Національна академія наук України
Інститут проблем моделювання в енергетиці

На правах рукопису

СИДОРОВ Микола Олександрович

ІНЖЕНЕРІЯ УТИЛІЗАЦІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМ УПРАВЛІННЯ ТА ЕОМ

05.13.09 — математичне та програмне забезпечення обчислювальних машин і систем

Автореферат дисертації на здобуття наукового ступеня доктора технічних наук

Київ
Інститут кібернетики імені В. М. Глушкова НАН України
1995

Робота виконана в Інституті кібернетики ім. В. М. Глушкова НАН України.

004

Офіційні опоненти: член-кореспондент, доктор фізико-математичних наук, професор
ЮЩЕНКО К. Л.,

доктор технічних наук, професор
НІКІТІН А. І.,

доктор технічних наук, професор
СТАСЮК А. І.

Провідна організація: Інститут прикладної інформатики.

Захист відбудеться «28» грудня 1995 р. о 14-00 год. на засіданні спеціалізованої вченої ради Д 01.91.01 при Інституті проблем моделювання в енергетиці НАН України за адресою:
252680 МСД, Київ 164, вул. Генерала Наумова, 15.

З дисертацією можна ознайомитися в науковій бібліотеці інституту.

Автореферат розісланий «__» _____ 199 р.

Учений секретар
спеціалізованої вченої ради СЕМАГІНА Є. П.

Загальна характеристика роботи

Актуальність. Дисертаційну роботу присвячено дослідженню задач вторинного використання, переробки та відновлення програмного забезпечення (ПЗ) систем керування та ЕОМ.

Теоретики і практики, розроблюючи для підвищення продуктивності програмування кращі засоби організації колективів і керування ними, підвищуючи ефективність процесів ПЗ, усуваючи або автоматизуючи окремі етапи у його виробництві, кожного разу знову розробляють ПЗ, не використовуючи накопичений досвід. Це знайшло своє відображення у моделях життєвого циклу ПЗ, котрі обмежуються фазою супроводження. Такі моделі є не тільки неповними з теоретичної точки зору, але й не відповідають сучасній практиці створення ПЗ, дають невірне уявлення про реальний "коловоріт" програмних ресурсів та витрат. Тому особливо актуальна у теперішній час проблема забезпечення тривалого результатизного застосування ПЗ та окремих програмних ресурсів, один раз уведених у життєвий цикл. Комплексне дослідження вторинного використання, переробки та відновлення ПЗ сприяє вирішенню зазначеної проблеми.

Стан проблеми. Перші роботи, що були спрямовані на використання програмного досвіду у розробці нового ПЗ, заходжувалися давно у різних галузях програмної діяльності. Передусім це стосується розробки й використання макросів, стандартних підпрограм, функцій у мовах програмування та модулей у системах обробки. Ці роботи розпочиналися у 70-х роках у галузі утворення макроасемблерних та компілюючих систем (**В.М.Глушков, К.Л. Ющенко, І.В.Вельбицький**) та інформаційних систем (**А.О.Стогній**). Тоді ж були сформульовані основні форми програмування (**А.П.Єршов**) та інтенсивно велися роботи за складальним (**В.В.Липаєв, К.М.Лаврищева**), синтезуючим (**Г.Є.Цейтлін, К.Л.Ющенко**) та конкретизуючим (**В.Н.Кас'янов, І.В.Поттосін**) програмуванням. Ідея вторинного використання застосовувалась у розробці мов програмування (**К.Л.Ющенко, І.В.Вельбицький**), розподілених систем (**P.Wegner**) та навчання (**ADA**). Відновленню та переробці ПЗ вітчизняними вченими приділялося менш уваги (**А.І.Нікітін**). За кордоном міцний та цілоспрямований імпульс такі роботи отримали після конференції НАТО з технології програмування (**McIlroy, 1969**). У 1983 році відбулася перша нарада з проблеми вторинного використання (**T.Biggerstaff**). Цей рік

вважається початком цілоспрямованого вирішення проблеми. Доменний аналіз для рішення цієї проблеми вперше був застосований у системі **Draco (J.Neighbors)**. Принципи класифікування беруть по .аток у роботах **R.Prieto-Dias**. Задача утворення компонентів, що вторинно використовуються на основі існуючого ПЗ, найбільш послідовно вирішується у роботах **G.Caldiera, V.Basilli**. Реверсивні методи, стосовно до проблеми, що вирішується, були використані у роботах **T.Biggerstaff, E.Chikofsky, J.Cross II**. Економічні питання вторинного використання ставилися вперше у роботах **B.Barnes, T.Bollinger**, а правові проблеми - у роботах **P.Samuelson**. З 1983 р. були проведені чотири конференції. Остання відбулася у 1994 р. у Бразілії.

Аналіз стану проблеми показує, що у теперішній час існують розрізнені методи, направлені на вирішення окремих зад ч. Об'єднання цих задач дозволить ефективніше застосовувати методи й засоби, ставити нові задачі, вирішення котрих забезпечить тривале й результативне застосування ПЗ і окремих програмних ресурсів. У зв'язку з цим актуальною є розробка комплексного підходу.

Мета та задачі дисертаційної роботи полягають у визначенні та дослідженні нового розділу в інженерії п ограмування, методи й засоби котрого забезпечують продовження використання програм шляхом їх відновлення, переробки та вторинного використання, а також зменшення вартості розробки та супроводження, прискорення вироблення ПЗ, підвищення якості програм за рахунок використання компонентів існуючого ПЗ у новому ПЗ. При цьому повинні бути вирішені такі задачі: сформулювати основні положення роділу інженерії програмування, який розробляється, вибрати методологію, дослідити принципи та засоби, на основі результатів досліджень розробити архітектури та реалізувати прототипи засобів для вирішення основних задач інженерії.

Методіка досліджень. Проблема розробки інженерії утилізації ПЗ розглядається у трьох аспектах - теоретичному, методологічному та прикладному. Теоретичний аспект полягає у формулюванні та дослідженні цілей, предмету, задач, принципів, аналізі існуючих методів та засобів с точки зору застосування їх у межах розділу. Методологічний аспект полягає у виборі та дослідженні такого підходу, який дозволив би виділити основні об'єкти та відношення, що складають суть рішення задач розділу інженерії програмування, який досліджується. Прикладний аспект полягає у розробці засобів, які дозволяють перевірити вірність прийнятих рішень та відробити практичні навики з їх реалізації.

Основні результати дисертаційної роботи. Визначено розділ інженерії програмування, що називається інженерією утилізації ПЗ. Інженерний напрямок забезпечує розробку та цілоспрямоване застосування методів, засобів та реалізацію процесів, що дозволяють з урахуванням технічних, економічних та соціальних факторів продовжити результативне використання ПЗ, один раз уведеного у життєвий цикл, шляхом застосування його у колишньому визначенні (після відновлення) або шляхом застосування його або компонентів, що входять до нього, по іншому призначенню (після переробки або вторинним використанням). Розроблено наукові основи, досліджено методи й засоби інженерії утилізації ПЗ. У межах роботи, стосовно до розділу, досліджено питання вторинного використання ПЗ, прямої та зворотної інженерії ПЗ, розуміння програм, застосування гіпертекстової технології у інженерії утилізації ПЗ, економічної оцінки ефективності утилізації ПЗ. Розроблено та перевірено на діючих прототипах методи побудови засобів зберігання, пошука та застосування компонентів, що вторинно використовуються; засобів розуміння ПЗ; засобів відновлення та переробки програм; засобів уявлення ПЗ.

Наукова новина. На основі комплексного підходу до задач вторинного використання, переробки та відновлення ПЗ з застосуванням доменного аналізу як методології розроблено новий розділ інженерії програмування - інженерія утилізації ПЗ.

Практична цінність. Методи й засоби, що розроблені й досліджені у дисертації, можуть використовуватися у практичному програмуванні, при масовому виробництві й супроводженні програм.

Автор захищає: 1) інженерію утилізації ПЗ як комплексний підхід до вторинного використання, відновлення та переробки ПЗ; 2) отримані моделі, підходи, засоби, схеми, що складають теоретичну базу інженерії утилізації; 3) розроблені на основі отриманих результатів прототипи інформаційно-програмних засобів, вирішуючих основні задачі утилізації ПЗ.

Реалізація роботи здійснювалася у межах таких НДР і ДКР, що виконувалися у різні часи в Інституті кібернетики ім. В.М.Глушкова НАН України під науковим керівництвом та при участі автора:

- НДР/ДКР "Дослідження та розробка експериментального зразка засобів програмування на основі принципу вторинного використання програмного забезпечення", 1989-1991 р.р., - г/р 952-89, м. Зеленоград, НВО ЕЛАС;

- НДР "Розробка технології утилізації інформаційно-програмного забезпечення систем управління та ЕОМ", 1991-1992 р.р. - Постанова ДК СРСР з науки і техніки, від 7 травня 1991 р. N 702;

- НДР "Вторинне використання інформаці...ю-програмного забезпечення систем управління та ЕОМ", 1992-1995 р.р. - Постанова ДК України з питань науки і технологій від 28.02.92, N 5;

- НДР "Розробка інформаційно-програмних засобів для створення та супроводження програм вбудованих систем на основі принципу вторинного використання" 1992-1995 р.р. - Постанова ДК України з питань науки й технологій від 04.05.92, N 12;

- НДР "Розробка інформа йно-програмного модуля, забезпечуючого гіпертекстову інформаційну технологію у державних автоматизованих системах інформатизації наукової діяльності", 1993-1996 р.р. - Постанова ДК України з питань науки і технології від 05.03.93, N 39;

- НДР "Дослідження технології програмування в інтересах утворення Единої автоматизованої системи управління Озброєних сил України", 1995-1998 р.р. - Програма фундаментальних та пошукових робіт в інтересах Міністерства оборони України, 1992 р.

Практично реалізовані компоненти складаються з таких систем:

- **"ГІПС Плюс"** - інструментальна система для утворення й обробки гіпертекстової інформації в інженерії програмування;

- **"ПУЛ"** - система для утворення, зберігання й використання компонентів, що вторинно використовуються, у мові Модула-2;

- **"Реверс"** - система для переробки програм на баз[illegible] **CASE-Аналітик**.

Отримані у дисертації результати покладено у основу реалізованих засобів для вторинного використання компонентів мови **Модула-2**, засобів розуміння початкового коду програм, засобів утворення й використання гіпертекстової інформації, засобів реверсивної інженерії та переробки програм (**Фортран, Паскаль, Модула-2**). Результати були використані у практиці розробки ПЗ для бортових ЕОМ (НТЦ ЕЛВІС, м.Зеленоград), використовуються у межах CASE технології (CASE-Аналітик, НТП Сітекс, м. Москва, і ЦІТ ДКПНіТ, м.Бердянськ), використовувалися у ПЗ ЕОМ Пошук-2 (НВО Електронмаш, м. Київ); в учбовому процесі (Київський педагогічний інститут) та у проведенні наукових досліджень (Інститут проблем

моделювання у енергетиці, м.Київ).

Публікації та апробації роботи. За темою дисертації автором опубліковано 25 наукових робіт. Роботи [4-6, 10, 15, 21, 23, 24] написано разом з колегами з Інституту кібернетики імені В.М.Глушкова НАН України, які реалізовували під керівництвом автора ПЗ засобів утилізації. У цих роботах автору належить постановка задач, розробка методів та архітектури засобів. Крім того автор приймав участь у реалізації ПЗ засобів, що описані. Роботу [15] написано разом із студенткою (Горобець С., Політехнічний університет, м.Київ), яка виконувала під керівництвом автора дипломну роботу.

Матеріали дисертаційної роботи докладалися на таких конференціях: "Прикладна інформатика автоматизованих систем проектування, управління, програмної експлуатації", м.Калінінград, 1987; "Синтез модульних систем обробки даних", Кишинів, 1988; "Проблеми управління", Ташкент, 1989; "Автоматизація проектування програмного забезпечення систем управління об'єктами, що рухаються", Харків, 1989; "Програмне забезпечення ЕОМ", Тверь, 1990; "Інструментальні засоби підтримки технологій розробки програмних засобів", Бердянськ, 1991; "Human-Computer Interaction-91", Москва, 1991; "Проблеми підвищення ефективності озброєння та військової техніки Військ ППО", Одеса, 1991; "Нові інформаційні технології у вищій школі", Москва, 1991; "WE-Human-Computer Interaction-92", С.-Петербург, 1992; "3'Human-Computer Interaction-93", Москва, 1993, "Методи математичного моделювання бойових дій військ та побудови інтелектуальних систем військового призначення", Київ, 1995, а також докладалися у різні часи на різноманітних секціях семінару АН України з проблеми "Кібернетика".

Об'єм роботи. Дисертаційну роботу викладено на **360** сторінках машинописного тексту, вона складається з вступу, шести частин, сімнадцяти глав, висновку, додатків (**72** сторінки) та списку цитованої літератури (**275** найменувань).

Зміст роботи

Вступ присвячено постановці проблеми, обмірковуванню її актуальності, короткому огляду отриманих результатів.

У першій частині дисертаційної роботи обговорюється проблема підвищення

ефективності утворення й супроводження великих програм та обирається для досліджень перспективний шлях її рішення. Перша частина складається з двох глав.

У першій главі розглядаються ключові проблеми програмування, відзначається неухильний зріст витрат на виробництво й супроводження програм і обговорюються шляхи зниження витрат. При зберіганні темпів росту витрат (12% у рік) у 1995 р. у світі на виробництво ПЗ буде потрібно 450 млрд. доларів. Підкреслюється, що збільшення витрат обумовлюється наявністю двох груп факторів: інтенсивних та екстенсивних. До перших належить, наприклад, фактор ускладнення вимог, які пред'являють до ПЗ. До других - фактор широкого розповсюдження ЕОМ. Крім того, збільшенню витрат сприяє особистий характер комп'ютерних програм. У зв'язку із збільшенням витрат стає все більш актуальною необхідність утворення й супроводження ефективних економічних програмних засобів та засобів, які своєчасно реалізуються. Навіть незначне підвищення продуктивності може привести до суттєвої економії. Аналіз розподілу витрат у фазах життєвого циклу (ЖЦ) ПЗ показує, що важливим є характер довгострокових витрат, пов'язаних з тим, що програми у процесі тривалого супроводження піддягають значним змінам. При цьому коректуючі зміни займають не найзначніше місце. Навпаки великі витрати на супроводження ПЗ, яке удосконалюється, та адаптивне супроводження ПЗ показують на те, що саме у процесі супроводження програмні ресурси накопичують досвід, який доцільно використовувати для зниження витрат як при утворенні таких додатків, так і у суміжних доменах. У теперішній час відомі декілька шляхів, слідуючи котрими можна зменшувати витрати та підвищувати продуктивність програмування. У главі на основі аналізу указаних шляхів як найбільш перспективне обирається вторинне використання існуючого ПЗ.

У другій главі висловлюються основні положення розділу інженерії програмування, розробленого у дисертаційній роботі на основі комплексного підходу до методів та засобів програмування, який називається інженерією утилізації ПЗ. Формулюються основні визначення й принципи, розглядаються методи й засоби, показано зв'язок з системними дослідженнями.

Утилізація ПЗ - це інженерний напрям, забезпечуючий розробку методів і засобів та реалізацію процесів, які дозволяють з урахуванням технічних, економічних та соціальних факторів продовжувати результативне використання ПЗ, один раз уведеного у ЖЦ і виконуваного. Продовження використання ПЗ здійснюється

шляхом застосування його в минулому визначенні (після відновлення) або шляхом застосування його або увіходжуючих до нього компонентів за іншим призначенням (після переробки або вторинним використанням).

Наведемо декілька прикладів утилізації ПЗ.

Приклад 1. У процесі супроводження ПЗ для виконання коректуючих модифікацій, як правило, використовуються швидкі зміни, тобто здіснюється виправлення безпосередньо початкового коду. Швидкі зміни призводять до того, що текст програми стає погано структурованим, а документація не відповідає новим можливостям програми. Це призводить до утрати ефективності експлуатації ПЗ та врешті решт, до припинення його використання. Щоб продовжити результативне використання ПЗ, треба змінити початковий код і (або) документацію. Це здійснюється шляхом їх відновлення методами утилізації.

Приклад 2. Коли супроводження програми стає невигідним або коли вона перестає задовольняти вимоги, що до неї пред'являють, її знімають з експлуатації та знищують. Проте у процесі тривалого супроводження у програмі акумулюється досвід, який доцільно використати знов у ЖЦ ПЗ, тому виділення з ліквідуємих програм окремих компонентів з метою їх вторинного використання дозволить зберегти накопичений досвід і зменшити витрати, які необхідні при утворенні таких нових додатків.

Приклад 3. Застосування CASE технологій підвищує продуктивність програмування. CASE дозволяє не тільки швидко й високоякісно створювати ПЗ, але й ефективно його супроводжувати. Проте у теперішній час сотні мільонів рядків програм, які експлуатуються на різних мовах, написано без застосування CASE. Щоб знизити витрати на їх супроводження і часто щоб продовжити їх ефективне використання, доцільно застосовувати переведення цих програм у ту чи іншу CASE - технологію.

Основні цілі утилізації ПЗ спрямовано на досягнення загальних цілей програмної інженерії, тобто розробка високоякісного ПЗ з меншими витратами, надійного у експлуатації та ефективного у супроводженні. Відміна міститься у підході до досягнень цих цілей, котрий головним чином засновується на вторинному використанні. Також як у програмній інженерії, в утилізації ПЗ досліджуються й загальносистемні і проблемно-орієнтовані аспекти ПЗ. Разом з тим перші результати застосування методів утилізації у розробці й супроводженні ПЗ показали, що

незважаючи на очевидні передумови, не усе ПЗ однаково ефективно утилізується, тому для зменшення витрат на утилізацію необхідні спеціальні дослідження.

Предметом інженерії утилізації є ПЗ та його компоненти у тих їх властивостях та відношеннях, котрі сприяють якомога більшому та результативнішому (ефективному) їх використанню у ЖЦ. Тому основна відміна методів утилізації від інших методів інженерії полягає у тому, що останні завжди передбачають розробку ПЗ знов, тоді як перші грунтуються на існуючому досвіді.

Для ефективного рішення задач утилізації ПЗ повинні бути вивчені ті властивості ПЗ (документації, специфікацій, архітектур, кодів, даних, тестів, мов програмування і т.д.) та його компонентів (процедур, модулів, об'єктів, класів і т. д.), які забезпечують високу утилізуємість. Вивченню підлягають також процеси, методи й засоби утворення й супроводження ПЗ та його компонентів з метою ефективної реалізації принципів утилізації ПЗ.

Принципи утилізації ПЗ - це ряд загальних принципів програмної інженерії. До них належать інкапсуляція, шаблонування, аналогія, інтеграція, класифікація, багаторівневість, системність. Як самостійний розділ інженерії ПЗ утилізація має системостворюючий принцип, це принцип вторинного використання ПЗ (ПВВ).

Задачі інженерії утилізації ПЗ тісно пов'язані з практичною реалізацією указаних принципів та особливо ПВВ. Основна задача інженерії утилізації ПЗ є створення компонентів, що вторинно використовуються, на основі існуючого ПЗ та застосування їх у діючому або новому ПЗ. Такі компоненти - це конструкції ПЗ, які несуть у собі досвід, акумульований у процесі розробки й супроводження ПЗ.

Методологія утилізації ПЗ. Для вирішення задач утилізації, використовуючи відповідні методи й засоби, необхідно привертати широкий і різноманітний досвід, накопичений в області ПЗ, що утилізується. Тому важливу роль в утилізації відіграє аналіз існуючого ПЗ. Предметом аналізу одночасно повинна бути не одна, а ряд реалізованих систем ПЗ (додатків). Треба привертати методи, які дозволяють виявляти рішення й обирати найбільш важливе для майбутніх додатків. У зв'язку з цим обрано доменний аналіз (ДА) як методологічну основу інженерії утилізації ПЗ. ДА - це методологія, відповідно якій інформація, один раз використана у розробці ПЗ, ідентифікується, організується та зберігається з метою вторинного використання її при утворенні нового ПЗ.

Мета ДА полягає не тільки у тому, щоб на основі аналізу різних додатків у

домені ідентифікувати, інкапсулювати й стандартизувати операції та відношення, що дійсно повторюються, але й у тім, щоб підібрати й структурувати загальну (допоміжну) інформацію, необхідну для утилізації програм на протязі ЖЦ.

У закінченні глави пропоновано модель утилізації ПЗ, яку засновано на відношеннях понять основних процесів утилізації - відновлення, вторинного використання й переробки ПЗ у контексті корисності ПЗ. Показано, що основним системостворюючим принципом утилізації є ПВВ, а основні процеси утилізації - це аспекти (сторони) процесу застосування ПВВ. При цьому відновлення ПЗ - це **"off-white box"** вторинне використання ПЗ, яке порушує конструкцію ПЗ, але не порушує його функціональність · використання компонентів ПЗ (те, що звичайно називають **reuse**) - це **"black box"** вторинне використання ПЗ як воне є, переробка ПЗ - це **"white box"** вторинне використання ПЗ, яке порушує і функціональний і конструктивний аспекти ПЗ.

У другій частині дисертаційної роботи досліджуються методи вторинного використання ПЗ. Друга частина складається з п'яти глав, присвячених відповідно ПВВ, компонентам, що вторинно використовуються, їх утворенню і застосуванню, засобам вторинного використання та економічному аспекту вторинного використання.

У третій главі розглядається ПВВ. Суть ПВВ полягає у застосуванні в новій ситуації раніше отриманих результатів, а вторинна використовуваність - це ступені свободи, з якою у новій ситуації можуть застосовуватися раніше отримані результати.

Вторинне використання ПЗ - це використання програмних знань або компонентів існуючого ПЗ у процесах побудови нового ПЗ.

Вторинна використовуваність ПЗ - це спроможність ПЗ або його компонентів до використання у проектуванні, конструюванні, документуванні та супроводженні нових програм. Компоненти ПЗ, у яких виявляється така спроможність, називаються компонентами, що вторинно використовуються - КВВ.

Застосування КВВ у ПЗ не обмежується початковим кодом, тому КВВ - це будь-який результат діяльності у рамках ЖЦ ПЗ, зберігаючий досвід та володіючий вторинною використовуваністю.

Дослідження показують, що у ПЗ застосування ПВВ - це більше, ніж просто техніка, придатна для утилізації одних систем і непридатна для інших. У ЖЦ ПЗ застосування ПВВ впливає на усі процеси ПЗ, тому воно має свої характерні

особливості. У главі досліджуються ці особливості.

У процесах застосування ПВВ важливу роль відіграють такі фактори: об'єкти вторинного використання (специфікації, вимоги, проект, документація, код і т.д.); межі вторинного використання (фазова, яка визначається місцем застосування ПВВ у ЖЦ ПЗ та організаційна, яка визначається організаційним контекстом застосування ПВВ, головним чином, місцем виробника і споживача КВВ у процесах розробки ПЗ); витрати на вторинне використання (пов'язані, по-перше, з рішенням задачі утворення КВВ, по-друге, з рішенням власно задач вторинного використання, які полягають у пошуку КВВ, розумінні їх функцій та особливостей, вбудовані КВВ у ПЗ, що знову створюється, по-третє, з необхідністю підвищення ефективності цих процесів).

У главі розглядаються процеси вторинного використання (утворення КВВ, пошук, порозуміння, настроювання і вбудовування КВВ у нове ПЗ) і проблеми вторинного використання - технічна, економічна, правова та соціальна.

Четверту главу присвячено КВВ. КВВ - це передусім програмний компонент (ПК), тому в главі розглядається поняття ПК. Наводиться декілька відомих визначень ПК (**C.Lins, G. Booch**). Розглядаються дві групи ПК - відкриті (макрос - **Асемблер**, локальний модуль - **Модула-2**, файл - **Сі** и т.д.) та закриті (стандартні підпрограми, процедури, модулі и т.д.). Показано розділення ПК на два типи: пасивні (постійні складові частини ПЗ, котрі не змінюються при використанні, наприклад, процедури) та активні (моделі частин ПЗ, які змінюються при використанні, наприклад, пакети типу generic у мові **Ада**). Показано існування форм ПК, які обумовлені тим, що для однієї функції можна побудувати родину ПК, які відрізняються часом виконання, потрібною пам'яттю і т.д. Звертається увага на існування ПК, віповідаючих певним абстракціям - операція, функція, дане, процес, мегамодуль.

Для кращого розуміння усього розмаїття ПК, деяких деталей їх будови, а також для ефективного рішення задач їх застосування необхідно використовувати класифікації, тому у главі та у **Додатку А** розглядається декілька відомих класифікацій ПК.

У главі уводиться поняття типу ПК, яке зв'язується є базами ЖЦ на основі принципу багаторівневого уявлення ПЗ, тому можуть мати місце ПК вимог, специфікацій, архітектури, проекту, коду, тестів та документації.

Далі у главі досліджується поняття КВВ ПЗ. У теперішній час відомо небагато визначень КВВ (**C. Lins, G. Booch, В.В.Липаев**). Беручи до уваги, що основні

особливості КВВ, які відрізняють їх від ПК, можна виявити у контексті процесів застосування ПВВ, у дисертаційній роботі уводиться модель КВВ, яка є розширенням відомої ЗК моделі. Розширення полягає у тому, що основні властивості ПК (Концепція - ПК вираз чітко визначеної функціональності, яка описується абстрактно; Конструкція - виявлення концепції ПК у вигляді однієї або декількох форм - реалізацій; Контекст - виявлення, по-перше, вимог релевантності середовища і концепції ПК - концептуальний контекст та, по-друге, релевантності середовища й конструкції ПК - конструкційний контекст) описуються поняттями, характеризуючими вторинну використаність (користь, спільність, примітивність, достатність і закінченість), які в свою чергу забезпечуються факторами, обов'язковими для досягнення вторинної використаності ПК. До цих факторів належать правові ("з...онність"), економічні ("вигідність"), технічні ("ефективність") та конструктивні ("якість"), досягнення яких пов'язане з рішенням основних проблем утилізації ПЗ. Далі у главі розглядається стиль уявлення КВВ (стиль проектування, стиль застосування мови й оформлення) і структура КВВ, яка визначається уведеною моделлю та уявляє собою три частини: опис КВВ (концепція), реалізація КВВ (конструкція) та інтерфейс КВВ (контекст). У закінченні глави розглядається задача класифікування КВВ, яка є однією з фундаментальних у вторинному використанні ПЗ. Ця задача вирішувалася традиційно у трьох аспектах. По-перше, необхідно вибрати або розроб ти найбільш ефективну техніку класифікування, по-друге, необхідно розробити відповідні інформаційно-програмні засоби, які забезпечують зберігання й пошук КВВ відповідно обраній класифікації, по-третє, необхідно утворити на базі обраної класифікаційної схеми та інформаційно-програмних засобів конкретну колекцію КВВ. У **Додатку Б** на прикладі трьох конкретних класифікацій КВВ (абстрактних типів даних у **Алі**, КВВ у рамках **IPSE**, класифікація **G.Booch**) розглядаються принципи класифікування КВВ і форми (по **G.Booch**) КВВ ПЗ.

У п'ятій главі розглядаються задачі утворення й застосування КВВ. Задачі утворення й застосування ПК у проц сі розробки ПЗ складають суть нових підходів програмної інженерії (наприклад, структурного й об'єктно-орієнтованого). КВВ у відміну від ПК зберігають досвід, накопичений у процесі розробки й супроводження ПЗ, та застосовують для підвищення ефективності програмування, якості проектів і т.д. не за рахунок розподілу великих програм на маленькі частини, а за рахунок вторинного застосування накопиченого у них досвіду. Утворення й застосування КВВ

вимагає реалізації ряда специфічних процесів (утворення КВВ, зберігання, розуміння, налагодження КВВ і т.д.), а витрати на їх виконання повинні бути меншими ніж ті, що потрібні для написання просто ПК знову. Тому утворення й застосування КВВ вимагають спеціальних підходів, а ефективність застосування ПВВ визначається тим, наскільки швидко задовольняються запити на пошук або утворення потрібного КВВ. Вибір шляху й засобів утворення й застосування КВВ є складна задача. У роботі пропонуються три засоби утворення КВВ: утилітарний, індуктивний, та дедуктивний. Далі у главі пропонується схема утворення КВВ. При масовій розробці КВВ необхідно оцінювати широкий спектр властивостей і вирішувати дві такі задачі: зменшення кількості ПК, які оцінюються вручну і які виділені з ПЗ; оцінювання властивостей ПК, забезпечуючих їх у вторинну використаність. І а першому рівні оцінюються ті властивості ПК, які дозволяють визначити кандидатів у КВВ ПЗ. Звичайно таких ПК близько 5-10%% від усіх виділених гранулятором з ПЗ, що утилізується. Це дозволяє виключити з подальшого розглядання ті ПК, котрі очевидно не вдовольняють вимоги, пред'явлені до КВВ. Тут може використовуватися атрибутна модель (**V.Basilli**). У другому рівні процес оцінювання має більш складний характер, бо властивості необхідно оцінювати точніше з урахуванням вимог нового ПЗ. Він базується на понятті моделі КВВ і включає до себе три стадії - ідентифікація ПК, моніторинг ПК, утворення КВВ.

Другу частину глави присвячено дослідженню шляхів та засобів застосування КВВ. У роботі пропонується підхід до упровадження ПВВ у організації, який передбачає рішення таких задач:

1. вивчення та підготовка організації до застосування ПВВ, вирішується шляхом розробки програми, яка повинна включати три розділи - моніторинг, розробка плану застосування ПВВ, реалізація плану;

2. утворення середовища розробки ПЗ з урахуванням ПВВ - вирішується розширенням моделі середовища розробки ПЗ, яке розглядається як **SDE(Software Development Environment) = ({S},{M},{P})**, де **{S}** - множина структур, тобто об'єктів або агрегатів об'єктів, які представляють ПЗ у процесі його розробки; **{M}** - множина механізмів, тобто мов, інструментів або фрагментів інструментів, котрі, оперуючи структурами, автоматизують процеси розробки ПЗ; **{P}** - множина вимог, котрі у процесі розробки ПЗ середовище пред'являє користувачу. Кожен компонент середовища утворюється з урахуванням вимог застосування ПВВ.

3. Утворення інфраструктури, яка необхідна для застосування ПВВ - вирішується утворенням усередині організації відповідної інфраструктури, яка підтримує реалізацію плану вторинного використання. Інфраструктура повинна складатися з двох компонентів - організаційного й технічного.

У шостій главі досліджуються засоби, необхідні для застосування ПВВ, які поділяються на мови програмування і системи для вторинного використання ПЗ. Розглядено основні властивості мов програмування (класифікація й композиція), що забезпечують застосування ПВВ, пропоновано методику для оцінювання властивостей вторинної використаності мов програмування композиційного типу і визначено показники вторинної використаності мов **FORTRAN, PL/1, PASCAL, BASIC, C, АДА** розглядено основні механізми, що забезпечують реалізацію КВВ у композиційних мовах (видимість, типізація, прихованість, параметризація, поліморфізм, "перепакування"), досліджено принципи й механізми побудови систем вторинного використання ПЗ, у **Додатку В** зроблено огляд конструкцій найбільш поширених систем.

У сьомій главі досліджується економічний аспект утилізації ПЗ. Однією з цілей застосування ПВВ є підвищення продуктивності програмування, однак організація застосування ПВВ потребує витрат. Економічний аспект застосування ПВВ стосується кількісних і якісних моделей, відображуючи процеси формування й раціонального розподілу цих витрат. У економічному аспекті застосування ПВВ розглядаються такі три напрямки:

1). Розробка загальних моделей застосування ПВВ;

2). Розробка моделей для кількісних та якісних оцінок витрат на застосування ПВВ, продуктивності й ефективності;

3). Розробка моделей підвищення ефективності застосування ПВВ.

Усі напрямки засновано на концепції, суть якої полягає у аналізі ефекту від застосування ПВВ на фоні витрат на розробку ПЗ знов, при цьому у теперішній час використовуються два підходи. Відповідно першому базова концепція у кількісних моделях виявляється у вигляді такої формули: **ПЗ = New + Reuse**, де **New** - частина ПЗ, утворена знову, **Reuse** - частина, яка вторинно використовується, тому загальне рівняння витрат приймає вигляд **C= (O-R)*d+ R*b**, де **C** - витрати на розробку ПЗ; **O** і **R** - об'єм усього ПЗ та його частини, що вторинно використовується, відповідно; **d** і **b** - витрати на написання ПЗ знову та вторинне використання частини **R** відповідно.

Другий є підхід "еталонного" проекту, суть якого виявляється формулою $ПЗ = ПЗ_{Const} + ПЗ_{Var}$, де $ПЗ_{Const}$ - постійна ("еталонна") частина ПЗ, $ПЗ_{Var}$ - перемінна його частина. Оцінка витрат в цьому підході будується на подібності витрат для подібних компонентів, а не на обчислюванні їх, наприклад, на основі продуктивності. Загальне рівняння витрат матиме такий вигляд: $C = C_{const} + C_{var}$, де C_{const} - вартість постійної еталонної частини додатку, а C_{var} - вартість тієї частини нового додатку, яка відрізняє його від еталонної.

У дисертаційній роботі при дослідженні економічного аспекту утилізації ПЗ приймається перший підхід. Для економічної оцінки застосуваання ПВВ уводяться показники застосування ПВВ (показник вторинного використання ресурсів, показник безвідхідності ПЗ) і у межах трьох пропонованих моделей (натуральна, ефективна натуральна й товарна) досліджуються вартість розробки ПЗ із застосовуванням ПВВ, продуктивність, доход від застосовування ПВВ, ефективність витрат на застосовування ПВВ, показник вторинного використання ресурсів, показник безвідхідності ПЗ. У закінченні надаються рекомендації для зменшення витрат на застосовування ПВВ.

У третій частині дисертації, яка складається з трьох глав, досліджуються методи утилізації, які забезпечують відновлення ПЗ. При цьому у восьмій главі розглядаються загальні питання відновлення ПЗ, дев'яту главу присвячено проблемі розуміння ПЗ, яка вирішується методами відновлення інформації, у десятій главі розглядаються методи й засоби відновлення програм.

У восьмій главі з урахуванням ПВВ розглядається взаємозв'язок двох розділів програмної інженерії, які у теперішній час все частіше використовуються разом - це пряма інженерія, яка вже давно досліджується, і менш відома зворотна інженерія, виявляється зв'язок між ними та розглядаються основні методи, які використовує зворотна інженерія, уводяться два аспекти відновлення ПЗ. Необхідність зворотної інженерії у теперішній час виникає у трьох випадках: при утворенні КВВ з існуючого ПЗ; при відновленні ПЗ, наприклад, у процесі супроводження; при переробці ПЗ, наприклад, у процесі міграції, або внаслідок змінювання господарсько-економічних зв'язків додатку, або для застосування нових технологій у додатку. До методів зворотної інженерії належать такі: відновлення проектної інформації, реінженерія, реструктуризація, редокументація. Стосовно ПЗ основні цілі зворотної інженерії полягають не в утворенні дубліката системи, а в отриманні інформації для кращого

порозуміння систем, з тим щоб підвищити ефективність супроводження, переробити систему або виділити з неї деякі компоненти, що відповідають заданим вимогам. У главі досліджуються аспекти відновлення ПЗ. Відновлення - це регенерація необхідних властивостей, утрачених ПЗ, з метою поновлення його нормального функціонування. У дисертаційній роботі відновлення ПЗ пропонується дослідити у двох таких аспектах: інформаційний - відновлення інформації про ПЗ; конструктивний - відновлення загублених властивостей конструкції ПЗ.

Дев'яту главу присвячено методам та засобам, призначеним для ефективної реалізації процесів порозуміння ПЗ. Відзначається, що інтерес до проблеми підсилювався тричі - при появі можливості обміну алгоритмами, написаними на мові високого рівня, при вирішенні задач супроводження великих програмних комплексів, при вторинному використанні існуючого ПЗ. Однією з головних задач у вирішенні проблеми розуміння ПЗ є відновлення інформації. У главі розглядено фактори, що впливають на сприйняття інформації у процесах утилізації, аспекти проблеми розуміння ПЗ, шляхи й методи її вирішення. Описано склад, структуру та особливості інформаційно-програмних засобів, забезпечуючих автоматизацію процесів розуміння ПЗ. Особлива увага приділяється засобам, розробленим на основі гіпертекстової технології.

У десятій главі розглядається задача відновлення ПЗ у конструктивному аспекті. Показується роль відновлення ПК у процесах ЖЦ, розглядаються методи й засоби відновлення конструкцій програм. Окремо представлено методи відновлення конструкцій програм для поновлення функціонування, які засновані на реверсивній інженерії. Методи та засоби відновлення конструкцій програм, що розглядені у главі, орієнтовані на початковий код і виконуються за допомогою його модифікації. Це пояснюється тим, що в основному усі конструктивні властивості, які забезпечують корисність ПЗ, можуть бути відновлені шляхом маніпулювання з конструкцією ПЗ, виявленням якої, зокрема, є початковий код. Усі методи можна поділити на дві групи. У першу входять методи відновлення початкової корисності, а в другу - методи відновлення контекстної корисності. Найбільш характерними для першої групи є методи відновлення стилю програмування, а для другої - методи, які забезпечують застосування ПВВ. Засоби відновлення конструкцій ПЗ звичайно використовують методи зворотної інженерії, що передбачає обов'язкове відновлення інформації про програму. До цих методів належать такі: структурне формування програм,

стандартизація стилю програмування, відновлення з використанням препроцесору, відновлення ПК з метою їх вторинного використання.

Четверта частина дисертаційної роботи складається з двох глав та присвячена дослідженню методів і засобів переробки ПЗ.

У одинадцятій главі розглядаються види й способи переробки програм. Переробка ПЗ - це перетворення існуючого ПЗ з метою отримання максимальної користі від нього при зніманні його з супровождения. Умови, що викликають переробку ПЗ, звичайно виникають при нових зовнішніх факторах, коли вимагаються його миграції в інші технічні й технологічні платформи або внаслідок змін ділових (господарсько-економічних) процесів, що відбуваються у даному домені (додатку), коли вимагається реінженерія господарьско-економічних зв'язків і, як наслідок, переробка ПЗ, або нарешті при раз'єднанні ПЗ з метою створення репозитарію КВВ. У главі особлива увага приділяється таким способам переробки програм, які застосовуються у реверсивній інженерії: логіко-базований, трансформаційний, аспектний, програмно-функціональний, заснований на кліше, рішення-орієнтований, заснований на обчислювальних планах.

У дванадцятій главі розлядаються засоби переробки програм, показується загальна схема засобів, описуються деякі реалізації засобів, показується уявлення інформації у засобах переробки ПЗ та розглядається зв'язок переробки програм і CASE систем. Дослідження методів вторинного використання (утворення КВВ), відновлення й переробки ПЗ показують, що значна частина засобів, реалізуючих ці методи, є загальною по архітектурі. Її основу складають два компоненти: екстрактор (на основі аналізу початкової інформації будує уявлення, яке відповідає мовному рівню знань про ПЗ), абстрактор (на основі побудованого екстрактором уявлення будує уявлення, які відповідають програмно-архітектурному і доменному рівням знань про ПЗ). У дисертації розглядаються особливості реалізацій указаних компонентів засобів переробки ПЗ, заснованих на уведених засобах переробки програм та описуються уявлення знань у них. У закінченні розглядається роль CASE у вирішенні задач відновлення й переробки ПЗ, відзначається її велике значення.

П'ята частина дисертаційної роботи складається з двох глав, її присвячено ДА - методології утилізації ПЗ. У цій частині дано основні визначення ДА, виконано огляд підходів до ДА - **Lubars, Kaptur, Prieto-Dias, FODA, Synthesis (Додаток Г)** та обрано підхід як методологію утилізації ПЗ, розглядено роль ДА у прямій та зворотній

інженерії та пропоновано методіку ДА для відновлення й переробки ПЗ.

Тринадцяту главу присвячено порівнянню підходів до ДА та вибору підходу, придатного як методологія утилізації ПЗ. У теперішній час відомі підходи до ДА відрізняються поглядом на місце ДА у процесах розробки ПЗ, методами реалізації ДА, цілями й результатами і т.д. Обираючи підхід, використовують такі фактори: вимоги процесу розробки ПЗ, стан існуючого ПЗ, економічні цілі ДА, стан доменних знань, наміри використовувати результати ДА. Кожний фактор знаходить виявлення через ряд критеріїв (наприклад, предмет ДА, фокус ДА, характер доменної моделі і т.д.), за значеннями котрих і обирається підхід. У главі розглядено фактори й критерії та обрано ДА (**R.Prieto-Dias**) для утилізації ПЗ.

У чотирнадцятій главі розглядається розширення обраного підходу ДА і пропоновано методику ДА для забезпечення процесів відновлення й переробки ПЗ. Основна ціль ДА полягає у отриманні структур ПЗ, які вторинно використовуються. Проте цієї інформації виявляється недостатньо для виконання процесів відновлення й переробки ПЗ. Реверсивні методи, які мають місце при виконанні цих процесів, вимагають додаткової специфічної інформації (наприклад, класифікацій понять, програмних кліше, компонентів ПЗ із зазначенням динамічних і функціональних властивостей і т.д.). Вона визначається тільки на основі аналізу існуючого ПЗ та знань, накопичених у домені, та може бути результатом ДА. У главі пропонується відповідне розширення інфраструктури, яка забезпечує ефективне застосування ПВВ у ЖЦ з урахуванням реверсивних процесів, та на прикладі показується методіка, що розширює обраний підхід до ДА.

Шоста частина дисертаційної роботи складається з трьох глав, її присвячено опису реалізації теоретичних та методологічних результатів, отриманих у роботі у вигляді додатків - гіпертекстова система **ГІПС Плюс**, яка забезпечує інформацією процеси відновлення ПЗ, система **ПУЛ** для утворення, зберігання й застосування КВЗ (**Модула-2**), прототип **РЕВЕРС** засобів вторинного використання, переробки та відновлення ПЗ.

У п'ятнадцятій главі розглядається застосування гіпертекстової технології у процесах утилізації ПЗ, надається класифікація гіпертекстових систем, що пропонуються на основі **Dexter Hypertext Reference Model** (сканують, зі схемою гіпертексту, на основі СУБД, на основі системи управління гіпертекстом з однорівневою та багаторівневою схемами), та обирається модель гіпертексту з

напіввизначеними вузлами, яка розроблена на основі предикатної моделі гіпертексту для застосування в утилізації ПЗ. У Додатку Д уводяться основні визначення гіпертекстових систем. Традиційних властивостей гіпертекстових систем для ефективного застосування їх у програмній інженерії виявляється недостатньо, тому у роботі на основі предикатної моделі гіпертексту запропоновано модель, що дозволяє утворювати й обробляти вузли, які використовують напіввизначені інформаційні об'єкти та .EXE і .COM файли, які гнучко підключаються до них. Цю модель реалізовано у гіпертекстовій системі **ГІпС Плюс**, яка є першою системою у СНД і Україні, для підтримки процесів розробки ПЗ, що забезпечує інтеграцію у середовищі гіпертексту гетерогенних документів, виконання програм різного походження, прототипування і т.д. У главі розглядається конструкція системи та її застосування у програмній інженерії, зокрема, для утворення засобів розуміння ПЗ, швидкого прототипування та інтеграції на основі ПВВ гетерогенної інформації.

У шістнадцятій главі описано систему **ПУЛ**, яка призначена для утворення, зберігання й застосування КВВ у мові **Модула-2**. Система здійснює прийняття, класифікацію, розміщення у репозитарії та ефективне ведення КВВ. Засоби надають такі можливості:

- утворення та ведення класифікаційної схеми для домена або окремого додатку;
- утворення, класифікацію та розміщення КВВ у репозитарії;
- ведення репозитарію КВВ;
- пошук у репозитарії необхідних КВВ та інформації про КВВ за вимогою користувачів;
- застосовування КВВ у ПЗ, яке розроблюється.

Для реалізації процесів зберігання й пошука КВВ у системі пропонується інтегрувальний фасетно-дескрипторний підхід (класифікуюча схема розробляється на основі фасетного аналітико-синтетичного методу, а індексування та інформаційний пошук виконуються у дескрипторному режимі), а ПЗ будується з ряду подсистем (утворення й ведення доменної інформації, концептуальної та зовнішньої схем, довідників, індексації й розміщення КВВ, локалізації й візуалізації КВВ, ведення репозитарію й бази даних). У главі розглядаються конструкція та окремі режими функціонування системи.

У сімнадцятій главі описано прототип засобів **РЕВЕРС**, які об'єднують у єдиний комплекс описані вище засоби і CASE систему, забезпечуючи тим самим усі процеси

утилізації ПЗ - вторинне використання, відновлення й переробку ПЗ.

У основу архітектури засобів покладено розширену концепцію "екстрактор-абстрактор", розглядену в роботі.

Відновлення. У прототипі відновлення розуміється як утворення семантично еквівалентних уявлень початкового коду у межах одного й того ж абстрактного рівня. Відновлення виконується за допомогою гіпертекстового інструментарію ГІПС Плюс, а результатом відновлення є перехресні посилання у початковому коді, структурна діаграма програми (блок-схема), діаграма управління, діаграма потоків даних і т.д. Абстрактор, призначений для відновлення інформації, використовує як вихідні дані модель програмних знань та будує модель для відновлення, яка уявляє собою гіпертекст у форматі гіпертекстової системи ГІПС Плюс.

Вторинне використання. У прототипі вторинне використання виявляється у виділенні із існуючого ПЗ тих ПК, які розглядаються як кандидати у КВВ, та подальшому оцінюванні ПК - кандидатів для отримання КВВ ПЗ. Таким чином основним процесом є виявлення конструкцій програм, які можуть бути вторинно використані, тому ціль процесів реверсивної інженерії полягає у підготуванні інформації для цього. Остаточний склад інформації залежить від конкретних механізмів визначення КВВ. Проте можна виділити інформацію двох видів:

- синтаксична, яку визначено класифікацією конструкцій програм, котрі можуть бути кандидатами у КВВ ПЗ;

- статистична, яка визначається даними про ті чи інші елементи початкового коду, котрі використатимуться при оцінюванні вторинної використаності.

Результат роботи абстрактору у вигляді КВВ надходить на вхід системи ПУЛ для розміщення у репозитарії з метою подальшого використання.

Переробка. У прототипі переробка виконується за допомогою CASE, тому цілі реверсивної інженерії визначаються вимогами, які пред'являє CASE до способу подання програмних конструкцій. Обравши конкретну CASE систему, необхідно, використовуючи її класифікацію конструкцій, установити відповідні конструкції програми, що переробляється. Переробка ПЗ, яке утилізується, у розробленому прототипі здійснюється за допомогою CASE засобів. Як CASE засоби використовується система CASE-Аналітик, розроблена підприємством Ейтекс.

Додатокдо дисертаційної роботи складається з шести частин (А, Б, В, Г, Д, Е), у яких подано: класифікації ПК і КВВ, описи 12 систем вторинного використання,

описи систем розуміння ПЗ, підходи до ДА, основні визначення гіпертекстової технології.

ВИСНОВКИ

Основні результати дисертаційної роботи:

1. Звернено увагу на фактори, що пояснюють зріст витрат на утворення великих програм: зовнішні, пов'язані із зміною вимог до програм та з широким розповсюдженням ЕОМ, та внутрішні, пов'язані з особистим характером програм як об'єктів розробки й супроводження; показано на необхідність перенесення уваги розроблювачів ПЗ з короткострокових на довгострокові витрати якомога тривалішого застосування розробленого ПЗ та його компонентів.

2. Сформульовано основні положення інженерії утилізації ПЗ - пропонованого комплексного підходу; пропоновано основні напрямки, у які слід групувати методи й засоби утилізації ПЗ; виконано аналітичний огляд принципів, застосованих у утилізації ПЗ та пропоновано методологію утилізації ПЗ; пропоновано модель утилізації ПЗ, яку засновано на відношеннях понять основних процесів утилізації - відновлення, вторинного використання та переробки ПЗ у контексті корисності ПЗ. Показано, що основним системоутворюючим принципом утилізації є принцип вторинного використання, а основні процеси утилізації - аспекти (сторони) процесу застосування принципу вторинного використання.

3. Сформульовано принцип вторинного використання ПЗ та уведено пов'язані з ним поняття (вторинне використання, вторинна використаність ПЗ, компоненти, що вторинно використовуються, фактори, які впливають на застосування принципу і т.д.); досліджено особливості застосування принципу у контексті життєвого циклу ПЗ й описано основні процеси застосування; розглядено проблеми вторинного використання ПЗ.

4. Пропоновано й досліджено модель компоненту, що вторинно використовується, котра являє собою якості, характеристики, фактори (умови) та відношення між ними, які відображують загальні вимоги, що ставлять до програмного компоненту у контексті процесів вторинного використання; запроваджено й описано принципи уявлення компонентів, що вторинно використовуються, шляхи їх утворення, пропоновано й описано засоби їх утворення,

схему процесу утворення компонентів, що вторинно використовуються, та схему упровадження принципу вторинного використання в організаціях, розробляючих ПЗ, описано основні особливості її складових частин - модель процесу підготовки організації до упровадження принципу, модель середовища розробки ПЗ та модель інфраструктури.

5. Виконано аналіз основних принципів, що застосовуються у мовах програмування для забезпечення вторинного використання програм (композиція і класифікація), пропоновано методіку та на її основі для найбільш розповсюджених мов виконано оцінку їх властивостей, забезпечуючих вторинну використаність; приведено вимоги до композиційної мови та на прикладі мови Модула-2 розглядено механізми, які використовуються при утворенні КВВ; зроблено аналітичний огляд реалізованих експериментальних систем, забезпечуючих вторинне використання ПЗ.

6. Зроблено аналітичний огляд основних напрямків моделювання економічного аспекту вторинного використання ПЗ; уведено показники для оцінки економічної доцільності застосовування принципу вторинного використання (показник вторинного використання ресурсів, показник безвідхідності ПЗ); уведено й досліджено три моделі процесів застосовування принципу вторинного використання, які дозволяють обирати шлях утилізації ПЗ й оцінювати доцільність витрат; приведено рекомендації для зменшення витрат на вторинне використання.

7. Надано характеристику зворотної інженерії у контексті проблеми, яка досліджується; зроблено аналітичний огляд методів зворотної інженерії та уведено поняття аспектів відновлення ПЗ, указано їх особливості; розглядено задачу розуміння ПЗ й шляхи її рішення методами зворотної інженерії, зроблено аналітичний огляд методів розуміння ПЗ та засобів, побудованих на їх основі; уведено та розглядено види переробки програм як дії, направлені на ефективне використання витрат, зроблених на ПЗ; зроблено аналітичний огляд способів обробки інформації у процесах переробки ПЗ й розглядено загальну схему засобів, реалізуючих процеси відновлення й переробки ПЗ; зроблено аналітичний огляд застосування засобів зворотної інженерії у переробці ПЗ, у кожному засобі визначено роль головних компонентів загальної схеми - екстрактора і абстрактора; розглядено роль CASE у реалізації процесів утилізації ПЗ.

8. Уведено основні поняття доменного аналізу, фактори й критерії, за якими для утилізації ПЗ можна досить обгрунтовано обирати підхід до доменного аналізу;

зроблено аналітичний огляд підходів до виконування доменного аналізу та обрано підхід шляхом використання факторів доменного аналізу і значень критеріїв, які визначаються особливостями процесів утилізації ПЗ; показано роль доменного аналізу у прямій та зворотній інженерії у контексті життєвого циклу ПЗ та розширено обраний підхід для задач відновлення й переробки ПЗ.

9. Зроблено аналітичний огляд нового гіпертекстового методу уявлення та обробки інформації і запропоновано класифікацію гіпертекстових систем на основі **Dexter Hypertext Referer ` Model**; розглядено застосування гіпертекстової технології у програмній інженерії та її зв'язок з CASE; пропоновано й досліджено модель гіпертексту з напіввизначеними вузлами, яка орієнтована на застосування у програмній інженерії.

10. Розроблено під керівництвом автора й реалізовано першу у колишньому СРСР інструментальну гіпертекстову систему **ГІПС Плюс**, призначену для вирішення задач програмної інженерії, описано її застосування у контексті проблеми, що розглядається; розроблено під керівництвом автора архітектуру та реалізовано з його участю інформаційно-програмну систему, яка забезпечує основні процеси вторинного використання компонентів, написаних у мові **Модула-2**; розроблено під керівництвом автора архітектуру та реалізовано прототип системи, забезпечуючий усі процеси утилізації ПЗ. Як інструментальні програми використано розроблені у дисертації засоби гіпертекстового уявлення та обробки інформації (відновлення), засоби утворення, зберігання, пошука й застосування компонентів, що вторинно використовуються (вторинне використання), CASE-Аналітик, АО Ейтекс (переробка).

Основні положення дисертації опубліковані в таких працях:

1. Сидоров Н.А. Повторное использование программного обеспечения// Кибернетика. - 1989. - N 3.- С. 46-50.

2. Сидоров Н.А. Утилизация программного обеспечения. Экономический аспект// Кибернетика и сист. анализ.- 1994.- N 2. - С. 151-167.

3. Сидоров Н.А. Экономические модели утилизации программного обеспечения// Кибернетика и вычисл. техника.- 1993.- N 99.- С. 104-113

4. Сидоров Н.А., Шерепа А.Н. Средства для утилизации программного обеспечения

// УСиМ.- 1990.- N 5.- С. 50-57.

5. Сидоров Н.А., Шерепа А.Н., Копач В.Б. Средства хранения и поиска повторно используемых компонентов // Там же.- 1991.- N 6.- С. 44-48.

6. Сидоров Н.А., Шерепа А.Н., Копач В.Б. Инструментарий для понимания утилизируемого программного обеспечения // Там же.-1992.- N 7/8.- С. 62-70.

7. Сидоров Н.А. Гипертекстовая система - новый инструмент программной инженерии //Там же.- 1993.- N 4.- С. 61-74.

8. Сидоров Н.А. Классификация гипертекстовых систем// Там же.- 1994.- N 6.- С. 37-53.

9. Сидоров Н.А. Повторное использование программного обеспечения на основе языка Ада // Программное обеспечение систем автоматизированного проектирования.-Рига: Рижск.политехн.ин-т, 1988.- С. 60-69.

10. Сидоров Н.А., Шерепа А.Н. Средства создания повторно используемых компонент программного обеспечения// Исследование процедур поддержки принятия решений в автоматизированных системах.- Киев: Ин-т кибернетики им.В.М. Глушкова АН УССР, 1989.- С. 40-45.

11. Сидоров Н.А. Свойства высокопроизводительных вычислительных систем // Математическое и электронное моделирование в информационном обеспечении технических систем.- Киев: КИИГА, 1989, С.38-42.

12. Сидоров Н.А. Инженерия утилизации программного обеспечения // Методы проектирования интеллектуальных прикладных программных систем.- Киев: Ин-т кибернетики им.В.М.Глушкова НАН Украины, 1992.- С.15-22.

13. Сидоров Н.А. Исследование повторного использования программного обеспечения.- М.: ВИНИТИ, 1987.-18 с.- Деп. 15.03.87, N 5457-В87

14. Сидоров Н.А. Повторное использование программ на основе языка Ада // Реализация и применение языка Ада.- Рига: Политехн. ин-т, 1987.- С. 73-74.

15. Sidorov N., Gorobetz S., Kopach V. Model of hypertext with the partly defined nodes //3-WE Human Computer Interaction'93.- Moscow: MICSTI, 1993.- P. 210-217

16. Сидоров Н.А. Повышение качества программного обеспечения информационных систем методом адаптивной настройки //Надежность и качество программного обеспечения.- Киев: Ин-т кибернетики им.В.М.Глушкова АН УССР, 1985.- С. 193-195.

17. Сидоров Н.А. Средства реализации информационно-справочных задач в системах

организационного типа// Проблемы автоматизации организвционного управления.- Тбилиси: ГКНТ СССР, 1985.- С. 275-276.

18. Сидоров Н.А. Адаптация программных систем управляемая синтаксическим описанием // Прикладная информатика систем проектирования.- Калиниград 1986.- С. 119.

19. Сидоров Н.А. Система повторного использования программного обеспечения// Там же.- Калининград, 1987.- С. 67.

20. Сидоров Н.А. Повт ное использование - метод утилизации модульного программного обеспечения// Синтез модульных систем обработки данных.- Кишинев, 1988.- С. 35.

21. Сидоров Н.А., Шерепа А.Н. Повторное использование программного обеспечения при разработке систем управления // Проблемы управления.- Москва: ИПУ АН СССР, 1989.- С. 85.

22. Сидоров Н.А. Средства повторного использования программного обеспечения// Всесоюз. совещ. по автоматизированному проектир. програм. обеспеч. систем управления движущимися объектами.- Харьков: ХАИ.- 1989.- С. 84.

23. Сидоров Н.А. Шерепа А.Н., Копач В.Б. Средства хранения и поиска программных компонентов в системе повторного использования программного обеспечения // Программное обеспечение ЭВМ.- Тверь: НПО Центрпрограммсистем, 1990.- С. 180-184.

24. Сидоров Н.А. Шерепа А.Н., Копач В.Б. Интерфейс гиперсреды для понимания утилизируемого ПО // Взаимодействие человека с компьютером.- М.: ММЦНТИ, 1991.- С. 280-283.

25. Сидоров Н.А. Применение гипертекста в исследовании повторной используемости программного обеспечения // Новые информационные технологии в высшей школе.- М.: МИЭТ, 1991.- С. 41.

Sidorov N.A. Control Systems and Computers Software Engineering Recycling. Thesis for a Doctor's degree in the field of "Software for Computers and Computer-Aided Systems - 05.13.09", Institute of Problems Modeling in Energetics, National Academy of Sciences, Kiev, Ukraine, 1995.

The following results are defended: software engineering recycling as a complex

approach to reuse, recovery and remaking of software systems; the received models, approaches, methods, schemes as a theoretical base of software engineering recycling; realized prototypes of software systems designed for the decision of main tasks of software engineering recycling.

The results received are to reduce expenditures for the software development and maintenance and to improve its quality and reliability.

Сидоров Н. А. Инженерия утилизации программного обеспечения систем управления и ЭВМ (рукопись). Диссертация на соискание ученой степени доктора технических наук по специальности 05.13.09 — математическое и программное обеспечение вычислительных машин и систем, Институт проблем моделирования в энергетике НАН Украины, Киев, 1995. Защищаются следующие результаты: инженерия утилизации программного обеспечения как комплексный подход к повторному использованию, восстановлению и переработке программных систем; полученные модели, подходы, способы, схемы, составляющие теоретическую базу инженерии утилизации; разработанные на основе полученных данных прототипы информационно-программных средств, практически решающие основные задачи утилизации программного обеспечения.

Применение полученных результатов позволит снизить затраты на создание и сопровождение программного обеспечения, повысить его качество и надежность.

Ключові слова: інженерія програмування, вторинне використання, реверсивна інженерія, гіпертекстова технологія.

Підп. до друку 02.11.95. Формат 60×84/16. Папір офс. Офс. друк. Ум. друк. арк. 1,46. Ум. фарб-відб. 1,51. Обл.-вид. арк. 1,50. Зам. 827. Тираж 100 прим.

Редакційно-видавничий відділ з поліграфічною дільницею
Інституту кібернетики імені В. М. Глушкова НАН України
252022 Київ 22, проспект Академіка Глушкова, 40