Издательский дом “Вильямс”, 2001. – 290с.
3. *Е.А. Высоцкая*. Задача распознавания написанного ключевого слова, как одна из задач, решаемых при выполнении аутентификации пользователей компьютерных систем по рукописному почерку. – Київ: НАН України. Збірник наукових праць. Моделювання та інформаційні технології, 2006р., випуск 36, стор. 67-76.

Поступила 10.02.2010р.

УДК 681

С.М. Головань, В.В. Нечипорук, Л.М. Щербак

ПРОБЛЕМИ ВИКОРИСТАННЯ І НАДІЙНІСТІ ВІЛЬНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

In work basic attention is spared free software and basic criteria are indicated on the basis of which the author of free software passes to the programs the users. Basic among these criteria is free distribution of source code, derivative product, inviolability of author source code, absence of discrimination and purview license. The questions of reliability of software are also considered at direction of exposure and correction of errors of software.

Вступ

Програмне забезпечення є функціональною складовою сучасних інформаційних технологій [1]. Тому роль програмного забезпечення постійно зростає при вирішені науково-технічних проблем у різних галузях народного господарства.

Особливості програмного забезпечення полягає в тому, що програмування створюється в одній формі – у вигляді вихідного тексту, а поширюється і використовується в іншій – у вигляді двійкової програми, машинних кодів, з яких неможливо однозначно відновити вихідний текст. Для того, щоб ефективно змінювати програму, виправляти помилки, чи навіть просто точно встановлювати, що і як робить програма, необхідно мати її вихідний текст, оскільки при компіляції в машинний код програма втрачає зручність до читання.

У ситуації коли програмне забезпечення є об’єктом продажу нарівні з предметами побуту, на нього автоматично поширюються вже не тільки закони наукової розробки, але й властивості матеріальних предметів, якими можливо торгувати, обмінюватися, право володіння і користування якими варто охороняти законодавчо. Програмне забезпечення потрапило в розряд інтелектуальної власності: тобто вихідний тест програми став об’єктом авторського права і охороняється як літературні твори у відповідності до Бернської конвенції про охорону літературних і художніх творів, Цивільного

кодексу України [2], Закону України “Про авторське право і суміжні права” та іншими нормативно-правовими актами України. Щоб захистити свої інтереси, виробники програмного забезпечення використовують ліцензії – вид договору між власником авторських прав і користувачем (покупцем) програмного забезпечення.

Операційна система інформаційної системи являє собою програмне забезпечення, що управляє й організовує роботу інформаційної системи. У якості прикладів операційної системи можна назвати DOS, Linux, Mac OS, Windows, UNIX наведений список далеко не повний, але ми свідомо обмежили його операційними системами, які використовуються найбільш часто. Операційні системи діють протягом усього часу, доки інформаційна система увімкнена.

Операційні системи можливо класифікувати по багатьом параметрам. Всі вони діляться на два види – платні і безкоштовні (умовно-безкоштовні), правомірно розділити їх на операційні системи з відкритим вихідним кодом (з правом вносити зміни або без права вносити зміни) та із закритим кодом тощо.

Використання вільного програмного забезпечення

Згідно з законодавством більшості країн світу, програмний продукт і його вихідний код за замовчуванням охороняється авторськими правами, яке дає право власнику (найчастіше право власником є організація, що наймає автора), повну владу над зміною, розповсюдженням, способом використання і поведінкою програми, включаючи випадки, коли вихідний код опубліковано. Сила власника настільки велика, що навіть вивчення або спроба виправлення помилок програми можуть переслідуватися кримінальним правом.

Щоб позбавити користувача від проблем, викликаних перекосом законодавства про охорону результатів інтелектуальної діяльності в бік право власника, самі право власники можуть передавати користувачеві права на вищевказані свободи дії. Це досягається шляхом випуску вихідного коду програмного забезпечення на умовах однієї з особливого виду ліцензій, які називаються, вільними ліцензіями.

Вільне програмне забезпечення – широкий спектр програмних рішень, в яких права користувача (“свобода”) не обмежені вивченням, розповсюдженням, запуском і зміною (удосконаленням) програм захищені юридично за допомогою вільних ліцензій.

Критерії визначають ті права, які автор вільної програми передає будь-якому користувачеві:

– програму можливо використовувати з будь-якою метою (“нульова свобода”);

– можливо вивчити, як програма працює і адаптувати її для своїх цілей (“перша свобода”);

– можливо розповсюджувати копії програми – на допомогу товаришу (“друга свобода”);

– програму можливо поліпшити і публікувати свою поліпшену версію – з тим, щоб принести користь всій спільноті (“третя свобода”). Умовою цього є доступність вихідного тесту програми.

Тільки та програма, що задовольняє всім принципам може вважатися вільною, тобто гарантовано відкритою та доступною. Треба підкреслити, що ці принципи обумовлюють тільки доступність програми для загального використання, критики та поліпшення, але ніяк не оговорюють пов’язані з поширенням програм грошові відносини, в тому числі не передбачають і безкоштовність. В англомовних текстах тут часто виникає плутанина, оскільки слово “free” по-англійському означає не тільки “вільне”, але й “безкоштовне” і нерідко вживається по відношенню до безкоштовного програмного забезпечення, яке розповсюджується без стягнення плати за використання, але яке недоступне для зміни , тому що його вихідні тексти не опубліковані. Таке безкоштовне програмне забезпечення зовсім не є вільним. Навпаки, вільне програмне забезпечення цілком можна поширювати, стягуючи при цьому плату, однак, дотримуючись при цьому критерію свободи, кожному користувачу надається право отримати вихідні тексти програм, змінювати їх і поширювати далі. Всяке програмне забезпечення, користувачам якого не надається такого права, є невільним – незважаючи від будь-яких інших умов.

Відкритий вихідний код означає більше ніж просто доступ до вихідного коду. Умови розповсюдження програмного забезпечення з відкритим вихідним кодом повинні відповідати наступним критеріям [3]:

– вільне розповсюдження. Ліцензія не повинна обмежувати право будь-яких фізичних або юридичних осіб продавати чи безкоштовно розповсюджувати програмний продукт як частину колекції програмних продуктів отриманих з різних джерел. Ліцензія не повинна вимагати відрахувань або інших виплат за подібне поширення;

– вихідний код. Програма повинна містити вихідний код і допускати розповсюдження у вигляді вихідного коду, так само як і скомпільованій формі. У тих випадках, в яких будь-яка форма продукту поширюється без вихідного коду, повинен існувати добре відомий спосіб отримання вихідного коду за ціну, що не перевищує обґрунтовану вартість відтворення – бажано безкоштовне скачування з Інтернету. Вихідний код повинен бути кращою формою в якій будь-який програміст зміг би модифікувати програму. Умисна зміна вихідного коду таким чином щоб вводив в оману, не допускається. Проміжні форми вихідного коду, такі як висновок транслятора не допускається.

– похідні продукти. Ліцензія повинна допускати модифікацію та створення похідних продуктів і повинна допускати їх поширення на таких же умовах як і оригінальний програмний продукт;

– недоторканість авторського вихідного коду. Ліцензія може обмежувати розповсюдження вихідного коду у модифікованій формі тільки якщо ліцензія допускає поширення “файлів-заплаток” з вихідним з метою

модифікації вихідного коду програми під час компіляції та створення програмного продукту. Ліцензія повинна явним чином дозволяти розповсюдження програмного продукту, створеного з модифікованого вихідного коду. Ліцензія може вимагати, щоб продукти мали найменування або номер версії відмінні від оригінального програмного продукту;

– відсутність дискримінації окремих осіб або груп осіб. Ліцензія не повинна дискримінувати будь-яких осіб або груп осіб;

– відсутність дискримінації областей застосування. Ліцензія не повинна накладати обмеження на застосування програмного коду в певних галузях діяльності – наприклад, вона не може обмежувати або забороняти використання програми в діловій сфері або в сфері генетичних досліджень;

– область дії ліцензії. Права, пов'язані з програмним продуктом повинні бути застосовані до всіх хто є одержувачем програмного продукту без необхідності укладання додаткової угоди або згоди на додаткову ліцензію з їхнього боку;

– ліцензія не повинна бути прив'язана до певного продукту. Права, пов'язані з програмним продуктом не повинні залежати від того, чи є даний продукт частиною певної колекції програмних продуктів. У випадку, якщо програмний продукт витягується з даної колекції та використовується чи розповсюджується у відповідності з умовами ліцензії, всі сторони на яких поширюється даний продукт повинні мати такі ж права як і ті, що дає оригінальна колекція;

– ліцензія не повинна обмежувати інші програмні продукти. Ліцензія не повинна обмежувати інші програмні продукти, що розповсюджуються разом з продуктом, який ліцензується. Наприклад, ліцензія не може вимагати всі інші програмні продукти, що розповсюджується на одному носії з даними, були програмним продуктом з відкритим вихідним кодом;

– ліцензія не повинна бути технологічно нейтральною. Тобто, ліцензія не повинна вимагати що-небудь від інтерфейсу або технологій, які застосовуються в похідній програмі.

Коротко зупинимось на надійності програмного забезпечення [4].

Програми для сучасних інформаційних систем можуть нараховувати чимало команд. Під час написання програм можуть по різним причинам появитися помилки. По цьому поводу шуткують, що нема програм без помилок, а є програми із не найденими помилками. Грубі помилки виявляються на стадії відпрацювання програми, але так як перевірити програму на всіх можливих режимах, як правило, не удається, то нема впевненості, що всі помилки в ній знайдені. При цьому природно є статистичний підхід до аналізу процесу виявлення помилок в програмі. Цей процес може бути охарактеризований функцією

$$\frac{f(t)}{R} \tag{1}$$

де $f(t)$ – кількість виявлених і виправлених помилок в одиницю часу в

програмі маючої R – команд

$$\frac{f(t)}{R} = \frac{d\varepsilon_n}{dt} = \frac{\varepsilon_n(t+\Delta t) - \varepsilon_n(t)}{\Delta t}, \tag{2}$$

де $\varepsilon_n(t)$ – кількість виявлених і виправлених помилок за час t в розрахунку на одну команду.

Відповідно,

$$\varepsilon_n(t) = \frac{1}{R}\int_0^t f(t)dt\,. \tag{3}$$

Функція $f(t)$ може бути дослідно визначена під час наладки програми шляхом фіксації кількості виявлених помилок. Задача визначення $f(t)$ спрощується, якщо припустити, що

$$f(t) = \frac{\varepsilon_0}{\tau_0} e^{-\frac{t}{\tau_0}}, \tag{4}$$

де ε_0 і τ_0 – параметри $f(t)$, які визначаються під час відпрацювання.

Тоді

$$\varepsilon_n(\tau) = \frac{1}{R}\int_0^\tau f(t)dt = \frac{\varepsilon_0}{R}(1 - e^{-\frac{\tau}{\tau_0}})\,. \tag{5}$$

При $\tau \to \infty$ $\varepsilon_n(\infty) = \frac{\varepsilon_0}{R}$ або $\varepsilon_0 = R\varepsilon_n(\infty)$. Звідки слідує, що ε_0 – це загальне число помилок в програмі перед початком наладки. Так як процес наладки не може бути тривалим, та в програмі завжди буде залишатися деяка кількість помилок

$$\varepsilon(\tau) = \frac{\varepsilon_0}{K} - \varepsilon_n = \frac{\varepsilon_0}{R} e^{-\frac{\tau}{\tau_0}}, \tag{6}$$

де $\varepsilon(\tau)$ – кількість знайдених помилок в розрахунку на одну команду. Якщо припустити, що помилки рівномірно розподілені по всій програмі, то імовірність Q появи помилки за час Δt буде пропорційна швидкодії δ інформаційної системи (середньому числу змін в одиницю часу команд) і кількості залишених в програмі помилок, тобто

$$Q(\tau) = \varepsilon(\tau)\delta\Delta t\,. \tag{7}$$

Проводячи аналогію між процесом появлення помилок і відмов об'єктів $(Q = \lambda\Delta t)$, можливо зробити висновок, що інтенсивність помилок $\varepsilon(\tau)$ не залежить від часу t і визначається тільки інтервалом Δt, на якому оцінюється імовірність появи помилки. Звідки, напрацювання на "відмову", котре обумовлене появою помилки в програмі буде складати

$$T(\tau) = \frac{1}{\varepsilon(\tau)\delta} = \frac{R}{\varepsilon_0 \delta} e^{\frac{\tau}{\tau_0}}. \quad (8)$$

Аналіз зміни $T(\tau)$ може служити основою для вибору часу τ відпрацювання програми, а наладка закінчується якщо величина $T(\tau)$ становиться достатньо великою.

У випадку, коли вдається оцінити матеріальні втрати C_n від появи помилки в розрахунках, то час τ відпрацювання можливо оцінити кількісно наступним способом. За час T_p роботи програми вона відмовить $\frac{T_p}{T(\tau)}$ разів, що викличе сумарну втрату $C_n \frac{T_p}{T(\tau)}$. Процес відпрацювання програми вимагає затрат машинного часу та інших затрат пов'язаних з ним. Якщо вартість одного часу відпрацювання позначити C_0, то за час τ таких затрат буде $C_0\tau$. Відповідно, загальна втрата C від похибки і затрат на відпрацювання програми будуть

$$C = \frac{C_n T_p}{T(\tau)} + C_0\tau = \frac{C_n T_p \varepsilon_0 \delta}{R} e^{-\frac{\tau}{\tau_0}} + C_0\tau. \quad (9)$$

звідки $\frac{dC}{d\tau} = \frac{-C_n N_p \varepsilon_0 \delta}{R\tau_0} e^{-\frac{\tau_M}{\tau_0}} + C_0 = 0$ або $\tau_M = -\tau_0 \ln \frac{C_0 R \tau_0}{C_n T_p \varepsilon_0 \delta}$,

де τ_M – час відпрацювання, котра забезпечить мінімум C.

У цьому випадку, коли необхідно виключити помилку в програмі бажано використовувати "резервування". Одна і та ж задача вирішується декількома програмами, кожна з який розробляється незалежними групами програмістів та в основу котрих покладені різні алгоритми, а результати розрахунків програм порівнюються та рахуються вірними коли вони співпадають. Так як поява помилки в програмах є подією маловірогідною, то співпадіння двох або більше таких подій (так то, що дві або більше груп програмістів зроблять помилку в програмах, які дадуть однакові помилки результатів) є подією практично неможливою.

Висновки

В роботі основну увагу приділено вільному програмному забезпеченню і вказані основні критерії на основі яких автор вільного програмного забезпечення передає програми користувачам. Основним серед цих критеріїв є вільне розповсюдження вихідного коду, похідний продукт, недоторканість авторського вихідного коду, відсутність дискримінації і області дії ліцензії. Також розглянуті питання надійності програмного забезпечення в напряму виявлення та виправлення похибок програмного забезпечення.

1. Процеси життєвого циклу програмного забезпечення (ISO/IEC 12207: 1995) : ДСТУ 3918-1999 – [Чинний від 2000-07-01]. – К. : Держстандарт України, 2000. VI. 49 с. (Державний стандарт України).
2. Цивільний процесуальний кодекс України [Електронний ресурс] Закон України від 18.03.2004 р. № 1618-IV. – Режим доступу: http.rada.gov.ua. – Заголовок з екрану.
3. Определение концепции открытого исходного кода – "http://www.free-soft/org/mirrors/www.opensource.org/docs/osd-russian.php".
4. *Воробйова Н. І.* Надійність комп'ютерних систем / *Воробйова Н. І., Корнійчук В. І., Савчук О. В.* – К.: Корнійчук, 2000. – 144 с.

Поступила 3.02.2010р.

УДК 683.06

Я.Равецки

ФОРМИРОВАНИЕ ВСПОМОГАТЕЛЬНЫХ ФУНКЦИЙ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ

Принципы построения структуры SAR, которая обеспечивает прогрессивное развитие объектов управления (OU), состоят не только в последовательном использовании таких базовых компонент как эволюционные модели процессов функционирования OU, процессов функционирования окружения OU, которые будем называть QOU и моделей определения риска MR, но и в использовании обратных связей между отдельными функциональными компонентами, а также, в использовании рекуррентных схем инициации текущих функциональных блоков.

В связи сформирование структуры SAR, необходимо рассмотреть все конструктивные элементы, которые обеспечивают определенную последовательность функционирования компонент SAR и общую схему реализации процесса анализа и преобразования PR в UR.

Сформированное PR инициирует активизацию SAR и соответственно MGR. Система SAR, реализующая анализ PR, осуществляет моделирование изменений параметров, которые характеризуют отдельные OU, а результаты такого моделирования оцениваются моделью оценки риска, который возникает в результате использования модифицированного PR, являющегося управляющим решением.

В первую очередь, необходимо отметить, что отличие модели SAR от модели MGR состоит в следующем. Модель $MGR = F(MER, MEF, MR)$, а модель системы SAR описывается соотношением: