Krassimir Markov, Vitalii Velychko, Oleksy Voloshin
(editors)

Natural and Artificial Intelligence

ITHEA
SOFIA
2010

Krassimir Markov, Vitalii Velychko, Oleksy Voloshin (ed.)

Natural and Artificial Intelligence

ITHEA®

Sofia, Bulgaria, 2010

ISBN 978-954-16-0043-9

First edition

Recommended for publication by The Scientific Concil of the Institute of Information Theories and Applications FOI ITHEA

This book is engraved in prof. Zinoviy Lvovich Rabinovich memory. He was a great Ukrainian scientist, co-founder of ITHEA International Scientific Society (ITHEA ISS). To do homage to the remarkable world-known scientific leader and teacher this book is published in Russian language and is concerned to some of the main areas of interest of Prof. Rabinovich.

The book is opened by the last paper of Prof. Rabinovich specially written for ITHEA ISS. Further the book maintains articles on actual problems of natural and artificial intelligence, information interaction and corresponded intelligent technologies, expert systems, robotics, classification, business intelligence; etc. In more details, the papers are concerned in:conceptual problems of the natural and artificial intelligent systems: structures and functions of the human memory, ontological models of knowledge representation, knowledge extraction from the natural language texts;network technologies;evolution and perspectives of development of the mechatronics and robotics;visual communication by gestures and movements, psihology of vision and information technologies of computer vision, image processing;object classification using qualitative characteristics;methods for comparing of alternatives and their ranging in the procedures of expert knowledge processing;ecology of programming – a new trend in the software engineering;decision support systems for economics and banking;systems for automated support of disaster risk management;and etc.

It is represented that book articles will be interesting for experts in the field of information technologies as well as for practical users.

General Sponsor: Consortium FOI Bulgaria (www.foibg.com).

Printed in Bulgaria

ISBN 978-954-16-0043-9

C\o Jusautor, Sofia, 2010

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ – ЭКОЛОГИЧЕСКИЙ ПОДХОД К ИССЛЕДОВАНИЯМ

Максим Луцкий, Николай Сидоров

Abstract: *The software ecology as a new branch of the software engineering is presented. Three directions of software ecology dessicions (green software, sustainable development, ecosystems) are looked.*

Keywords: *software, software engineering, software ecosystem, reverse engineering.*

ACM Classification Keywords: *software, software engineering, distribution, maintence and enchaincment .*

Введение

В статье рассматривается применение экологического подхода к исследованию программного обеспечения и формулируются основные положения нового раздела инженерии программного обеспечения – экологии программного обеспечения.

Можно указать три причины для применения экологических исследований к программному обеспечению. Первые две причины связаны с концепцией устойчивого развития и определяются влиянием программных продуктов и их производств на окружающую среду. Третья причина вызвана необходимостью наблюдений за программным обеспечением как за организованной системой в контексте реального мира.

Первая причина, основывается на предположении, что программные продукты и их производство в индустрии программного обеспечения, как в других инженерных отраслях влияют на окружающую среду и следовательно на биосферу [1, 2]. Это влияние может быть вредным и неуправляемым.

Вторая причина, основывается на парадоксе развития научно-технического прогресса, суть которого заключается в том, что приходится осуществлять деятельность по ликвидации результатов деятельности. При этом, в любой отрасли деятельности человека, и в индустрии программного обеспечения также, понятна и разумна постановка задачи – утилизировать как можно больше из того, что стало не нужно [3, 4].

Третья причина основывается на наблюдении, что эффективное планирование развития и обслуживания программных продуктов требует понимания их места в реальном мире, с учетом их взаимодействий с реальным миром, внутри продукта, и внутри элементов продукта [5].

Применение экологического подхода в программном обеспечении

Под экологическим подходом будем понимать методологию исследований, в основе которой лежит рассмотрение объекта, как части окружающей среды, как правило в форме экосистемы, с учетом устойчивого развития.

Распространение общих принципов и требований экологичности на процессы жизненного цикла программного обеспечения объясняется тем, что в настоящий момент программное обеспечение является продуктом инженерной технической деятельности [6]. Индустрия программного обеспечения потребляет значительные ресурсы, а программные продукты, функционируют во всех хозяйственных отраслях, поэтому прямо или косвенно индустрия и продукты оказывают значительное влияние на окружающую среду [3]. Кроме того, программное обеспечение потенциально опасных технических объектов, в случае его неправильного функционирования, может быть причиной прямых экологических катастроф.

В контексте концепции устойчивого развития [2, 7], программное обеспечение является его активом вследствие того, что оно продукт производственной деятельности человека. Известно, что производственная деятельность человечества все чаще вступает в противоречие с процессами, поддерживающими устойчивый круговорот в биосфере. В. Вернадский указывал на необходимость решения задачи перехода от стихийных взаимодействий человека и биосферы к сознательным, которые превращают биосферу в ноосферу и обеспечивают устойчивое развитие [8]. Сейчас, эта задача вновь актуальна в контексте программы совместных действий, в интересах устойчивого развития, и программное обеспечение играет важную роль в ее решении вследствие того, что оно является следующим:

- - основой всех информационных и коммуникационных технологий;
- - наукоемким образованием;
- - продуктом коллективной, все чаще глобальной деятельности.

В контексте экологического подхода программное обеспечение рассматривается, как технический объект, взаимодействующий с окружающей средой. Такой аспект рассмотрения относится к той части экологии, которая иногда обозначается «относящейся к окружающей среде» (environmental) [7]. Экологизация программного обеспечения фактически делает отрасль разработки программных продуктов объектом исследований инженерной экологии техносферы [3, 9]. Основные цели такой экологизации – это сохранение природных ресурсов и защита окружающей среды. Достижение целей требует постановки и решения соответствующих задач.

Применение экологических исследований в индустрии программного обеспечения показывает, что их распространение идет в трех основных направлениях:

- первое, связанно с применением к программному обеспечению общих принципов и требований экологичности производства и использования технических объектов [10, 11]:
- второе, связанно с реализацией ресурсосберегающих и безотходных производств программного обеспечения [4, 12, 13];
- третье, связанно с применением к программному обеспечению экологических исследований, рассматривая его как экосистему [14, 15].

Экологическое производство и использование программного обеспечения

Наиболее ярким проявлением первого направления является GreenSoftware – «Зеленое» программное обеспечение [10, 11]. В рамках этого направления основной акцент делается на использовании программного обеспечения, как средства прямого или косвенного уменьшения вредного влияния на окружающую среду. Например, использование электронной почты вместо бумажной, видеоконференций вместо поездок. Кроме того, рассматриваются влияния на окружающую среду при производстве и эксплуатации программного обеспечения, например, эффективное энергопотребление, уменьшение выделения CO и CO_2 в атмосферу и разогрева оборудования, других подобных аспектов взаимодействия между окружающей средой и вычислительными средствами на всех этапах жизненного цикла программного обеспечения. В рамках GreenSoftware важным также является минимизация использования основных (объем памяти, время и быстродействие процессора) и дополнительных (объем внешней памяти, распределение каналов) вычислительных ресурсов.

Наиболее яркими примерами такого подхода являются стратегия корпорации IBM по разработке GreenSoftware [10] и стратегия GreenAsuskомпании Asus [11].

При разработке программного обеспечения корпорация IBM предлагает следующие подходы, большую часть из которых она использует в качестве стратегии IBM - GreenSoftwareStrategy, при разработке собственного «зеленого» программного обеспечения:

- сокращение деловых поездок, используя on-line коммуникации;
- загрузка недостаточно загруженных серверов, для уменьшения потребления энергии и рабочей площади;
- планирование графика выполнения рабочей нагрузки во время умеренной нагрузки, для того, чтобы использовать энергию меньшей стоимости;
- эффективное управление ресурсами жизненных циклов;
- оптимизация разных приложений для уменьшения ресурсов и энергии, которые потребляются;
- объединение и укрупнение ресурсов для уменьшения необходимой площади и упрощения вычислительной инфраструктуры;
- оптимизация нагревания, вентиляции и кондиционирования воздуха, на рабочих местах для сокращения потребления энергии;
- эффективное управление хранением данных;
- сокращение потребления энергии при уменьшении рабочей нагрузки;
- оптимизация бизнес-процессов для сокращения потребления энергии и операционных затрат;
- сокращение использования бумажных носителей путем введения в бизнес-процессы электронных документов.

Стратегия GreenAsus была сформулирована в 2000 году и делится на четыре части. Первая, Green-Design, должна обеспечивать создание и внедрение в производство «зеленых» компонентов, которые состоят из легко восстанавливаемых материалов и эффективно утилизируются. Вторая, Green-Manufacturing обеспечивает разработку и внедрение экономичных и экологичных технологий на производстве. Например, компания полностью отказалась от использования свинца, а до 2009 года собиралась прекратить применение галогенов. Третья часть, Green-Procurement, регламентирует логистику товаров, использование материалов и компонентов от внешних производителей. Кроме этого, стимулируются поставщики на пересмотр своего отношения к проблемам экологии. Четвертая часть, Green-Marketing решает задачи утилизации старых компьютеров. Компания рассматривает последнюю часть важной и в 2010 году собирается начинать свою деятельность по переработке электроники в Украине.

Другие компании начинают решать задачи этого направления. Например, Intelтакже пересматривает свое отношение к проблемам экологии, разрабатывая новый стандарт, который минимизирует вредное влияние на окружающую среду на всех этапах производства.

Ресурсосберегающее и безотходное производство программного обеспечения

Ресурсосберегающее производство – это производство и реализация продуктов с минимальным расходом вещества и энергии на всех этапах производственного цикла и с наименьшим воздействием на человека и природные системы. Основой ресурсосберегающего производства являются ресурсосберегающие технологи [9].

Рассмотрим существующие ресурсы инженерии программного обеспечения (принципы, процессы, конструкции), которые отвечают экологическому подходу и могут использоваться как основа для

построения ресурсосберегающих и безотходных технологий разработки и сопровождения программного обеспечения (табл. 1).

Таблица 1. Методы, принципы, конструкции экологического программного обеспечения

№ п/п	Методы	Подпрограммное (процедурное) программирование	Объектно-ориентирорваное программирование	Модульное программирование
1	Конструкция	Подпрограмма (закрытая, открытая)	Класс (C++, C #)	Модуль (Модула 2,3), пакет (Ada)
2	Принципы реализации конструкций	Параметризация	Полиморфизм, наследование	Раздельная компиляция
3	Принципы использования конструкций	Процедурная абстракция абстракция выполнения, шаблонирование	Повторное и многократное использование (унификация)	Принципы использования конструкций
4	Фундаментальная абстракция	Абстрактный тип даних		

Первой такой конструкцией была подпрограмма, которую рассматривали как средство сокращения текста программы. Гибкость конструкции обеспечивала параметризация. Вскоре взгляд на подпрограмму начал изменяться, и её стали рассматривать, как средство структурной организации программ. Приспособление программы к ожидаемым изменениям постановки задачи или сопровождения сводилось к замене одного или нескольких тел подпрограмм или к изменению значений некоторых фактических параметров в обращении к подпрограммам. Поэтому, применялись подпрограммы как процедурные абстракции, абстракции выполнения (закрытые подпрограммы) или шаблоны (открытые подпрограммы). В качестве метода применения подпрограммы использовалось подпрограммное или процедурное программирование. Позднее, следуя стремлению улучшить конструкцию для реализации абстрактных типов данных были созданы модуль и класс.

В 1984 году, при развертывании исследований связанных с повторными использованными программного обеспечения обращалось внимание на то, что с годами количество новых применений вычислительных машин уменьшается [16, 17]. Это свидетельствовало о том, что сопровождаемое программное обеспечение, которое впоследствии стали называть наследуемым, содержало опыт, и его следовало повторно использовать при создании нового программного обеспечения. С учетом этого жизненный цикл программного обеспечения был расширен дополнительными процессами (рис. 3). С одной стороны, он был дополнен доменным анализом. Его цель, путем анализа опыта, накопленного в домене строить повторно используемые решения, для применения в разработке нового программного обеспечения [18]. С другой стороны, в жизненный цикл, в контексте фазы ликвидации были введены процессы, которые связаны с утилизацией программного обеспечения (рис.4, таб. 2) [13].

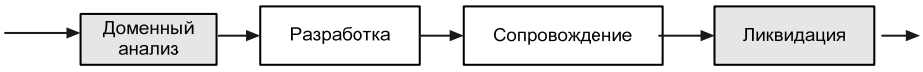

Рис. 3 Расширение жизненного цикла программного обеспечения

Утилизация включает три процесса – повторное использование, восстановление и переработку наследуемого программного обеспечения, а все неутилизированные компоненты уничтожаются (рис. 4).

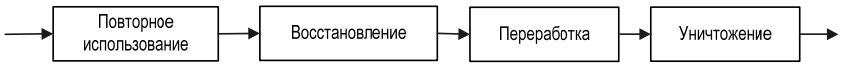

Рис. 4 Содержание фазы ликвидации

Таблица 2 Процессы утилизации

Процессы утилизации	Общая характеристика процессов утилизации	Методы инженерии программного обеспечения	Средства инженерии программного обеспечения
Повторное использование	Создания компонентов, хранение и применение компонентов	Классификация, хранение, поиск	Среда программирования, мониторинговые системы, экспертные системы
Восстановление	Анализ программного обеспечения, преобразование программного обеспечения на одном уровне	Синтаксического семантического анализа прямой и обратной инженерии	Анализаторы преобразователи абстрактори экстракторы
Переработка	Преобразование программного обеспечения на разных уровнях представления. Анализ программного обеспечения	Синтаксического и семантического анализа, реверсивной инженерии	CASE, преобразователи, абстрактор, экстрактор.

Реализация принципа повторного использования привела к появлению понятия реверсивной инженерии программного обеспечения, соединение которой с прямой инженерией дало понятие реинженерии и позволило в экологическом аспекте рассматривать разработку и сопровождение программного обеспечения, как его круговорот (рис. 5) [4] (подкова – «horseshoe» [5]).

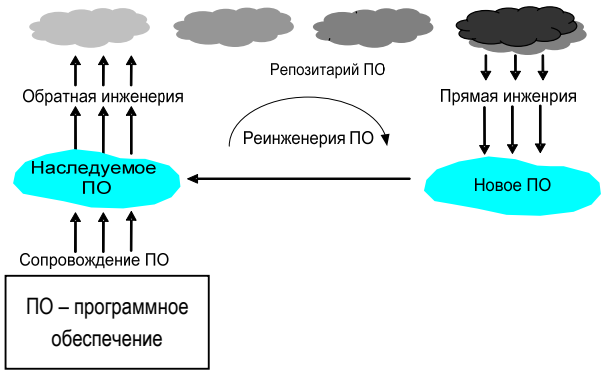

Рис. 5 «Круговорот» программного обеспечения

В доменном анализе и утилизации программного обеспечения сложилось несколько способов создания (табл. 3) и применения (рис. 6) [4] компонентов повторного использования.

Таблица 3. Создание компонентов программного обеспечения

Характеристика	Способы создания компонентов программного обеспечения		
	Утилитарный	Индуктивный	Дедуктивный
Характер способа	Экстенсивный	Интенсивный	Интенсивный
Метафора	«Свалка»	«Фабрика компонентов»	«Высокая технология»
Тип компонентов	Пассивные	Большие пассивные	Большие активные
Техника применения	Композиционная	Композиционная и адаптивная	Адаптивная и композиционная
Метод разработки компонентов	Утилизация	Доменный анализ, снизу вверх	Структуроное проектирование, сверху вниз
Тип применения	Случайный	Случайный систематический	систематический
Средства переработки	Не нужны	Нужные	Не нужные

и возобновления			
Мощность инфраструктуры применения	Высокая	Высокая	Низкая
Формы повторного использования	«Черный ящик»	«Белый ящик»	«Черный ящик»

Индуктивный и дедуктивный способы создания компонентов впоследствии позволили превратить компоненты повторного использования в компоненты многократного использования, которые сейчас составляют основу компонентной разработки программного обеспечения.

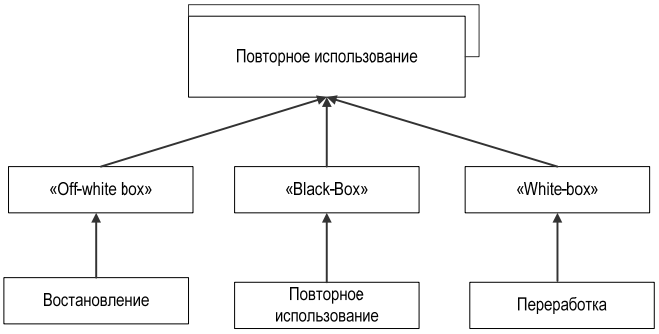

Рис. 6 Аспекты применеия принципов повторного использования

Появились также две парадигмы применения компонентов: адаптивная и сборочная. В соответствии с адаптивной парадигмой – применяется модель программного обеспечения, похожая на фрейм, где компоненты заполняют слоты. В соответствии с сборочной парадигмой – компоненты объединяются программным кодом, как «клеем».

Таким образом, видно, что в инженерии программного обеспечения есть принципы, процессы и конструкции и принципы, которые могут использоваться при построении безотходных, ресурсосберегающих технологий и продуктов.

Программное обеспечение как экосистема

В работе [5] относительно программного обеспечения показано следующее:

- изменение (развитие) – непременное свойство программ, обусловленное наличием обратных связей и связанное с законами эволюции программ;

- метасистема, в рамках которой развивается программа включает продукты деятельности, процессы и организацию, содержит большое количество обратных связей, стабилизирующих внутренних механизмов, влияющих на процессы планирования, управленияи повышения их эффективности;

- эффективное планирование и обслуживание программы требует понимания ее места в метасистеме, а также взаимодействий как между элементами, так и внутри них.

При этом, программы, о которых идет речь, по классификации SPE являются E – программами, а метасистема, – это их внешняя среда – реальный мир [5] .

Рассматривая программные системы в контексте реального мира как экосистемы с учетом рассмотренных выше, особенностей к их исследованиям целесообразно применить экологический

подход. Тогда, одна из основных задач исследований будет состоять в сборе, накоплении, систематизации и анализе информации о количественном характере взаимосвязей внутри программной системы, между программной системой и внешней средой для получения следующих результатов (рис. 7):

- оценка качества исследуемых программных систем – как экосистем;

- влияние причин, изменений компонентов, источников и факторов воздействия;

- прогноз устойчивости программных систем как экосистем и допустимости изменений.

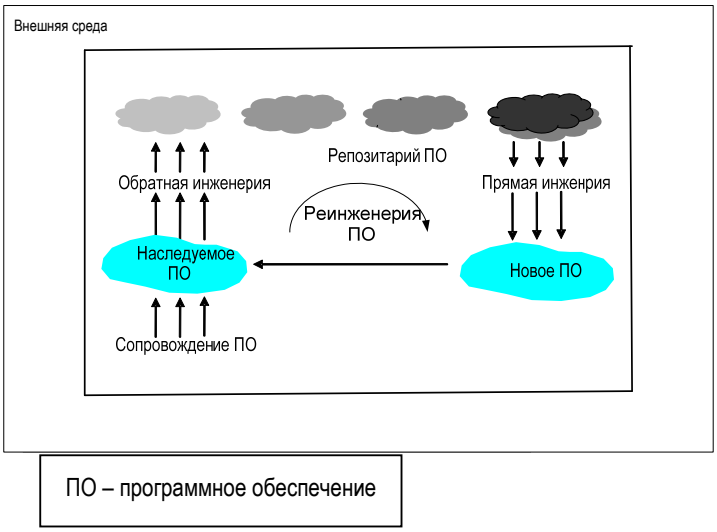

Рис. 7 Е – программа в реальном мире

Таким образом, для выполнения исследований должен быть организован мониторинг программной системы в контексте внешней среды, а исследование программной системы как экосистемы должно осуществляться на основе системного подхода, например, в следующих аспектах:

- морфологическом – исследуется устройство системы;

- функциональном – исследуются функции компонентов системы, обычно в терминах входных, выходных и управляющих параметров, возмущающих воздействий и параметров состояния.

Принимая во внимание тот факт, что исследуется наследуемая программная система, важную роль в экологическом исследовании будет играть реверсивная инженерия.

Заключение

Таким образом, принимая во внимание представление направленные экологических исследований программного обеспечения, в целом, видимо, можно говорить об экологии программного обеспечения как разделе инженерии программного обеспечения, хотя и понимая, что наиболее полно это касается «энвайроиментологии», как основы рационального использования ресурсов в целях устойчивого развития.

Практическим результатом экологических исследований было бы создание научно-обоснованных технологий создания и применения программного обеспечения, основанных на оптимальном использовании ресурсов, в том числе и природных, обеспечивающих их поддержку, восстановление и контролированное устойчивое развитие.

Такой результат может быть получен только путем широкой пропаганды экологического подхода как в среде разработчиков, так и пользователей программного обеспечения.

Библиография

[Баландин, Вернадский, 1999] Р.К. Баландин, В.Н. Вернадский: жизнь, мысль, бессмертие. М., «Знание», 1979.

[Сидоров, 2006] М.О.Сидоров. Экология программного обеспечения. – Материалы Всеукраинской конференции аспирантов и студентов «Инженерия программного обеспечения» – К.: НАУ, 2006. – С. 41 - 48

[Мазур, Молданов, 2006] И.И. Мазур, О.И. Молданов. Курс инженерной |экологии: - М.: Высш.шк., 1999.-447 с| |.

[Сидоров, Шарепа, 1990] Н.А. Сидоров, А.Н. Шарепа. Средство для утилизации программного обеспечения // УсиМ.- 1990.- №5.- с. 50-54.

[Lehman, Belady,1985] M.M. Lehman, L.A. Belady. Program Evolution.- Academic Press.- 1985.- 532 p.

[Sommerville, 2001] I. Sommerville. Software Engineering.- Addition- Wesley.- 2001.- 625 p.

[Угольницкий, 2002] Г.Л. Угольницкий. Иерархические управления устойчивым развитием социальных организаций. – Общественная наука и современность. - №3. – 2002. – с. 133 – 140.

[Вернадский, 1994] В.Н. Вернадский. Несколько слов о неосфере. – Успехи современной биологии. - №18. – вып. – 2. –с. 113 – 120.

[2000] Словарь терминов по экологии.- 2000 г.

[IBM Software, 2008] IBM Software: A green strategy for your entire organization. IBM Software for a greener world June. 2008. NY 10589. U.S.A. Produced in the United| States| of| America|. May| 2008.

[Турнов, 2008] Н. Турнов. Зеленый свет| для ASUS| // PC| WEEK/UE.- #24|(93).- 2008.

[Сидоров, 1989] Н.А. Сидоров. Повторное использование| программного| обеспечения// Кибернетика. – 1989. - №3 – с.46-51

[Сидоров, 1994] Н.А. Сидоров. Утилизация программного обеспечения экономический аспект // Кибернетика и системный аналіз.- 1994.- №.3.- с. 151-166.

[Messershmitt] D.G. Messershmitt , C. Szyperski. Software Ecosystems: Understanding an Indispensable Technology and Industry. – MIT press. – 2003. – 233 p.

[Lungu] M.F. Lungu. Reverse Engineering Software Ecosystems. – Doct. Diss. – USI.- 2009. – 208 p.

[Boehm, 1987] B.W. Boehm.| Improving| Software| Productivity| // Computer|, v. 20, n. 9, 1987, г. 43-57

[Biggerstaff, Foreword, 1984] T. Biggerstaff|. Foreword| //IEEE Trans|, on| Software| Engeneer|. v. 10, n. 5, - в 1984 г. 474-476

[Prieto-Diaz|, 1987] R. Prieto-Diaz| Domian.| Analysis| For| Reusability| // Compsac| 87 Oct|., 7-9.-1987.- г. 23-30

Информацияобавторах

Maksim Lyzkiy – *the 1-st deputy of rector, National aviation university, Cand. Of Sc. (techn.).*

Major Field of Scientific Research: Software engineering, information technology, education.

Nikolay Sidorov – *head of software engineering department, National aviation university, Doct. of sc. (thechn.).*

Major Fields of Scientific Research: Software engineering, software education.