УДК 004.414

О. М. Трунов, С. О. Волкова

Миколаївський державний гуманітарний університет ім. Петра Могили комплексу "Києво-Могилянська академія"

МОДЕЛЮВАННЯ НАДІЙНОСТІ СТРУКТУРОВАНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

В статті представлено та проаналізовано модель оцінювання надійності програмного забезпечення (ПЗ), що враховує властивості команди розробників та спеціалістів з тестування ПЗ, які працюють над реалізацією структурованого програмного продукту (ПП). Представлена методика дозволяє досліджувати динаміку виникнення дефектів у часі, ймовірність появи критичного дефекту, що визначає ймовірність виникнення відмови модуля ПЗ та системи в цілому, густину ймовірності виникнення відмов та ін. Сформовано функцію надійності для модуля ПЗ та системи в цілому. Приведено класифікацію типів дефектів ПЗ за складністю та представлено розрахунки коефіцієнтів, які впливають на якість та визначають надійність ПЗ. Запропоновано рекомендації щодо підвищення надійності структурованого ПП.

Ключові слова: *програмне забезпечення (ПЗ), тестування ПЗ, надійність, якість, ймовірність безвідмовної роботи, дефект, відмова.*

Вступ. Тенденція зростання складності сучасного програмного забезпечення (ПЗ) є стійкою через широке застосування інформаційних технологій в усіх сферах життя суспільства. Більш того, врахування динамічно зростаючої складності програмних продуктів (ПП), високої вартості та низької якості їх створення не виключає можливість виникнення помилок в ПЗ, що призводить до порушення його працездатності, зниження продуктивності роботи та відмов.

Особливий статус та ступінь контрольованості надається відмовам ПЗ критичного застосування. Відмови медичних програмно-апаратних комплексів (МПАК) є досить критичними, тому що вони в першу чергу пов'язані з можливим негативним впливом на стан здоров'я пацієнта. Детальна класифікація систем критичного застосування та особливості забезпечення їх необхідного рівня якості та надійності розглянуто в роботах [1-3].

Загалом, *проблема надійності програмного забезпечення* розглядається в двох аспектах: забезпечення та оцінювання надійності. Аналіз предметної галузі дозволяє стверджувати, що більшість досліджень присвячені першому аспекту даної проблеми, а відповіді на питання щодо оцінки надійності комп'ютерних програм чітко не визначені. Разом з тим очевидно, що надійність програми набагато важ-

ливіша інших традиційних її характеристик, однак ніякої загальноприйнятої кількісної міри надійності ПЗ досі не існує.

Джерелом ненадійності програм служать помилки, які знаходяться в них, і якщо помилки відсутні, то програма абсолютно надійна. Всі заходи щодо забезпечення надійності програм спрямовані на те, щоб звести до мінімуму, або виключити взагалі помилки при розробці, і якомога раніше їх виявити та видалити після розробки програмного продукту. Варто помітити, що безпомилкові програми, звичайно ж, існують, однак сучасні програмні системи занадто складні для того, щоб бути наділеними такою характеристикою як безпомилковість [4].

Джерелами помилок та загрозами надійності ПЗ є: а) внутрішні: помилки проектування, помилки алгоритмізації, помилки програмування, недостатня якість засобів захисту, помилки в документації та ін.; б) зовнішні: помилки користувачів, збої та відмови апаратури, недостовірність інформації в каналах зв'язку, зміни конфігурації системи та інші [5].

Метою статті є моделювання процесу оцінювання надійності ПЗ в часі для прогнозування таких вихідних характеристик ПЗ як: динаміка виникнення дефектів у часі, ймовірність появи критичного дефекту, ймовірність виникнення відмови, густина ймовірності виникнення відмов та ін.

Введемо визначення таких показників надійності ПЗ як: дефект, помилка та відмова ПЗ. Дефект ПЗ – це явна або гіпотетична причина відмов системи, тобто відхилення від результатів коректного обслуговування користувачів ПЗ. Помилка ПЗ – запис елемента програми або тексту програмної документації, використання якої приводить або може привести до невірного результату. Відмова ПЗ – подія, що полягає у прояві непрацездатності ПЗ [6-8]. Вищеописані показники нерозривно пов'язані між собою причинно-наслідковим зв'язком (*рис. 1*). Даний зв'язок є багатоланковим, що характерно для сучасного ПЗ. При кодуванні програміст випадково вносить дефект у код. Далі при роботі ПЗ цей дефект проявляється та призводить до помилки ПЗ, що у свою чергу призведе до відмови системи.

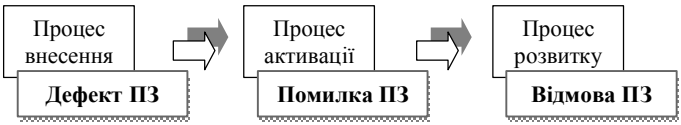

Рис. 1. *Взаємозв'язок понять дефект, помилка, відмова ПЗ*

Важливим етапом життєвого циклу програмного забезпечення, що визначає якість і надійність системи є тестування. Тестування – процес виконання програм з метою виявлення помилок. Для структурованого ПЗ характерні наступні етапи модульного тестування ПЗ: автономне тестування, контроль окремого програмного модуля окре-

мо від інших модулів системи; тестування взаємозв'язків, контроль зв'язків між частинами системи (модулями, компонентами, підсистемами); тестування функцій, контроль виконання системою автоматизованих функцій; комплексне тестування, перевірка відповідності системи вимогам користувачів; тестування повноти та коректності документації, виконання програми в строгій відповідності з інструкціями; тестування конфігурацій, перевірка кожного конкретного варіанта поставки системи та ін. [9, 10].

Однією з важливих ознак життєвого циклу ПЗ є зміна у часі кількості дефектів, які виявляються завдяки тестуванню продукту та роботі по їх усуненню. У більшості робіт ці залежності задаються апріорно. Здійснимо спробу встановити аналітичні залежності.

Постановка задачі. Позначимо N – кількість дефектів, тоді їх зміна у часі буде пропорційною, по-перше, загальній їх кількості, по-друге, визначатися: $f_i(t)$ – ймовірністю того, що команда спеціалістів з тестування знайде дефекти; $\psi_i(t)$ – ймовірністю того, що команда розробників виправить знайдені дефекти. Останні в свою чергу, визначаються факторами, що характеризують засоби тестування та кваліфікацію команд χ_i та μ_j, де χ_i – коефіцієнт пропорційності, що характеризує вплив процесу знаходження дефектів командою спеціалістів з тестування, μ_j – коефіцієнт пропорційності, що характеризує вплив процесу усунення дефектів у сусідніх модулях командою розробників.

Система рівнянь має наступний вигляд:

$$\begin{cases} dN_i = -\chi_i N_i f_i(t)\psi_i(t)dt + \sum\limits_{j=1, j\neq i}^{m} \mu_j N_j f_j(t)\psi_j(t)dt, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots, \\ dN_m = -\chi_n N_n f_n(t)\psi_n(t)dt + \sum\limits_{j=1, j\neq i}^{m} \mu_j N_j f_j(t)\psi_j(t)dt. \end{cases} \quad (1)$$

Розв'язання задачі. Розв'язок системи диференційного рівняння з урахуванням граничних умов $N = N_0$ та часу від t до $t + \Delta t$ має наступний вигляд:

$$N_i(t) = (2 \cdot N_{0i} - N_0) \cdot e^{-\int \chi_i f_i(t)\psi_i(t)dt} + \sum_{j=1, j\neq i}^{m} N_{0j} \cdot e^{\int (\mu_j - \chi_i) f_j(t)\psi_j(t)dt}, \quad (2)$$

при чому $\chi_i \leq 1, \mu_j \leq 1$ та $\mu_j < \chi_i$; N_{0i} – кількість дефектів в i-му модулі, N_0 – загальна кількість дефектів, що містить ПЗ у початковий момент часу t_0 при $N_0 \geq N_{0i}$, $N_0 = \sum\limits_{i=1}^{m} N_{0i}$.

Сформулюємо загальні припущення моделі, а також припущення типу та класу моделей надійності ПЗ. *Загальні припущення* (модулі системи являються незалежними; ПЗ проходить тестування в умовах близьких до реальних умов експлуатації; дефекти видаляються негайно; прояв всіх дефектів відбувається незалежно один від одного; дефекти одного типу складності проявляються з однаковою ймовірністю). *Припущення типу* (всі виявлені дефекти видаляються, однак в процесі корегування ПЗ можуть бути внесені). *Припущення класу/сімейства* (інтенсивність прояву дефектів непостійна, а є функцією від часу і залежить від характеристик команди χ_i та μ_j).

Змоделюємо процес зміни кількості дефектів у ПЗ з використанням пакету MatLab (*рис. 2.*). Маємо програмний продукт, який складається з $m = 10$ модулів. Над розробкою даного ПЗ працюють дві команди:

– спеціалістів з тестування ПЗ, характеристикою якої є ймовірність знаходження дефектів $f_i(t) = [f_1(t), f_2(t), \ldots, f_m(t)]$ для $m = 10$ модулів;

– розробників ПЗ, характеристикою якої є ймовірність виправлення дефектів $\psi_i(t) = [\psi_1(t), \psi_2(t), \ldots, \psi_m(t)]$ для $m = 10$ модулів відповідно.

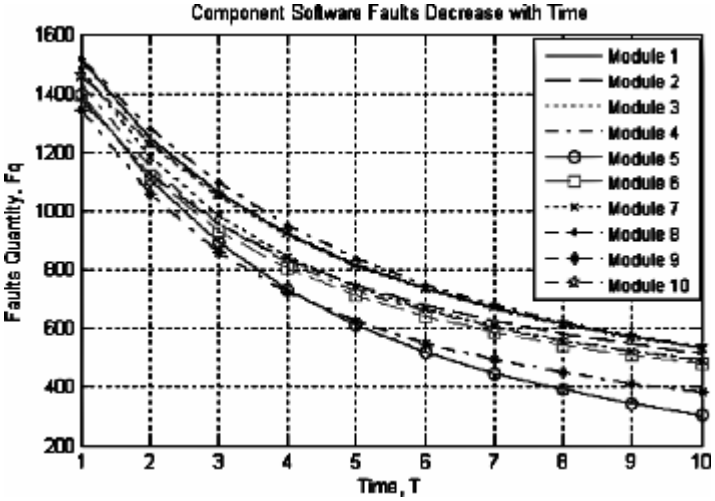

Рис. 2. *Динаміка зміни кількості дефектів ПЗ у часі* $N_i(t)$

Крім того, коефіцієнти χ_i та μ_j отримані за допомогою накопичення статистичних даних про попередні програмні продукти, також формулюються для кожного модуля відповідно: $\chi_i = [\chi_1, \chi_2, \ldots, \chi_m]$

та $\mu_j = [\mu_1, \mu_2, \ldots, \mu_m]$. Вектор значень початкової кількості дефектів для $m = 10$ модулів задається як

$$N_{0i}(t) = [N_{01}, N_{02}, N_{03}, N_{04}, N_{05}, N_{06}, N_{07}, N_{08}, N_{09}, N_{010}].$$

Опишемо множину вхідних даних для запропонованої моделі типу "білий ящик" для оцінки надійності ПЗ:

$$Input = \{N_{0i}, N_0, \chi_i, \mu_j, f_i(t), \psi_j(t)\}.$$

Множина вихідних шуканих значень має вигляд: $Output = \{N_i(t)\}$.

Більшість провідних фахівців в предметній галузі пов'язують поняття надійності програмних засобів з надійністю апаратних засобів (АЗ). Це і є причиною можливості використання деяких термінів і показників надійності АЗ при дослідженні якості ПЗ [1, 4, 5, 8]. Наприклад, одним з найважливіших показників надійності, що представляють інтерес для практики, є ймовірність безвідмовного функціонування ПЗ. Тому здійснимо спробу однозначно пов'язати дефекти ПЗ та його відмови.

Ймовірність того, що через дефекти в *i*-му модулі виникне відмова ПЗ залежить від типу дефекту. Визначимо основні типи дефектів, такі як: *critical* (критичні – призводять до відмови ПЗ), *major* (середньої важливості – впливають на коректність функціонування ПЗ), *minor* (незначні – не впливають на коректність функціонування програмного забезпечення).

Далі опишемо множину дефектів для *i*-го модуля:

$$N_i(t) = K_{mn} N_i^{mn}(t) + K_{mj} N_i^{mj}(t) + N_i^{cr}(t),\ i = \overline{1, m}, \qquad (3)$$

де K_{mn} та K_{mj} – коефіцієнти, які визначають вплив на якість програмного продукту. Для дефектів критичного типу, що визначають надійність ПЗ коефіцієнт впливу дорівнює 1.

Для системи, яка складається з *m* модулів множина дефектів має вигляд:

$$\sum_{i=1}^{m} N_i(t) = K_{mn} \sum_{i=1}^{m} N_i^{mn}(t) + K_{mj} \sum_{i=1}^{m} N_i^{mj}(t) + \sum_{i=1}^{m} N_i^{cr}(t). \qquad (4)$$

Якщо проаналізувати причини виникнення відхилень від коректного функціонування програмного забезпечення, то зрозуміло, що множина $V_i(t)$ складається з трьох підмножин: $V_i(t) = \{V_{mn}, V_{mj}, V_{cr}\}$. Описуючи взаємозв'язок дефектів $N(t)$, кількість відхилень від коректної роботи $V(t)$ та відмов $F(t)$ ПЗ в динаміці з моменту часу t до $t + \Delta t$ маємо наступні співвідношення:

$$V_{0i}(t) = V_{0mn} + V_{0mj} + V_{0cr} \sim \sum_{q=1}^{3} V_{0iq}(t); \qquad (5)$$

$$V_i(t) = V_{mn} + V_{mj} + V_{cr} \sim \sum_{q=1}^{3} N_{iq}(t). \quad (6)$$

Визначимо коефіцієнти, що визначають вплив на якість та надійність програмного продукту, такі як:

1) кількості повних відхилень системи – $K_{cr} = 1$;

2) кількості відхилень, що свідчать про некоректне функціонування системи: $K_{mj} = V_{mj} \Big/ \sum_{i=1}^{m} V_i$;

3) кількості відхилень, що не впливають на коректність функціонування системи: $K_{mn} = V_{mn} \Big/ \sum_{i=1}^{m} V_i$. Слід зауважити, що для визначення кількості відмов необхідно покласти $K_{mj} = K_{mn} = 0$, так як дані коефіцієнти визначають лише вплив дефектів на якість ПЗ, а не є причиною виникнення відмов системи.

Ймовірність виникнення відмови у *i*-му модулі у момент часу *t*:

$$P_i^V = \sum_{j=1}^{m_i} P_{ij}^{calls} P_{ij}^{Ncr}, \quad (7)$$

де m_i – кількість структурованих одиниць в *i*-му модулі, P_{ij}^{Ncr} – ймовірність наявності критичного дефекту у *j*-й структурній одиниці *i*-го модуля, P_{ij}^{calls} – ймовірність звернень до *j*-ї структурної одиниці *i*-го модуля (визначається алгоритмом та структурою ПЗ).

Ймовірність наявності критичного дефекту у j-й структурній одиниці i-го модуля:

$$P_{ij}^{Ncr} = \frac{N_i^{cr}}{m_i}, \text{ при чому } P_{ij}^{Ncr} = \begin{cases} 0, N_i^{cr} < 1 \\ 1, N_i^{cr} \geq 1. \end{cases} \quad (8)$$

Ймовірність відмови структурованого ПЗ, що складається з *m* модулів визначається наступним чином:

$$P_{system}^V = \sum_{i=1}^{m} P_i^V = \sum_{i=1}^{m} P_i^{calls} P_i^{Ncr}. \quad (9)$$

Якщо модуль містить різні структурні одиниці, наприклад для процедурних мов програмування (МП) – підпрограми, процедури, оператори та ін.; для об'єктно-орієнтованих мов програмування – об'єкти, класи та ін., тоді *ймовірність відмови і-го модулю ПЗ* визначається за аналогією з (7) з урахуванням ієрархії структури ПЗ та мови програмування:

$$P_i^V = \sum_{j=1}^{m_i} P_{ij}^{calls} P_{ij}^{Ncr} + \sum_{q=1}^{q_i} P_{iq}^{calls} P_{iq}^{Ncr} + \ldots + \sum_{f=1}^{f_i} P_{if}^{calls} P_{if}^{Ncr} , \qquad (10)$$

де P_{iq}^{calls} – ймовірність звернення до q-ї підпрограми, P_{iq}^{Ncr} – ймовірність наявності критичного дефекту в q-й підпрограмі; P_{if}^{calls} – ймовірність виклику f-ї функції, P_{if}^{Ncr} – ймовірність наявності критичного дефекту в f-й функції.

Далі *сформуємо функцію ймовірності виникнення відмов для структурованого ПЗ*, що складається з m модулів та приведена на *рис. 3*.

$$P_{system}^V(t) = \sum_{i=1}^{m} P_i^V(t) = \sum_{i=1}^{m} \sum_{j=1}^{m_i} P_{ij}^{calls} P_{ij}^{Ncr} , \qquad (11)$$

Для i-го модуля функція надійності, що визначає *ймовірність безвідмовного функціонування ПЗ* формується таким чином:

$$R_i(t) = 1 - P_i^V = 1 - \sum_{i=1}^{m} P_i^{calls} P_i^{Ncr} , \; i = \overline{1, m} . \qquad (12)$$

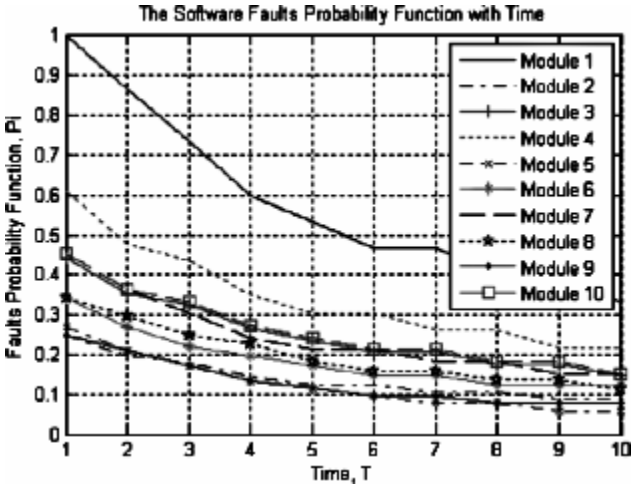

Рис. 3. *Функція ймовірності наявності* N_i^{cr} *в і-му модулі* $P_{ij}^{Ncr}(t)$

Сформуємо *функцію надійності для структурованого ПЗ* з m модулів, як $R_{system}(t) = 1 - P_{system}^V$, що в повному вигляді:

$$R_{system}(t) = 1 - \sum_{j=1}^{m_i} P_{ij}^{calls} P_{ij}^{Ncr} + \sum_{q=1}^{q_i} P_{iq}^{calls} P_{iq}^{Ncr} + \ldots + \sum_{f=1}^{f_i} P_{if}^{calls} P_{if}^{Ncr} . \qquad (13)$$

Густина ймовірності відмов – це ймовірність виникнення відмови в інтервалі часу від часу t до $t+\Delta t$:

$$I(t)=\lim_{\Delta t\to 0}\frac{P_i^V(t+\Delta t)-P_i^V(\Delta t)}{\Delta t},\ i=\overline{1,m}. \qquad (14)$$

Тобто, густина ймовірності виникнення відмов (функція ризику) для структурованого ПЗ, яке складається з m модулів описується наступною формулою:

$$I(t)=\sum_{i=1}^{m}\sum_{j=1}^{m_i}\frac{P_i^{calls}}{m_i}\frac{dN_i(t)}{dt},\ i=\overline{1,m}. \qquad (15)$$

Обговорення результатів. Як результат моделювання процесу оцінювання надійності структурованого ПЗ, необхідно відзначити наступні обмеження щодо використання запропонованої моделі:

– якщо розробники ПЗ виправляють менше дефектів їх ніж виявляють спеціалісти з тестування, тобто $\psi_i(t)\le f_i(t)$, то відповідно збільшується (накопичується) загальна кількість дефектів $Output=\{N_i(t)\}$, що в першу чергу підвищує ймовірність виникнення відмови та зменшує надійність ПЗ;

– чим менші ймовірності виявлення $f_i(t)$ та видалення $\psi_i(t)$ дефектів, тим більше загальна кількість дефектів $Output=\{N_i(t)\}$, тобто властивості команди в першу чергу впливають на надійність ПЗ.

Запропонована модель демонструє динаміку зменшення кількості дефектів у часі $N_i(t)$, що пов'язано з негайним виявленням дефектів шляхом використання технологій тестування ПЗ.

Як рекомендації щодо підвищення надійності ПЗ варто відзначити методи попередження відмов ПЗ, що можуть використовуватися на окремих етапах його розробки, такі як: методи, що призначені для зменшення складності системи; методи негайного виявлення та усунення дефектів на кожному кроці ЖЦ ПЗ та ін.

Висновки. В результаті проведеного дослідження представлено та проаналізовано модель оцінювання надійності ПЗ. Варто відмітити новизну запропонованої моделі, що дозволяє врахувати властивості команди спеціалістів з тестування $f_i(t)$ та розробників ПЗ $\psi_i(t)$.

Розраховано показники моделювання надійності структурованого ПЗ:

– динаміка виникнення дефектів у часі $N_i(t)$;

– ймовірність появи критичного дефекту N_i^{cr} для *i*-го модуля $P_{ij}^{Ncr}(t)$, що в першу чергу визначає ймовірність виникнення відмови *i*-го модуля $P_i^V(t)$ та системи в цілому $P_{system}^V(t)$;

– густина ймовірності виникнення відмов $I(t)$.

На основі даних розрахунків сформовано функцію надійності, тобто ймовірності безвідмовної роботи *i*-го модуля ПЗ $R_i(t)$ та системи $R_{system}(t)$.

Крім того, приведена класифікація типів дефектів та представлено розрахунки коефіцієнтів, які визначають вплив на якість та визначають надійність ПЗ. Сформовано рекомендації щодо підвищення надійності структурованого ПЗ.

В перспективі подальших досліджень варто більш чітко розрізняти поняття відмови, яка виникає одноразово, та яка виникає постійно внаслідок перенавантаження ПЗ, наприклад під час навантажувального тестування.

Список використаних джерел:

1. Липаев В. В. Надёжность программных средств. Серия "Информатизация России на пороге 20 века". – М.: СИНТЕГ, 1998. – 232 с.
2. Оценка качества, надежности и безопасности программного обеспечения информационно-управляющих систем АЭС: модели, методики и средства / В. С. Харченко, Б. М. Конорев, О. М. Тарасюк и др. // Тр. Международный симпозиум "Измерения, важные для безопасности в реакторах". – Москва, 2003. – С.11-12.
3. Volkova S. O., Trunov O. M. Medical systems: quality and safety in robotic surgery // Collected Abstracts 10th All-Russian Scientific and Technological Conference with International Participation "Extreme Robotics". – April 8-9. – Saint-Petersburg, 2008. – P. 74-75.
4. Майерс Г. Надежность программного обеспечения / Пер. с англ.; под ред. В. Ш. Кауфмана. – М.: Мир, 1980. – 360 с.
5. Тейер Т., Липов М., Нельсон Э. Надежность программного обеспечения / Пер. с англ. – М.: Мир, 1981. – 323 с.
6. ДСТУ 30004-95. Надійність техніки. Методи оцінки показників надійності за експериментальними даними. – К.: Держстандарт України, 1995. – 42 с.
7. ДСТУ 2861-94. Надійність техніки. Аналіз надійності. Основні положення. – К.: Держстандарт України, 1994. – 33 с.
8. Lyu M. R. Handbook of Software Reliability Engineering. – McGraw-Hill Company, 1996. – 805 p.
9. Канер С., Фолк Д., Нгуен Е. К. Тестирование программного обеспечения / Пер с англ. – К.: DiaSoft, 2000. – 544 с.
10. Майерс Г. Искусство тестирования программ. – М.: Финансы и статистика, 1982. – 176 с.

The presented model for software reliability estimation is analyzed. The model is based on team's individual and professional characteristics, which works on the software development and consists of developers and testers. The methodology allows investigating the dynamic of software defects in time, the probability of critical defect rising, which defines the probability of software failure, and probability density of software failure.

The classification of software defect types by complexity is presented. The calculations of coefficients that define software quality and reliability are demonstrated. The software reliability function is formed. The recommendations for structured software reliability increasing are proposed.

Key words: *software, testing, reliability, quality, faultness probability, defect, fault.*

Отримано: 20.05.2008

УДК 681.513.2

М. Ф. Ус, Н. Л. Костьян

Восточноевропейский университет экономики и менеджмента, г. Черкассы

МОДЕЛИРОВАНИЕ АДАПТИВНЫХ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ ЭЛЕКТРОННОГО ОБУЧЕНИЯ ДЛЯ ДИСТАНЦИОННОГО ОБРАЗОВАНИЯ

У роботі досліджуються методи адаптації інтелектуального тьютора до індивідуальних особливостей користувача в процесі електронного навчання. Для зберігання глибинної структури різних форм представлення інформації пропонується використовувати семантичну мережу ключових понять, що розроблена для конкретної предметної області знань. Для побудови семантичної мережі використана концепція об'єктно-орієнтованих баз даних. Проектування бази даних мережі реалізовано з використанням уніфікованої мови моделювання UML 2.0. Визначено параметри для подальшого розвитку семантичної мережі. Розроблений сценарій побудови схематичного опорного конспекту навчального матеріалу для одного з когнітивних типів користувача з метою підвищення ефективності навчання. Програмна реалізація алгоритму виконана з використанням мови структурованих запитів SQL і мови об'єктно-орієнтованого програмування Java.

Ключевые слова: *тьютор, когнитивный профиль, семантическая сеть, дефиниция, коэффициент значимости, коэффициент степени неопределенности.*

Введение. Интеллектуальные тьюторские системы – обширный класс электронных систем обучения. Главная идея этих систем – имитация обучающего поведения тьютора-человека. Для поддержки процесса обучения тьютор использует специальные знания четырех основных типов: о предмете обучения, о стратегиях и методах обучения, о студенте, о коммуникативных функциях. Идеальная интеллектуальная система обучения должна представлять и использовать все