XII Konferencja PLOUG
Zakopane
Październik 2006

Państwo jako system informatyczny, czyli dlaczego nasze prawo musi być złe?

Tomasz Wyrozumski

BMS Creative
e-mail: tw@bms.krakow.pl

Abstrakt
Artykuł jest swoistym eksperymentem polegającym na porównaniu państwa do potężnego systemu informatycznego, w którym prawo pełni rolę oprogramowania. Staramy się przyjrzeć procesowi tworzenia prawa i skonfrontować go z najbardziej ogólnymi zasadami budowania oprogramowania komputerowego. Z takiego podejścia wyciągamy niezbyt optymistyczne wnioski, lecz spoglądając cały czas okiem informatyka próbujemy wskazać pewne sposoby poprawy sytuacji. Zwracamy przy tym uwagę na bardzo ważną kwestię kompatybilności „sprzętu" i „oprogramowania". Nie licząc specjalnie na zainteresowanie tematem polityków odwracamy na koniec tok rozumowania i zastanawiamy się, jakie konkretne korzyści może odnieść projektant systemów informatycznych z analizowania szerokiego spektrum analogii pomiędzy prawem i oprogramowaniem.

1. L'informatique

To co Anglosasi zwykli określać mianem *computer science*, we Francji nazwano – jak się wydaje, znacznie trafniej – *l'informatique*. Dobrze się stało, że ten właśnie termin przyjął się w języku polskim, gdyż informatyka winna być postrzegana właśnie jako nauka o przetwarzaniu informacji, a nie nauka o komputerach. A jeżeli już koniecznie o komputerach, to nie tylko tych, opartych na technologii krzemowej, nawet jeśli wydają się nam one (z różnych zresztą powodów) najbardziej interesujące. Czasami warto wejść na nieco wyższy poziom abstrakcji i przyjrzeć się też innym układom przetwarzającym informację – choćby tylko po to, by nabrać pewnego dystansu do własnych codziennych zajęć.

Niniejszy artykuł poświęcamy pewnemu specyficznemu systemowi informatycznemu, jakim jest państwo – dla ustalenia uwagi: współczesne, należące do tzw. cywilizacji zachodniej. Mamy nadzieję, że czytelnik wybaczy nam przy tym pewien chaos semantyczny, wynikły ze świadomego zresztą mieszania pojęć prawnych, politycznych i informatycznych. Jest to typowe dla rozważań o charakterze interdyscyplinarnym, podobnie zresztą jak brak profesjonalizmu ich autorów w przynajmniej jednej z dziedzin, których rozważania te dotyczą.

Przechodząc *ad rem* zauważmy, że wspólną cechą państw jest działanie według pewnych reguł, zwanych potocznie prawem, zaś przez informatyków określanych mianem oprogramowania. Odpowiednikiem sprzętu komputerowego jest społeczeństwo realizujące instrukcje zawarte w oprogramowaniu. Oczywiście każdy sprzęt ulega czasem awariom i robi też to, czego robić absolutnie nie powinien, bądź nie robi tego, co powinien robić. Generalnie jednak systemy działają, a do spektakularnych awarii, w rodzaju tej, jaka miała miejsce w Niemczech, w latach trzydziestych ubiegłego wieku, dochodzi na szczęście stosunkowo rzadko. Nie oznacza to też jednak bynajmniej, że wszystko jest zawsze jak należy, a użytkownicy wpadają w zachwyt. Przeciwnie, narzekają, i to pod każdą długością i szerokością geograficzną. Nie powinno nas to jednak dziwić – w końcu każdy informatyk wie, że użytkowników w ogóle trudno zadowolić, choć z drugiej strony trzeba zrozumieć, iż nie wystarcza im, by komputer nie zawieszał się co chwilę i nie kopał prądem.

Załóżmy jednak, że mamy już nasz „jakoś działający" system: maszynę i oprogramowanie, przy czym maszyna jest dana a priori i choć ulega pewnym modyfikacjom (co ciekawe, również pod wpływem oprogramowania!) to jednak modyfikacje te mają na ogół charakter długotrwały i ewolucyjny oraz pozostają raczej poza bezpośrednią kontrolą administratorów. Zupełnie inaczej jest z oprogramowaniem – jego twórcy mogą uczynić zaskakująco wiele.

Wiadomo, że aby komputer działał, potrzebny jest system operacyjny – specjalne oprogramowanie pośredniczące pomiędzy sprzętem a aplikacjami użytkownika, zarządzające zasobami maszyny z jednej, a zadaniami z drugiej strony. Analogię, choć oczywiście niepełną, stanowi tu konstytucja – ustawa zasadnicza, regulująca funkcjonowanie państwa. Systemy operacyjne współczesnych społeczeństw opierają się na idei wybitnego osiemnastowiecznego teoretyka omawianej kategorii zagadnień informatycznych, barona de Montesquieu, który wpadł na pomysł, aby odseparować od siebie trzy obszary funkcjonalne: środowisko programistyczne, zwane władzą ustawodawczą, część odpowiedzialną za wykonywanie zadań (władza wykonawcza) oraz procesy zajmujące się detekcją konfliktów, wywłaszczaniem wątków itp., określane mianem władzy sądowniczej. Sam system operacyjny może być przy tym na bieżąco modyfikowany za pomocą udostępnianych przezeń narzędzi, jednak posiada wbudowane, wewnętrzne mechanizmy obronne, które utrudniają programistom wprowadzanie nierozważnych zmian – nic dziwnego, ewentualne błędy mogą doprowadzić do poważnych problemów ze stabilnością.

W dalszej części niniejszych rozważań skoncentrujemy się na pierwszym ze wspomnianych obszarów, przyglądając się procesowi tworzenia ustaw na konkretnym przykładzie naszego kraju i analizując poszczególne stadia owego procesu.

2. Inżynieria prawa, czyli jak się tworzy oprogramowanie?

Na początku, jak wiadomo, musi pojawić się u użytkowników mniej lub bardziej uświadomiona potrzeba. Najlepiej, jeśli wynika ona z chęci poprawy organizacji pracy. Widzimy, że robimy coś mało efektywnie, bardzo się przy tym męczymy itd., a przecież z pomocą odpowiedniego oprogramowania można by lepiej. Bywa, że podglądniemy coś u innych – ot, za miedzą wdrożyli i są bardzo zadowoleni. Niekiedy wcale nie mamy ochoty robić tego samego, co kooperanci, ale oni nas po prostu do tego zmuszają. Dostosujemy się w celu zachowania kompatybilności. Zdarza się też jednak i inna sytuacja: oto ambitny dyrektor zaczyna lansować jakieś rozwiązanie z powodów, delikatnie mówiąc, pozamerytorycznych. Może chodzi mu o to, by zaistnieć w mediach, otrzymać prestiżową nagrodę, zdobyć kolejne punkty w życiorysie, a może po prostu łączy go jakiś układ z dostawcą... Pobudki mogą być różne, ale wiadomo, że istnieje pewna klasa przedsięwzięć informatycznych, które już w zamierzeniu bynajmniej nie służą dobru szerokich rzesz użytkowników. Poświęcimy jej nieco miejsca w dalszej części niniejszych rozważań.

Wracając do tematu, kolejny etap – analiza, służy między innymi szczegółowemu zdefiniowaniu wymagań użytkowników, określeniu kontekstu aplikacji (zwanej też ustawą), a także przeprowadzeniu studium wykonalności. Tu inżynierowie systemu (czyli państwa) na ogół sięgają po pomoc tzw. „niezależnych" konsultantów i wszystko byłoby dobrze, gdyby nie właśnie ów cudzysłów przy *niezależności*. Szczegółowe opracowania analityczne zamawiane są często u tych konsultantów, którzy swoją „fachowością" (znów cudzysłów) z góry zagwarantują taki, a nie inny rezultat badań. W kontekście interesujących nas zagadnień nazywa się to zwykle „realizacją zapotrzebowania politycznego". Odnotujmy na marginesie, że słowo *polityka* pochodzi od greckiego *polis* (państwo) i według Arystotelesa oznacza sztukę rządzenia państwem, nakierowaną na dobro wspólne.

Po analizie przychodzi oczywiście czas na zaprojektowanie aplikacji, przy czym tworzony przez polityków projekt na ogół na bieżąco przetwarzany jest w prototypowy kod przez programistów zwanych legislatorami. Są to zazwyczaj zawodowi prawnicy, którzy starają się pisać program poprawnie, zgodnie ze wszystkimi regułami sztuki, tak by nie „pogryzł się" z systemem operacyjnym w szczególności, ale też z innymi aplikacjami. Nie jest to oczywiście proste i na ogół wymaga dokonania modyfikacji w tych ostatnich poprzez dopisanie odpowiednich linii kodu. Prototyp podlega wstępnemu zatwierdzeniu. Jeśli tworzy go opozycja, wiąże się to na ogół z mniejszymi formalnościami, jednak w przypadku tzw. przedłożeń rządowych, przyjęta metodyka narzuca ściśle określony tryb postępowania. Prototyp trafia do tzw. uzgodnień międzyresortowych, tzn. jego kopie lądują w poszczególnych ministerstwach, które nanoszą nań odpowiednie poprawki – niestety nierzadko sprzeczne ze sobą. Ostatecznie zwyciężą te, których autorzy będą mieli większą siłę przebicia (nie mylić z siłą argumentacji). Czasami prototyp poddawany jest też tzw. konsultacjom społecznym, jednak te możemy pominąć, jako mało istotne dla całego procesu tworzenia oprogramowania.

Wersja uzgodniona przez resorty trafia wreszcie pod obrady Komitetu Rady Ministrów, który decyduje, czy poprawiać ją dalej, czy też przedstawić komisji prawnej Rządowego Centrum Legislacyjnego, czyli specjalnemu zespołowi programistów, który ostatecznie zbada formalną poprawność kodu. Oczywiście nie jest to proste, bo każda ustawa, podobnie jak i program na „zwykły" komputer, jest bardzo skomplikowaną konstrukcją logiczną, działającą w realiach narzuconych przez system operacyjny i współpracującą ściśle z innymi równie skomplikowanymi konstrukcjami logicznymi. Problem wydaje się więc znacznie bardziej skomplikowany niż dowodzenie poprawności algorytmów, które zresztą jak dobrze wiadomo, w ogólnym przypadku jest nie-

wykonalne. Efekt prac komisji, mówiący, że ustawa jest logicznie spójna, nie może więc w żadnym razie posiadać statusu twierdzenia matematycznego, ale co najwyżej, przypuszczenia popartego pewnym wysiłkiem intelektualnym.

Co dzieje się dalej ze skorygowanym i ostatecznie pobłogosławionym przez komisję prototypem? Trafia na posiedzenie Rady Ministrów, by – jeśli nie zajdą jakieś nieoczekiwane zdarzenia natury pozamerytorycznej (kłótnia w koalicji itp.) – zostać przyjętym i skierowanym pod obrady Sejmu. Owo szerokie gremium projektantów reprezentuje zwykle całe spektrum poglądów na temat nowo tworzonej aplikacji, jej przydatności użytkownikom, niezbędnych funkcjonalności, tudzież efektów wdrożenia. Ponieważ trudno jest uzgodnić coś w zespole składającym się z 460 specjalistów, ustawę po tzw. pierwszym czytaniu, czyli wstępnej analizie kodu, kieruje się do komisji, która bada go dogłębniej, czasami pyta nawet przy tym o zdanie przyszłych użytkowników, ale przede wszystkim wprowadza poprawki – i to wcale nie kosmetyczne. Niekiedy ingerują one głęboko w samą analizę wymagań i dotyczą zupełnie nowych, nieplanowanych wcześniej funkcjonalności, wywracając do góry nogami całą misterną konstrukcję logiczną oprogramowania. Bezpośrednie prace programistyczne przeprowadzają apolityczni fachowcy, tzn. sejmowi legislatorzy. Ich zadaniem jest pisać formalnie poprawny kod, realizując postulaty projektantów. Bywa jednak, że czasem i tu coś zazgrzyta. Pamiętamy doskonale dodanie do pewnej aplikacji owej nieszczęsnej linii *lub czasopisma*, która uzupełniała program o cechy pożądane przez tzw. grupę trzymającą władzę. W rzeczywistości zdarzają się jednak – i to podobno nierzadko – wpadki całkowicie niezamierzone. Legislatorzy korygują logikę prawa nie rozumiejąc czasami merytorycznego charakteru pewnych jego zawiłości i chcąc je uprościć, wypaczają w efekcie sens i wywołują zupełnie nieoczekiwane efekty.

Załóżmy jednak, że wszystko poszło dobrze, i komisja przegłosowała swoją wersję aplikacji. Teraz mamy drugie czytanie i dalszy ciąg zmian. Tym razem poprawki zgłasza się z sali (na której zasiadać może nawet 460 wspomnianych specjalistów, choć to akurat zdarza się rzadko) i przegłosowuje. Podobno przed wojną nie było to możliwe. Na tym etapie propozycje komisji Sejm mógł przyjąć w całości lub w całości odrzucić. Zabezpieczano w ten sposób ustawy przed destrukcją, jaką musi powodować pisanie programu metodą burzy mózgów, nawet najdoskonalszych.

Po opuszczeniu Sejmu aplikacja trafia do Senatu, czyli Rady Starszych. W założeniu powinien być to zespół projektantów znacznie bardziej doświadczonych, myślących racjonalniej i pewnie trochę mniej skłóconych między sobą. Mniejsza jest też jego liczebność – zaledwie (!) 100 osób. Proceduje on jednak w sposób analogiczny jak Sejm, nanosi własne poprawki i w końcu zwraca temu ostatniemu odpowiednio skorygowaną aplikację. Teraz Sejm przyjmuje lub odrzuca owe zmiany, przy czym, nie wchodząc w szczegóły proceduralne, można powiedzieć, że poprawki znacznie łatwiej jest przyjąć niż odrzucić. Może się też oczywiście zdarzyć, że Sejm przyjmie poprawki, ale nie wszystkie. Wtedy to, co wyszło z Senatu jako w miarę spójna konstrukcja logiczna, ma wszelkie szanse się „rozjechać". Np. na formatkach pozostaną jakieś przyciski, jednak nie będą już one aktywne, albo, co gorsza, będą robić zupełnie coś innego, niż użytkownik mógłby oczekiwać.

Gdy aplikacja jest już gotowa, przychodzi czas na zaprezentowanie jej użytkownikom, a ściśle mówiąc jednemu użytkownikowi, zwanemu niekiedy Pierwszym Obywatelem lub Prezydentem. Byłoby naturalną rzeczą, gdyby reprezentował on interesy szerokiego gremium odbiorców i mógł kompetentnie wypowiedzieć się, czy oprogramowanie jest dobre, czy złe, czy mu się do czegoś przyda, czy też tylko utrudni życie. Tak jednak nie jest. Aplikację zatwierdza zawsze ta sama osoba, niezależnie od tego, czy w rzeczywistości będzie jej używać, czy też może ten akurat obszar tematyczny zupełnie, ale to zupełnie jej nie interesuje. Prezydent może oczywiście (choć nie musi!) konsultować się z rzeczywistymi użytkownikami, a także polecić zbadanie kodu swoim fachowym doradcom (co akurat czyni chyba nieco częściej). Ostatecznie ma on trzy możliwości do wyboru: albo aplikację zatwierdzić, albo zawetować, albo też odesłać do zbadania pod kątem kompatybilności z systemem operacyjnym (tym zajmie się Trybunał Konstytucyjny, oczywiście).

Zatwierdzenie oznacza oddanie użytkownikom, weto – odesłanie do sejmowych projektantów, którzy albo przyznają mu rację niwecząc swój ogromny wysiłek i wyrzucą program do kosza, albo też, wykazując sporą determinację (2/3 głosów), odrzucą weto. Jeśli prezydent odeśle aplikację do Trybunału, ten znów będzie bardzo szczegółowo (i chyba znacznie kompetentniej od innych) analizował kod, aż w końcu zdecyduje: kod jest poprawny i zostanie uruchomiony, bądź też zawiera rażące błędy i musi zostać skorygowany.

Uważny czytelnik powinien postawić teraz zasadnicze pytanie: wszystko ładnie, pięknie, ale gdzie testy? No właśnie. Wygląda na to, że program trafia w ręce użytkowników całkowicie nieprzetestowany, bo trudno testami nazwać analizowanie poprawności kodu. I w tym miejscu można by udzielić ostatecznej odpowiedzi na tytułowe pytanie: *dlaczego nasze prawo musi być złe?* Po prostu dlatego, że nie jest testowane. W istocie każda ustawa stanowi całkiem skomplikowaną konstrukcję logiczną, przypominającą program: definicje typów, deklaracje zmiennych, instrukcje warunkowe, tu i ówdzie pętla, odwołania do innych programów (ustaw), wreszcie zaszyte interpretery zewnętrznych skryptów konfiguracyjnych, tworzonych przez uprzywilejowanych użytkowników, należących do kręgu władzy wykonawczej. Mamy tu na myśli oczywiście rozporządzenia i zarządzenia, których wydanie ustawy na ogół przewidują. Czy zatem coś takiego może w ogóle działać poprawnie jeśli nie przejdzie ani jednego testu? Czy może być pozbawione logicznych błędów, nie mówiąc o sprzecznościach z innymi ustawami? Nawet początkujący programista odpowie: nie. W istocie bowiem cały ambaras z oprogramowaniem bierze się stąd, że tworzą je ludzie, który są omylni, a ich sprawność w analizowaniu zdań logicznych maleje, bodaj czy nie eksponencjalnie wraz ze wzrostem złożoności tych zdań. Nawet najlepsi prawnicy należą do tego samego gatunku, co programiści (*homo sapiens*) i trudno przypuścić, aby ich mózgi nie podlegały analogicznym ograniczeniom. Każdy, kto wie, ile czasu zabiera przetestowanie przeciętnej aplikacji i ile różnych błędów wpuszcza się do niej nawet najbardziej starannie pisząc kod, złapie się za głowę uświadomiwszy sobie, że prawo powstaje z pominięciem tego ważnego etapu. Tak jest. Pytanie jednak, czy musi tak być?

3. Dwa systemy prawa

We współczesnym świecie zachodnim funkcjonują dwa systemy prawa. Pierwszy to oparty na koncepcji rzymskiej system prawa stanowionego, znany nam z autopsji. Przepisy tworzy się „za biurkiem", a następnie stosuje w praktyce. Zadaniem sądów jest „podciągać" sytuacje rzeczywiste pod odpowiednie paragrafy i postępować zgodnie z nimi, przy czym uważa się, że prawo jest tym doskonalsze im mniej pozostawia swobody interpretacyjnej. System drugi to *common law*, które wywodzi się ze średniowiecznej Anglii, a dziś funkcjonuje z powodzeniem w Wielkiej Brytanii, Stanach Zjednoczonych i szeregu krajów Brytyjskiej Wspólnoty Narodów. System ten bazuje na precedensach, co oznacza, że prawo tworzone jest głównie na salach sądowych. Każdy precedens to jakaś wzięta z życia sytuacja, przy czym ciągłość owego prawa precedensowego sięga daleko wstecz – we współczesnych sądach amerykańskich prawnicy powołują się niekiedy na sprawy, które toczyły się przed angielskimi trybunałami w siedemnastym wieku! Ustawy tworzy się po to, by porządkować owo precedensowe prawo, względnie regulować nowe, bardzo konkretne sytuacje.

Prawnicy mogliby zapewne toczyć długie, pełne wnikania w szczegóły dyskusje na temat tego, która z owych koncepcji jest lepsza (autor miał okazję przysłuchiwać się niegdyś fachowym wywodom), lecz informatykowi jedno spostrzeżenie nasuwa się nieodparcie. Otóż w modelu prawa stanowionego oddaje się użytkownikom całkiem spore aplikacje, jak już zauważyliśmy, całkowicie nieprzetestowane. Na użytkownikach sprawdza się sensowność i skuteczność przepisów, a dopiero potem ewentualnie koryguje te, które okażą się wadliwe, bądź niefunkcjonalne. Model anglosaski należałoby porównać do sytuacji, w której zamiast dużych aplikacji buduje się niewielkie programiki, mające spełnić jakieś ściśle określone zadania. Trudniej wtedy o omyłki, a i

same testy przeprowadza się na wybranych jednostkach, co w oczywisty sposób minimalizuje społeczne koszty błędów. Nie ma tu miejsca na tzw. twarde metodyki, a procedury są znacznie mniej skomplikowane od tej, którą przedstawiliśmy w poprzednim ustępie. Nasuwa się oczywiście skojarzenie z tzw. *extreme programming*, jednak wydaje nam się, że jeszcze lepszą analogią jest budowanie systemu szytego na miarę wewnętrznymi siłami użytkownika. Jak wiadomo, jakiś czas temu dość rozpowszechniony wśród naszych przedsiębiorstw był taki właśnie model. Zatrudniające sporą ilość pracowników działy informatyki zajmowały się tworzeniem i utrzymywaniem unikalnych, dedykowanych rozwiązań. Spójna koncepcja, przejrzysta architektura, czy technologiczna innowacyjność ustępowały na ogół miejsca zaspokajaniu konkretnych potrzeb użytkowników. Projekty nie miały dobrze zdefiniowanych budżetów ani harmonogramów, a poszczególne aplikacje niejednokrotnie tworzone były *ad hoc* i zszywane razem w jedną całość. Zarządy postępowały w myśl zasady: *chcesz mieć buty, zatrudnij na etacie szewca*, co z ekonomicznego punktu widzenia wydaje się oczywiście mało uzasadnione.

Koncepcja prawa stanowionego kojarzy się bardziej z wdrażaniem systemu gotowego, bądź też tworzeniem go przez wyspecjalizowaną firmę zewnętrzną. Oczywiście, na prezentacjach zawsze wszystko wygląda idealnie, natomiast w rzeczywistości, jak wiadomo, bywa różnie. Niejednokrotnie użytkownicy tęsknią do starej prowizorki, a wlokącemu się w nieskończoność projektowi towarzyszą ciche westchnienia pracowników: „było jak było, ale jakoś działało". Co jest zatem lepsze? Czy należy od fundamentów budować piękny i solidny pałac, czy też uznać, że wygodniej mieszka się w prowizorycznym baraku z niezliczoną ilością przybudówek?

Co do informatyki w węższym tego słowa znaczeniu, na pewno nie radzimy nikomu zatrudniać na etacie szewca, ale o tym więcej w ostatnim ustępie.

4. Ewolucja i rewolucja

Można śmiało powiedzieć, że jednym z największych nowoczesnych dokonań w dziedzinie prawa stanowionego był niewątpliwie napoleoński *Code Civil de Français* z 1804 roku, który zresztą stał się podstawą późniejszych regulacji w wielu krajach Europy. Nieodparcie nasuwa się tu jednak pewne skojarzenie. Otóż zarówno Francja i jak i Wielka Brytania mogą dziś poszczycić się demokratyczno-liberalnymi ustrojami (demokracja oznacza rządy większości, a liberalizm – o czym się niekiedy zapomina – poszanowanie praw mniejszości, a w szczególności jednostki). Oba kraje dochodziły jednak do tego nieco innymi drogami. Anglicy doświadczyli rewolucyjnej zawieruchy znacznie wcześniej niż Francuzi, a i jej przebieg był jakby spokojniejszy. Rewolucja Francuska postawiła sobie natomiast za zadanie stworzyć od podstaw nowy ład, negując wszystko co było przed nią. Zerwano z tradycją, wyzerowano nawet licznik lat i zmieniono nazwy miesięcy, choć na szczęście, takie barbarzyństwo nie mogło przetrwać próby czasu. Pragmatyczni i obdarzeni mniejszym temperamentem Brytyjczycy od dawna wyznają zasadę, że ewolucja, choć mniej spektakularna niż rewolucja, na dłuższą metę jest bardziej skuteczna, co w ludowej wersji streszcza się mądrym powiedzeniem: *Co nagle, to po diable*. Nawiasem mówiąc, zabawne, że ktoś bardzo mocno kojarzony ze słowem *ewolucja*, a mianowicie Charles Robert Darwin, był właśnie Brytyjczykiem!

W istocie, jeśli już musimy być skazani na używanie nieprzetestowanego prawa, to byłoby lepiej gdyby przynajmniej kolejne jego wersje służyły korygowaniu nieuchronnych błędów oraz wprowadzały niezbędne, drobne, modyfikacje. Niezwykle ważne jest zachowanie pewnej ciągłości, a nie wywracanie wszystkiego do góry nogami po każdych wyborach, jak próbuje się to czynić tu i ówdzie.

5. Sprzęt a oprogramowanie

Każdy informatyk wie doskonale, że oprogramowanie, aby działało, musi być kompatybilne ze sprzętem. Systemu Windows nie uruchomi się na procesorze SPARC, a program napisany w języku C, nawet po przekompilowaniu, nie będzie działał pod systemem UNIX, jeśli na przykład odwołuje się do bibliotek DLL. Jak wygląda ta sama kwestia w przypadku państw? Przyjrzyjmy się jednemu z najsprawniejszych systemów operacyjnych, jakim bez wątpienia okazała się Konstytucja Stanów Zjednoczonych. W ciągu ponad 200 lat wprowadzono do niej zaledwie 26 korekt, co nie tylko w informatyce stanowi ewenement. Konstytucja okazała się doskonale skalowalna i dziś sprawnie steruje funkcjonowaniem największego mocarstwa świata, nie sprawiając przy tym żadnych problemów wydajnościowych. Uważna lektura tego dokumentu pokazuje, jak bardzo jest on lakoniczny i konkretny zarazem. Za dumnymi słowami *We the People* następują artykuły niezwykle jasno precyzujące ustrojowe zasady działania państwa, zaś wolności obywatelskie, o których uczy się w szkołach, są w istocie treścią późniejszych poprawek.

Wyobraźmy sobie teraz niewielkie afrykańskie państwo targane nieustannymi wewnętrznymi konfliktami, rebeliami, przewrotami, wojnami domowymi i nie wiadomo, czym jeszcze. Dlaczegóż nie zawieźć tam Konstytucji Stanów Zjednoczonych mówiąc: „Macie, używajcie, teraz już będzie dobrze!" I cóż z tego, że ów rewelacyjny system operacyjny dostępny jest za darmo? Wiadomo, nie zadziała. I to nawet po wykonaniu niezbędnych prac dostosowawczych. Okazuje się zatem, że prawo z jednej strony reguluje działanie społeczeństwa, z drugiej jednak, by funkcjonowało, musi być jego emanacją. Zaskakujące, jak niekompatybilne są poszczególne społeczeństwa, mimo iż składają się z jednostek biologicznie bardzo podobnych. Wydaje się zresztą, że zapominają o tym niekiedy Amerykanie, pełni wiary, iż demokracja stanowi wartość samą w sobie, niezależnie od lokalnych tradycji kulturowych, czy religijnych, a jej brak w różnych miejscach naszego globu daje się zawsze wytłumaczyć przejściowymi awariami. Niestety, liczne doświadczenia i obserwacje zdają się raczej dowodzić zgoła odmiennej tezy: dla bardo wielu nacji demokracja jest czymś zupełnie obcym. Z drugiej strony, jeden z najlepiej działających komputerów (tak pod względem sprzętu jak i oprogramowania) został zbudowany z komponentów wywodzących się z całkowicie różnych kultur. Naród amerykański to, jakby nie było, z pochodzenia Irlandczycy, Włosi, Żydzi, Polacy, Chińczycy i wielu, wielu innych. Jakim cudem ta hybryda w ogóle funkcjonuje? A w dodatku tak sprawnie?

6. Wirusy

Jak powszechnie wiadomo, od lat zmorą użytkowników komputerów jest niechciane oprogramowanie, pisane przez złych ludzi w różnych niecnych celach. Czasami motywem jest kradzież danych, czasami terroryzm polityczny, czasem po prostu zwykła bezinteresowna złośliwość. Twory, o których mowa, doczekały się swoistej kategoryzacji, w której jedną z klas stanowią tzw. wirusy, czyli niewielkie fragmenty kodu wykonywalnego, doklejające się do innych programów i prowadzące destrukcyjną działalność. Wydaje się, że ową zapożyczoną z biologii nazwę można by zupełnie dobrze przenieść na grunt zagadnień dotyczących prawa. Jeśli bowiem spojrzeć na ustawę (lub akt niższego rzędu) jak na program komputerowy, to trzeba dopuścić też możliwość doklejenia doń jakichś dodatkowych instrukcji. Może się to stać zarówno na etapie samej legislacji, czyli tworzenia programu, jak i później, podczas jego wykonywania.

W pierwszym przypadku mamy oczywiście na myśli świadome działania osób oficjalnie zaangażowanych w proces konstruowania prawa, mające doprowadzić do umieszczenia w nim takich zapisów, które umkną uwadze opinii publicznej i wprowadzą do aktów prawnych zapisy całkowicie sprzeczne z oficjalnymi intencjami ich twórców. Za przykład posłużyć może owo słynne, wspomniane już wcześniej *lub czasopisma*, które miało konkretnym osobom przynieść konkretne, materialne korzyści. Niestety, wydaje się, że takich wirusów można by w naszym prawie znaleźć

znacznie więcej, a ich detekcja jest o tyle trudna, że korzyści częściej dotyczą całych grup społecznych niż ludzi nazwanych z imienia i nazwiska. Niekiedy zresztą psucie prawa odbywa się w sposób tak spektakularny i bezpardonowy (by nie powiedzieć bezczelny), że porównanie z komputerowymi wirusami wydaje się mało adekwatne. W końcu te ostatnie tworzy się potajemnie, w zaciszu informatycznego warsztatu, nie zaś przed kamerami, wyjaśniając tłumnie zebranym dziennikarzom, jakie to wspaniałe zmiany w oprogramowaniu wprowadza się głosując nad nimi aż do skutku.

Znacznie mniej spektakularna jest druga kategoria wirusów, tych bardziej podobnych w zachowaniu do znanych nam doskonale komputerowych odpowiedników. Chodzi tu o różnego rodzaju instrukcje, czy wytyczne wydawane przez organa administracji państwowej podległym im urzędom, a więc bardzo ważnym elementom społecznego procesora. Dotyczą one zazwyczaj interpretacji niejasnych przepisów i choć próżno dla nich szukać delegacji ustawowych, doklejają się de facto do aktów prawnych, wyrządzając nierzadko wymierne szkody licznym użytkownikom oprogramowania. Autor miał sam okazję paść ofiarą tego typu wirusa, gdy jakiś czas temu zmuszony został odpowiednimi przepisami do aktualizacji danych spółki. I choć z przepisów tych jasno wynikało, że cała operacja powinna być bezpłatna, a stosowne ministerstwo potwierdziło to na swoich stronach internetowych, to jednak lokalne władze uparły się, że jest inaczej. Podobno miasto Kraków zarobiło na owym procederze kilkaset tysięcy złotych.

Oczywiście, tak jak przed wirusami komputerowymi można się zabezpieczać stosując odpowiednie oprogramowanie, tak i na wirusy administracyjne są sposoby. System operacyjny oferuje zabezpieczenie w postaci dostępu do sądów, przed którymi można dochodzić swych racji. Niestety, ten typ zadań procesor realizuje niesamowicie wolno i opornie – z powodów, które mogłyby być tematem oddzielnego referatu z dziedziny szeroko rozumianej informatyki. Rzeczywisty koszt eliminacji wirusów bywa zwykle wielokrotnie wyższy niż wyrządzone przezeń (jednostkowe) straty, a to sprawia, że z wirusami często łatwiej jest żyć niż je tępić. Niekiedy oczywiście straty bywają bardzo znaczne – upadłość przedsiębiorstw, likwidacja setek miejsc pracy, osobiste tragedie niewinnych ludzi. Wtedy słabość zabezpieczeń manifestuje się w całej pełni.

7. Demokracja bezpośrednia

Ci, którzy uznają demokrację za najlepszą formę sprawowania władzy (autor skłania się raczej ku monarchii oświeconej – jedyny problem, to skąd wziąć oświeconego monarchę?) nierzadko nie mogą się też oprzeć niewątpliwemu urokowi tzw. *demokracji bezpośredniej*, a więc ustroju, w którym o ważnych sprawach decydują wszyscy obywatele, nie zaś wybrani przez nich reprezentanci. Jedną z manifestacji takiej właśnie metody rządzenia są spotykane w wielu krajach (a w Szwajcarii najczęściej) referenda. Odwoływanie się do opinii publicznej wydaje się czymś bardzo szlachetnym, jednak trudno nie odnieść wrażenia, iż sprowadza się ono do podejmowania decyzji przez osoby niekompetentne. Jeśli np. gospodyni domowa (z całym szacunkiem dla jej ciężkiej pracy) ma wypowiedzieć się na temat przyszłości elektrowni atomowych, to raczej małe szanse, że wcześniej poświęci czas na zapoznanie się choćby z podstawami fizyki jądrowej, problemami współczesnej energetyki, zasadami ekonomii itd. Raczej ulegnie prostym i zrozumiałym argumentom, które ją przekonają, a takich najprawdopodobniej dostarczą organizacje o różnych odcieniach zieloności, szermujące na prawo i lewo hasłem *Czernobyl*. Jest to trochę tak, jakby aplikację projektowali bezpośrednio użytkownicy, którzy niby najlepiej przecież wiedzą, czego oczekują. Prawda, tylko czy znają się na projektowaniu?

Autor był kiedyś świadkiem następującego wydarzenia. W pewnej, nie takiej znów małej firmie zatrudniono studenta, aby napisał program *Dziennik Korespondencji*. Pomysł taki sobie, ale niestety często spotykany. Miało być tanio, szybko i dobrze. Gdy powstał prototyp, sekretarka ze łzami w oczach pobiegła do szefa, tłumacząc, że gdyby miała tego używać, to już na nic innego nie starczyłoby jej czasu. Ten zaczął więc naprędce wymyślać, co by tu zmienić na formatkach,

żeby było lepiej, ale w rzeczywistości w ogóle nie o to chodziło – firma potrzebowała systemu zarządzania obiegiem dokumentów, a nie dziennika korespondencji. Nieszczęsny autor tłumaczył później znajomym, że on chciał jak najlepiej, ale nie zna się na funkcjonowaniu przedsiębiorstw. Wolałby pisać coś, co ktoś bardziej doświadczony porządnie by zaprojektował. Analogicznie jest z ustawami. Po to ludzie głosują na polityków, żeby ci profesjonalnie zajmowali się tworzeniem prawa w ich imieniu. A że nie zawsze potrafią, to już całkiem inna kwestia...

8. Litera i duch

Jest też coś – i dla porządku trzeba o tym też wspomnieć – co odróżnia prawo od oprogramowania. Tym czymś jest tzw. *duch* – mówi się o literze i duchu prawa. W istocie chodzi o to, że społeczny komputer nie działa w sposób tak idealnie jednoznaczny jak maszyna zbudowana w oparciu o technologię krzemową. Dzięki temu zresztą nieprecyzyjne i nieprzetestowane prawo może w ogóle funkcjonować, nie wywołując natychmiast komunikatów w rodzaju *general protection fault* czy *access violation*. Specyfika ludzkiego mózgu, pozwalająca rozumieć nie tylko języki formalne, ale również (i to chyba nawet lepiej!) – naturalne, sprawia, że czytając dowolny tekst, nie tyle analizujemy składnię i semantykę poszczególnych zdań, co raczej staramy się zrozumieć intencje autora. Pytanie, czy trafnie. Na słynne *Co poeta miał na myśli?* chyba tylko w szkole może być jednoznaczna odpowiedź. Jeśli pytanie brzmi *Co miał na myśli ustawodawca?* przechodzimy właśnie do kwestii związanej z duchem prawa, a mówiąc bardziej precyzyjnie – do jego wykładni innej, niż czysto językowa. Jest to praktyka normalna i powszechnie przyjęta, co więcej, konieczna, choć de facto nawiązująca jakby trochę do koncepcji *common law*. Nie byłoby w niej zresztą nic złego, gdyby nie niebezpieczeństwo wynikające z tego, że interpretując czyjś zamiar zawsze można to zrobić opacznie. Okazuje się więc, że na jakość ustanowionego już prawa mają wpływ kompetencje stosujących go prawników. Może to oczywiście przypominać nieco sytuację, gdy ten sam program różnie zachowuje się uruchamiany w różnych środowiskach, choć tu akurat naszym zdaniem analogia nie jest najlepsza. Konkretne decyzje zależą bowiem nie tylko od rodzaju organizacji społecznej, czyli komputera jako całości, lecz wręcz od poszczególnych osób, ich wiedzy lub osobistych interesów. Doskonałym przykładem może być tu tzw. klauzula obejścia prawa, wprowadzona w ordynacji podatkowej, a następnie częściowo zakwestionowana przez Trybunał Konstytucyjny. Pozwalała ona urzędnikom fiskusa kwestionować konkretne zapisy umów między przedsiębiorstwami, jeśli ich zdaniem służyć miały one jedynie obniżeniu podatku. Chodziło więc o odwołanie się nie tyle do litery, co do ducha wspomnianych umów, czyli zgodnych intencji zawierających je stron. Problem w tym, że urzędnicy fiskusa, badający owe intencje, otrzymują premie za wykrycie przypadków zaniżenia zobowiązań podatkowych i w dodatku wcale nie muszą tych premii zwracać, jeśli właściwy sąd orzeknie, iż nie mieli racji.

9. Okno debuggera

Monteskiusz byłby zapewne nieco zaskoczony, gdyby dowiedział się, że do jego genialnej koncepcji trójpodziału władzy dodano coś, co nazwano władzą czwartą, choć przecież formalnie na nazwę taką nie zasługuje. Media, bo o nich oczywiście mowa, spełniają niezwykle ważną rolę w systemie, zabezpieczając go, choć nie bezpośrednio, przed różnego rodzaju degeneracjami. Stanowią one swoisty debugger, czyli program który śledzi wykonanie innych programów i pozwala wykrywać w nich usterki. Choć sam nie usuwa błędów i nie naprawia automatycznie sprzętu ani oprogramowania, to jednak bardzo pomaga w diagnozowaniu problemów, a czasem nawet umożliwia ich uniknięcie, np. pokazując zawczasu, jak podczas funkcjonowania jakiejś aplikacji powoli, ale systematycznie rośnie stos. I nawet jeśli sam napisany jest trochę topornie, to jednak trudno by znaleźć programistę, który nie korzystał z jego pomocy, nie śledził krok po kroku wykonania kodu, względnie nie zastawiał w nim – nomen omen – pułapek. Mimo wszystkich zarzutów, które stawia się dziennikarzom, od niechlujstwa poprzez niekompetencję, aż po stronni-

czość, przerażenie ogarnia na samą myśl, co stałoby się z niejednym państwem, gdyby nie oni! Pomyślmy tylko, ile demokracji niepostrzeżenie przekształciłoby się w despotyczne dyktatury? Przecież wiadomo, że nawet najlepszy system operacyjny nie pomoże, jeśli np. zacznie szaleć jeden lub dwa tranzystory w procesorze, a przy tym w dodatku zawiedzie pamięć...

10. Informatykom ku przestrodze

Czas wreszcie na pewne podsumowanie naszych rozważań, które w zamierzeniu miały być bardziej formą rozrywki intelektualnej, niż spójnym naukowym wywodem. Należałoby zapewne dedykować je politykom, ale wątpimy, aby któryś z nich przypadkiem przeczytał ten tekst, a gdyby nawet już tak się stało, to zapewne i tak nic by z tego nie wynikło. Pozostają więc informatycy, którzy, mimo wszystkich narzekań użytkowników komputerów, mają chyba jednak większe od polityków zasługi dla dobra ogólnego. Nie zamierzamy przekonywać ich do testowania aplikacji, bo to przecież oczywistość. Nie chcemy również lansować jakiejś nowej metodyki wzorowanej na procesie legislacji, bo naszym zdaniem standardy tworzenia oprogramowania znacznie przewyższają standardy tworzenia prawa. Jeśli więc na coś chcielibyśmy zwrócić ich uwagę, to na koncepcję anglosaskiego *common law* i owo ewolucyjne podejście do prawa. Tak naprawdę bowiem nie ma na tym łez padole nic trwałego. Prowizorką jest ewoluujący Wszechświat, przechodząca kolejne epoki geologiczne Ziemia, życie ludzkie, dzieła człowieka, itd. Różna jest tylko skala prowizorek. Prowizoryczny (i to bardzo!) charakter mają też systemy informatyczne. Nie namawiamy tu jednak nikogo, by pisać je w związku z tym byle jak. Chodzi nam raczej o pewne podejście. Informatyka jest sztuką tworzenia prowizorek i nie ma w tym nic złego, tak samo jak nie ma nic złego w anglosaskim prawie, które dzień w dzień podlega ciągłej ewolucji w wyniku mozolnej, acz twórczej pracy całej rzeszy sędziów, adwokatów, prokuratorów i przysięgłych. Prowizorki należy poprawiać, by były one lepszymi prowizorkami, czasem wymieniać na inne prowizorki, ale zawsze ostrożnie. Gdy chce się zrobić coś nowego i ma to być dobre, raczej nie należy z dnia na dzień wywracać wszystkiego, co było do góry nogami i kwestionować dorobku poprzedników. Rewolucje są z reguły krwawe i w dodatku zwykle nic sensownego z nich nie wynika. Budując bądź wdrażając nowy system, tworząc lub lansując nową technologię, należy iść do przodu małymi krokami i na ile się da, unikać rewolucyjnych zmian. Dopiero jeśli się nie da, znaczy, że są one naprawdę potrzebne. Ktoś mógłby powiedzieć, że nawołujemy do powstrzymania postępu technicznego. Nic podobnego. Chodzi nam tylko o to, że nie każdy wizjonerski pomysł i nie każda nowa moda generują rzeczywisty postęp i rzeczywisty pożytek dla ogółu, choć zwykle przysparzają jakichś korzyści ich autorom. By dać konkretny przykład, odnotujmy, że ze sporym zdumieniem obserwujemy od paru lat rozwój technologii trójwarstwowych opartych o przeglądarki. O ile są one niemal jedynym sensownym pomysłem na tworzenie aplikacji serwerowych, z których korzystać ma bliżej nieokreślona liczba użytkowników rozsianych po całym świecie, o tyle wykorzystywanie ich w wewnętrznych systemach wspomagających pracę przedsiębiorstw można wytłumaczyć wygodą administratorów, ale przecież nie użytkowników, bo konia z rzędem temu, kto udowodni ergonomiczną wyższość przeglądarki nad tradycyjnymi okienkami! A wszystkie argumenty o przystosowaniu do pracy w sieciach rozległych świadczą tylko o niezrozumieniu elementarnych pojęć. Ale cóż, rewolucja ma swoje prawa i zawsze musi *ruszyć z posad bryłę świata*, jak głosi zapomniana już nieco pieśń. Świadomi historycznych doświadczeń, zdecydowanie zalecamy ostrożność. Również informatykom.