'owskie odpowiedniki. Wśród wyróżnionych kategorii figurowały: „biznes", „komunikacja", „bazy danych", „development" i „Web development", „nauka i edukacja", „gry", „aplikacje do obróbki graficznej", „Internet oraz technologie sieciowe", „multimedia", „bezpieczeństwo", „aplikacje systemowe". Podobną funkcję spełniał serwis **Get The Free Version** (http://getthefreeversion.com/). Na stronie serwisu podawane były programy OSS, które funkcjonalnością nie odbiegały od swoich własnościowych odpowiedników.

Funkcjonowały też bazy tematyczne, a rejestrowane tam oprogramowanie open source przeznaczone było dla konkretnych grup użytkowników, do konkretnych celów, znajdowało zastosowanie w określonej branży. Jedną z takich baz był **FOSS4Lib**, czyli wolne i otwarte oprogramowanie przeznaczone dla bibliotek (http://foss4lib.org/), inną jest **OSS4LIB** (http://www.oss4lib.org/), inną jeszcze – np. **FOSS4Trans** (http://traduccionymundolibre.com/wiki/Main_Page), czyli baza aplikacji zawierająca różnego rodzaju narzędzia językowe, a więc przeznaczona przede wszystkim dla tłumaczy. Bazą produktów OSS przeznaczonych specjalnie dla biznesu jest **EOS Directory** (http://eos.osbf.eu/start/). Katalog ten indeksuje oprogramowanie bezpłatne, aktualne i przeznaczone dla przedsiębiorców. Dla potrzeb edukacji stworzono np. **Shool Forge** (https://schoolforge.net/). Nazwa nawiązuje wprost do kuźni SOURCE FORGE. Celem działania School Forge jest ułatwienie współpracy pomiędzy niezależnymi organizacjami, które wykorzystują, rozwijają i propagują rozwiązania open source w edukacji.

Gdy użytkownik zainteresowany jest odnalezieniem gier OSS, a ściślej gier FOSS, to powinien, poza przeprowadzeniem kwerend na stronach wymienionych projektów, udać się pod następujące adresy: **LGames** (http://lgames.sourceforge.net/), **Libre Game Wiki** (http://libregamewiki.org/Main_Page), **Caiman** (http://www.caiman.us/); **Liberated Games** (http://www.liberatedgames.com/), **Games for Linux** (http://games.linux.sk/), **The Linux Game Tome** (http://www.happypenguin.org/).

Przedstawione powyżej adresy warto zebrać i przedstawić w postaci tabeli (tab. 5).

Tabela 5

TYP BAZY	NAZWA	ADRES
Kuźnie	SOURCE FORGE	http://sourceforge.net/
Meta-kuźnie	OHLOH	http://www.ohloh.net/
	FLOSSmole	http://flossmole.org/
	Melquiaqdes	http://melquiades.flossmetrics.org/
	SWiK	http://swik.net/

Katalogi, listy, bazy	Free Code	http://freecode.com/
	eProgramy	http://www.eprogramy.net/
	GARA	http://www.gara.com/free-software/
	Open Source Living	http://osliving.com/archive/entertainment/
	Open Source Scripts	http://www.opensourcescripts.com/
	OSload	http://osload.com/
	Webi	http://www.webi.org/
	Zwodnik	http://www.zwodnik.pl/
	My Open Source	http://www.myopensource.org/
	The Free Software Directory	http://directory.fsf.org/wiki/Main_Page
	OSALT	http://www.osalt.com/
	Get The Free Version	http://getthefreeversion.com/
	Wikipedia	http://en.wikipedia.org/wiki/List_of_free_and_open_source_software_packages
	Wikipedia	http://en.wikipedia.org/wiki/Portal:Free_software
	Helionica	http://encyklopedia.helion.pl/index.php/Lista_program%C3%B3w_open-source
	Downloadpedia	http://downloadpedia.org/Main_Page
	FOSS4Lib	http://foss4lib.org/
	OSS4LIB	http://www.oss4lib.org/
	FOSS4Trans	http://traduccionymundolibre.com/wiki/Main_Page
	EOS Directory	http://eos.osbf.eu/start/

Shool Forge	https://schoolforge.net/
LGames	http://lgames.sourceforge.net/
Libre Game Wiki	http://libregamewiki.org/Main_Page
Caiman	http://www.caiman.us/
Liberated Games	http://www.liberatedgames.com/
Games for Linux	http://games.linux.sk/
The Linux Game Tome	http://www.happypenguin.org/

Kuźnie, metakuźnie oraz bazy, listy i katalogi aplikacji open source [opracowanie własne][251].

Niekiedy odpowiednie OSS prezentuje się również w formie książkowej. Tu można wskazać pracę Cona Zymaricsa z 2008 roku, pt. *Free software for schools v8.12*, zawierającą zestawione tematycznie krótkie omówienia kilkudziesięciu wybranych aplikacji OSS, które zdaniem autora mogą znaleźć zastosowanie w działalności edukacyjnej[252]. Indeks około trzydziestu stron WWW zawierających oprogramowanie OSS wskazano też w specjalnym przewodniku dla australijskich agencji rządowych[253].

Gdy mimo wszystko nie odnajdziemy odpowiedniej aplikacji OSS, pozostaje jeszcze skorzystać z uniwersalnego repozytorium wszelkich informacji, tj. World Wide Web. Narzędzia wyszukiwawcze WWW są coraz doskonalsze i tylko od kompetencji internautów uzależnione jest odnalezienie pożądanych aplikacji OSS.

Podsumowując, trzeba przyznać rację Jamesowi Kretchmarowi, że nie każda przestrzeń internetowa zawierająca aplikacje OSS zapewnia również jakiekolwiek wsparcie. Idealnym rozwiązaniem byłoby pobieranie programów ze stron WWW, na których można uzyskać również pomoc przy pracy

[251] Niestety niektóre ze wskazanych stron w maju 2014 roku już nie istniały, tak więc tabela i zaprezentowany rozdział ma już po części wartość historyczną.

[252] C. Zymaris, *Free software for schools v8.12* [online], [dostęp: 25.05.2012]. Dostępny w WWW: http://cc.com.au/files/Free-Software-for-Schools.pdf.

[253] Por. *A Guide to open source software for Australian government agencies* [online], [dostęp: 28.05.2012]. Dostępny w WWW: http://cc.com.au/sites/cc.com.au/files/A_Guide_to_Open_Source_Software.pdf, s. 50.

z danym programem. W wielu jednak przypadkach trzeba nastawić się na to, że tzw. support może być bardzo ograniczony[254]. W przypadku aplikacji komercyjnych, techniczne wsparcie jest zapewnione poprzez opłaty, które wnoszą użytkownicy kupujący te programy. Zrozumiałym jest, że wraz z darmowymi aplikacjami open source nie idzie w parze stała (obligatoryjnie zapewniona) komunikacja z informatykami, którzy nad daną aplikacją pracowali (pracują). Niekiedy więc znalezienie pożądanego oprogramowania jest jednym problemem, natomiast kolejnym jest zrozumienie, jak działa program i poznanie jego pełnej funkcjonalności, co może spadać na barki użytkownika końcowego. Przewagą i zaletą aplikacji wolnych i otwartych jest to, że wszystkie one tworzone są w tzw. otwartych standardach open source. Innymi słowy, jeżeli coś jest programem open source, to jest tak opracowane i przygotowane, że zasady działania programu są transparentne. Nikt celowo nie ogranicza poznania funkcjonalności aplikacji.

* * *

Jak już wspomniałem, każda z baz po swojemu grupuje i dzieli dostępne aplikacje. Do tych kategoryzacji można by zaproponować nieco inny podział i wskazać takie kategorie, jak: „Narzędzia do tworzenia oprogramowania", „Systemy i narzędzia systemowe", a w tym podkategorię „Bezpieczeństwo", „Aplikacje biurowe", „Multimedia" (audio, wideo, obraz), a w tym podkategorię „Programy do obróbki graficznej", „Internet i narzędzia do pracy w Internecie", a w tym podkategorię „Komunikacja", „Edukacja" (nauka i technika), „Zarządzanie informacją", „Biznes", „Rozrywka i gry", „Biblioteki". Podobnie jak inne podziały i ten jest subiektywny, wynikający z innych pobudek.

Funkcjonujące klasyfikacje i bazy można przez to różnie oceniać. Najbardziej dopracowane i rozbudowane są te największe pod względem liczby indeksowanych aplikacji i jednocześnie te najbardziej popularne, jeśli idzie o liczbę ich użytkowników, przy czym zauważalna jest tendencja do monopolizacji. Choć każde powszechnie używane narzędzie OS posiada własną niezależną przestrzeń w Internecie, z której program można pobrać czy też za pośrednictwem której można wspomóc pracę nad programem, to większość popularnych programów open source i tak trafia do któregoś z pretendujących do roli centralnych katalogów czy kuźni.

---

[254] Por. J. M. Kretchmar, *Open source network administration*, Upper Saddle River 2004, s. 6.

5. WYBRANE APLIKACJE OPEN SOURCE

W niniejszym rozdziale wskażę zostaną przykładowe aplikacje open source'owskie. Z powodu dużej liczby programów, jak również ciągłej aktualizacji starszych wersji programów oraz tworzenia coraz to nowszych, wybiorę zostaną jedynie aplikacje reprezentatywne dla wyszczególnionych kategorii, które omówię bardziej szczegółowo. Dodatkowo, w celach informacyjnych, wyliczę dziesięć innych programów z każdej grupy. W zasadzie każda baza czy forge OSS posiada własne kategorie, według których porządkuje gromadzone aplikacje. Niewątpliwie i ten wyróżniony podział jest subiektywny, zdaje się jednak, że obejmuje on większość używanych powszechnie grup programów. Oczywiście można by wzorem innych dużych serwisów wprowadzać dodatkowe uszczegółowienia i podziały w obrębie kategorii aż do wyczerpania wszystkich możliwości, jednakże w tym przypadku nie będzie to konieczne. Książka niniejsza nie pretenduje do roli przewodnika po bogactwie aplikacji OSS, lecz ma być wprowadzeniem do zagadnienia open source w ogóle. Stąd ograniczyłem się jedynie do wskazania kilku głównych kategorii (systemy operacyjne, oprogramowanie biurowe, aplikacje do zarządzania informacją, programy edukacyjne, aplikacje internetowe, programy multimedialne, oprogramowanie rozrywkowe) przydatne w bibliotekach. Na rynku dostępnych jest bardzo dużo źródeł (artykuły, książki-monografie, podręczniki) poświęconych wyodrębnionym programom oraz grupom programów. Czytelnik zainteresowany konkretnymi aplikacjami bez trudu je odnajdzie.

Wybierając prezentowane aplikacje kierowałem się kryteriami wskazanymi w rozdziale 3.1 (tab. 4), takimi jak np. duża liczba ściągnięć, pozytywne opinie na temat aplikacji itd.

Dla każdego z programów w celach identyfikacyjno-informacyjnych podałem logo, jeżeli program je posiada, dokładną nazwę własną programu, numer polecanej wersji programu, niekiedy dodatkowo z datą jej publikacji, zrzut ekranu (ang. *screenshot*) uruchomionej aplikacji, datę pracy z programem, przynajmniej jeden adres URL wskazujący lokalizację programu, specyfikację aplikacji. Ponadto przedstawiłem zostanie opis funkcjonalności programu, zadań, które wypełnia, oraz korzyści płynących z jego użytkowania. Pominę każdorazowe przypominanie o takich oczywistych korzyściach, jak brak konieczności ponoszenia opłat za licencje do programu (wszystkie przedstawione aplikacje udostępniane były na licencjach typu GPL), szybko pojawiające się nowe wersje i ulepszenia oraz te cechy aplikacji open source, o których była mowa powyżej.

Wskazywane będą te aplikacje, które są dalej rozwijane lub takie, których rozwijanie zostało zakończone, ale funkcjonują w dopracowanych wersjach. Wybrane i opisane poniżej oprogramowanie działa pod popularnymi systemami należącymi do grupy OSS i PS (ściślej – dostępne są odpowiednie wersje programu przeznaczone dla wszystkich systemów operacyjnych). Starałem się również wybierać te aplikacje, które posiadały polski język interfejsu.

W opisie pominąłem się również instrukcje instalacji programów, gdyż mogą się one różnić w zależności od przeznaczenia programu (użytek domowy, komercyjny) oraz urządzenia, na którym program może być instalowany. Dodatkowo zakładam, że czytelnicy posiadają podstawową wiedzę informatyczną z zakresu obsługi komputerów i pracy w środowisku internetowym.

5.1. SYSTEMY OPERACYJNE

Popularną grupę systemów operacyjnych stanowią systemy Linux. Dokładniej, Linux jest to rodzina uniksopodobnych systemów operacyjnych opartych o jądro Linux. Na bazie jądra Linux buduje się dopiero konkretne, tzw. dystrybucje Linuxa, czyli kompletne systemy operacyjne[255].

[255] Por. *Wikipedia. Wolna encyklopedia* [online], [dostęp: 17.02.2012]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Linux.

W tym miejscu warto zreferować pokrótce historię rozwoju Linuxa. Na początku lat 90. XX wieku w przestrzeni publicznej wciąż brakowało jądra systemu operacyjnego. Fundacja Free Software Foundation nieustannie podejmowała w tym czasie działania zmierzające do opracowania takowego. Można tu wymienić projekt GNU HURD, który pozostał jednak systemem eksperymentalnym – nie nadawał się do wykorzystania w codziennej pracy. Pożądany system musiał ponadto odznaczać się dostępnością do kodu źródłowego, a więc najlepiej powinien być objęty licencją GPL. Wówczas funkcjonowały darmowe systemy, jednakże licencje, na których były udostępnione ograniczały możliwość wykorzystania ich kodu. Systemem, który nadawał się do tego zadania w pełni był MINIX (kopia systemu UNIX stworzona w 1987 pod kierownictwem profesora Andrei Tanenbauma na Vrije Universiteit w Amsterdamie). System był darmowy, powszechnie dostępny i używany do celów edukacyjnych, a jego kod można było dowolnie zmieniać. Jednym ze studentów, który postanowił uczyć się programowania systemowego na MINIX-ie był właśnie Linus Torvalds, który, pracując z kodem źródłowym systemu, doprowadził do stworzenia prostego i funkcjonalnego jądra systemu operacyjnego. Dodatkowo, w 1992 roku Torvalds zastąpił napisaną przez siebie wcześniej licencję kodu Linuksa na GPLv2, co przypieczętowało dalszy rozwój tego systemu. Linux jest najbardziej chyba rozpoznawaną marką open source, choć z całym przedsięwzięciem związane są też pewne kontrowersje. Fundacja Wolnego Oprogramowania podkreśla, że Linux jest tylko jądrem, które wymaga dodatkowego oprogramowania, np. z projektu GNU. A zatem użytkownik końcowy Linuksa w dużej mierze pracuje z programami napisanymi przez innych informatyków niż Torvalds i jego współpracownicy[256].

Podsumowując, Linux powstał w sierpniu 1991 roku, kiedy to Torvalds zdecydował się stworzyć (a w zasadzie zmodyfikować uniksopodobny system MINIX) dla własnych edukacyjnych celów taki system operacyjny, który spełniałby jego indywidualne wymagania. Poza tym chciał system ten po prostu usprawnić. Zamierzenie swoje zainicjował, wysyłając post do grupy dyskusyjnej skupionej przy MINIX-ie[257]. W rezultacie początkowy pomysł przerodził się w ruch, gdyż programiści skupieni wokół MINIX-a oraz inni, którzy od dawna pracowali na rzecz tworzenia do-

[256] Pisząc akapit skorzystałem z książki: P. Gawrysiak, *Cyfrowa rewolucja*..., s. 316-321.

[257] Pierwszą część akapitu opracowałem na podstawie książki: K. K. Mookhey, N. Burghate, *Linux. Security, audit and control features,* Rolling Meadows 2005, s. 3.

stępnych za darmo systemów uniksopodobnych, szybko przyłączyli się do projektu Torvaldsa[258].

Rosnąca popularność Linuxa zaowocowała zmianą filozofii pracy nad tym systemem. Mianowicie chciano, aby Linux dostępny był również dla osób, które nie posiadały wiedzy informatycznej. Samo jądro Linuxa było więc w takiej perspektywie niewystarczające. Potrzebne było dołączenie do niego pakietu programów, które łącznie zapewniłyby sprawną pracę komputera. Ponadto cały taki pakiet znacznie przyspieszał dalsze prace, ponieważ kolejni programiści i użytkownicy otrzymywali od razu zbiór niezbędnych narzędzi, które dalej można było ulepszać. Taki pakiet określa się mianem *dystrybucji Linuxa*[259].

To, co się powszechnie określa mianem *Linux* jest to *de facto* jądro Linuxa wraz z dodatkowym oprogramowaniem, a większość tego oprogramowania powstawała pod auspicjami Projektu GNU[260]. Bez Projektu GNU prawdopodobnie nigdy nie powstałby Linux, z kolei bez Linuxa już dziś Projekt GNU mógłby utracić swój potencjał i nie rozwinąć się. Obydwa projekty wzajemnie się stymulują i napędzają[261]. Jądro Linux wraz z oprogramowaniem open source dystrybuowane jest przez różne firmy i organizacje na całym świecie. Instytucje te zapewniają równocześnie wsparcie dla konkretnej sygnowanej przez siebie dystrybucji. Najpowszechniejszymi dystrybucjami Linuxa są Red Hat, Debian, SuSE oraz Mandrake. Dodatkowo na fundamencie takiej dystrybucji tworzy się dalsze wersje systemu, tj. kolejne dystrybucje[262]. Dla przykładu, wskazany poniżej system Ubuntu jest oparty na Debianie. Przy okazji warto podkreślić, że jest to właśnie przykład zastosowania licencji typu GPL. Kolejni użytkownicy mogą poprawić istniejący system (dystrybucję), w tym przypadku Debiana, i zaproponować swoją własną, w tym przypadku Ubuntu[263].

---

[258] Drugą część akapitu przygotowałem w oparciu o pracę: J. Masters, R. Blum, *Professional Linux programming*, Indianapolis 2007, s. 3.

[259] Uwagi wskazane w akapicie przywołałem z książki: J. Masters, R. Blum, op. cit., s. 4.

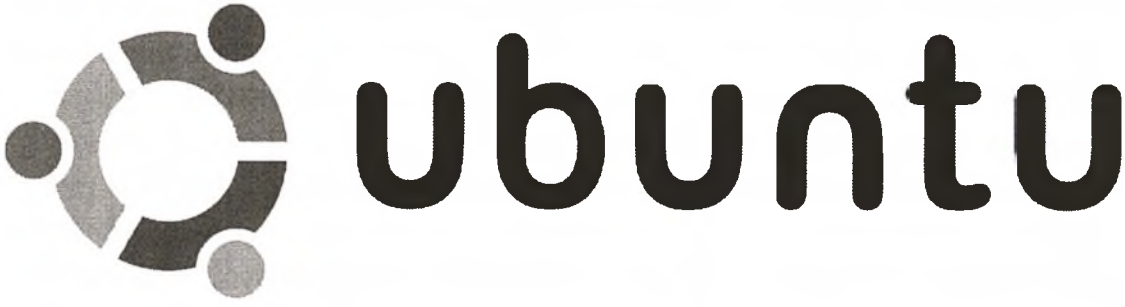
[260] Por. K. K. Mookhey, N. Burghate, op. cit., s. 3-4.

[261] Por. J. Masters, R. Blum, op. cit., s. 2.

[262] Por. K. K. Mookhey, N. Burghate, op. cit., s. 7.

[263] Por. D. M. Williams, *Czym Ubuntu różni się od Debiana?* [online], [dostęp: 17.02.2012]. Dostępny w WWW: http://jakilinux.org/linux/debian/czym-ubuntu-rozni-sie-od-debiana/. Debian powstał w 1990 roku, natomiast Ubuntu w 2004 roku. Por. C. Negus, F. Caen, *Ubuntu Linux toolbox: 1000+ commands for Ubuntu and Debian power users*, Indianapolis 2008, s. 2.

Systemem (dystrybucją Linuxa), który nieco przybliżę, jest właśnie Ubuntu (fot. 7).



Fot. 7. Logo Ubuntu[264].

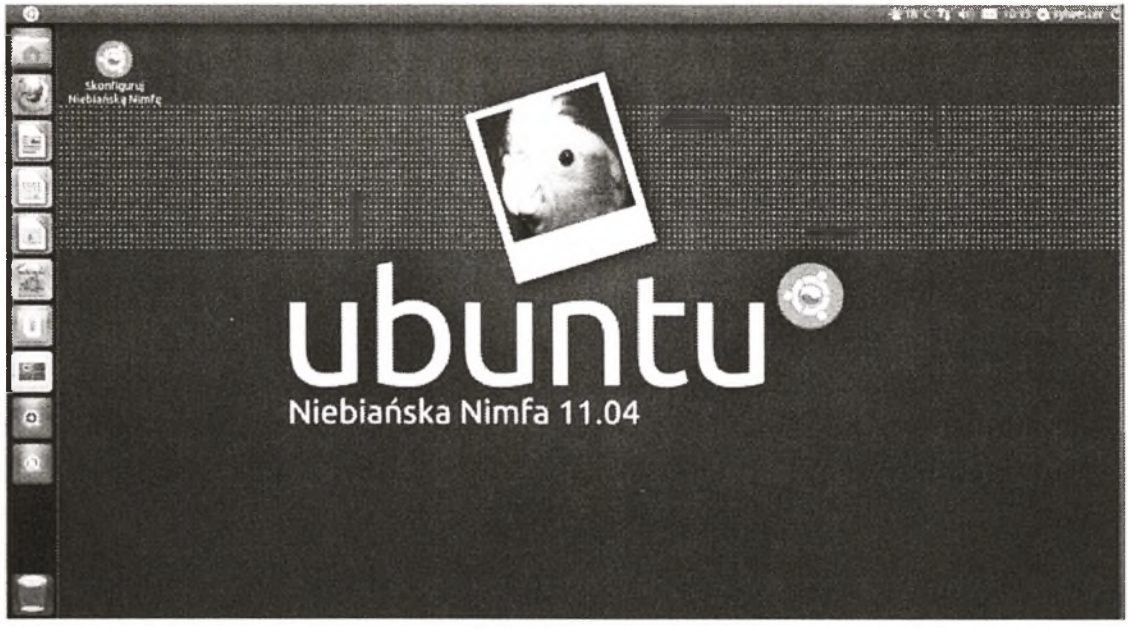
Nazwa własna	Ubuntu Niebiańska Nimfa
Numer wersji	11.04 PL
Data publikacji	28 kwietnia 2011
Data pracy z programem	15 grudnia 2011
Adres URL	http://ubuntu.pl/pobierz.php

W porównaniu do swojego „dziecka" Debian jest bardzo dopracowaną, bogatą i rozbudowaną dystrybucją Linuxa, która może spełnić wszelkie, jak na system operacyjny, oczekiwania. Ubuntu natomiast, w porównaniu z Debianem, jest dystrybucją skierowaną do mniejszej grupy odbiorców, która jest przeznaczona dla wszystkich. Ubuntu jest prostym, w pozytywnym tego słowa znaczeniu, systemem operacyjnym, który z powodzeniem można wykorzystywać do codziennej pracy w domu czy biurze. Zapewnia więc pełną funkcjonalność systemu operacyjnego. Rozpowszechniany w rozmiarze mieszczącym się na jednej płycie DVD, dostarczany jest w wersji live oraz instalacyjnej. Użytkownik wybiera, czy chce system najpierw sprawdzić bez instalacji czy od razu zainstalować na dysku komputera.

Zdaniem Davida Williamsa, Ubuntu można polecić jeszcze z kilku względów. Przede wszystkim dlatego, że kolejne wersje tej dystrybucji wydawane są dość często (co sześć miesięcy), dzięki czemu zapewnia się stałą

[264] Logo pobrano z: *Fall in love in New Ubuntu logo* [online], [dostęp: 17.02.2012]. Dostępny w WWW: http://frendhi.wordpress.com/2010/03/05/fall-in-love-in-new-ubuntu-logo/.

poprawę funkcjonalności. Jest systemem całkowicie dopracowanym i przyjaznym użytkownikowi. Ubuntu instaluje się i używa, tj. nie potrzeba na wstępie doinstalowywać kolejnych aplikacji. Celem projektantów Ubuntu jest tworzenie popularnego systemu dla przeciętnych użytkowników komputerów. Debian zaś zdaje się wysoce elastycznym systemem dla ekspertów, który mogą oni dowolnie dostosowywać do swoich własnych potrzeb[265]. Do dziś Ubuntu stał się bardzo popularną (najpopularniejszą), a przy tym stabilną i bezpieczną, dystrybucją Linuxa. Zapewnia użytkownikom wsparcie w różnych obszarach, np. jest systemem wielojęzycznym, a środowisko graficzne jest bardzo dopracowane, tzn. jest atrakcyjne wizualnie[266], o czym można się przekonać patrząc na zrzut ekranu (fot. 8). Przyjemny interfejs staje się jeszcze jednym powodem zachęcającym do pracy w tym systemie.



Fot. 8. Zrzut ekranu systemu Ubuntu Niebiańska Nimfa[267].

Inne projekty:

Anakonda (http://www.rs-anakonda.org/) – polski system ERP (ang. *enterprise resource planning*, tj. wspierający zarządzanie przedsiębiorstwem).

---

[265] Por. D. M. Williams, op. cit.

[266] Por. C. Negus, F. Caen, op. cit., s. 2-3.

[267] Zrzut pobrano z: *Niebiańska Nimfa, czyli polski remiks Ubuntu 11.04* [online], [dostęp: 27.02.2012]. Dostępny w WWW: http://www.ubucentrum.net/2011/06/niebianska-nimfa-czyli-polski-remiks.html.

Android (https://developers.google.com/android/?csw=1) – przeznaczony dla urządzeń mobilnych system operacyjny oparty na jądrze Linuksa.

Debian http://www.debian.org/) – system operacyjny (dystrybucja Linuksa).

Fedora (http://fedoraproject.org/) – system operacyjny (dystrybucja Linuksa).

FreeBSD (http://www.freebsd.org/) – system operacyjny z rodziny Unix.

Mandriva (http://www.mandriva.com/en/) – system operacyjny (dystrybucja Linuksa).

Mint (http://www.linuxmint.com/) – system operacyjny (dystrybucja Linuksa).

NetBSD (http://www.netbsd.org/) – system operacyjny z rodziny Unix.

PHPOS (http://www.phpos.pl/) – webowy system operacyjny.

Suse (http://www.opensuse.org/pl/) – system operacyjny (dystrybucja Linuksa).

5.2. OPROGRAMOWANIE BIUROWE

Projekt GNU oraz następnie projekt Linux wpłynęły na rozpowszechnienie wolnego i otwartego oprogramowania, realizacja tych projektów przyczyniła się do powołania OSI, a tym samym do jeszcze większego rozpropagowania samej idei open source. Mając do dyspozycji wolne i otwarte systemy operacyjne, a także zapewnione wsparcie rzeszy programistów i internautów, można było (od tamtego momentu) z łatwością tworzyć kolejne wspólne projekty wypełniające rozmaite nisze. Jedną z takich inicjatyw było stworzenie funkcjonalnego pakietu biurowego, stanowiącego alternatywę dla komercyjnych odpowiedników. W ten sposób w 1998 roku powstał projekt KOffice, czyli zestaw aplikacji biurowych przeznaczony dla wszystkich dostępnych systemów operacyjnych. Na bazie tego pakietu stworzono w 2010 roku pakiet Calligra Suite. Ich cechą szczególną jest mały rozmiar, co zapewnia szybkie ściąganie z sieci, instalację oraz uruchamianie zainstalowanych komponentów.

Drugim w kolejności i, jak się zdaje, najpopularniejszym jak dotąd, pakietem biurowym udostępnianym na licencji typu GPL był pakiet Open

Office (obecnie pod nazwą Apache OpenOffice), który po raz pierwszy pojawił się w 2001 roku (fot. 9).

Fot. 9. Logo pakietu biurowego Open Office[268].

Nazwa własna	Apache Open Office
Numer wersji	3.4.1
Data publikacji	sierpień 2012
Data pracy z programem	8 maja 2014
Adres URL	http://www.openoffice.org/pl/product.download.html

Open Office jest pakietem biurowym składającym się z szeregu komponentów przeznaczonych do realizacji konkretnych zadań biurowych. W skład pakietu wchodzi aplikacja przeznaczona do pracy z dokumentami tekstowymi, tj. procesor i edytor tekstu. Program pozwala na edytowanie plików w popularnych formatach, jak również na zapisywanie ich w różnych formatach przeznaczonych do pracy z pakietami biurowymi OSS, jak i PS. Wyliczanie funkcji programu zostanie pominięte, gdyż umożliwia on wykonywanie wszystkich możliwych operacji niezbędnych przy redagowaniu dokumentu tekstowego. Poza komponentem przydatnym do edycji tekstu Open Office dysponuje komponentami przeznaczonymi do tworzenia prezentacji multimedialnych, rozbudowanych arkuszy kalkulacyjnych, prostych i złożonych rysunków, baz danych oraz formuł matematycznych (fot. 10). Wzorem edytora tekstu pozostałe komponenty pakietu umożliwiają zapis i edycję plików w popularnych formatach przeznaczonych dla rozmaitych pakietów biurowych (OSS i PS). Osobom posiadającym pod-

[268] Logo pobrano z: *A brand refresh for OpenOffice.org* [online], [dostęp: 20.02.2012]. Dostępny w WWW: http://www.openoffice.org/trademark/brandrefresh.html.

stawową wiedzę z zakresu obsługi komputerów, a do takich skierowana jest ta książka, nie jest konieczne szczegółowe opisywanie funkcji i zadań, jakie pakiety biurowe mogą realizować, stąd kwestii tej nie będę pogłębiać.

Fot. 10. Widok okna powitalnego pakietu Open Office.

Pierwotnie Open Office tworzony był i rozwijany przez firmę Sun Microsystems. Firma ta została w 2010 roku przejęta przez firmę Oracle, która ostatecznie zdecydowała się na zaprzestanie dalszego rozwijania tego pakietu biurowego. W związku z polityką nowego właściciela oprogramowania Open Office, w tym samym roku większość programistów wspierających Open Office odeszła z firmy i założyła The Document Fundation. Po nieudanych staraniach przejęcia prawa do pakietu Open Office, nowo powstała Fundacja rozpoczęła tworzenie nowego pakietu biurowego, tj. Libre Office. Pakiet ten szybko zyskał wsparcie kilku dużych firm z branży IT. Ostatecznie firma Oracle ogłosiła zaprzestanie rozwijania pakietu Open Office, jednocześnie przekazując ciężar prac nad jego dalszym funkcjonowaniem społeczności skupionej wokół OpenOffice.org[269], a ściślej Apache Software Foundation(stąd dziś funkcjonuje pod nazwą Apache OpenOffice).

[269] W akapicie wykorzystałem dane pochodzące z: D. Szymański, *Oracle porzuca swój OpenOffice.org* [online], [dostęp: 27.02.2012]. Dostępny w WWW: http://www.benchmark.pl/aktualnosci/Oracle_porzuca_rozwoj_OpenOffice.org-34468.html.

Powstały w 2010 roku Libre Office (fot. 11) jest pakietem biurowym dostępnym m.in. wraz z dystrybucją Linuxa – Ubuntu. Libre Office jest wieloplatformowym pakietem biurowym powstałym na bazie kodu pakietu Open Office. Wieloplatformowość oznacza, że możliwa jest nie tylko edycja plików dostępnych w popularnych formatach, utworzonych w pakietach biurowych OSS i PS, lecz także zapisywanie plików w rozszerzeniach, które później można otworzyć w innych pakietach biurowych. Podobnie jak Open Office, składa się z aplikacji służącej do edycji tekstu, programu przeznaczonego do tworzenia prezentacji, arkuszy kalkulacyjnych, baz danych, formuł matematycznych oraz edytora grafiki.

Fot. 11. Logo pakietu biurowego Libre Office[270].

Nazwa własna	Libre Office
Numer wersji	3.5.0
Data publikacji	14 lutego 2012
Data pracy z programem	27 lutego 2012
Adres URL	http://www.libreoffice.org/download/

Zarówno pakiet Open Office, jak i Libre Office dostępne są w wersjach stacjonarnych, czyli przeznaczonych do instalacji na konkretnym urządzeniu, jak również w wersjach portable (przenośnych, podręcznych), a więc takich, których nie trzeba instalować. Wersja przenośna zapisywana jest

[270] Logo pobrano z: *Libre Office 3.4 również na wynos* [online], [dostęp: 27.02.2012]. Dostępny w WWW: http://www.komputerswiat.pl/nowosci/programy/2011/32/libreoffice-34-rowniez-na-wynos.aspx.

na pamięciach zewnętrznych, np. USB i stamtąd od razu uruchamiana. Ponadto pakiety te są wielojęzyczne, tj. posiadają wiele wersji językowych (ponad trzydzieści), co świadczy o ich wciąż rosnącej popularności.

Open Office wyznaczył też pewien standard, który uznaje się za alternatywę dla pakietów PS i który, co zrozumiałe, staje się teraz udziałem Libre Office. Mam na myśli format Open Document (skrót ODF). By lepiej zrozumieć ideę open document format, należy przypomnieć, że wszystkie dokumenty przechowywane są w konkretnych plikach – dokumentach tekstowych, arkuszach kalkulacyjnych itd. Innymi słowy, dane te przechowywane są w postaci cyfrowej, w konkretnym formacie pliku. Podczas korzystania z pakietu PS możliwe jest transferowanie danych pomiędzy różnymi plikami (dokument tekstowy, prezentacja, baza danych, arkusz kalkulacyjny). Natomiast w momencie, gdy plik stworzony dla jednego pakietu biurowego będzie się chciało odczytać w innym pakiecie biurowym, mogą pojawić się problemy. Kod programu jest chroniony, podobnie i zawartość, a zatem dane użytkownika są zaszyfrowane, co uniemożliwia ich odczytanie w innej aplikacji. Chcąc je odczytać w innym programie, plik z danymi należałoby wcześniej zapisać w formacie takim, jak np. RTF, który można odczytywać w różnych aplikacjach biurowych. Jednakże konwersja do postaci RTF skutkuje niekiedy utratą pewnych właściwości danych, tj. formatowania. Inną jeszcze kwestią jest ta związana z rozwijaniem danego formatu przez właściciela praw do niego. Może się bowiem okazać, że za kilka lat konkretna firma zaprzestanie rozwijania i wspierania formatu, który wcześniej opracowała i rozpowszechniała, co zaowocuje tym, że dane użytkowników staną się niemożliwe do odczytania. Remedium na te problemy stanowi format Open Document (skrót ODF)[271].

ODF jest to otwarty i wolny standard formatu plików pakietów biurowych. Obejmuje dokumenty tekstowe, arkusze kalkulacyjne, prezentacje i bazy danych. Jest, co zrozumiałe, alternatywą dla pakietów PS. Specyfikacja tego formatu została rozwinięta przez firmę Sun Microsystems. Standard został opracowany przez OASIS, na bazie XML-owego formatu pakietu Open Office, a w maju 2006 roku stał się standardem ISO (ISO/IEC 26300). ODF jest używany zarówno przez PS, jak i OSS. Odkąd celem funkcjonowania ODF jest zagwarantowanie długoletniego dostępu do danych zawartych w plikach, a zatem zniesienie technologicznej barie-

[271] Przy opracowaniu akapitu posiłkowałem się książką: J. D. Eisenberg, *OASIS Open Document essentials. Using OASIS OpenDocument XML*, Airlie Beach 2005, s. 1.

ry informacyjnej, odtąd ODF stał się sprawą publiczną i polityczną. ODF jest bowiem zalecany przez ustawodawstwo wielu krajów[272].

„Jednym z argumentów zachęcających do wdrażania wolnego oprogramowania jest fakt wykorzystywania go przez administrację i inne instytucje publiczne w wielu państwach Unii Europejskiej. W tym celu powstał m.in. dokument eEuropa+ 2003, promujący wykorzystywanie wolnego oprogramowania"[273]. W Polsce kwestii tej dotyczy Rozporządzenie Rady Ministrów z dnia 11 października 2005 roku w sprawie minimalnych wymagań dla systemów teleinformatycznych (Dziennik Ustaw – Dz. U. rok 2005 nr 212, poz. 1766). W tabeli „Formaty danych zapewniające dostęp do zasobów informacji udostępnianych za pomocą systemów teleinformatycznych używanych do realizacji zadań publicznych" umiejscowiono Open Document Format for Office Applicaiton (otwarty format dokumentów aplikacji biurowych), tworzony pod auspicjami organizacji OASIS[274].

Inne aplikacje:

AbiWord (http://www.abiword.org/) – procesor (edytor) tekstu.

Appache Open Office (http://www.openoffice.org/) – pakiet biurowy.

Feng Office Community Edition (http://www.fengoffice.com/web/) – pakiet wspierający (współ)pracę biurową online.

GNOME Office (https://wiki.gnome.org/) – pakiet biurowy dla środowiska GNOME.

Joeffice (http://www.joeffice.com/) – pakiet biurowy.

---

[272] Dane przedstawione w akapicie uzyskałem z pracy: M. Bijsterbosch, K. van Godtsenhoven, P. Hochstenbach, R. Russell, M. Vanderfeesten, *Framework for interoperability*, [in:] *Emerging standards for enhanced publications and repository technology. Survey on technology*, ed. M. Vernooy-Gerritsen, Amsterdam 2009, s. 183-184. „Otwarty Format Dokumentów (ODF) daje realne podstawy do tworzenia uniwersalnych dokumentów, które będzie można tworzyć, wyświetlać i przetwarzać niezależnie od platformy sprzętowo-systemowej [...]. Oparty jest na języku XML i to właśnie ta technologia wyznacza obecny kierunek rozwoju, jeśli chodzi o zarządzanie czy przetwarzanie danymi [...]". P. Tkacz, *Otwarte formaty dokumentów i ich znaczenie przy wymianie informacji*, „Zeszyty Naukowe Wyższej Szkoły Zarządzania Ochroną Pracy w Katowicach" 2006, nr 1, s. 135.

[273] K. Wasiucionek, *Wolne oprogramowanie w bibliotekach – problemy i perspektywy*, „Przegląd Informacyjno-Dokumentacyjny" 2009, nr 3, s. 12.

[274] Podobne zalecenia formułuje Parlament Europejski: *European Parliament wants Open Document exchange format for electronic business* [online], [dostęp: 2.08.2013]. Dostępny w WWW: http://press.ffii.org/Press%20releases/European%20Parliament%20wants%20Open%20Document%20exchange%20format%20for%20electronic%20business.

NeoOffice (http://www.neooffice.org/neojava/en/index.php) – pakiet biurowy dla systemów OS X.

Oxygen Office Professional (http://sourceforge.net/projects/ooop/) – pakiet biurowy.

Scribus (http://www.scribus.net/canvas/scribus) – program do składu tekstu (DTP).

Simple Groupware (http://www.simple-groupware.de/cms/) – oprogramowanie do pracy grupowej online.

Tiki Wiki CMS Groupware (http://sourceforge.net/projects/tikiwiki/) – CMS oraz oprogramowanie biurowe online.

5.3. APLIKACJE DO ZARZĄDZANIA INFORMACJĄ

Poza jądrem otwartego systemu potrzebne są programy, dzięki którym możliwa jest codzienna komputerowa praca. Wśród nich istotną grupę stanowią programy służące do zarządzania informacją. Truizmem jest twierdzenie, że informacja jest dziś centralną kategorią i najwyższym dobrem rozwijających się społeczeństw, a skuteczne narzędzia pozwalające nią zarządzać są tego rozwoju gwarantem. Po dodaniu do tego zalet technologii open source, powstają idealne wprost warunki do realizacji wszelkich procesów informacyjnych odbywających się w i przy udziale sieci internetowej.

Termin *zarządzanie informacją* pojawił się w piśmiennictwie naukowym na przełomie lat 60. i 70. XX wieku, kiedy to założono m.in. takie czasopisma, jak „Information & Management", „Information Storage and Retrieval" (później „Information Processing and Management") czy „MIS Quarterly". Z kolei w latach 80. do dyskursu naukowego wprowadzono również termin *zarządzanie wiedzą* – utożsamiany z terminem *zarządzanie informacją*[275].

Od końca lat 60., jak podaje Wiesław Babik, obserwuje się dynamiczny wzrost publikacji poświęconych zagadnieniu zarządzania informacją[276].

---

[275] Por. B. Stefaniak, *Bibliometria w zarządzaniu informacją*, [w:] *Zarządzanie informacją w nauce*, red. D. Pietruch-Reizes, Katowice 2008, s. 23-24.

[276] Por. W. Babik, *Informacja naukowa jako przedmiot zarządzania*, [w:] *Zarządzanie informacją w nauce*, red. D. Pietruch-Reizes, Katowice 2008, s. 43.

W publikacjach tych nie tylko próbuje się wyjaśnić znaczenie samego terminu, lecz także postuluje się, czym zarządzanie informacją powinno się zajmować. W dobie powstawania społeczeństw opartych na informacji i wiedzy wyłonił się w obszarach aktywności ludzkiej nowy kierunek działań, zmierzający do opanowania informacji i jej nadmiaru, tj. właśnie zarządzanie informacją. Jak dotąd nie stworzono jednak takiej definicji tego terminu, która wszystkim by odpowiadała, a także zostałaby przystosowana do potrzeb teorii i praktyki[277].

W *Słowniku encyklopedycznym informacji, języków i systemów informacyjno-wyszukiwawczych* podano, że zarządzanie informacją to 'sterowanie przebiegiem procesów informacyjnych mające na celu ich optymalizację'[278]. Wiesław Babik podaje z kolei taką eksplikację: „zespół działań, dzięki którym informacja dociera tam, gdzie powinna – do odpowiedniej osoby/instytucji, w odpowiednie miejsce, o odpowiedniej porze, o odpowiedniej jakości, w odpowiedniej postaci/formie, lub jest potencjalnie dostępna dla określonych użytkowników"[279]. Określenie *zarządzanie informacją* odnosi się więc w szczególności do wyszukiwania, selekcjonowania, oceniania (wartościowania), opracowywania, gromadzenia, przechowywania, udostępniania informacji. Widać wyraźnie, że większość programów komputerowych można zaliczyć w poczet aplikacji służących do zarządzania informacją. Mając to na uwadze, podam przykład tylko jednego z nich. Będzie to program FileZilla (fot. 12).

Fot. 12. Logo programu FileZilla[280].

[277] Ibidem, s. 43.

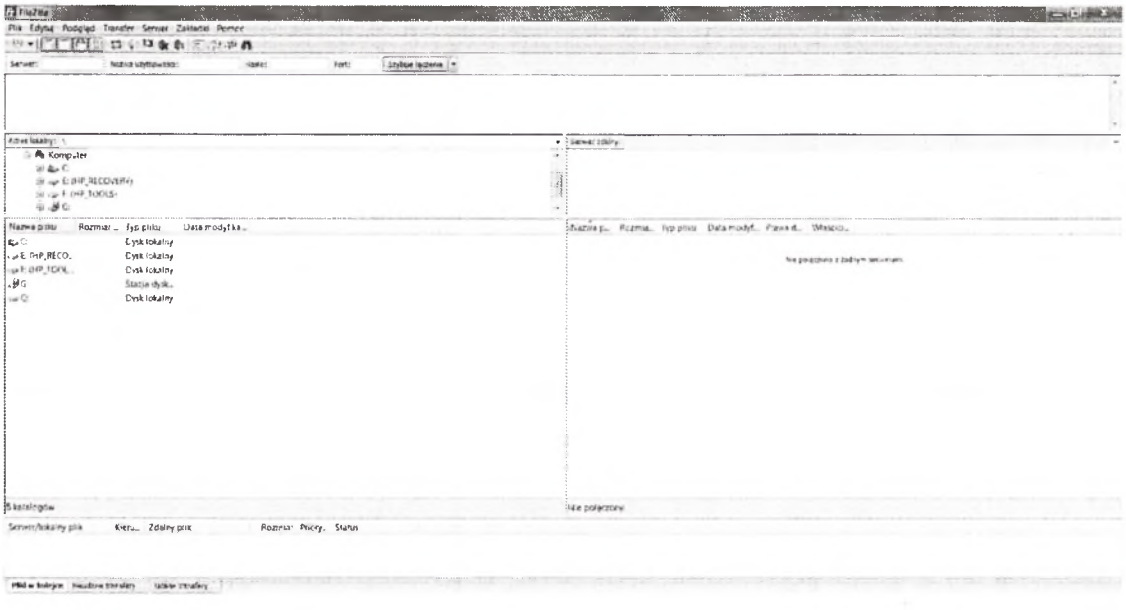
[278] *Słownik encyklopedyczny informacji, języków i systemów informacyjno-wyszukiwawczych*, oprac. B. Bojar, Warszawa 2002, s. 307.

[279] W. Babik, op. cit., s. 47.

[280] Logo pobrano z: *FILEZILLA* [online], [dostęp: 6.08.2012]. Dostępny w WWW: http://filezilla.com.pl/.

Nazwa własna	FileZilla
Numer wersji	3.8
Data publikacji	28 marca 2014
Data pracy z programem	6 maja 2014
Adres URL	http://filezilla-project.org/

FileZilla jest to klient FTP obsługujący protokoły FTP, FTPS, SFTP. Ponadto współpracuje z protokołem IPv6 oraz serwerami Proxy i obsługuje połączenia szyfrowane. Jest więc to program komputerowy, którego cechą główną jest możliwość łączenia się z serwerami FTP w celu pobierania i wysyłania plików. Komunikacja z serwerem wymaga posiadania odpowiedniego konta oraz danych domeny (nazwa użytkownika, hasło, serwer). Oprócz tego FileZilla posiada wiele wersji językowych. Umożliwia pobieranie plików, w tym zatrzymywanie i kontynuowanie pobierania. Posiada także wersję portable (nie wymaga instalacji) oraz wersję serwer (fot.13).



Fot.13. Zrzut ekranu działającego programu FileZilla.

Inne aplikacje:
Drupal (https://drupal.org) – system zarządzania treścią (CMS).
e107 (http://e107.org) – system zarządzania treścią (CMS).

Gekosale (http://www.gekosale.pl) – oprogramowanie sklepu internetowego.

Joomla (http://www.joomla.org) – system zarządzania treścią (CMS).

Launchpad (https://launchpad.net) – strona internetowa oraz program sieciowy przeznaczone do rozwoju i zarządzania projektami.

MODx (http://modx.com) – system zarządzania treścią (CMS).

Pligg (http://pligg.com) – system zarządzania treścią (CMS).

Ralph (http://ralph.allegrogroup.com/) – system służący do zarządzania infrastrukturą serwerową.

TYPO3 (http://typo3.org) – system zarządzania treścią (CMS).

WordPress (http://wordpress.org) – system zarządzania treścią (CMS).

5.4. TECHNOLOGIE DLA EDUKACJI

Ruch open source, a także ideały leżące u podstaw tego społecznego fenomenu *per se* są nośnikami wartości pedagogicznych. Open source wyrósł w duchu współpracy i dzielenia się własnymi pomysłami, dokonaniami i wiedzą z innymi członkami społeczności i współgra z ideałami edukacji. Już choćby to w pełni uzasadnia potrzebę edukowania (na każdym szczeblu) o tym zjawisku. Jak się zdaje, najlepszym sposobem uczenia jest praktyka. W tym przypadku – rzeczywiste wykorzystanie i pracowanie na open source'owskich technologiach informacyjnych i komunikacyjnych, dlatego warto wprowadzić je do wszelkich zajęć. Z tego typu narzędzi warto używać platform służących do prowadzenia procesu dydaktycznego, takich jak np. e-learningowe (Moodle, OLAT, Sakai Project, iTunesU). Za ich pośrednictwem lub *via* inne technologie można stosować dalsze narzędzia dydaktyczne o otwartym i wolnym kodzie, np. Anagramarama (edukacyjna gra polegająca na odszukiwaniu anagramów), Anki (progam przeznaczony do uczenia się zapamiętywania), Celestia (program pozwalający eksplorować wszechświat), Open Teacher (program ułatwiający naukę języka obcego), Tux Math (program przeznaczony do nauki matematyki) itd. W każdej bazie OSS można odnaleźć kategorię edukacja lub edukacja i nauka, w której gromadzone są odesłania do konkretnych technologii informacyjnych open source znajdujących zastosowanie w działalności edukacyjnej.

Wśród portali i serwisów poświęconych oprogramowaniu open source przeznaczonemu dla edukacji można wskazać Education software for

schools: free software, open software (http://www.schoolforge.net), Seul.org Home Page (http://www.seul.org/), OpenSourceSchools.org (http://www.opensourceschools.org/) czy WiOO w szkole (http://wioowszkole.org/). Serwisy powołano m.in. po to, aby ułatwić współpracę pomiędzy organizacjami promującymi otwarte zasoby w edukacji stworzyć szereg narzędzi przydatnych w działalności edukacyjnej i upowszechniać wolne i otwarte oprogramowanie w obszarze edukacji i nauki[281].

Wartą bliższego przyjrzenia się jest platforma przeznaczona do nauczania na odległość – Moodle (fot. 14).

Fot. 14. Logo Moodle[282].

Nazwa własna	Moodle
Numer wersji	2.5.1
Data publikacji	8 lipca 2013
Data pracy z programem	2 sierpnia 2013
Adres URL	https://moodle.org/downloads/

Moodle jest to *de facto* zestaw narzędzi umożliwiający prowadzenie zajęć (kursów, szkoleń) w formie e-learningowej. Projekt rozwijany jest globalnie, podobnie jak inne tego typu inicjatywy. Całość instaluje się na komputerze (serwerze) obsługującym PHP oraz bazy SQL (np. MySQL). Słowo *moodle* jest w zasadzie akronimem od Modular Object Oriented Dynamic Learning Environment[283], a więc jest to „środowisko nauczania

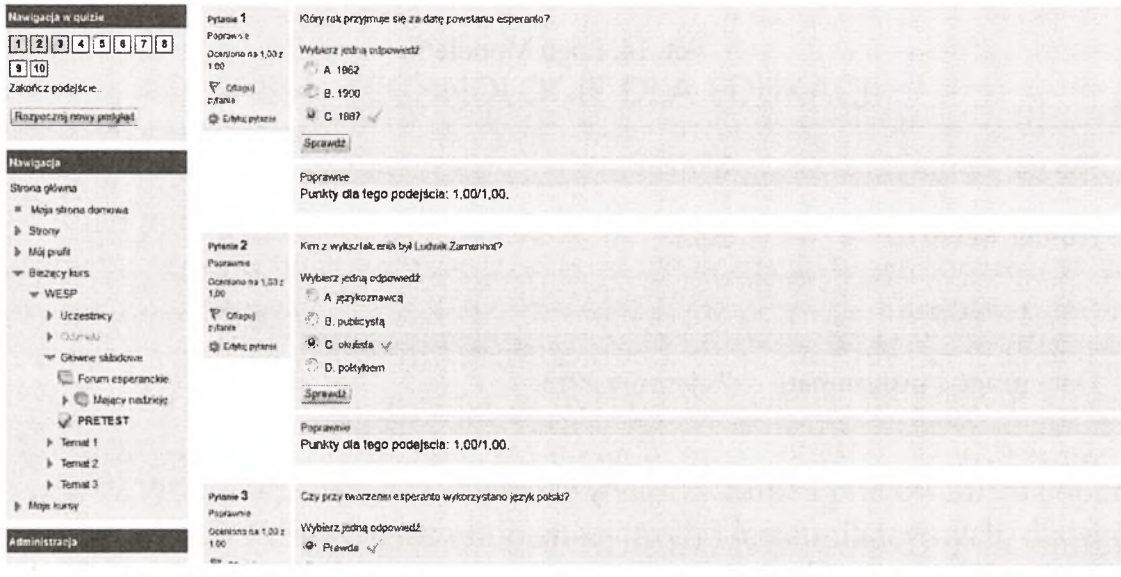
[281] W akapicie umieściłem informacje posiłkując się pracą: M. Jarocki, *Open Source – alternatywa w edukacji*, [w:] *Seminarium „Otwarte Zasoby Edukacyjne", Toruń 07.10.2009*, Elektroniczna Biblioteka Pedagogiczna SBP [online], [dostęp: 20.10.2009]. Dostępny w WWW: http://e-pedagogiczna.edu.pl/upload/file/zasoby/konferencje/torun/jarocki.pdf.

[282] Logo pobrano z: *Moodle* [online], [dostęp: 2.08.2013]. Dostępny w WWW: https://moodle.org/.

[283] Por. *About Moodle* [online], [dostęp: 2.08.2013]. Dostępny w WWW: http://docs.moodle.org/25/en/About_Moodle.

zdalnego za pomocą sieci teleinformatycznych, dostępne przez przeglądarkę internetową"[284].

Po zainstalowaniu Moodle osoba zakładająca kurs ma do dyspozycji szereg edukacyjnych narzędzi, jak np. forum, strony wiki, tworzenie quizów, możliwość odsyłania do wszelkich zasobów sieciowych, tworzenie stron HTML itd. Innymi słowy, może przygotować materiały tekstowe, audio, audiowizualne, pytania i zadania dla kursantów i wreszcie w sposób zautomatyzowany poddać ocenie wykonane przez nich prace. Dodatkowo cały czas zapewniony jest stały kontakt użytkowników z nauczycielem prowadzącym dany kurs. Jako przykład zademonstrowałem fragment niegdyś prowadzonego przeze mnie kursu poświęconego nauce języka esperanto (fot. 15).



Fot. 15. Zrzut ekranu rozwiązanego częściowo quizu dotyczącego znajomości języka esperanto.

Aplikacje OSS przeznaczone dla edukacji można podzielić również według obszarów zastosowań czy dyscyplin. Daje się bowiem wydzielić programy w kategoriach[285]: astronomia (Celestia, Stellarium, KStars), sztuka

[284] *Wikipedia. Wolna encyklopedia* [online], [dostęp: 2.08.2013]. Dostępny w WWW: http://pl.wikipedia.org/wiki/Moodle.

[285] Por. *50 open source tools that replace popular educational apps* [online], [dostęp: 5.06.2012]. Dostępny w WWW: http://www.datamation.com/feature/50-Open-Source--Tools-That-Replace-Popular-Education-Apps-3888901.htm.

(Tux Paint), chemia (Kalzium), gry edukacyjne (GCompris, ChildsPlay), języki obce (ZWDisplay, Zkanji), geografia (WorldWind, Marble), matematyka (GraphCalc, gnuplot, TTCalc, Maxima), fizyka (Step), nauka pisania na klawiaturze (TuxType, Klavaro, TypeFaster Typing Tutor).

W edukacji można wykorzystywać również open source hardware, czego dowodem może być przykład interaktywnej tablicy multimedialnej[286]. Poza wyżej wymienionymi powodami skłaniającymi do pracy z narzędziami OS, podstawowym pozostaje koszt wdrożenia tych narzędzi. Fakt, że bez trudu można odnaleźć darmowe aplikacje OS przechyla szalę na ich korzyść i przemawia za używaniem w obszarze edukacji.

Inne aplikacje:

Apereo OAE (http://www.oaeproject.org/) – środowisko nauczania zdalnego.

Audacity (http://audacity.sourceforge.net/?lang=pl) – edytor i rekorder audio.

BigBlueButton (http://bigbluebutton.org) – program do organizowania konferencji webowych oraz wspomagania nauczania na odległość.

Canvas (http://www.instructure.com/) – środowisko nauczania zdalnego (LMS, czyli Learning Management System, pol. *system zarządzania nauczaniem*).

Etherpad (http://etherpad.org/) – edytor dokumentów online.

Gibbon (http://gibbonedu.org/) – system zarządzania procesem edukacyjnym.

ILIAS (www.ilias.de) – system zarządzania nauczaniem (LMS).

Matterhorn (http://opencast.org/matterhorn/) – platforma służąca do zarządzania zasobami edukacyjnymi audio, wideo.

OpenMeetings (http://openmeetings.apache.org/) – program do organizowania konferencji webowych.

ownCloud (http://owncloud.org/) – program do tworzenia osobistej chmury oraz zarządzania dokumentami i ich edytowania.

[286] Por. P. Peszko, *Open-source hardware po polsku, czyli jak radzić sobie z brakiem sprzętu* [online], [dostęp: 29.05.2012]. Dostępny w WWW: http://blog.2edu.pl/2011/07/open-source-hardware-po-polsku-czyli.html.

5.5. APLIKACJE INTERNETOWE

Przypomnę, że powstanie, upowszechnienie się i rozwój Internetu (w tym rozmaitych sieciowych technologii informacyjnych, np. protokołów komunikacyjnych) odbywał się w warunkach swobodnej współpracy rzesz wynalazców, a więc w środowisku w pełni otwartym. To podłoże odcisnęło swoje piętno na sferze IT. Skoro początki były open source'owskie, to łatwiej było powrócić do korzeni i znowu tworzyć aplikacje w open source'owski sposób, dowodem czego jest dzisiejszy rynek technologii informacyjnych i komunikacyjnych. Muszę przyznać, że nie czuję się w pełni kompetentny do omawiania zagadnienia wskazanego nagłówkiem tego podrozdziału. Przykładowe aplikacje przedstawione poniżej niech będą małym egzemplum tej ogromnej grupy oprogramowania. Zdaje się bowiem, że dziś, jeśli nie wszystkie, to zdecydowana większość programów komputerowych mogłaby być zakwalifikowana do tej grupy.

Poniżej wskażę dwie aplikacje, tj. Thunderbird, czyli program do zarządzania pocztą elektroniczną, oraz Pidgin – wieloplatformowy komunikator internetowy.

Fot. 16. Logo programu Thunderbird[287].

Nazwa własna	Thunderbird
Numer wersji	13.0
Data publikacji	5 czerwca 2012
Data ściągnięcia	6 czerwca 2012
Adres URL	http://www.mozilla.org/pl/thunderbird/

[287] Logo pobrano z *Thunderbird* [online], [dostęp: 4.08.2013]. Dostępny w WWW: http://www.mozilla.org/thunderbird/img/tb5/download/download-logo.png.

Thunderbird to funkcjonalny klient pocztowy oraz przeglądarka grup dyskusyjnych. Pocztę można otwierać w kartach (wzorem przeglądarek internetowych), co pozwala na przeglądanie wielu maili jednocześnie. Istnieje możliwość doinstalowywania dodatków takich, jak np. filtr niechcianej poczty czy kalendarz (dodatki dodaje się poprzez menu narzędzia). Thunderbird pozwala archiwizować pocztę, a ponadto posiada ochronę przed złośliwym oprogramowaniem, spamem i phisingiem (wyłudzanie poufnych informacji osobistych).

Drugim programem jest komunikator Pidgin.

Fot. 17. Logo programu Pidgin[288].

Nazwa własna	Pidgin
Numer wersji	2.10.4
Data publikacji	6 maja 2012
Data ściągnięcia	6 czerwca 2012
Adres URL	http://www.pidgin.im/

Wieloplatformowy komunikator internetowy Pidgin obsługuje różne protokoły transmisyjne. Pozwala używać wielu różnych komunikatorów jednocześnie, np. AIM, Bonjour, Gadu-Gadu, Google Talk, Groupwise, ICQ, IRC, MSN, MySpaceIM, MXit, QQ, SILC, SIMPLE, Sametime, XMPP, Yahoo!, Zephyr. Dzięki aplikacji można więc prowadzić komunikację synchroniczną na wielu platformach i nie trzeba instalować wielu różnych komunikatorów. Ponadto Pidgin obsługuje wtyczki, posiada moduł przesyłania plików oraz ma wiele wersji językowych. Funkcjonuje również w wersji przenośnej.

[288] Logo pobrano z: *PIDGIN* [online], [dostęp: 6.05.2014]. Dostępny w WWW: http://pidgin.com.pl/.

Inne aplikacje:

Chromium (http://dev.chromium.org/home) – przeglądarka internetowa.

DownThemAll (https://addons.mozilla.orgpl/firefox/addon/downthemall/) – menedżer plików (dodatek do Firefoxa).

Epiphany (http://wiki.gnome.org/Apps/Web) – przeglądarka internetowa.

FireFTP (http://fireftp.net/) – klient FTP (dodatek do Firefoxa).

LimeSurvey (http://www.limesurvey.org/en/) – internetowy system ankiet.

Lightning Calendar (http://www.mozilla.org/en-US/projects/calendar/) – dodatek do programu Thunderbird służący organizacji i zarządzaniu kalendarzem.

Links (http://www.jikos.cz/~mikulas/links/) – przeglądarka internetowa.

Mozilla Firefox (http://www.mozilla.org/pl/firefox/new/) – przeglądarka internetowa.

NetSurf (http://www.netsurf-browser.org/) – przeglądarka internetowa.

SeaMonkey (http://www.seamonkey-project.org/) – przeglądarka internetowa.

5.6. PROGRAMY MULTIMEDIALNE

Warto przypomnieć, że problematykę dotyczącą aplikacji multimedialnych można włączyć do zagadnień *edukacji medialnej*. Zadowalającej definicji tego terminu jak na razie nie ustanowiono[289]. Wyznacza się natomiast zakres, który edukacja medialna powinna objąć. Wskazuje się, że należy uczyć „kultury korzystania z mediów", a w sytuacji postępującej konwergencji – także „zastosowań komputerów w obróbce i animowaniu obrazów, tworzeniu filmów, emisji audycji radiowych (radio internetowe) itp."[290].

Zakres ten można jeszcze bardziej doprecyzować. Otóż za sprawą serwisów udostępniających krótkie filmy ma się do czynienia nie tyle z media literacy (edukacja medialna), ile raczej z visual literacy (edukacja wizu-

[289] „Wielofunkcyjność i wielopostaciowość mediów sprawia, że edukację medialną bardzo trudno jest zdefiniować oraz umieścić na rozległej i ciągle się zmieniającej mapie oddziaływań wychowawczych". J. Jastrzębski, *Edukacja medialna*, [w:] *Edukacja medialna. Teksty i preteksty*, red. I. Borkowski, Wrocław 2004, s. 28.

[290] Ibidem, s. 28-29.

alna), co oznacza *de facto* potrzebę nauczania ściślej multimedia literacy (edukacja multimedialna). Odpowiednikiem czytania i pisania w text literacy jest w multimedia literacy myślenie krytyczne. Podczas realizacji takich przedsięwzięć rozwija się wyobraźnię, logiczne myślenie, planowanie i zdolności analityczne oraz uczy się rozwiązywania danego problemu[291].

Generalnie kluczowe staje się więc ukazanie szerokiego spektrum zastosowań komputerów podłączonych do internetowego środowiska hipertekstualnego w aspekcie poszukiwania, odbierania, tworzenia, przetwarzania, zapisywania, przesyłania, emitowania różnych komunikatów, czyli po prostu informacji (zarządzanie informacją).

Wśród służących do tego aplikacji wskażę przede wszystkim program do odtwarzania plików multimedialnych (audio, wideo), czyli tzw. player, jakim jest np. Miro.

Fot. 18. Logo programu Miro[292].

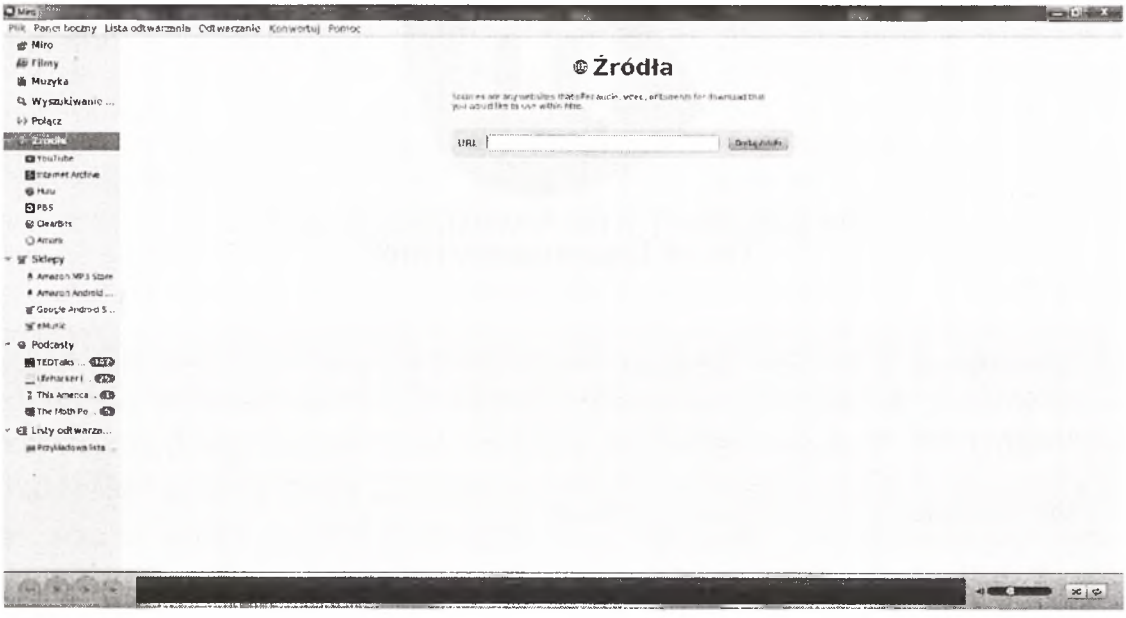
Nazwa własna	Miro
Numer wersji	6.0
Data publikacji	16 kwietnia 2013
Data ściągnięcia	6 maja 2014
Adres URL	http://www.getmiro.com/

Miro to odtwarzacz plików audio i wideo. Odtwarza większość popularnych formatów plików. Program ma bezpośredni dostęp do interneto-

[291] Akapit przygotowałem w oparciu o pracę: W. L. Sutton, *Literacy replacing libraries? A chance to turn the tide*, „School Libraries in Canada" 2006, vol. 25, no 3, s. 13.

[292] Logo pobrano z: *Miro – The New Democracy Web TV Player : Open Source Online Video* [online], [dostęp: 4.08.2013]. Dostępny w WWW: http://www.webtvwire.com/miro-the-new-democracy-web-tv-player-open-source-online-video/.

wych serwisów wideo, w tym telewizji internetowej (także w jakości HD), serwisów podcastów, sklepów internetowych, w których można kupić m.in. utwory muzyczne, a także serwisów, z których można pobierać pliki multimedialne. W programie można tworzyć biblioteki wszystkich multimedialnych plików zgromadzonych na komputerze, tworzyć playlisty, ustawiać odtwarzanie ciągłe lub w kolejności. Program zapamiętuje moment przerwania odtwarzania pliku i po ponownym uruchomieniu rozpoczyna odtwarzanie od miejsca, w którym zostało ono przerwane. Jest więc w pełni funkcjonalnym playerem, który w zupełności wystarczy do odtwarzania najpopularniejszych formatów plików (np. MPEG, Quicktime, AVI, H.264, Flash Video, Windows Media Video, XVID, MP3, MP4, WebM). Jest zatem remedium na tzw. wojnę formatów oraz konieczność instalowania dużej ilości programów przeznaczonych do odtwarzania określonych formatów plików.



Fot. 19. Zrzut ekranu otwartej aplikacji Miro.

Inne aplikacje:

Art of Illusion (http://www.artofillusion.org/) – program do modelowania obrazów i animacji 3D.

Banshee (http://banshee.fr/) – odtwarzacz multimedialny.

Blender (http://www.blender.org/) – program graficzny do tworzenia animacji filmowych.

FreeMind (http://freemind.sourceforge.net/wiki/index.php/main-page) – program do tworzenia map myśli.

Gwenview (http://gwenview.sourceforge.net/) – przeglądarka zdjęć.

HandBrake (http://handbrake.fr/) – konwerter wideo.

Krita (http://krita.org/) – edytor grafiki rastrowej.

Open Movie Editor (http://www.openmovieeditor.org/) – program do edycji wideo.

VLC (http://www.videolan.org/vlc/) – odtwarzacz multimedialny.

Wings 3D (http://www.wings3d.com/) – program do modelowania obiektów 3D.

5.7. OPROGRAMOWANIE ROZRYWKOWE

Aplikacje rozrywkowe, tj. przeznaczone do zabawy, są chyba najbardziej rozbudowaną grupą oprogramowania, zwłaszcza własnościowego. Na aplikacjach realizujących funkcję ludyczną można bowiem wypracowywać pokaźne zyski. Mimo że w tym względzie będą górować aplikacje PS, to jednak funkcjonują na rynku również programy OSS pełniące funkcje ludyczne i zapewniające rozrywkę. W tej kategorii aplikacje umieszcza się w dwóch grupach: *game* oraz *entertainment* (pol. gry i rozrywka)[293].

Sam ruch open source *per se* sprzyja realizacji zadań rozrywkowych. Ciekawą i śmiałą tezę postawił Fredrik Hallberg, którego zdaniem technologie peer-to-peer nie tylko doprowadziły do tego, że wymiana plikami stała się bardzo popularna, lecz także przyczyniły się do popularyzacji procesu wymiany wszelkimi pomysłami, tj. wpłynęły w znaczący sposób na rozwój kreatywności[294]. Odtąd nowości stawały się szybciej dostępne ogółowi. W ten sposób daje się wyjaśnić nie tylko popularność ruchu open source – ludzie mogą dzielić się treściami (uploads), które sami tworzą, dodatkowo pozostają w stałym kontakcie z innymi użytkownikami (connectivity) – ale i wpływ OS na rozwój Internetu i oprogramowania w ogóle. Zabawą i rozrywką jest samo dzielenie się swoimi pomysłami[295].

---

[293] Por. J. Feller, *Perspectives on free and open source software*, Cambridge, London 2005, s. 157.

[294] Por. F. Hallberg, *Open source entertainment – how media and entertainment might thrive in a networked economy* [online], [dostęp: 1.06.2012]. Dostępny w WWW: http://www.brightmedia.se/open%20entertainment.pdf, s. 1.

[295] Por. Ibidem.

Człowiek uczy się przez naśladownictwo, a do tego potrzebne jest wskazywanie wzorów. Komunikacja sieciowa znacznie ułatwiła przekazywanie własnych idei, które, po zyskaniu aprobaty, szybciej mogą się rozchodzić pomiędzy ludźmi. Poza pomysłami tworzy się także takie narzędzia, które ułatwiają społeczną wymianę. Są one proste w obsłudze, dlatego homo ludens może bez trudu to zadanie realizować.

Dlaczego jeszcze warto polecać aplikacje OSS? Przede wszystkim dlatego, że gwarantują one możliwość swobodnego wykorzystywania ich funkcjonalności w pełni, do własnych potrzeb i we własnej pracy. Oczywiście można skorzystać z gotowych rozwiązań, tj. grać w gry OSS, ale można też przy użyciu narzędzi OSS tworzyć nowatorskie projekty. Idzie nie tylko o tworzenie gier, lecz również np. filmów. Pracując na programach open source'owskich przeznaczonych do tworzenia animacji można zmontować i wykreować film animowany. Jednym z takich jest *Big Buck Bunny*[296], stworzony w całości w aplikacji Blender[297]. Innym przykładem filmu wykonanego przy użyciu Blendera jest *Elephants Dreams*[298].

W pojemnej kategorii „rozrywka" umieścić można w zasadzie większość aplikacji, które przydają się w codziennej pracy na komputerze, a zatem programy do edycji i tworzenia materiałów filmowych, wizualnych, audialnych oraz te przeznaczone do odczytywania ebooków[299]. Poza aplikacjami pozwalającymi odczytywać popularne formaty plików warto wskazać jeszcze program Bookworm (http://bookworm.oreilly.com/), który umożliwia odczyt plików zapisanych w formacie ePub.

Podsumowując: sam ruch open source pełni funkcje rozrywkowe. Wśród aplikacji należy wskazać gry, aplikacje ułatwiające kontakt z wytworami kultury (filmem, muzyką, literaturą) oraz narzędzia umożliwiające tworzenie programów i różnego rodzaju dzieł (utworów).

---

[296] Film dostępny jest w serwisie You Tube. *Big Buck Bunny animation (1080p HD)* [online], [dostęp: 1.06.2012]. Dostępny w WWW: http://www.youtube.com/watch?v=XSGBVzeBUbk.

[297] Opracowując akapit, wykorzystałem pracę: C. Josephes, *Open source entertainment* [online], [dostęp: 1.06.2012]. Dostępny w WWW: http://news.oreilly.com/2008/06/open-source-entertainment.html. Program Blender można pobrać ze strony: http://www.blender.org/.

[298] *Elephants Dreas [HD] FULL MOVIE short film (2006)* [online], [dostęp: 1.06.2012]. Dostępny w WWW: http://www.youtube.com/watch?v=eFQxRd0isAQ.

[299] Por. S. Dean, *10 free, open source digital entertainment resources and roundups* [online], [dostęp: 1.06.2012]. Dostępny w WWW: http://ostatic.com/blog/10-free-open-source-digital-entertainment-resources-and-roundups.

Jako przykład może posłużyć gra strategiczna czasu rzeczywistego 0 A. D. Jest to gra wojenno-ekonomiczna osadzona w realiach historycznych. Gracz ma za zadanie rozwijanie wybranej cywilizacji (m.in. Rzymianie, Kartagińczycy, Celtowie, Persowie) zarówno pod względem gospodarczym, jak i militarnym. Gra oferuje standardowe funkcjonalności.

Fot. 20. Logo gry 0 A.D.[300].

Nazwa własna	0 A.D.
Numer wersji	13 alpha
Data publikacji	2 kwietnia
Data ściągnięcia	5 sierpnia 2013
Adres URL	http://play0ad.com/

Inne aplikacje:

Bos Wars (http://www.boswars.org) – gra strategiczna czasu rzeczywistego.

Enigma (http://www.nongnu.org/enigma/) – puzzle.

FlightGear (http://www.flightgear.org/) – symulator lotniczy.

Freeciv (http://freeciv.wikia.com/wiki/main_page/) – turowa gra strategiczna.

MegaGlest (http://megaglest.org/) – gra strategiczna czasu rzeczywistego.

OpenTTD (http://www.openttd.org/) – symulator metropolii.

[300] Logo pobrano z: *0 A.D. (video game)* [online], [dostęp: 5.08.2013]. Dostępny w WWW: http://en.wikipedia.org/wiki/0_A.D._%28video_game%29.

Red Eclipse (http://redecplise.net/) – FPS (ang. First Person Shooter), tzw. strzelanka.

SokoSolve (http://sokosolve.sourceforge.net/) – puzzle.

SuperTux (http://supertux.lethargik.org/) – dwuwymiarowa gra platformowa.

The Battle for Wesnoth (http://www.wesnoth.org/) – turowa gra strategiczna.

5.8. APLIKACJE PRZYDATNE W PRACY BIBLIOTECZNEJ

Działalność biblioteczna w sposób niemalże naturalny wiąże się z ideałami ruchu open source choćby dlatego, że u jej podłoża leży „uwalnianie" wszelkich informacji i treści[301]. Rola bibliotek polega na udostępnianiu wszelkich kwantów wiedzy. Wcześniej niekiedy trzeba je zgromadzić w macierzystej placówce lub skierować użytkownika do innych bibliotek, ewentualnie wypożyczyć lub uzyskać dostęp do zasobów z innych instytucji. Każdorazowo idzie więc o upowszechnianie treści. Idealną sytuacją byłoby, gdyby klient biblioteki lokalnie miał wgląd we wszystko, co znajduje się w bibliotekach globalnie rozproszonych. Poniekąd próbuje się to realizować poprzez masową digitalizację zbiorów oraz ich udostępnienie *via* Internet. Cele praktyki bibliotekarskiej są więc zbieżne z celami ruchu wolnego i otwartego oprogramowania. Jedne i drugie nastawione są na uwalnianie danych, informacji, treści. Ruch open source jest przez to sojusznikiem bibliotek, zwłaszcza publicznych i naukowych. Obydwa typy działalności mają podobną filozofię, m.in. promowanie nauki oraz uczenie się poprzez swobodny dostęp do wiedzy.

Po drugie, działalność biblioteczna jest *de facto* działalnością non profit, a więc obniżanie i niwelowanie kosztów własnych zapewnia bibliotekom nie tylko oszczędności, tak kluczowe w dzisiejszym świecie, lecz także, a może przede wszystkim, pozwala lepiej spożytkować uzyskane i zaoszczędzone środki, np. na zakup zbiorów. Zadanie to można zrealizować, wyposażając placówki biblioteczne w oprogramowanie OS, poczynając od systemu operacyjnego, poprzez pakiet biurowy, na aplikacjach służących do zarządzania informacją czy multimedialnych kończąc.

---

[301] Por. A. Rhyno, *Using open source systems for digital libraries*, Westport 2004, s. XIV.

Już od połowy lat 90. w amerykańskich bibliotekach zaczęły pojawiać się wolne i otwarte aplikacje wspierające procesy automatyzacji. Nie były to jeszcze programy przeznaczone do zadań *stricte* bibliotecznych, ile raczej programy komputerowe ogólnego przeznaczenia, np. systemy operacyjne. Dopiero pod koniec XX wieku działania ukierunkowano na stworzenie narzędzi konkretnie bibliotekarskich[302].

Po trzecie wreszcie, dzisiejsza działalność biblioteczna (katalogi biblioteczne, zintegrowane systemy biblioteczne, aplikacje ułatwiające działania promocyjne itp.) w dużej mierze opiera się o wykorzystywanie rozmaitych technologii informacyjnych i komunikacyjnych. Bez technologii trudno byłoby realizować podstawowe zadania, tj. gromadzenie, opracowywanie i udostępnianie zbiorów. W wielu krajach infrastruktura technologiczna jest nieodpowiednia, a komercyjne rozwiązania są zbyt kosztowne, co wpływa hamująco na rozwój podstawowej praktyki bibliotekarskiej. Dlatego warto sięgać do rozwiązań wolnych i otwartych, które stają się remedium niwelującym główne problemy[303].

Do wdrażania rozwiązań open source w działalności bibliotecznej niezbędna jest specjalistyczna wiedza i umiejętności. Programy OS można i powinno się dostosowywać do określonej pracy. Specyficzną sferą działalności jest właśnie bibliotekarska, toteż personel powinien zostać wyposażony w niezbędne umiejętności, które pozwoliłyby mu wprowadzać technologie OS do realizacji codziennych zadań. Dziś w większości przypadków biblioteki ograniczają się do zakupu oprogramowania własnościowego, co nie oznacza, że jest ono przez zespół biblioteki użytkowane prawidłowo i z zastosowaniem wszystkich funkcji. Ruth Samuels i Henry Griffy sugerują, z czym wypada się zgodzić, że zasadniczo rozwiązania open source funkcjonują bez profesjonalnego informatycznego wsparcia, a także nie towarzyszą im szkolenia w zakresie funkcjonalności danej aplikacji. Oszczędności ekonomiczne, które na pierwszy rzut oka są pokaźne, mogą w konsekwencji przerodzić się w konieczność poniesienia znacznych opłat za pomoc techniczną ze strony zewnętrznych podmiotów[304]. Dla przykładu, za Johnem Hellingiem można podać, że biblioteka

---

[302] Akapit na podstawie pracy: K. Wasiucionek, op. cit., s. 13-14.

[303] Warto w tym miejscu zapoznać się z działaniami prowadzonymi przez fundację EIFL (*Electronic Information for Libraries)* m.in. na polu FOSS. Por. *EIFL* [online], [dostęp: 02.04.2014]. Dostępny w WWW: http://www.eifl.net/foss.

[304] Por. R. G. Samuels, H. Griffy, *Evaluation open source software for use in library initiatives. A case study involving electronic publishing*, „Libraries and the Academy" 2012, vol. 12, no 1, s. 44.

publiczna Bloomfield-Eastern Greene County przechodząc z komercyjnego systemu bibliotecznego na Evergreen, zaoszczędziła czternaście tysięcy dolarów. Dodatkowo zniknęło ograniczenie co do liczby komputerów, na których można było oprogramowanie instalować, co znacznie usprawniło pracę biblioteki[305]. Podkreślę wyraźnie, że pewne koszty należy ponieść. Nie będą to jednak środki finansowe przeznaczone na zakup produktów programistycznych, lecz czas i praca, które należy przeznaczyć na rozwój danego programu, jego utrzymanie i szkolenia w zakresie jego funkcjonalności[306].

Open source to nie tylko sposób tworzenia oprogramowania, lecz także, a może przede wszystkim, tworzenie społeczności skupionych wokół zadań, które razem chcą realizować w zgodzie z ideałami ruchu open source. Open source'owska społeczność biblioteczna zawiązana jest m.in. przy serwisach Oss4lib (http://www.oss4lib.org/) oraz Code4Lib (http://code4lib.org/)[307]. Celem ich działania, zwłaszcza serwisu Oss4lib, jest stworzenie lepszych i wolnych systemów bibliotecznych, a w prowadzonych działaniach można wyszczególnić takie obszary tematyczne, jak OPAC-i, zintegrowane systemy biblioteczne, narzędzia bibliograficzne, protokoły, zarządzanie metadanymi, systemy informacyjno-wyszukiwawcze[308]. W ten sposób biblioteki znalazły się w gronie podmiotów zaangażowanych w rozwój oprogramowania. Tym samym też granica pomiędzy działalnością biblioteczną a informacją naukową i systemami informacyjno-wyszukiwawczymi uległa nieco zatarciu, scalając te obszary w ramach jednego modelu[309].

---

[305] Por. J. Helling, *Saving by sharing. Using open-source and shared catalogs to do more with less*, [in:] *The frugal librarian. Thriving in tough economic times*, ed. C. Smallwood, Chicago 2011, s. 121.

[306] Por. C. F. Intner, *Homework help from the library. In person and online*, Chicago 2011, s. 67. Na marginesie dodam, że Carlo Daffara oszacował, iż europejska ekonomia stosując rozwiązania open source oszczędza rocznie około 114 miliardów euro. Por. G. Hillenius, *Contribution of open source to Europe's economy. 450 billion per year* [online], [dostęp: 26.03.2014]. Dostępny w WWW: https://joinup.ec.europa.eu/ews/contribution-open-source-europes-economy-450-billion-year.

[307] Por. J. K. Samyal, S. Sumi, *Creation of online library using KOHA open source software. A case study of Chitkara University*, [in:] *Open-source solution in education. Theory and practice*, ed. J. B. Browning, Santa Rosa 2010, s. 182.

[308] Por. A. Pyati, *Open source software and libraries*, [in:] *Information technology in librarianship. New critical approaches*, ed. G. J. Leckie, J. Buschman, Westport 2009, s. 209.

[309] Por. A. K. Pace, *The ultimate digital library. Where the new information players meet*, Chicago 2003, s. 21.

Warto również wspomnieć o założonej w 2005 roku przez społeczność bibliotekarzy firmie LibLime (http://www.liblime.com/). Celem jej działania jest pomoc pracownikom bibliotek naukowych, publicznych, specjalnych oraz muzeów, szkół, organizacji non profit, przedsiębiorstw itp. we wdrażaniu rozwiązań open source'owskich[310].

Poza tym, różne organizacje non profit podejmują podobne do bibliotecznych przedsięwzięcia, korzystając z dorobku teoretyczno-praktycznego bibliotek. Jako przykład dobrej praktyki można wskazać inicjatywę Internet Archive, w ramach której w 2003 roku przy udziale wielu bibliotek narodowych rozpoczęto tworzenie narzędzi open source'owskich oraz standardów formatów przeznaczonych do archiwizowania Webu[311].

Wśród bibliotecznych aplikacji open source wylicza się repozytoria cyfrowe, np. Digital Asset Factory, DSpace, Fedora[312], CLOCKSS (Lots of Copies Keep Stuff Safe)[313], specjalne metawyszukiwarki, np. CUFTS, LibraryFind, VuFind, katalogi OPAC, np. Blacklight, Kochief (wcześniej Fac-Back-OPAC), MARC Module for Drupal, Scriblio, SOPAC[314], MicroLCS[315] oraz zintegrowane systemy biblioteczne, np. Evergreen, KOHA[316], OPALS (OPen-source Automated Library management System)[317], FireFly, Avanti, WEBLIS[318], GSDL (The Greenstone Digital Li-

---

[310] Por. K. Wasiucionek, op. cit., s. 15.

[311] Por. G. Mohr, *Archival tools to match the Web. Open, international, comprehensive*, [in:] *Asian digital libraries. Looking back 10 years and forging new frontiers. 10th International Conference on Asian Digital Libraries, ICADL 2007, Hanoi, Vietnam, December 10-13, 2007. Proceedings*, ed. D. Goh et al., Berlin, New York 2007, s. 7-8.

[312] Por. A. Kozbial, *Sorting through digital preservation systems*, [in:] *More technology for the rest of us. A second primer on computing for the non-IT librarian*, ed. N. Courthey, Santa Barbara 2010, s. 121.

[313] Por. A. Crosetto, *The use and preservation of e-books*, [in:] *No shelf required. E-books in libraries*, ed. S. Polanka, Chicago 2011, s. 132.

[314] Por. A. Kozbial, op. cit., s. 121.

[315] Por. J. K. Samyal, S. Sumi, op. cit., s. 183.

[316] Implementację systemu KOHA w ramach krakowskiego konsorcjum bibliotek omówiono w artykule: I. Drabik, R. Kopaczka, *Przystosowanie i implementacja otwartego systemu bibliotecznego Koha*, „Biuletyn EBIB" 2014, nr 2 [online], [dostęp: 02.04.2014]. Dostępny w WWW: http://open.ebib.pl/ojs/index.php/ebib/article/view/227.

[317] Por. P. J. Newberg, *Cataloging for non-English-speaking and preliterate children*, [in:] *Cataloging correctly for kids. An introduction to the tools*, ed. S. S. Intner, J. F. Fountain, J. R. Weihs, D. A. Fritz, J. Beall, Chicago 2011, s. 178.

[318] Por. J. K. Samyal, S. Sumi, op. cit., s. 186-188.

brary System)[319], Mateusz[320], a także przeznaczone dla małych bibliotek Emilda i OpenBiblio, brazylijska Gnuteca, filipiński PhpMyLibrary, czy francuski PMB[321]. Aplikacje można pobierać, modyfikować i dowolnie ich używać. Trzeba jednakże zaznaczyć, że w przypadku OPALS użytkownik ponosi roczną opłatę za jego użytkowanie, natomiast nie płaci za sam program[322]. W sieci, jak również w formie drukowanych publikacji, dostępne są podręczniki dokładnie wprowadzające w pracę z poszczególnymi wymienionymi powyżej aplikacjami.

Warto wskazać jeszcze oparty na Ubuntu system operacyjny Digitlab, na który, poza samymi narzędziami systemowymi, składa się zestaw wolnych i otwartych aplikacji niezbędnych w procesie digitalizacji zbiorów bibliotecznych. Digitlab powstał w ramach projektu ACCESS IT Plus. Posiada on również polską wersję językową. Obraz ISO można pobrać ze strony: http://dl.psnc.pl/download/digitlab/digitlab-1.0-desktop-i386.iso[323].

Niektórzy bibliotekarze słusznie chyba zauważają, że oprogramowanie open source kojarzone jest w środowisku bibliotekarskim z hakerami ubranymi w podarte koszulki, którzy w garażu tworzą przeznaczone tylko dla siebie programy. Utożsamianie open source z hakerami daje się uzasadnić i usprawiedliwić, jednak negatywny obraz hakera utrwalony w społecznej świadomości nie jest zgodny z rzeczywistością. Ciąży on jednak częściowo na oprogramowaniu OS, skutecznie zniechęcając do wykorzystywania rozwiązań OS w codziennej działalności[324].

Warto w tym kontekście przywołać wyniki badań przeprowadzonych w latach 2007-2012 w środowisku bibliotecznym, choć przeprowadzający je podkreślili, że mogą one posłużyć do obrony skrajnych tez. Badania dostarczyły bowiem dowodów, że w bibliotekach powoli rośnie zainteresowanie rozwiązaniami OS, jeśli idzie o systemy biblioteczne. Z drugiej jednak stro-

---

[319] Por. M. M. Hasan, K. Takeuchi, H. Isahara, V. Sornlertamvanich, *Digital libraries in Asian languages – A TCL initiative,* [in:] *Digital libraries. Technology and management of indigenous knowledge for global access,* ed. T. M. T. Sembok, H. B. Zaman, H. Chen, S. Urs, S. H. Myaeng, Berlin, London 2003, s. 365-372.

[320] Por. K. Wasiucionek, op. cit., s. 22.

[321] Por. E. Balnaves, *Open source library management systems. A multidimensional evaluation,* „Australian Academic & Library Research" 2008 vol. 39, no 1, s. 3.

[322] Por. D. Webber, A. Peters, *Integrated library systems. Planning, selecting, and implementing*, Santa Barbara 2010, s. 7.

[323] Opis systemu Digitlab na podstawie: B. Wróż, *Digitlab* [online], [dostęp: 8.08.2013]. Dostępny w WWW: http://dl.psnc.pl/2012/09/23/digitlab/.

[324] Uwagi zawarte w akapicie uzyskałem z książki: A. Kozbial, op. cit., s. 121.

ny poziom wykorzystania OSS jest w tym zakresie bardzo przeciętny, gdyż utrzymują się jednocześnie opinie, że tego typu OSS nie będzie w przyszłości szeroko implementowane w działalności bibliotecznej[325]. Odpowiedź na pytanie, dlaczego produkty OS tak wolno rozchodzą się wśród użytkowników, jest prosta: nie ma żadnej wielkiej akcji promocyjno-reklamowej, na którą określony podmiot przeznaczyłby znaczne środki pieniężne, jak ma to miejsce w przypadku podobnych towarów komercyjnych[326].

Implementacja technologii open source w bibliotekach prowadzi do wyłaniania się zjawiska, które można określić mianem otwartych bibliotek (ang. *open libraries*)[327]. Pozostaje mieć nadzieję, że w świecie coraz bardziej zdominowanym przez restrykcyjne prawo, idea otwartości będzie znajdować coraz szersze zastosowanie, zwłaszcza w środowisku naukowej działalności uniwersyteckiej oraz, co za tym idzie, bibliotecznej.

Niemalże wszystkie aplikacje OS mogą znaleźć zastosowanie w działalności bibliotekarskiej, nie tylko te przeznaczone do zadań czysto bibliotecznych (np. katalogi online), lecz wszystkie programy komputerowe. Działalność biblioteki to nie tylko praca na zapleczu ze zbiorami (informacji, wiedzy, treści), to także, a może przede wszystkim, frontowa praca z użytkownikiem. Od rezultatów uzyskanych w relacji biblioteka – klient zależy przecież powodzenie przedsięwzięcia, jakim jest biblioteka. Praca z użytkownikiem polega nie tylko na udostępnianiu zbiorów i dostarczaniu potrzebnych kwantów wiedzy i informacji, ale również na informowaniu m.in. o fundamentach ruchu open source, dostępnych aplikacjach OS itp. Niezwykle ważne jest, aby w świecie zmierzającym małymi krokami w kierunku tzw. społeczeństwa wiedzy i informacji, biblioteki przyłączały się do działań na rzecz otwartości zasobów i źródeł informacji, wiedzy, treści. Jedną z możliwych dróg realizacji tego zadania jest propagowanie wolnego i otwartego oprogramowania oraz zachęcanie do stosowania zasad towarzyszących OS w codziennym życiu, także w sferach na pierwszy rzut oka dalekich od obszaru produktów z branży IT.

W odczuciu bibliotekarzy może funkcjonować również takie przekonanie, że użytkowanie OSS wymaga znajomości kodu źródłowego programu, bez której nie ma możliwości włączania się w jego propagowanie.

---

[325] Przedstawione wyniki badań w opraciu o: M. Breeding, A. Yelton, *Librarians' assessments of automation systems. Survey results, 2007-2010*, Chicago 2011, s. 21.

[326] Por. M. Lesk, *Understanding digital libraries*, Amsterdam, Boston 2005, s. 172.

[327] Por. D. R. Miller, K. S. Clarke, *Putting XML to work in the library. Tools for improving access and management*, Chicago 2003, s. 97.

Zdaniem Nicole Engard założenie to jest błędne. Jeżeli bowiem tworzone są programy OS sprofilowane na potrzeby bibliotekarskie, to przecież właśnie bibliotekarze powinni programy te przetestować, ponieważ to oni mają je później użytkować. Poza tym, bibliotekarze jako osoby, które zawodowo zajmują się zarządzaniem informacją, są predestynowani do testowania różnych rozwiązań z obszaru technologii informacyjnych, nie tylko *stricte* bibliotecznych (większość programów OSS może znaleźć zastosowanie w działalności bibliotecznej). Kolejną kwestią jest tworzenie instrukcji użytkowania aplikacji. Zdaniem Engard, najlepsze instrukcje mogą przygotować ci, którzy dany program przetestowali (i tu widzi zadanie dla bibliotekarzy), ale także ci, którzy znają się na zarządzaniu informacją, czyli wiedzą, jak skutecznie opracować podręcznik użytkowania programu. Osoby takie mogą również wskazać, jak poprawić funkcjonalność menu danego programu tak, aby było ono, a przez to i cały program, przyjazne użytkownikowi[328].

Tym sposobem przed w pełni wykwalifikowanymi pracownikami bibliotek malują się ogromne i ciekawe perspektywy pracy. Można sobie wyobrazić specjalistów bibliotecznych od konkretnych grup programów, których celem będzie zgłaszanie uwag związanych z funkcjonalnościami aplikacji (np. rozwiązania nawigacyjne) czy też pomaganie w opracowywaniu podręczników i kursów (np. online) obsługi programów itp.

Inne aplikacje:

Bib Citer (http://sourceforge.net/projects/bibciter/) – menedżer bibliografii.

Internet Archive Bookreader (https://openlibrary.org/dev/docs/bookreader) – program do czytania online zeskanowanych książek.

JabRef (http://jabref.sourceforge.net/) – menedżer bibliografii.

Kete (http://katipo.co.nz/software/kete.html) – program do tworzenia bibliotek, archiwów i repozytoriów cyfrowych.

NewGenLib NGL (http://www.verussolutions.biz/web/) – zintegrowany system biblioteczny.

ReservesDirect (https://code.google.com/p/reservesdirect-csu/) – oprogramowanie do zarządzania zasobami bibliotecznymi (wypożyczanie, rezerwowanie).

The Collection Workflow Integration System CWIS (https://scout.wisc.edu/cwis) – program do zarządzania danymi.

---

[328] Por. N. C. Engard, *Practical open source software for libraries*, Oxford 2010, s. XXIII.

The Library à la Carte (http://alacart.library.oregonstate.edu) – system zarządzania treścią (CMS).

Userful PreBook™ (http://userful.com/products/userful-pre-book) – oprogramowanie do zarządzania zasobami bibliotecznymi (wypożyczanie, rezerwowanie).

Web User Booking System WUBS (http://sourceforge.net/projects/wubs/) – oprogramowanie do zarządzania zasobami bibliotecznymi (wypożyczanie, rezerwowanie).

* * *

Na rynku technologii informacyjnych funkcjonuje ogromny, bo ponad półmilionowy zbiór aplikacji OS. Niektóre są już dopracowane, inne – w trakcie testów, inne jeszcze – w fazie projektowania. To bogactwo znacznie utrudnia, a w zasadzie uniemożliwia w ogóle, zaprezentowanie w formie podręcznikowoprzewodnikowej choćby reprezentatywnej próbki tej grupy programów. Rozdział niniejszy nie pretendował też do roli wyboru najważniejszych aplikacji OS. Jak już pisałem, powstał w celach porządkowych i informacyjnych, aby wskazać przykładowe programy, a także po prostu wykazać obecność OSS na konkretnych polach, biorąc dodatkowo pod uwagę fakt, że opublikowana dziś w statycznej formie książki lista programów OS (nawet wraz z ich szczegółowym omówieniem), po chwili może mieć już tylko wartość historyczną. Rynek IT jest jednym z bardziej dynamicznie rozwijających się, toteż próba ujęcia go w sztywne ramy drukowanych stron nie może się udać.

Na zakończenie dodam, że rozdziału tego w ogóle mogłoby nie być. Jedynie logiczna, konsekutywna spójność wypowiedzi wymagała wyliczenia chociaż kilku aplikacji OSS. Wiedząc już, że programy OS są obecne i stale ich przybywa, a także znając podwaliny ruchu open source, jego determinanty, założenia i różne realizacje, mogę przejść w ostatniej części książki do podsumowania, zakończenia i nakreślenia dalszych perspektyw rozwoju tego sektora.

6. ZAKOŃCZENIE - PERSPEKTYWY

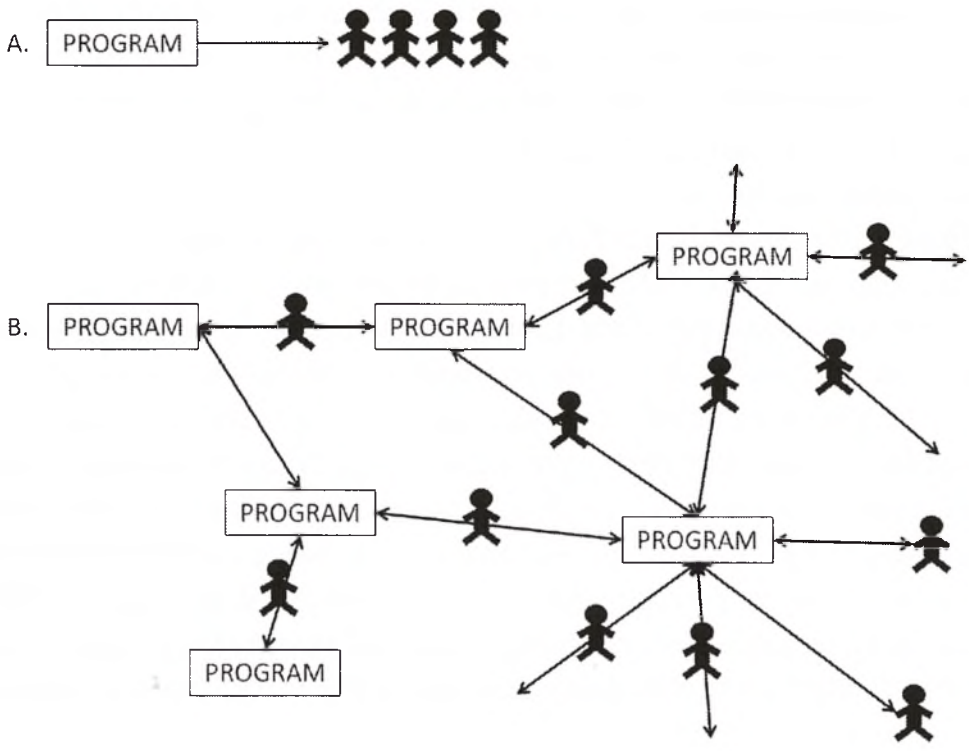
W ostatnim, podsumowującym rozdziale postaram się przedstawić perspektywy rozwoju i dalszego funkcjonowania ruchu wolnego i otwartego oprogramowania. Do tego celu posłużę się w znacznej mierze literaturą przedmiotu (głównie angielskojęzyczną), w której reprezentanci różnych gałęzi wiedzy i praktycznej działalności zaprezentowali rozmaite opinie na temat tego zjawiska, co – mam nadzieję – doprowadzi do uzyskania jasnej konkluzji.

Obrońcy OSS, np. Brian Fitzgerald, wskazują, że reprezentuje ono tak istotną zmianę w modelu tworzenia oprogramowania w ogóle, że jest w stanie rozwiązać problem tzw. kryzysu oprogramowania, który dostrzega się zwłaszcza w grupie rozmaitych systemów. W grupie tej odnotowuje się m.in. takie bolączki, jak to, że ten typ software'u tworzy się bardzo długo, jest przy tym bardzo kosztochłonny, a gdy zostanie dostarczony odbiorcom, nie działa tak, jak powinien. Z kolei systemy OSS tworzone są dużo szybciej, taniej, a wszystkie błędy i niedociągnięcia usuwane są w zasadzie natychmiastowo. Stąd apologeci nurtu OS stwierdzają, że jest to przyszłościowy model tworzenia programów, który może (tylko ten w zasadzie) przyczynić się do budowania społeczeństwa informacyjnego[329].

Powszechnie wiadomo, że każdy program komputerowy (a system operacyjny w szczególności) zawiera błędy (tzw. *bugs*) i że nie ma takich programów, które zawierałyby wszystkie niezbędne dla użytkow-

[329] Por. B. Fitzgerald, *Has open source software a future?* [in:] *Perspectives on free and open source software*, ed. J. Feller, B. Fitzgerald, S. A. Hissam, K. R. Lakhani, Cambridge 2005, s. 93.

nika funkcje. Innymi słowy, dany program posiada niedociągnięcia (w pewnych aspektach działa nieprawidłowo) oraz brakuje mu różnych funkcjonalności, a więc zawsze można coś poprawić, dodać, uzupełnić, skorygować, ulepszyć itp. Używana aplikacja znajduje się zatem w nieustannym procesie doskonalenia. Tworzy się nowe wersje, usuwa usterki, kolportuje poprawki. Aby program zmodyfikować, trzeba mieć dostęp do kodu źródłowego (copyleft)[330]. Poza tym, na mocy prawa musi być dopuszczone wprowadzanie do obiegu poprawek do programu. Jeżeli prawo (copyright) zamyka drogę do rozpowszechniania kolejnych działających skuteczniej wersji programu, to *de facto* znacznie utrudnia i spowalnia proces korygowania błędów, a w rzeczywistości w ogóle uniemożliwia poprawienie niedociągnięć i usunięcie ograniczeń. Rozwój oprogramowania w modelu zamkniętym i otwartym można zobrazować następującym schematem (rys. 6).



Rys. 6. Rozwój oprogramowania w modelu zamkniętym i otwartym [opracowanie własne].

[330] Na zawieranie przez program błędów zwracał uwagę: K. Siewicz, *Towards an improved regulatory framework of free software. Protecting user freedoms in a world of software communities and eGovernments*, Leiden 2010, s. 22-23.

Sytuacja A jest ilustracją modelu własnościowego (zamkniętego), gdzie komunikacja przebiega jednokierunkowo, od kogoś, kto stworzył i wprowadził program na rynek, do użytkowników, którzy mogą jedynie z programu korzystać. Mogą oni również zgłaszać uwagi do producenta, aby skorygował pewne błędy, lecz sami nie mają wglądu do kodu źródłowego, a nawet jeśliby go otrzymali, to nie mogą wprowadzać własnych poprawek i ich dalej dystrybuować.

Sytuacja B jest natomiast ilustracją otwartego modelu open source'owskiego, gdzie komunikacja zachodzi wielokierunkowo. Ktoś wprowadza do obiegu dany program OS, użytkownik może z programu skorzystać, wprowadzić doń określone poprawki czy po prostu go po swojemu zmodyfikować. Następnie owoc swojej pracy może dalej przekazać innym użytkownikom, którzy znowu mogą – w miarę własnych potrzeb i możliwości – aplikację tę modyfikować. Odkąd kod źródłowy jest otwarty, a licencja zapewnia możliwość (a nawet w niektórych sytuacjach niejako narzuca) modyfikowania kodu i upowszechniania rezultatów tych modyfikacji, odtąd wprowadzanie kolejnych wersji programu jest w zasadzie niczym nieograniczone. Na ilustracji zrezygnowano ze zdywersyfikowania kolejnych wersji programu, np. poprzez opis cyfrowy (program 1, program 2 itd.), gdyż użytkownik nie jest zobligowany do wprowadzania nowej wersji aplikacji. W jego gestii leży podjęcie lub nie tego zadania, może też np. ten sam program tylko poznawać i powielać.

Konieczna jest tutaj jedna uwaga. Rozsądek podpowiada, że kod programu nie zawsze powinien być swobodnie dostępny i otwarty dla wszystkich. Wystarczy pomyśleć o systemach bankowych, instytucjach finansowych, administracji publicznej, instytucjach medycznych itp. W tych przypadkach kod powinien być raczej niewidoczny i niedostępny dla wszystkich, jego dostępność stwarza bowiem zagrożenie dla ich prawidłowego funkcjonowania. Ciekawą w tym względzie tezę przedstawił Ross Anderson. Tak naprawdę nie ma znaczenia, czy program jest otwarty czy zamknięty. Z jednej bowiem strony otworzenie kodu pozwala poznać jego strukturę po to, aby się do niego włamać lub go uszkodzić. Z drugiej – otworzenie pozwala równie szybko stworzyć dogodne zabezpieczenia przed tymi atakami[331]. Jak się zdaje, oprogramowanie jest wtedy wartościowe, kiedy można je modyfikować, ulepszać,

[331] Por. R. Anderson, *Open and closed systems are equivalent (that is, in an ideal world)*, [in:] *Perspectives on free and open source software*, ed. J. Feller, B. Fitzgerald, S. A. Hissam, K. R. Lakhani, Cambridge 2005, s. 128.

poprawiać, rozbudowywać, usuwać niedociągnięcia itp., a zatem sektor OSS jest wielce obiecujący.

Warto tutaj dodać, że oprogramowanie open source jest bardziej odporne na ataki wirusowe. Zarażenie komputera pracującego na Linuksie jest znikome. W jednej ze statystyk podano, że funkcjonuje około tysiąca złośliwych narzędzi mogących zaszkodzić pecetowi z Linuksem, podczas gdy tego typu oprogramowania przeznaczonego dla popularnych komercyjnych systemów funkcjonuje około osiemnastu milionów. Różnice te biorą się z popularności obydwu grup systemów. Na rynku wciąż mało komputerów osobistych pracuje na Linuksie, natomiast na komercyjnych odpowiednikach pracuje ich zdecydowanie więcej. Sposób działania obydwu grup systemów jest też różna. W systemach komercyjnych wystarczy odebrać wiadomość mailową z wirusem i wykonać kliknięcie myszką. W Linuksie proces ten jest o wiele bardziej skomplikowany. Przed zainstalowaniem czegokolwiek trzeba podać hasło lub mieć specjalne prawa do uruchomienia jakiejkolwiek aplikacji[332].

Niektórzy informatycy dostrzegają niebezpieczeństwo w zaangażowaniu większej liczby programistów. Sugerują bowiem, że więcej osób skupionych na jedynym projekcie wcale nie musi oznaczać zredukowania czasu potrzebnego do napisania programu. Wręcz przeciwnie, większa liczba osób może ten czas znacznie i niepotrzebnie wydłużyć[333]. Można to pokrótce omówić na hipotetycznym projekcie X, który właśnie zainicjowano. Załóżmy, że podczas prac nad projektem X kilku programistów napisze kod przeznaczony dla tego samego komponentu X. Może się też okazać, że dany programista napisze więcej kodu niż jest to wymagane do stworzenia X lub jakiejś części X. Może się też pojawić wiele zbieżnych rozwiązań dla X, co będzie oznaczać konieczność sprawdzenia ich przez pozostałych programistów pracujących nad X. Niekiedy programiści stworzą kod dla już istniejących modułów X, tym samym dublując pracę. Stworzenie interfejsu dla aplikacji X stanowi kolejne wyzwanie, ponie-

---

[332] Pisząc akapit skorzystałem z pracy: A. Nowak, *Wirusy dla Linuksa? Nie daj się zastraszyć* [online], [dostęp: 7.08.2013]. Dostępny w WWW: http://di.com.pl/news/47738,4.html. Kevin Mitnick, guru w kwestiach bezpieczeństwa programów i danych, znany z wcześniejszych nielegalnych działań polegających na łamaniu rozmaitych zabezpieczeń systemów i programów, w swojej książce stwierdził, że głównym problemem jest i będzie czynnik ludzki. Por. K. Mitnick: *Sztuka podstępu*. Gliwice 2003, s. 21.

[333] Por. S. Rusovan, M. Lawford, D. L. Parnas, *Open source software development. Future or fad?* [in:] *Perspectives on free and open source software*, ed. J. Feller, B. Fitzgerald, S. A. Hissam, K. R. Lakhani, Cambridge 2005, s. 109.

waż każdy programista może zaprezentować swoje własne rozwiązanie, co wiąże się z większą ilością czasu na dyskutowanie, który interfejs jest lepszy. I choć wskazuje się to jako wielką barierę w tworzeniu aplikacji[334], to, według mnie, jest to przykładem czegoś odwrotnego. Taka wzmożona komunikacja sprzyja przecież znalezieniu rozwiązania idealnego. Piszący programista lub grupa (posiadająca zapewne jakiegoś lidera) są ograniczeni tym indywidualnym spojrzeniem i w zasadzie realizują zleconą pracę (lub własny zamysł). W takim modelu nietrudno o usterki. Jeden punkt widzenia jest z założenia ograniczony, a w ideały OS wpisana jest ich wielość. Projekty tworzone w szczerej współpracy zdają się rokować lepiej. Zresztą podobnie rzecz ma się z projektami komercyjnymi, gdzie pracuje grupa programistów, którzy otrzymują wynagrodzenie za wykonanie czegoś, co będzie działać. W obydwu przypadkach nie powinno być miejsca na oszukiwanie. Cel jest zbieżny – stworzyć coś, co będzie poprawnie funkcjonować. Choć realizuje się to zgodnie z innymi scenariuszami, gdyż inna jest organizacja pracy, to podporządkowane są one tej samej logice.

Przyjęcie perspektywy ekonomicznej (biznesowej) pozwala zauważyć, że nie ma wypracowanej strategii rozwoju OS. Zasilający projekty OS reprezentują swoje własne interesy, motywacje, wizje i każdy realizuje je po swojemu. Bez, nazwijmy to, scentralizowania całego procesu twórczego i zorganizowania systematycznej pracy, ruch open source może nie rozwijać się tak, jak komercyjne przedsięwzięcia[335].

Zdaje się jednak, że OSS jest bardziej konkurencyjne od PS. OSS zapewnia, że pochodzące z zewnątrz innowacyjne rozwiązania są adaptowane do projektów i tworzonych w izolacji wszelkich aplikacji. Wiele firm i organizacji korzysta z rozwiązań open source, jeśli nie w pełni, to choćby z elementów, które potem są wykorzystywane przy budowie PS, co dla obszaru open source oznacza pozytywny scenariusz. Open source wywarło głęboki wpływ na branżę oprogramowania (i inne) w ogóle. Otwarte standardy wykorzystywane są np. przy tworzeniu wszelkich startupów i mało kto wyobraża sobie funkcjonowanie w branży IT bez OSS. Oprogramowanie OSS stało się już pewnym standardem[336].

Biorąc pod uwagę fakt, że OSS i PS wzajemnie się stymulują, o czym była mowa w podsumowaniu rozdziału 3.1 (*Porównanie oprogramo-*

---

[334] Ibidem.

[335] Pisząc akapit powołałem się na pracę: B. Fitzgerald, *Has open source...*, s. 100.

[336] Akapit w oparciu o książkę: J. D. Campbell, T. Kreidl, D. Pace, op. cit., s. 12-16.

wania open source z oprogramowaniem własnościowym), należy przyjąć, że ta swoista zależność obydwu sektorów jest gwarantem dalszego rozwoju OSS. Wolne i otwarte oprogramowanie znajduje zastosowanie w rozwiązaniach komercyjnych jeszcze w takim sensie, co należy dobitnie podkreślić, że pozwala na swej podstawie rozpoczynać działalność gospodarczą. Chcąc założyć serwis internetowy służący e-commerce, przedsiębiorca może zakupić PS lub użyć OSS. Na początku prowadzenia działalności każde dodatkowe wydatki podnoszą znacznie koszty całego przedsięwzięcia, a zatem warto korzystać z rozwiązań darmowych. Rozważyć trzeba jeszcze fakt, że w tym kontekście darmowe bynajmniej nie oznacza gorsze, lecz równie dobre, a niekiedy i lepsze. Innym problemem pozostaje ograniczone wsparcie techniczno-informatyczne.

Gdyby wszelkie oprogramowanie uznać tylko za narzędzie, tak zdaje się chce ruch open source oraz OSI, a na pewno ruch związany z Fundacją Wolnego Oprogramowania, to wtedy można by swobodniej z niego korzystać. To tak, jak z narzędziami stolarskimi, które służą określonym celom. Co by się stało, gdyby zacząć je patentować? W pewnym momencie programy zaczęto patentować, przez co same narzędzia, *de facto* środki do określonych zadań, stały się celem samym w sobie. Program stał się celem w tym sensie, że wykonanie go, zostało wielkim komercyjnym przedsięwzięciem policzonym na uzyskanie dodatnich finansowych rezultatów. Zamiast skupiać się na przeznaczeniu tych narzędzi, zaczęto skupiać się na nich samych. Marketing i ogromne inwestycje w reklamę i promocję programów stają się głównym gwarantem powodzenia całego przedsięwzięcia. Stąd zapewne pierwsze wersje PS są pełne błędów i niedociągnięć, które dopiero w trakcie korzystania przez użytkowników wychodzą na jaw i są potem wraz z kolejnymi wersjami lub poprawkami usuwane. OSS z kolei powstaje tak samo, lecz zdjęcie ciężaru konieczności wypracowania zysku finansowego (zwrot poniesionych kosztów i zarobek) pozwala na to, aby – tak, jak zwykle się to z wszystkimi narzędziami dzieje – nieustannie je modyfikować i poprawiać, aż do uzyskania idealnego wzoru. Tworzący OSS nie musi się martwić o to, że oprogramowanie na początku będzie źle działać. Po to są inni członkowie społeczności, aby wspólnie pracować nad tymi narzędziami, tj. poprawiać je i ulepszać. Społeczność tworzy narzędzia z zamierzeniem wykorzystania ich do określonych, społecznie uzasadnionych zadań. Inna sprawa, że licencje open source zdejmują ciężar gwarancji prawidłowego działania kodu z aplikacji lub jej części, które są nimi licencjonowane.

Kolejnym problemem jest ten, na który zwrócił uwagę Brian Fitzgerald, czyli niewystarczające realizowanie projektów wertykalnie, tj. napisanie kodu pod konkretną aplikację i doprowadzenie do zbudowania tej aplikacji. Dużo więcej kodu transferuje się horyzontalnie, tj. tworzy się kod, który może zasilać różne projekty. W przeciwieństwie do ruchu ściśle komercyjnego wysiłek skupia się zasadniczo na tworzeniu kodu służącego rozwojowi programów, a nie na tworzeniu konkretnych programów[337]. Z drugiej jednak strony, np. Karl Popp wskazał, że w przyszłości wyłonią się nowe modele biznesowe oparte o działania open source, takie jak oprogramowanie open source na żądanie, oprogramowanie open source w środowisku chmury[338].

Z kulturowej perspektywy patrząc, można dostrzec jeszcze jeden problem natury socjologicznej. Wielu współtwórców ruchu OS przystąpiło do niego dlatego, że był on czymś funkcjonującym poza głównym nurtem, poza mainstreamem. W momencie, gdy coraz więcej zaczęto mówić o open source i inwestować w nie znacznie sumy pieniędzy, ruch ten stał się *de facto* częścią głównego nurtu. W konsekwencji może to doprowadzić do odwrotnego zjawiska i część osób zniechęcić do dalszej pracy na jego rzecz. W przeszłości prace nad projektami OS były prowadzone bardziej w ukryciu, nie przyciągały uwagi szerokich mas, przez co można je było wykonywać w spokoju. Dziś, gdy OS zyskuje coraz większy zasięg i coraz więcej osób zadaje pytania (*via* email, forum, grupy dyskusyjne itp.), osoby tworzące OS są coraz bardziej uwikłane w czasochłonne procesy komunikacyjne. Może się więc okazać, że dawniej prowadzone w spokoju prace przybierają formy bardziej stresujące. A nic tak nie zniechęca do działania, jak stres[339].

Rację miał zapewne Stallman, twierdząc, że nie wystarczy samo propagowanie idei wolnego oprogramowania, choć przyznał, że jest to pierwszy krok na tej drodze (po to też m.in. powstała właśnie niniejsza książka). Drugim natomiast jest praca na rzecz wolnego oprogramowania, a przez to dokładanie kolejnych cegiełek do nowej jakościowo budowli, którą jest OSS[340]. Dodałbym jeszcze krok trzeci, mianowicie: użytkowanie tych pro-

---

[337] Por. B. Fitzgerald, *Has open source...*, s. 102.

[338] Por. K. A. Popp, *Advances in software economics. A reader on business models and partnering*, Berlin 2011, s. 39.

[339] W akapicie przedstawiłem uwagi zawarte w pracy: Ibidem, s. 104-105.

[340] Por. S. Williams, *Free as in freedom. Richard Stallman's crusade for free software*, Sebastopol 2002, s. 119-120.

gramów. Wiele mówi się o samej idei otwartości, coraz więcej programów się tworzy, lecz wydaje mi się, że wciąż za mało aplikacji OSS się używa. Na tej drodze leżą ciągle te same przeszkody, bariery mentalne: podstawową jest natura ludzka i związane z nią obawa przed nowym, trudności w odzwyczajaniu się od dawnych nawyków, tj. korzystania z innych programów, niechęć uczenia się nowych funkcji i sposobów ich realizacji w nowych narzędziach, a co za tym idzie, mylna argumentacja deprecjonująca zalety OSS.

Można przywołać tutaj jeszcze wyniki badań z 2013 roku przeprowadzonych przez Linux Professional Institute Central Europe we współpracy z firmą CTS (Customized Training Solutions), dotyczących wykorzystania Linuksa w polskich firmach. Badania wykazały m.in., że większość respondentów: planowała w najbliższym roku zakupić serwery oparte na Linuksie oraz wdrożyć rozwiązania oparte na open source, już w czasie badań korzystała z baz danych opartych na narzędziach open source (mysql, postgresql), znaczna część pytanych planowała również zatrudnienie specjalistów od Linuksa, zwłaszcza administratorów systemów[341]. Prywatny sektor jest więc zainteresowany stosowaniem technologii informacyjnych open source, co jest dobrą wróżbą na najbliższe lata.

Jeśli idzie natomiast o sektor państwowy (budżetowy), to jako przykład dobrej praktyki można wskazać działania rządu estońskiego. Według danych Simm Sikkut z National ICT Policy Advicer Government Office z 2013 roku Estonia wdrożyła, m.in. współpracując z sektorem prywatnym, cały szereg rozmaitych technologii opartych na open source w obszarze spraw wewnętrznych i administracji, powszechnej edukacji, opieki zdrowotnej itd., uzyskując liczne korzyści (znaczną oszczędność pieniędzy, usprawnienie komunikacji pomiędzy obywatelami a urzędami, np. 95% deklaracji podatkowych realizowanych jest poprzez Internet)[342]. Przyglądając się rodzimemu rynkowi można zauważyć, że w drugim półroczu 2012 roku „skala wydatków na nowoczesne rozwiązania informatyczne podmiotów realizujących zadania publiczne podlegających uPzp jest w Polsce na tak wysokim poziomie (1/3 wartości całego rynku ICT),

[341] Por. *Raport na temat wykorzystania Linuksa w Polsce* [online], [dostęp: 27.03.2014]. Dostępny w WWW: www.lpice.eu/pl/strona-glowna/informacje-prasowe/raport-na-temat-wykorzystania-linuksa-w-polsce.html.

[342] Por. M. Olber, *Rząd Estonii polega głównie na wolnym oprogramowaniu* [online], [dostęp: 27.03.2014]. Dostępny w WWW: http://osworld.pl/rzad-estonii-polega-glownie-na-wolnym-oprogramowaniu/.

że w znacznej mierze kształtuje kondycję i kształt krajowego rynku informatycznego"[343]. Choć jednak, jak wyjaśniają twórcy raportu, w postępowaniach przetargowych wcale nie są wybierane produkty wolne i otwarte, mimo że są zdecydowanie tańsze i pozwalają uniknąć efektu *vendor lock-in*, czyli uzależnienia się od produktów jednego producenta[344]. Perspektywa maluje się jednak obiecująco. Zapotrzebowanie na rozwiązania IT jest ogromne, rynek również, nic (poza przyzwyczajeniami) więc nie stoi na przeszkodzie, aby w najbliższych latach chętniej sięgać po narzędzia z grupy FLOSS.

Ogrom rozwiązań komercyjnych i równie ogromny obszar rozwiązań niekomercyjnych pozwalają wyciągnąć odmienne wnioski. Programy można tworzyć w duchu open source. Mogą być one równie dobre, a nawet i lepsze od komercyjnych odpowiedników. Poza tym potencjał innowacyjny tkwiący w stale powiększającej się społeczności sieciowej zdaje się zapewniać nieskończone zasilanie nowymi pomysłami i rozwiązaniami. Procesy społeczne wskazują, że internauci chcą się dzielić i wymieniać potrzebnymi kwantami informacji i wiedzy, co rokuje pomyślnie na przyszłość.

* * *

Na zakończenie ośmielę się wyrazić nadzieję, że niniejsza książka choć po części może przyczynić się do wyjaśnienia niektórych kwestii związanych z nurtem open source. Gdyby jeszcze udało się jej wprowadzić w zagadnienia OS kogoś, kto wcześniej z tą problematyką się nie zetknął, to radość autora byłaby dużo większa. Celem książki było przybliżenie głównych i podstawowych kwestii związanych z open source w sposób możliwie przystępny. W opinii autora problemy przedstawione na kartach książki są wystarczające, aby osiągnąć zamierzony cel.

---

[343] A. Michałek-Budzicz, R. Brzychcy, R. Michalski, *Analiza rynku zamówień publicznych na narzędzia informatyczne w okresie od 1 lipca do 31 grudnia 2012 roku*, Poznań 2013 [online], [dostęp: 02.04.2014]. Dostępny w WWW: http://pppit.org.pl/publikacje/raport_2012-2013.pdf, s. 4.

[344] Por. Ibidem, s. 4-5.

BIBLIOGRAFIA:

1. *50 open source tools that replace popular educational apps* [online], [dostęp: 5.06.2012]. Dostępny w WWW: http://www.datamation.com/feature/50-Open-Source-Tools-That-Replace-Popular-Education-Apps-3888901.htm.
2. *A Guide to open source software for Australian government agencies* [online], [dostęp: 28.05.2012]. Dostępny w WWW: http://cc.com.au/sites/cc.com.au/files/A_Guide_to_Open_Source_Software.pdf.
3. *A sample of journals using Open Journal Systems* [online], [dostęp: 19.11.2012]. Dostępny w WWW: http://pkp.sfu.ca/ojs-journals.
4. *About Moodle* [online], [dostęp: 2.08.2013]. Dostępny w WWW: http://docs.moodle.org/25/en/About_Moodle.
5. Amant Kirk St., Still Brian, *Open source software. Technological, economic, and social perspectives*, Hershey 2007.
6. Anderson Ross, *Open and closed systems are equivalent (that is, in an ideal world)*, [in:] *Perspectives on free and open source software*, ed. Joseph Feller, Brian Fitzgerald, Scot A. Hissam, Karim R. Lakhani, Cambridge 2005, s. 127-142.
7. *Arduino playground* [online], [dostęp: 31.01.2012]. Dostępny w WWW: http://www.arduino.cc/playground/Projects/ArduinoUsers.
8. Avrin Leila, *Scribes, script and books. The book arts from antiquity to the renaissance*, Chicago 1991.
9. Babik Wiesław, *Informacja naukowa jako przedmiot zarządzania*, [w:] *Zarządzanie informacją w nauce*, pod red. D. Pietruch-Reizes, Katowice 2008, s. 33-49.
10. Bahareth Mohammad, *Isay. Kings of the internet*, 2010.
11. Balnaves Edmund, *Open source library management systems. A multidimensional evaluation*, „Australian Academic & Library Research" 2008, vol. 39, no 1, s. 1-13.

12. *Beagle board* [online], [dostęp: 27.03.2014]. Dostępny w WWW: http://beagleboard.org/.
13. Benkler Yochai, *The wealth of networks. How social production transforms markets and freedom*, New Haven, London 2006.
14. Bergenholtz Henning, Nielsen Sandro, Tarp Sven, *Introduction*, [in:] *Lexicography at a crossroads. Dictionaries and encyclopedias today, lexicographical tools tomorrow*, ed. Henning Bergenholtz, Sandro Nielsen, Sven Tarp, Bern 2009, s. 7-15.
15. Berners-Lee Tim, *Weaving the Web. The original design and ultimate destiny of the World Wide Web*, New York 2000.
16. Best Jr. Richard A., Cumming Alfred, *Open source intelligence (OSINT). Issues for Congress*, [in:] *Intelligence issues and developments*, ed. Terrance M. Paulson, New York 2008, s. 75-97.
17. Bijsterbosch Magchiel, van Godtsenhoven Karen, Hochstenbach Patric, Russell Rosemary, Vanderfeesten Maurice, *Framework for interoperability*, [in:] *Emerging standards for enhanced publications and repository technology. Survey on technology*, ed. Marjan Vernooy-Gerritsen, Amsterdam 2009, s. 107-193.
18. Bolter J. David, *Writing space. Computers, hypertext and the remediation of Print*, ed. 2., Mahwah 2009.
19. Breeding Marshall, Yelton Andromeda, *Librarians' assessments of automation systems. Survey results, 2007-2010,* Chicago 2011.
20. Campbell John D., Kreidl Tobias, Pace Douglas, *Why central IT must embrace open source*, [in:] *Open-source solution in education. Theory and practice*, ed. J. Burton Browning, Santa Rosa 2010, s. 1-20.
21. Carlson Robert H., *Biology is technology. The promise, peril, and new business of engineering life*, Cambridge 2010.
22. Castells Manuell, *Społeczeństwo sieci*, Warszawa 2010.
23. Ceruzzi Paul E., *A history of modern computing*, ed. 2, London 2003.
24. Chao Lee, *Utilizing open source tools for online teaching and learning. Applying Linux technologies*, Hershey 2009.
25. Coleman Sean, *Open-source as an alternative to commercial software. Final report 583*, Tempe 2009.
26. Colford Scot, *Free and open source software,* [in:] *More technology for the rest of us. A second primer on computing for the non-IT librarian*, ed. Nancy Courtney, Santa Barbara 2010, s. 109-124.
27. Couch Carl J., *Oral technologies. A cornerstone of ancient civilization?*, „The Sociological Quarterly" 1989, vol. 30, no 4, s. 587-602.
28. Couch Carl J., Maines David R., Chen Shing-Ling, *Information technologies and social orders*, New York 1996.
29. *Creative Commons Polska* [online], [dostęp: 25.01.2012]. Dostępny w WWW: http://creativecommons.pl/faq/#15.

30. Crosetto Alice, *The use and preservation of e-books*, [in:] *No shelf required. E-books in libraries*, ed. Sue Polanka, Chicago 2011, s. 125-134.
31. *Czym jest wolne oprogramowanie?* [online], [dostęp: 22.01.2012]. Dostępny w WWW: http://www.gnu.org/.
32. Daffara Carlo, *The small/medium enterprises guide to open source software*, ed. 4., 2009 [online], [dostęp: 28.05.2012]. Dostępny w WWW: http://smeguide.conecta.it/smeguide.pdf.
33. Dale Nell, Lewis John, *Computer science illuminated*, ed. 4., Sudbury 2011.
34. Das Subrata K., *High-level data fusion*, Boston 2008.
35. Dasgupta Subhasish, *Encyclopedia of virtual communities and technologies*, Hershey 2006.
36. Dean Sam, *10 free, open source digital entertainment resources and roundups* [online], [dostęp: 1.06.2012]. Dostępny w WWW: http://ostatic.com/blog/10-free-open-source-digital-entertainment-resources-and-roundups.
37. Deek Fadi P., McHugh James A. M., *Open source technology and policy*, Cambridge 2008.
38. *Definicja oprogramowania open source* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://www.free-soft.org/mirrors/www.opensource.org/docs/osd-polish.php.
39. Deily Mary-Ellen Phelps, *The education week guide to K-12 terminology*, San Francisco 2009.
40. DiBona Chris, Ockman Sam, Stone Mark, *Open sources. Voices from the open source revolution*, Beijing, Sebastopol 1999.
41. Dixon Rod, *Open source software law*, Norwood 2004.
42. Drabik Iwona M., Kopaczka Rafał, Szczudło Ilona, *Przystosowanie i implementacja otwartego systemu bibliotecznego Koha*, „Biuletyn EBIB" 2014, nr 2 [online], [dostęp: 02.04.2014]. Dostępny w WWW: http://open.ebib.pl/ojs/index.php/ebib/article/view/227.
43. Edwards Lewin A. R. W., *Open-source robotics and process control cookbook. Designing and building robust, dependable real-time systems*, Amsterdam, Boston 2005.
44. Egyedi Tineke M., van Wendel de Joode Ruben, *Standardization and other coordination mechanisms in open source software*, [in:] *Advanced topics in information technology standards and standardization research*, ed. Kai Jakobs, Hershey 2005, s. 71-90.
45. *EIAO* [online], [dostęp: 31.01.2012]. Dostępny w WWW: http://www.eiao.net/.
46. Eisenberg David J., *OASIS Open Document essentials. Using OASIS OpenDocument XML*, Airlie Beach 2005.
47. Eisenstein Elizabeth L., *On revolution and the printed word*, [in:] *Revolution in history*, ed. Roy Porter, Mikuláš Teich, Cambridge 1986, s. 186-205.

48. Eisenstein Elizabeth L., *The printing press as an agent of change*, Cambridge 1994.
49. Eisentein Elizabeth L., *The printing revolution in early modern Europe*, ed. 2., Cambridge 2005.
50. *Elektrobiblioteka* [online], [dostęp: 31.07.2013]. Dostępny w WWW: http://www.elektrobiblioteka.net/.
51. *Elektrobiblioteka / Electrolibrary* [online], [dostęp: 31.07.2013]. Dostępny w WWW: http://vimeo.com/47656204.
52. Engard Nicole C., *Practical open source software for libraries*, Oxford 2010.
53. *European Parliament wants Open Document exchange format for electronic business* [online], [dostęp: 2.08.2013]. Dostępny w WWW: http://press.ffii.org/Press%20releases/European%20Parliament%20wants%20Open%20Document%20exchange%20format%20for%20electronic%20business.
54. Feller Joseph, *Perspectives on free and open source software*, Cambridge, London 2005.
55. Feller Joseph, Fitzgerald Brian, *Understanding open source software development*, London 2002.
56. Fink Martin, *The business and economics of Linux and open source,* Upper Saddle River 2003.
57. Fitzgerald Brian et al., *Adopting open source software. A practical guide*, Cambridge 2011.
58. Fitzgerald Brian, *Has open source software a future?,* [in:] *Perspectives on free and open source software*, ed. Joseph Feller, Brian Fitzgerald, Scott A. Hissam, Karim R. Lakhani, Cambridge 2005, s. 93-106.
59. Gawrysiak Piotr, *Cyfrowa rewolucja. Rozwój cywilizacji informacyjnej*, Warszawa 2008.
60. Gawrysiak Piotr, *Cyfrowe wykluczenie treści*, [w:] *Informacja w sieci. Problemy, metody, technologie*, red. Barbara Sosińska-Kalata, Ewa Chuchro, Włodzimierz Daszewski, Warszawa 2006, s. 117-124.
61. German Daniel M., González-Barahona Jesús M., *An empirical study of the reuse of software licensed under the GNU General Public License*, [in:] *Open source ecosystems. Diverse communities interacting. 5th IFIP WG 2.13 International Conference on Open Source Systems, OSS 2009, Skövde, Sweden, June 3-6, 2009. Proceedings*, ed. Cornelia Boldyreff et.al., Berlin, New York 2009, s. 185-198.
62. Ghosh Dilip K., *Governance for development. Issues and strategies*, [in:] *Good governance. Initiatives in India*, ed. Etakula Vayunandan, Dolly Mathew. New Delhi 2003, s. 28-38.
63. Gilles James, Cailliau Robert, *How the Web was born*, New York 2000.
64. Goban-Klas Tomasz, *Cywilizacja medialna. Geneza, ewolucja, eksplozja*, Warszawa 2005.

65. Gogołek Włodzimierz, *Komunikacja sieciowa. Uwarunkowania, kategorie i paradoksy*, Warszawa 2010.
66. Goldman Ron, Gabriel Richard P., *Innovation happens elsewhere. Open source as business strategy*, Amsterdam, Boston 2005.
67. Golden Bernard, *Succeeding with open source*, Boston 2004.
68. Grabowska Marta, *Biblioteka cyfrowa w środowisku wirtualnym*, [w:] *Biblioteki cyfrowe. Projekty, realizacje, technologie*, red. Jadwiga Woźniak-Kasperek, Jerzy Franke, Warszawa 2007, s. 21-33.
69. Griffey Jason, *Gadgets and gizmos. Personal electronics and the library*, Chicago 2010.
70. Góralska Małgorzata, *Perspektywy e-booków w kontekście rozwoju komputerów jako urządzeń uniwersalnych i specjalistycznych*, [w:] *Biblioteka, książka, informacja, Internet 2010*, red. Zbigniew Osiński, Lublin 2010, s. 77-92.
71. Góralska Małgorzata, *Piśmienność i rewolucja cyfrowa*, Wrocław 2012.
72. Górniak-Kocikowa Krystyna, *Revolution and the library*, „Library Trends" 2001, vol. 49, no 3, s. 454-461.
73. Hahn Robert W., *Government policy toward open source software. An overview* [online], [dostęp: 28.01.2012]. Dostępny w WWW: http://www.brookings.edu/press/books/chapter_1/governmentpolicytowardopensourcesoftware.pdf.
74. Hallberg Fredrik, *Open source entertainment – how media and entertainment might thrive in a networked economy* [online], [dostęp: 1.06.2012]. Dostępny w WWW: http://www.brightmedia.se/open%20entertainment.pdf.
75. Harrer Andreas, Zeini Sam, Ziebarth Sabrina, *Visualisation of the dynamic for longitudinal analysis of computer-mediated social networks-concept and exemplary cases,* [in:] *From sociology to computing in social networks. Theory, foundations and applications*, ed. Nasrullah Memon, Reda Alhajj, Vienna, New York 2010, s. 119-134.
76. Hasan Maruf M., Takeuchi Kazuhiro, Isahara Hitoshi, Sornlertamvanich Virach, *Digital libraries in Asian languages – A TCL initiative,* [in:] *Digital libraries. Technology and management of indigenous knowledge for global access,* ed. Tengku M. T. Sembok, Halimah B. Zaman, Hsinchun Chen, Shalini Urs, Sung H. Myaeng, Berlin, London 2003, s. 365-372.
77. Hashagen Ulf, Keil-Slawik Reinhard, Norberg Arthur L., *History of computing. Software issues*, Berlin 2002.
78. *Helionica. Sieciowa encyklopedia informatyki* [online], [dostęp: 28.01.2012]. Dostępny w WWW: http://encyklopedia.helion.pl/.
79. Helling John, *Saving by sharing. Using open-source and shared catalogs to do more with less,* [in:] *The frugal librarian. Thriving in tough economic times,* ed. Carol Smallwood, Chicago 2011, s. 120-126.

80. Hillenius Gijs, *Contribution of open source to Europe's economy. 450 billion per year* [online], [dostęp: 26.03.2014]. Dostępny w WWW: https://joinup.ec.europa.eu/news/contribution-open-source-europes-economy-450-billion-year.
81. Hirst Nigel, Brocklebank Mike, Ryder Martyn, *Containment systems. A designed guide,* Rugby 2002.
82. Hobart Michael E., Schiffman Zachary S., *Information ages. Literacy, numeracy, and the computer revolution*, Baltimore 2000.
83. *How to find FOSS (Free Software and Open Source Software)* [online], [dostęp: 27.05.2012]. Dostępny w WWW: http://how-to.wikia.com/wiki/Howto_find_Free_Software_and_Open_Source_software_(FOSS).
84. *How to find open source software* [online], [dostęp: 16.05.2012]. Dostępny w WWW: http://sosopensource.com/34.html.
85. Innis Harold A., *Empire and communication*, Toronto 2007.
86. Intner Carol F., *Homework help from the library. In person and online*, Chicago 2011.
87. Jacob Matthias, Boneh Dan, Felten Edward, *Attacking by obfuscated cipher by injecting faults,* [in:] *Digital rights management. ACM CCS-9 workshop, DRM 2002, Washington, DC, USA, November 18, 2002. Revised papers,* ed. Joan Feigenbaum, Berlin 2003, s. 16-31.
88. Jakubowski Marcin, *Open-source blueprints for civilization* [online], [dostęp: 26.03.2014]. Dostępny w WWW: www.ted.com/talks/marcin_jakubowski.
89. Jarocki Mariusz, *Open Source – alternatywa w edukacji,* [in:] *Seminarium „Otwarte Zasoby Edukacyjne", Toruń 07.10.2009,* Elektroniczna Biblioteka Pedagogiczna SBP [online], [dostęp: 20.10.2009]. Dostępny w WWW: http://e-pedagogiczna.edu.pl/upload/file/zasoby/konferencje/torun/jarocki.pdf.
90. Jastrzębski Jerzy, *Edukacja medialna*, [w:] *Edukacja medialna. Teksty i preteksty*, red. Igor Borkowski, Wrocław 2004, s. 28-36.
91. Jełowicki Jan, *Zajęcia z podstaw informatyki* [online], [dostęp: 7.11.2012]. Dostępny w WWW: http://karnet.up.wroc.pl/~jasj/cwiczenia/kwpp4.html.
92. Jenkins Henry, *Kultura konwergencji. Zderzenie starych i nowych mediów*, Warszawa 2007.
93. Johnson Loch K., *Handbook of intelligence studies*, London, New York 2007.
94. Jordan Tim, *Hacking. Digital media and technological determinism*, Cambridge 2008.
95. Josephes Chris, *Open source entertainment* [online], [dostęp: 1.06.2012]. Dostępny w WWW: http://news.oreilly.com/2008/06/open-source-entertainment.html.

96. Kanwal Preet, Gupta Anu, Singla Ravinder K., *Open source software development. Exploring research perspectives*, [in:] *Emerging trends in computing, informatics, systems sciences and engineering*, ed. Tarek Sobh, Khaled Elleithy, New York 2013, s. 607-618.
97. Karvinen Tero, Karvinen Kimmo, *Make Arduino bots and gadgets. Learning by discovery*, Beijing, Sebastopol 2011.
98. *Katedra i bazar* [online], [dostęp: 25.01.2012]. Dostęp w WWW: http://www.linux-community.pl/node/4.
99. Kavanagh Paul, *Open source software. Implementation and management*, Amsterdam 2004.
100. Keen Andrew, *Kult amatora. Jak internet niszczy kulturę*, Warszawa 2007.
101. Kelty Christopher M., *Two bits. The cultural significance of free software*, Druham 2008.
102. Kisilowska Małgorzata, *Przestrzeń informacyjna jako termin informatologiczny*, „Zagadnienia Informacji Naukowej" 2011, nr 2, s. 35-52.
103. Klincewicz Krzysztof, *Innowacyjność projektów open source*, „Studia i Materiały" 2006, nr 2, s. 7-13.
104. Koch Stefan, *Free/open source software development*, Hershey 2005.
105. Kołcio Dominik, Sobecki Janusz, *Wykorzystanie WWW w bibliotekach i ośrodkach informacji naukowotechnicznej*, „Praktyka i Teoria Informacji Naukowej i Technicznej" 1995, nr 4, s. 8-13.
106. Kotuła Sebastian D., *Od Biblioteki Aleksandryjskiej do World Wide Web*, „Biblioteka" 2012, nr 16, s. 115-137.
107. Kotuła Sebastian D., *Prace Paula Otleta a World Wide Web*, „Biblioteka" 2013, nr 17, s. 153-167.
108. Kotuła Sebastian D., *WEB 2.0 – współczesny paradygmat Internetu*, [w:] *Oblicza Internetu. Architektura komunikacyjna Sieci*, red. Marek Sokołowski, Elbląg 2007, s. 181-188.
109. Kozbial Ardys, *Sorting through digital preservation systems*, [in:] *More technology for the rest of us. A second primer on computing for the non-IT librarian*, ed. Nancy Courthey, Santa Barbara 2010, s. 95-123.
110. Krakowiak Ludwik, *Który system bezpieczniejszy – Linux czy Windows?* [online], [dostęp: 8.08.2013]. Dostępny w WWW: http://www.idg.pl/news/360790/ktory.system.bezpieczniejszy..linux.czy.windows.html.
111. Krakowiak Ludwik, *Ubuntu zamiast Windows. Dzień 4 – nie ma Photoshopa* [online], [dostęp: 8.08.2013]. Dostępny w WWW: http://www.pcworld.pl/news/372518/Ubuntu.zamiast.Windows.Dzien.4..nie.ma.Photoshopa.html.
112. Kretchmar James M., *Open source network administration*, Upper Saddle River 2004.
113. Kuckenburg Martin, *Pierwsze słowo. Narodziny mowy i pisma*, Warszawa 2006.

114. Kumar Anup, *Internet and information technology*, New Delhi 2002.
115. Landy Gene K., Mastrobattista Amy J., *The IT/digital legal companion. A comprehensive business guide to software, Internet, and IP law*, Burlington 2008.
116. Laurent Andrew M. St., *Understanding open source & free software licensing*, Sebastopol 2004.
117. Lee Jyh-An, *Nonprofit organizations and the intellectual commons*, Cheltenham, Northampton 2012.
118. Lerner Josh, Tirole Jean, *Economic perspectives on open source*, [in:] *Perspectives on free and open source software*, ed. Joseph Feller et al., Cambridge 2005, s. 47-78.
119. Lesk Michael, *Understanding digital libraries*, Amsterdam, Boston 2005.
120. Lessig Lawrence, *Wolna kultura* [online], [dostęp: 3.02.2012]. Dostępny w WWW: http://www.futrega.org/wk/.
121. Levene Mark, *An introduction to search engines and web navigation*, ed. 2., Hoboken 2010.
122. Lievrouw Leah A., *Alternative and activist new media*, Cambridge 2011.
123. *Licencja BSD k/3 – proste wytłumaczenie, limitacje oraz przywileje* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://funkcje.net/view/5/12225/index.html.
124. *Licencja BSD & licencja FNU GPL* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://forum.webhosting.pl/licencja-BSD-and-lhcenbja-F--t2474.html&mode=threaded&pid=9586.
125. Lindberg Van, *Intellectual property and open source*, Beijing 2008.
126. Locke John, *Open source solution for small business problems*, Hingham 2004.
127. Lowe Charlie, *A brief history of open source. Working to make knowledge free* [online], [dostęp: 22.01.2012]. Dostępny w WWW: http://english.ttu.edu/kairos/6.2/binder.html?news/opensource.htm.
128. Lucas Michael W., *Absolute OpenBSD. UNIX for the practical paranoid*, San Francisco 2003.
129. Lull James, *Culture-on-demand. Communication in a crisis world*, Malden 2007.
130. *Manifest GNU* [online], [dostęp: 23.01.2012]. Dostępny w WWW: http://www.gnu.org/gnu/manifesto.pl.html.
131. Madey Greg R., *Open source software*, [in:] *Berkshire encyclopedia of human-computer interaction*, ed. William S. Bainbridge, Great Barrington 2004, s. 531-537.
132. Mapuva Jephias, *Defining the role of online education in Today's World*, [in:] *Marketing online education programs. Frameworks for promotion and communication*, ed. Egur Demiray, N. Serdar Server, Hershey 2011, s. 159-183.
133. Margolis Michael, *Arduino cookbook*, Beijing 2011.

134. Maruta Marcin, *Tekst licencji gpl v3 po polsku plus prośba o pomoc* [online], [dostęp: 24.01.2012]. Dostępny w WWW: http://itlaw.computerworld.pl/index.php/2008/03/10/tekst-licencji-gpl-v3-po-polsku-plus-prosba-o-pomoc/.
135. Masters Jon, Blum Richard, *Professional Linux programming*, Indianapolis 2007.
136. McAndrew Alasdair, *Introduction to cryptography with open-source software*, Boca Raton 2011.
137. McQuail Denis, *Mass communication theory*, ed. 6., London 2010.
138. Meeker Heather J., *The open source alternative. Understanding risks and leveraging opportunities*, Hoboken 2008.
139. Metcalfe Randy, *Top tips for selecting open source software* [online], [dostęp: 3.06.2012]. Dostępny w WWW: http://www.oss-watch.ac.uk/resources/tips.xml.
140. Michalski Ryszard, „Open Source" oczami Waldemara Pawlaka [online], [dostęp: 28.01.2012]. Dostępny w WWW: http://pppit.org.pl/?a=142.
141. Michałek-Budzicz Agata, Brzychcy Rafał, Michalski Ryszard, *Analiza rynku zamówień publicznych na narzędzia informatyczne w okresie od 1 lipca do 31 grudnia 2012 roku*, Poznań 2013 [online], [dostęp: 02.04.2014]. Dostępny w WWW: http://pppit.org.pl/publikacje/raport_2012-2013.pdf.
142. Miller Dick R., Clarke Kevin S., *Putting XML to work in the library. Tools for improving access and management*, Chicago 2003.
143. Mohr Gordon, *Archival tools to match the Web. Open, international, comprehensive*, [in:] *Asian digital libraries : looking back 10 years and forging new frontiers. 10th International Conference on Asian Digital Libraries, ICADL 2007, Hanoi, Vietnam, December 10-13, 2007. Proceedings*, ed. Dion Goh et al., Berlin, New York 2007, s. 7-8.
144. Moody Glyn, *Rebel code. The inside story of Linux and the open source revolution*, Cambridge 2001.
145. Mookhey K. K., Burghate Niesh, *Linux. Security, audit and control features*, Rolling Meadows 2005.
146. Moschovitis Christos J., Poole Hilary V. et. al., *The Internet. [Volume 3], chronology. A historical encyclopedia*, Santa Barbara 2005.
147. Muffatto Moreno, *Open source. A mutlidisciplinary approach*, London 2006.
148. Nambisan Satish, Sawhney Mohanbir, *The global brain. Your roadmap for innovating faster and smarter in a networked world*, Upper Saddle River 2008.
149. Negus Christopher, Caen Francois, *Ubuntu Linux toolbox. 1000+ commands for Ubuntu and Debian power users*, Indianapolis 2008.
150. *Netcraft* [online], [dostęp: 2.01.2013]. Dostępny w WWW: http://news.netcraft.com/.
151. Neteler Markus, Mitasova Helena, *Open source GIS. A GRASS GIS approach*, Boston 2004.

152. Newberg Pamela J., *Cataloging for non-English-speaking and preliterate children*, [in:] *Cataloging correctly for kids. An introduction to the tools*, ed. Sheila S. Intner, Joanna F. Fountain, Jean R. Weihs, Deborah A. Fritz, Julianne Beall, Chicago 2011, s. 173-178.
153. Nowak Adrian, *Wirusy dla Linuksa? Nie daj się zastraszyć* [online], [dostęp: 7.08.2013]. Dostępny w WWW: http://di.com.pl/news/47738,4.html.
154. Olber Michał, *Rząd Estonii polega głównie na wolnym oprogramowaniu* [online], [dostęp: 27.03.2014]. Dostępny w WWW: http://osworld.pl/rzad-estonii-polega-glownie-na-wolnym-oprogramowaniu/.
155. Olson Richard G., *Technology and science in ancient civilization*, Santa Barbara 2010.
156. *Open access – otwarta nauka* [online], [dostęp: 6.02.2012]. Dostępny w WWW: http://otwartanauka.cel.agh.edu.pl/course/view.php?id=2.
157. *Open Cores* [online], [dostęp: 18.11.2012]. Dostępny w WWW: http://opencores.org/.
158. *Open hand project* [online], [dostęp: 27.03.2014]. Dostępny w WWW: www.openhandproject.org.
159. *Open source ecology* [online], [dostęp: 26.03.2014]. Dostępny w WWW: opensourceecology.org.
160. *Open source software assessment methodologies* [online], [dostęp: 15.05.2012]. Dostępny w WWW: http://en.wikipedia.org/wiki/Open_source_software_assessment_methodologies.
161. *D. Szymański, Oracle porzuca swój OpenOffice.org* [online], [dostęp: 27.02.2012]. Dostępny w WWW: http://www.benchmark.pl/aktualnosci/Oracle_porzuca_rozwoj_OpenOffice.org-34468.html.
162. O'Reilly Tim, *Hardware, software, and infoware* [online], [dostęp: 2.01.2012]. Dostępny w WWW: http://oreilly.com/openbook/opensources/book/tim.html.
163. O'Reilly Tim, *Ten myths about open source* [online], [dostęp: 2.01.2013]. Dostępny w WWW: http://www.oreillynet.com/pub/a/oreilly/opensource/news/myths_1199.html.
164. Overly Michael, *The open source handbook*, Silver Spring 2003.
165. Oxer Jonathan, Blemings Hugh, *Practical Arduino. Cool projects for open source hardware*, Berkeley 2009.
166. Pace Andrew K., *The ultimate digital library. Where the new information players meet*, Chicago 2003.
167. Perens Bruce, *The open source definition* [online], [dostęp: 2.02.2012]. Dostępny w WWW: http://oreilly.com/openbook/opensources/book/perens.html.
168. Peszko Piotr, *Open-source hardware po polsku, czyli jak radzić sobie z brakiem sprzętu* [online], [dostęp: 29.05.2012]. Dostępny w WWW: http://blog.2edu.pl/2011/07/open-source-hardware-po-polsku-czyli.html.

169. Poole Hilary W., Lambert Laura, Woodford Chris, Moschovitis Christos J. P., *The Internet. A historical encyclopedia*, Santa Barbara 2005.
170. Popp Karl A., *Advances in software economics. A reader on business models and partnering*, Berlin 2011.
171. Pyati Ajit, *Open source software and libraries*, [in:] *Information technology in librarianship. New critical approaches*, ed. Gloria J. Leckie, John Buschman, Westport 2009, s. 205-220.
172. Questier Frederik, Schreus Wim, *Open courseware and open scientific publication*, [in:] *How open is the future? Economic, social & cultural scenarios inspired by free & open-source software*, ed. Marleen Wynants, Jan Cornelis, Brussels 2005, s. 111-132.
173. Quiggin Thomas, *Seeing the invisible. National security intelligence in an uncertain age*, Hackensack 2007.
174. *Raport na temat wykorzystania Linuksa w Polsce* [online], [dostęp: 27.03.2014]. Dostępny w WWW: www.lpice.eu/pl/strona-glowna/informacje-prasowe/raport-na-temat-wykorzystania-linuksa-w-polsce.html.
175. Reagle Jr. Joseph M., *Good faith collaboration. The culture of Wikipedia*, Cambridge 2010.
176. Reijswoud Victor V., Jager Arjan D., *Free and open source software for development. Exploring expectations, achievements and the future*, Monza 2008.
177. Rhyno Art, *Using open source systems for digital libraries*, Westport 2004.
178. Richter Susanne, *Critique for the open source development model*, München 2007.
179. Rusovan Srdjan, Lawford Mark, Parnas David L., *Open source software development. Future or fad?*, [in:] *Perspectives on free and open source software*, ed. Joseph Feller, Brian Fitzgerald, Scott A. Hissam, Karim R. Lakhani, Cambridge 2005, s. 107-121.
180. Sampathkumar K. S., *Understanding FOSS Version 3.0i revised*, 2009.
181. Samuels Ruth G., Griffy Henry, *Evaluation open source software for use in library initiatives. A case study involving electronic publishing*, „Libraries and the Academy" 2012, vol. 12, no 1, s. 41-62.
182. Samuelson Pamela, *A case study on computer programs*, [in:] *Global dimensions of intellectual property rights in science and technology*, ed. Michael B. Wallerstein, Mary E. Mogee, Roberta A. Schoen, Washington 1993, s. 284-318.
183. Samyal Jatinder. K., Sumi Suman, *Creation of online library using KOHA open source software. A case study of Chitkara University*, [in:] *Open-source solution in education. Theory and practice*, ed. J. Burton Browning, Santa Rosa 2010, s. 177-209.
184. Sassoon Rosemary, Gaur Albertine, *Signs, symbols and icons. Pre-history to the computer age*, Exeter 1997.

185. Shrum Wesley et.al., *Past, present, and future of research in the information society*, New York 2006.
186. Siewicz Krzysztof, *Towards an improved regulatory framework of free software. Protecting user freedoms in a world of software communities and eGovernments*, Leiden 2010.
187. Singh Ashok K., *Science and technology for civil service*, New Delhi 2008.
188. *Słownik encyklopedyczny informacji, języków i systemów informacyjno-wyszukiwawczych*, oprac. Bożenna Bojar, Warszawa 2002.
189. Sojer Manuel, *Reusing open source code. Value creation and value appropriation perspectives on knowledge reuse*, Wiesbaden 2011.
190. *Sourceforge* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://sourceforge.net/directory/?q=licenses.
191. Stallman Richard M., *Dlaczego termin „free software" jest lepszy niż „open source"* [online], [dostęp: 1.08.2013]. Dostępny w WWW: http://www.gnu.org/philosophy/free-software-for-freedom.pl.html.
192. Stallman Richard M., *Projekt GNU* [online], [dostęp: 3.03.2012]. Dostępny w WWW: http://www.gnu.org/gnu/thegnuproject.html.
193. Stefaniak Barbara, *Bibliometria w zarządzaniu informacją*, [w:] *Zarządzanie informacją w nauce*, red. Diana Pietruch-Reizes, Katowice 2008, s. 17-32.
194. Suber Peter, *Open access*, Cambridge 2012.
195. Suber Peter, *Open access overview* [online], [dostęp: 5.02.2012]. Dostępny w WWW: http://www.earlham.edu/~peters/fos/overview.htm.
196. Suber Peter, *Timeline of the open access movement* [online], [dostęp: 5.02.2012]. Dostępny w WWW: http://www.earlham.edu/~peters/fos/timeline.htm.
197. Sutton Wendy L., *Literacy replacing libraries? A chance to turn the tide*, „School Libraries in Canada" 2006, vol. 25, no 3, s. 12-15.
198. Szafranek Krzysztof, *Licencjonowanie otwartego oprogramowania* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://szafranek.net/works/articles/opensourcelicenses/.
199. Szymański Damian, Oracle porzuca swój OpenOffice.org [online] [dostęp: 27.02.2012]. Dostępny w WWW: http://www.benchmark.pl/aktualnosci/Oracle_porzuca_rozwoj_OpenOffice.org-34468.html.
200. Tanaka Yuzuru, *Meme media and meme market architectures. Knowledge media for editing, distributing, and managing intellectual resources*, Piscataway 2003.
201. Tarkowski Alek, Hofmokl Justyna, Wilkowski Marcin, *Digitalizacja oddolna. Partycypacyjny wymiar procesu digitalizacji dziedzictwa* [online], [dostęp: 11.11.2011]. Dostępny w WWW: http://www.nina.gov.pl/docs/kultura2.0/digitalizacja-oddolna.-partycypacyjny-wymiar-procesu-digitalizacji-dziedzictwa..pdf.

202. Tavani Herman T., *Ethics and technology. Controversies, questions, and strategies for ethical computing*, Hoboken 2011.
203. *The encyclopedia of new media*, ed. Steve Jones, New York 2003.
204. *The Internet. A historical encyclopedia*, ed. Hilary W. Poole, Laura Lambert, Chris Woodford, Christos J. P. Moschovitis, Santa Barbara 2005.
205. *The third culture. S. Brand, K. Kelly, G. Dyson. A Edge conversation in Munich. Introduction: Adrian Kreye; moderator John Brockman*, „Edge" 2011, no 338 [online], [dostęp: 18.01.2012]. Dostępny w WWW: http://www.edge.org/documents/archive/edge338.html.
206. *The Open Graphics Projects* [online], [dostęp: 18.11.2012]. Dostępny w WWW: http://wiki.opengraphics.org/tiki-index.php.
207. *The open source definition (annotated)* [online], [dostęp: 30.01.2012]. Dostępny w WWW: http://www.opensource.org/osd.html.
208. Tkacz Piotr, *Otwarte formaty dokumentów i ich znaczenie przy wymianie informacji*, „Zeszyty Naukowe Wyższej Szkoły Zarządzania Ochroną Pracy w Katowicach" 2006, nr 1, s. 125-135.
209. Tuomi Ilkka, *Networks of innovation. Change and meaning in the age of the Internet*, Oxford 2002.
210. van den Boomen Marianne, Schäfer Mirko T., *Will the revolution be open-sourced? How open source travels through society*, [in:] *How open is the future? Economic, social & cultural scenarios inspired by free & open-source software*, ed. Marleen Wynants, Jan Cornelis, Brussels 2005, s. 31-67.
211. von Humboldt Wilhelm, *O myśli i mowie. Wybór pism z teorii poznania, filozofii dziejów i filozofii języka*, Warszawa 2002.
212. *W świetle GNU* [online], [dostęp: 3.03.2012]. Dostępny w WWW: http://pl.wikibooks.org/wiki/W_%C5%9Bwietle_GNU.
213. Walczak Adam, *Open source projects – how are they organized and financed* [online], [dostęp: 26.03.2014]. Dostępny w WWW: www.slideshare.net/walec51/open-source-projects-how-are-they-organized-and-financed.
214. Warford J. Stanley, *Computer systems*, ed. 4., Sudbury 2010.
215. Webber Desiree, Peters Andrew, *Integrated library systems. Planning, selecting, and implementing*, Santa Barbara 2010.
216. Wasiucionek Krzysztof, *Wolne oprogramowanie w bibliotekach – problemy i perspektywy*, „Przegląd Informacyjno-Dokumentacyjny" 2009, nr 3, s. 7-21.
217. Wierzbicki Marcin, *Arduino – Twój mały interaktywny przyjaciel* [online], [dostęp: 19.11.2012]. Dostępny w WWW: http://www.medialarts.pl/download/kadra/skrypty/interakcje/arduino.pdf.
218. Więcek Łukasz, *Od czego zacząć zabawę z Arduino?* [online], [dostęp: 18.11.2012]. Dostępny w WWW: http://majsterkowo.pl/od-czego-zaczac-zabawe-z-arduino/.

219. Wheeler David A., *Generally recognized as mature (GRAM) OSS/FS programs* [online], [dostęp: 16.05.2012]. Dostępny w WWW: http://www.dwheeler.com/gram.html.
220. *Wikipedia. The free encyclopedia* [online], [dostęp: 23.01.2012]. Dostępny w WWW: http://en.wikipedia.org/.
221. *Wikipedia. Wolna encyklopedia* [online], [dostęp: 27.07.2013]. Dostępny w WWW: http://pl.wikipedia.org/.
222. Williams David M., *Czym Ubuntu różni się od Debiana?* [online], [dostęp: 17.02.2012]. Dostępny w WWW: http://jakilinux.org/linux/debian/czym-ubuntu-rozni-sie-od-debiana/.
223. Williams Rob, *Real-time systems development*, Oxford, Burlington 2006.
224. Williams Sam, *Free as in freedom. Richard Stallman's crusade for free software*, Sebastopol 2002.
225. Williams Sam, *W obronie wolności : krucjata hakera na rzecz wolnego oprogramowania* [online], [dostęp: 23.01.2012]. Dostępny w WWW: http://stallman.helion.pl/.
226. Wojciechowski Jacek, *Biblioteczna wartość naddana*, Kraków 2006.
227. Woodard C. Jason, West Joel, *Strategic responses to standardization. Embrace, extend or extinguish*, [in:] *Project-based organizing and strategic management*, ed. Gino Cattati et al., Bingley 2011, s. 263-287.
228. Woods Dan, Guliani G.autam *Open source for enterprise. Managing risks, reaping rewards*, Beijing 2005.
229. Wróż Bogna, *Digitlab* [online], [dostęp: 8.08.2013]. Dostępny w WWW: http://dl.psnc.pl/2012/09/23/digitlab/.
230. *Wykorzystanie wolnego i otwartego oprogramowania w rządowej administracji publicznej. Raport z badania ilościowego dla Fundacji Wolnego i Otwartego Oprogramowania, marzec 2010* [online], [dostęp: 28.01.2012]. Dostępny w WWW: http://pppit.org.pl/publikacje/badanie_pentor.pdf.
231. Wynants Marleen, Cornelis Jan, *How open is the future? Economic, social & cultural scenarios inspired by free & open-source software*, Bruksela 2005.
232. Zalewski Karol, *Czym jest free oraz open source software?* [online], [dostęp: 28.07.2013]. Dostępny w WWW: http://blog.4zal.net/2009/08/06/czym-jest-free-oraz-open-source-software/.
233. Zarzycki Paweł, *Udostępnianie oprogramowania w modelu open source w branży elektroniki użytkowej*, „Zarządzanie Zmianami. Zeszyty Naukowe" 2010, nr 4, s. 20-38.
234. Zymaris Con, *Free software for schools v8.12* [online], [dostęp: 25.05.2012]. Dostępny w WWW: http://cc.com.au/files/Free-Software-for-Schools.pdf.

Dr SEBASTIAN DAWID [illegible] jest adiunktem w Instytucie Informacji Naukowej i Bibliotekoznawstwa Uniwersytetu Marii Curie-Skłodowskiej w Lublinie. Absolwent Informacji Naukowej i Bibliotekoznawstwa na UMCS. Stopień doktora nauk humanistycznych w zakresie bibliologii i informatologii uzyskał w 2013 r. na Wydziale Historycznym UW. Jego zainteresowania naukowe skupiają się głównie na kulturze książki w środowisku cyfrowo-sieciowym Internetu i WWW oraz technologiach informacyjnych w działalności bibliotecznej i informacyjnej. Autor ponad 50 publikacji, m.in. *Komunikacja bibliologiczna wobec World Wide Web*, Lublin 2013.

W XXI w. jednym z prężniej rozwijanych i rozwijających się sektorów w branży IT jest open source. Choć coraz częściej obecne są w różnych sferach aktywności ludzkiej, to jednak wciąż mało poznane. Na rodzimym gruncie informatologii brak było do tej pory opracowania przybliżającego w przystępny sposób tę problematykę. Praca napisana została przez humanistę – bibliologa i informatologa – z myślą o/i z przeznaczeniem dla humanistów i bibliotekarzy, zwłaszcza tych o zainteresowaniach informatycznych i informatologicznych. Książka ma więc charakter wprowadzający w obszar open source na potrzeby szeroko pojętej humanistyki.

Seria wydawana przez Wydawnictwo
STOWARZYSZENIA BIBLIOTEKARZY POLSKICH
we współpracy
Z INSTYTUTEM INFORMACJI NAUKOWEJ I STUDIOW BIBLIOLOGICZNYCH
UNIWERSYTETU WARSZAWSKIEGO

ISBN 978-83-64203-33-6

Cena 28 zł