УДК 681.3.06

А.М. Ковальчук, асист.
В.Г. Левицький, к.т.н.
Житомирський інженерно-технологічний інститут

РОЗРОБКА АДАПТИВНОГО ІНТЕРФЕЙСУ КОРИСТУВАЧА ПРОГРАМНОЇ СИСТЕМИ ЧИСЕЛЬНОГО АНАЛІЗУ МАТЕМАТИЧНИХ ЗАДАЧ

В даній роботі розглянуто проблему розробки адаптивного інтерфейсу користувача програмної системи чисельного аналізу математичних задач. Запропоновано метод автоматизованого формування адаптивного інтерфейсу користувача програмної системи на основі формального опису предметної області чисельного аналізу математичних задач. Вказано на доцільність використання програмних компонент та компонентної архітектури для створення програмного засобу.

Вступ

Розробка будь-якого програмного забезпечення прикладного характеру завжди робить актуальною проблему організації взаємодії людини та комп'ютера. Як правило, актуальність такої проблеми виражається в необхідності задовольняти зростаючі вимоги користувачів до можливостей програмних систем. Для вирішення питань, що виникають в процесі спілкування користувача та обчислювальної системи, потрібно розглянути шляхи організації існуючого прикладного програмного забезпечення.

Аналіз особливостей сучасних програмних систем дає змогу зробити висновок, що майже всі такі системи складаються з двох основних частин: перша вирішує задачі предметної області програмного забезпечення (ПЗ), а саме: прикладні задачі, загальносистемні задачі тощо, друга відповідає за діалог користувача та обчислювального засобу. Друга частина програмного забезпечення називається інтерфейсом користувача (ІК) програмної системи. Отже, в даному випадку вирішення проблеми організації взаємодії людини та комп'ютера зводиться до проектування інтерфейсу користувача програмного засобу [1–8].

Існує декілька основних напрямків організації інтерфейсу користувача прикладного програмного забезпечення (більшість з них, в деякій мірі, вже стали стандартами для розробників ПЗ) [9]. Всі ці напрямки умовно можна розділити на такі, що в процесі взаємодії комп'ютер–людина передбачають використання:

- проблемно-орієнтованої мови програмування, яка є єдиним засобом опису та аналізу задач предметної області;
- графічного інтерфейсу користувача, який є альтернативою попередньому варіанту розвитку взаємодії користувача та ПЗ;
- проблемно-орієнтованої мови програмування та графічного інтерфейсу користувача, які об'єднуються на певних умовах.

Обидва шляхи мають декілька позитивних та негативних рис, а саме:

- використання в системі спеціальної мови програмування надає гнучкості та універсальності програмній системі (ПС), але вимагає від користувача зусиль, іноді не потрібних, на вивчення та використання такої мови;
- графічні засоби спілкування дають можливість розробити інтерфейс програмного засобу, максимально наближеного до предметної області, проте, як правило, такі системи недостатньо відкриті та універсальні.

В наш час рівень розвитку обчислювальної техніки дозволяє поєднувати різні способи взаємодії людина–комп'ютер. Отже, потрібно знайти методи, що дозволять створити архітектуру та спроектувати інтерфейс користувача програмної системи, яка буде мати графічний інтерфейс користувача та мову опису задачі, максимально наближену до предметної області (у нашому випадку задачі чисельного аналізу математичних задач).

В процесі вирішення поставленої задачі потрібно враховувати особливості предметної області програмної системи (специфіку опису та аналізу задачі, способи представлення результатів тощо) та загальні вимоги до сучасних ПС (уніфікованість, адаптивність та інтерактивність інтерфейсу користувача, функціональну відкритість програмного забезпечення тощо). Процес проектування адаптивного інтерфейсу користувача ПЗ чисельного аналізу математичних задач складається з наступних етапів:

- аналіз предметної області та створення формального опису задачі предметної області;
- створення діаграм поведінки та варіантів використання ПС;
- проектування архітектури ІК програмної системи та створення компонентної моделі ІК;
- створення методики автоматизованого формування ІК за формалізованим описом предметної області.

Далі детально розглянута методика формування адаптивного інтерфейсу користувача, яка була використана при створенні програмного комплексу “DSR Open Lab 1.0” [9-10].

Проектування адаптивного інтерфейсу користувача програмної системи

Проектування інтерфейсу користувача вимагає аналізу вимог, що визначені предметною областю та загальних вимог до програмної системи. Формально задачу чисельного аналізу математичних задач можна представити як множину видів математичних задач $F = \{F_1, F_2, \ldots, F_n\}$, де кожен елемент зв’язаний з певною, індивідуальною для кожного елементу F_i, підмножиною видів аналізу $A = \{A_1, A_2, \ldots, A_m\}$. Будь-який з елементів A_j може бути представлено у вигляді підмножини чисельних методів $M = \{M_1, M_2, \ldots, M_p\}$, за допомогою яких проводиться чисельний аналіз. У загальному випадку для кожного чисельного методу існує унікальна послідовність маніпуляцій для того, щоб описати задачу, провести її аналіз, отримати та переглянути результати досліджень. Проведення таких маніпуляцій вимагає введення в інтерфейс користувача програмної системи відповідної множини візуальних засобів або використання внутрішньої мови. Таким чином, програмна система, що орієнтована на вирішення великої кількості видів математичних задач, повинна мати величезну кількість візуальних засобів або надскладну внутрішню мову. Інтерфейси, що мають велику кількість візуальних засобів (такі, як Mathcad) чи занадто ускладнену мову (Matlab та MATEMATICA), є незручними у використанні. Для вирішення даної проблеми автором була використана ідея адаптивного розподілу управління процесом вирішення математичної задачі між лінгвістичним та візуальним інтерфейсами програмної системи. В подальшому розвитку даної ідеї було запропоновано розділити весь процес аналізу математичної задачі на декілька етапів, кожен з яких відповідає за вирішення окремої частини задачі. Кількість етапів (тобто ступінь деталізації процесу опису, аналізу, представлення результатів та надання допомоги) залежить від виду задачі, що вирішується.

Отже, для кожного елементу з множини чисельних методів M інтерфейс користувача може бути представлений у вигляді множини інтерфейсів I, де елементи I – це інтерфейси користувача на кожному етапі аналізу задачі. Таким чином, інтерфейс користувача для i-ї математичної задачі, j-го виду аналізу, k-го методу аналізу може бути представлений у вигляді $I_{jik} = \{F_i, A_j, M_k, I, R\}$, де R – множина попередніх маніпуляцій користувача. Множину інтерфейсів I визначаємо як $\{V, L, Q\}$, де V – підмножина візуальних інтерфейсів, L – підмножина мовних інтерфейсів, а Q – непуста скінчена множина станів ІК. Всі представлені в програмній системі види аналізу об’єднані в підмножини $G = \{G_1, G_2, \ldots, G_p\}$, яким відповідають однакові підмножини візуальних інтерфейсів.

Процес адаптації ІК до математичної задачі, яку вирішує користувач, використовує також ряд множин, що вміщують інформацію довідникового характеру щодо ієрархії задач, видів та методів аналізу, візуальних інтерфейсів і підсистем формування ІК:

1. Ієрархія математичних задач і відповідних елементів їх візуального представлення складає множину кортежів $D_F = \{<IDM, P, O, N, M, I>\}$, де IDM – унікальний номер, що ідентифікує математичну задачу ($F_{IDM} \in F$); P – номер, що вказує на ідентифікатор батьківської задачі більш високого рівня ієрархії ($F_P \in F$); O – порядковий номер задачі серед задач такого ж рівня ієрархії; N – назва задачі; I – графічне представлення задачі.

2. Ієрархія видів аналізу для математичних задач і відповідних елементів їх візуального представлення складає множину кортежів $D_A = \{<IDA, IDM, P, O, N, I, G, S>\}$, де IDA – унікальний номер, що ідентифікує вид аналізу для математичної задачі ($A_{IDA} \in A$); IDM – номер, що ідентифікує задачу ($F_{IDM} \in F$); P – номер, що вказує на ідентифікатор батьківського виду аналізу більш високого рівня ієрархії ($A_P \in A$); O – порядковий номер виду аналізу серед задач такого ж рівня ієрархії; N – назва виду аналізу; I – графічне представлення виду аналізу; G – номер підмножини видів аналізу, яким відповідають однакові підмножини візуальних інтерфейсів; S – підмножина станів ІК ($S \subset Q$).

3. Ієрархія математичних методів проведення аналізу для математичних задач і відповідних елементів їх візуального представлення складає множину кортежів $D_M = \{<MM, IDA, MS, N, I>\}$, де MM – унікальний номер математичного методу ($M_{MM} \in M$); IDA – номер, що ідентифікує вид аналізу для математичної задачі ($A_{IDA} \in A$); MS – номер підмножини математичних методів, які використовуються в даному методі; N – назва математичного методу; I – графічне представлення математичного методу.

4. Ієрархія взаємного використання математичних методів складає множину кортежів $D_{MS} = \{<MM, MS>\}$, де MM – номер математичного методу ($M_{MM} \in M$); MS – номер підмножини математичних методів, до складу якої входить даний метод.

5. Зв’язки станів і множин підсистем формування ІК складають множину кортежів $D_{SK} = \{<SI, K, O, N, I>\}$, де SI – стан ІК ($SI \in Q$); K – номер, що визначає підмножину підсистем формування ІК; O – пріоритет підмножини підсистем формування ІК в даному стані; N – назва підмножини підсистем формування ІК; I – графічне представлення (зображення) підмножини підсистем формування ІК.

6. Візуальні інтерфейси підсистем формування ІК складають множину кортежів $D_V = \{<KI, V>\}$, де KI – унікальний номер підсистеми формування ІК; V – візуальний інтерфейс користувача.

Розглянемо процес адаптації інтерфейсу програмної системи шляхом автоматизованого формування I_{jik} з відповідної множини візуальних інтерфейсів V та множини маніпуляцій R; виділимо основні етапи розв'язку задачі, елемента множини формування візуальних інтерфейсів V та критерії адаптації інтерфейсу. Визначимо, що множина V складається з двох підмножин V_{fx} та V_{ad}, де V_{fx} — підмножина незмінного ІК, а V_{ad} — підмножина ІК, яка адаптує інтерфейс програмної системи до виду задачі, що вирішується.

Тут і далі використовуємо йота-оператор $(\iota x)P(x)$ для позначення вибору точно одного об'єкта, який задовольняє заданому предикату $P(x)$. Якщо не існує жодного такого об'єкта або таких об'єктів більше, ніж один, вираз не визначено.

1) *Вибір виду математичної задачі*. Підмножина візуального інтерфейсу $Vm \subset V_{fx}$ представляє множину задач F у вигляді позначеного впорядкованого дерева (так зване дерево математичних задач), вершинами якого є елементи множини кортежів D_F. При цьому:

– значення кореня дерева задається як

$$(\iota x)[x \in D_F \wedge x = <IDM, P, O, N, M, I> \wedge P = 0];$$

– якщо піддерево d_l, над яким домінує y, прямий нащадок кореня дерева, має корінь x, то

$$(\forall x \forall y)[\, y \in D_F \wedge y = <IDM_1, P_1, O_1, N_1, M_1, I_1> \wedge x \in D_F \wedge$$
$$x = <IDM_2, P_2, O_2, N_2, M_2, I_2> \wedge IDM_1 = P_2];$$

причому d_l складається з єдиної вершини, позначеної x, тільки якщо виконується умова

$$x \in D_F \wedge x = <IDM_1, P_1, O_1, N_1, M_1, I_1> \wedge \neg (\exists z)[\, z \in D_A \wedge$$
$$z = <IDM_2, P_2, O_2, N_2, M_2, I_2> \wedge IDM_1 = P_2].$$

Підмножина маніпуляцій користувача $Rm \subset R$ містить засоби навігації по дереву математичних задач, вибору моделі або відмови від її вибору. Критерієм допустимості вибору є відсутність нащадків для вибраного елемента дерева. За видом вибраної користувачем математичної задачі (елементу множини кортежів $d_k \in D_F$) з множини візуальних інтерфейсів V_{fx} формується підмножина візуальних інтерфейсів $Va \subset V_{fx}$ програмної системи та допустимих маніпуляцій користувача $Ra \subset R$ для стану ІК вибору виду аналізу. Таким чином, перший крок адаптивної організації ІК можна описати функціональним відношенням ChFA : $(D_F, Vm, Rm) \rightarrow (F_i, Va, Ra)$, де $i = (\iota IDM)[d_k = <IDM, P, O, N, M, I>]$.

2) *Вибір виду аналізу задачі*. Підмножина візуального інтерфейсу $Va \in V_{fx}$ представляє підмножину видів аналізу $A_j \in A$ у вигляді позначеного впорядкованого дерева, вершинами якого є елементи множини кортежів D_A. При цьому:

– значення кореня дерева вводиться як фіктивний вид аналізу

$$(\iota x)[x \in D_A \wedge x = <IDA, IDM, P, O, N, I, G, S> \wedge IDA = 0 \wedge i = IDM],$$

де F_i — математична задача, вибрана на попередньому кроці адаптації ІК;

– якщо піддерево d_l, над яким домінує y, прямий нащадок кореня дерева, має корінь x, то

$$(\forall x \forall y)[\, y \in D_A \wedge y = <IDA_1, IDM_1, P_1, O_1, N_1, I_1, G_1, S_1> \wedge x \in D_A \wedge$$
$$x = <IDA_2, IDM_2, P_2, O_2, N_2, I_2, G_2, S_2> \wedge IDA_1 = P_2 \wedge i = IDM_1 \wedge i = IDM_2];$$

причому d_l складається з єдиної вершини, позначеної x, тільки якщо виконується умова

$$x \in D_A \wedge x = <IDA_1, IDM_1, P_1, O_1, N_1, I_1, G_1, S_1> \wedge \neg (\exists z)[\, z \in D_A \wedge$$
$$z = <IDA_2, IDM_2, P_2, O_2, N_2, I_2, G_2, S_2> \wedge IDA_1 = P_2 \wedge i = IDM_1 \wedge i = IDM_2].$$

З множини $A_j \in A$ користувачеві надається можливість провести один або декілька видів аналізу, номери яких визначаються за виглядом вибраних листків дерева видів аналізу (множина кортежів $D_{FU} \subset D_A$), для яких повинна виконуватись умова рівності відповідних множин станів ІК:

$$(\forall x \forall y)[x \in D_{FU} \wedge y \in D_{FU} \Rightarrow x = <IDA_1, IDM_1, P_1, O_1, N_1, I_1, G_1, S_1> \wedge$$
$$y = <IDA_2, IDM_2, P_2, O_2, N_2, I_2, G_2, S_2> \wedge$$
$$i = IDM_1 \wedge i = IDM_2 \wedge G_1 = G_2 \wedge S_1 = S_2].$$

Підмножина інтерфейсів користувача $Vs \subset V_{ad}$, що формується для множини D_{FU}, використовується для подальшого візуального представлення та аналізу задачі. Критерієм вибору інтерфейсів з множини V_{ad} до відповідної підмножини Vs є виконання умови:

$$(\forall V)[V \in Vs \Rightarrow (\exists x \exists y \exists z)(x \in D_{FU} \wedge x = <IDA, IDM, P, O, N, I, G, S> \wedge$$
$$y \in D_{SK} \wedge y = <SI, K, O_1, N_1, I_1> \wedge z \in D_V \wedge z = <KI, V> \wedge SI \in S \wedge KI \in K)].$$

Таким чином ми адаптуємо множину візуальних інтерфейсів V_{ad} та допустимих маніпуляцій користувача R до виду вибраної користувачем підмножини видів аналізу. Отже, другий крок адаптивної організації ІК можна описати функціональним відношенням ChAGr : $(F_i, Va, Ra) \rightarrow (D_{FU}, Vs, Rg)$, де $Rg \subset R$ – підмножина подальших допустимих дій користувача, яка визначається за виглядом множини

візуальних інтерфейсів *Vs*. Крім того, вибір елементів множини *Vs* визначає також лінгвістичні інтерфейси, асоційовані з відповідними візуальними інтерфейсами.

Метою наступних трьох кроків адаптивної організації ІК є виділення з множини *Vs* підмножин візуальних інтерфейсів: *а*) опису умови математичної задачі $VDs \subset Vs$; *б*) аналізу (розв'язання) задачі $VAs \subset Vs$ та *в*) представлення результатів аналізу задачі $VRs \subset Vs$.

3) *Опис задачі*. Уточнення підмножини візуальних інтерфейсів опису умови математичної задачі *VDs* відбувається за допомогою сформованої на попередньому кроці множини D_{FU}. При цьому використовується одномісний предикат is-description, областю значень змінної якого є Q — множина станів ІК. Значення is-description(x) дорівнює "істина", якщо стан x інтерфейсу користувача використовується для етапу опису умови математичної задачі, та "неправда" – в іншому випадку.

Таким чином, множина $VDs \subset Vs$ формується з умови:

$$(\forall V)[V \in VDs \Rightarrow (\exists x \exists y \exists z)(x \in D_{FU} \wedge x = \langle IDA, IDM, P, O, N, I, G, S\rangle \wedge y \in D_{SK} \wedge y = \langle SI, K, O_1, N_1, I_1\rangle \wedge z \in D_V \wedge z = \langle KI, V\rangle \wedge SI \in S \wedge KI \in K \wedge \text{is-description}(SI))].$$

Даний крок адаптації ІК описується функціональним відношенням, що має вигляд ChDT : (D_{FU}, *Rg*) → (*VDs*, *Rg1*), де $Rg1 \subset Rg$ – підмножина подальших допустимих дій користувача, яка визначається за виглядом множини візуальних інтерфейсів *VDs*.

4) *Аналіз задачі*. Уточнення підмножини візуальних інтерфейсів аналізу (розв'язання) задачі $VAs \subset Vs$ відбувається за допомогою сформованої на другому кроці множини D_{FU}. При цьому використовується одномісний предикат is-solve, областю значень змінної якого є Q — множина станів ІК. Значення is-solve(x) дорівнює "істина", якщо стан x інтерфейсу користувача використовується для етапу аналізу задачі, та "неправда" – в іншому випадку.

Таким чином, множина $VAs \subset Vs$ формується з умови:

$$(\forall V)[V \in VAs \Rightarrow (\exists x \exists y \exists z)(x \in D_{FU} \wedge x = \langle IDA, IDM, P, O, N, I, G, S\rangle \wedge y \in D_{SK} \wedge y = \langle SI, K, O_1, N_1, I_1\rangle \wedge z \in D_V \wedge z = \langle KI, V\rangle \wedge SI \in S \wedge KI \in K \wedge \text{is-solve}(SI))].$$

Даний крок адаптації ІК описується функціональним відношенням, що має вигляд ChM : (D_{FU}, *Rg*) → (*MA*, *VAs*, *Rg2*), де $Rg2 \subset Rg$ – підмножина подальших допустимих дій користувача, яка визначається за виглядом множини візуальних інтерфейсів *VAs*, *MA* — множина математичних методів, які використовуються у видах аналізу з множини D_{FU} ($MA \subset M$). Множина *MA* формується за умовою:

$$(\forall MM)[M_{MM} \in MA \Rightarrow (\exists x \exists y)(x \in D_{FU} \wedge x = \langle IDA, IDM, P, O, N, I, G, S\rangle \wedge y \in D_M \wedge y = \langle MM, IDA_1, MS, N_1, I_1\rangle \wedge IDA_1 \in IDA)].$$

Множина математичних методів *MA* представляється в ІК у вигляді позначеного впорядкованого дерева, вершинами якого є елементи множини кортежів D_M. При цьому:

– значення кореня дерева вводиться як фіктивний метод

$$(\iota x)[x \in D_M \wedge x = \langle MM, IDA, MS, N, I\rangle \wedge (\exists y)(y \in D_{FU} \wedge y = \langle IDA_1, IDM, P, O, N, I, G, S\rangle \wedge IDA = IDA_1) \wedge MS = 0];$$

– якщо піддерево d_l, над яким домінує y, прямий нащадок кореня дерева, має корінь x, то

$$(\forall x \forall y)[\, y \in D_M \wedge y = \langle MM_1, IDA_1, MS_1, N_1, I_1\rangle \wedge x \in D_M \wedge x = \langle MM_2, IDA_2, MS_2, N_2, I_2\rangle \wedge (\exists z)(z \in D_{MS} \wedge z = \langle MM_2, MS_1\rangle) \wedge (\exists a)(a \in D_{FU} \wedge a = \langle IDA_3, IDM_3, P_3, O_3, N_3, I_3, G_3, S_3\rangle \wedge IDA_3 = IDA_1) \wedge (\exists b)(b \in D_{FU} \wedge b = \langle IDA_4, IDM_4, P_4, O_4, N_4, I_4, G_4, S_4\rangle \wedge IDA_4 = IDA_2)];$$

причому d_l складається з єдиної вершини, позначеної x, тільки якщо виконується умова

$$x \in D_M \wedge x = \langle MM_1, IDA_1, MS_1, N_1, I_1\rangle \wedge MS_1 = 1 \wedge (\exists y)[y \in D_{FU} \wedge y = \langle IDA_2, IDM_2, P_2, O_2, N_2, I_2, G_2, S_2\rangle \wedge IDA_1 = IDA_2].$$

5) *Представлення результатів*. Уточнення підмножини візуальних інтерфейсів представлення результатів аналізу задачі *VRs* відбувається за допомогою сформованої на другому кроці множини D_{FU}. При цьому використовується одномісний предикат is-report, областю значень змінної якого є Q — множина станів ІК. Значення is-report(x) дорівнює "істина", якщо стан x інтерфейсу користувача використовується для етапу представлення результатів аналізу задачі.

Таким чином, множина $VRs \subset Vs$ формується з умови:

$$(\forall V)[V \in VRs \Rightarrow (\exists x \exists y \exists z)(x \in D_{FU} \wedge x = \langle IDA, IDM, P, O, N, I, G, S\rangle \wedge y \in D_{SK} \wedge y = \langle SI, K, O_1, N_1, I_1\rangle \wedge z \in D_V \wedge z = \langle KI, V\rangle \wedge SI \in S \wedge KI \in K \wedge \text{is-report}(SI))].$$

Даний крок адаптації ІК описується функціональним відношенням, що має вигляд ChREP : (D_{FU}, *Rg*) → (*VRs*, *Rg3*), де $Rg3 \subset Rg$ – підмножина подальших допустимих дій користувача, яка визначається за виглядом множини візуальних інтерфейсів *VRs*.

За результатами проведеної формалізації адаптивної організації ІК до процесу аналізу математичних задач було створено діаграму станів програмного засобу, яка є однією з діаграм необхідних для створення концептуальної моделі інтерфейсу користувача програмної системи (рис. 1 та рис. 2). На

діаграмі зображено множину станів інтерфейсу програмного засобу чисельного аналізу математичних задач в процесі діалогу з користувачем (відображені підмножини можливих маніпуляцій користувача).

Згідно з наведеною вище процедурою адаптації ІК, множина підсистем формування ІК користувача *Ks*, яка використовується при вирішенні математичної задачі для видів аналізу з множини D_{FU}, формується за умовою:

$$(\forall KI)[KI \in Ks \Rightarrow (\exists x \exists y \exists z)(x \in D_{FU} \wedge x = <IDA, IDM, P, O, N, I, G, S> \wedge$$
$$y \in D_{SK} \wedge y = <SI, K, O_1, N_1, I_1> \wedge z \in D_V \wedge z = <KI, V> \wedge SI \in S \wedge KI \in K)].$$

Аналіз предметної області чисельного аналізу математичних задач, що одні й ті ж підсистеми формування ІК можуть бути використані одразу для задач декількох видів, тобто потужність перетину множин *Ks*, побудованих для різного виду множин D_{FU}, у загальному випадку ненульова (зазвичай, це досить значна величина для вирішення багатьох математичних задач). Це підтверджує припущення про доцільність використання для візуального відображення процесу аналізу множини програмних підсистем – засобів відображення, які далі будемо називати програмними компонентами. Можна сказати, що виділення множини програмних підсистем візуального відображення задачі, з яких формується ІК програмного засобу в цілому, надає можливість створення архітектури інтерфейсу користувача, яка передбачає розділення на підмножини інтерфейсів для кожного стану ПС.

Для більшої зручності використання програмної системи в дослідженнях і навчальному процесі та з методичних міркувань структурування завдання на аналіз, пропонується використовувати три підмножини станів програмної системи:

- “Обчислення” – режим аналізу задачі;
- “Навчання” – режим аналізу та вивчення задачі або програмної системи;
- “Дослідження” – режим, що дозволяє проводити дослідження процесу аналізу задачі, використовуючи різноманітні види чисельних методів та алгоритмів, отримувати додаткові види вихідних результатів.

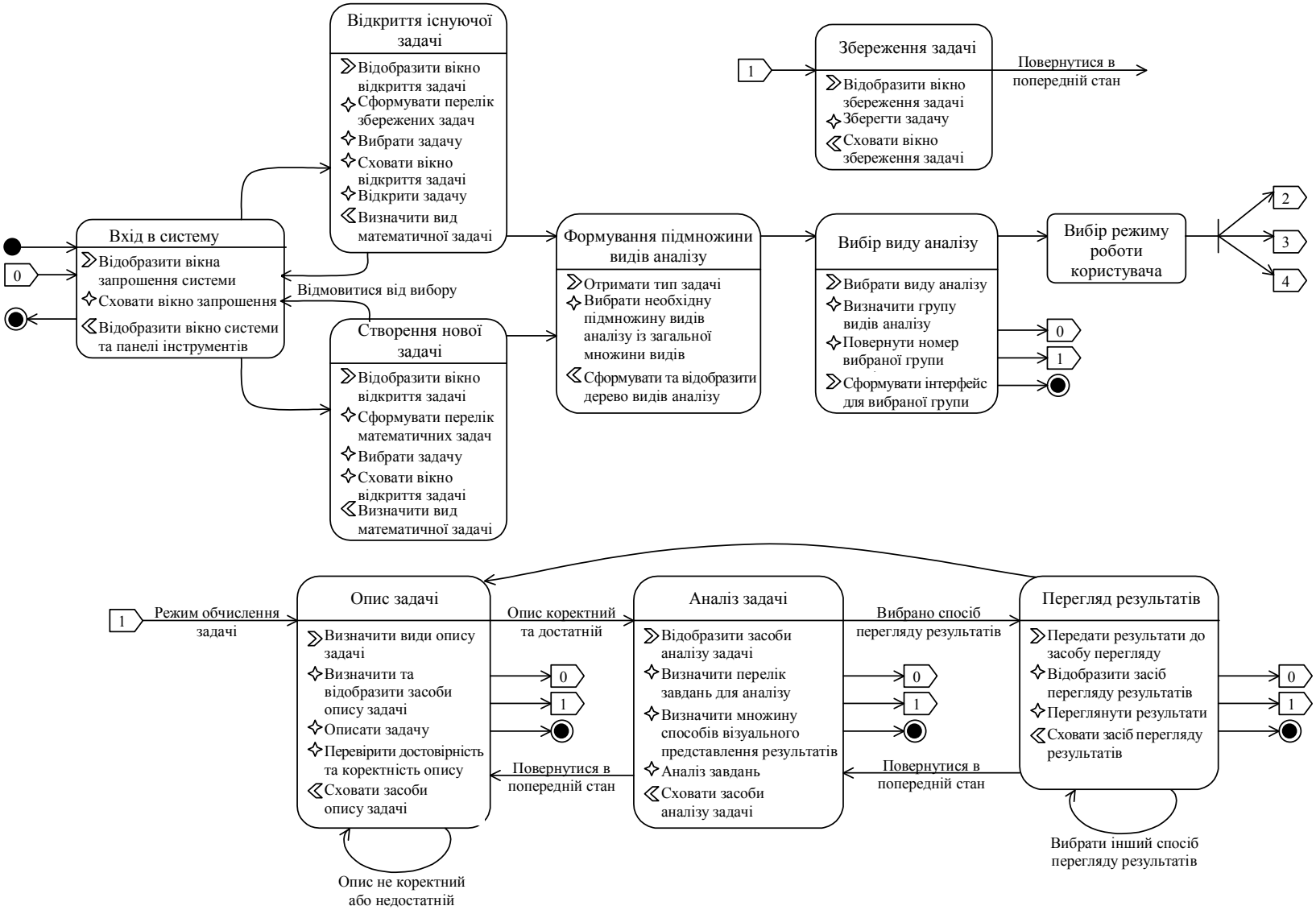

Рис. 1. Діаграма станів інтерфейсу користувача, частина 1

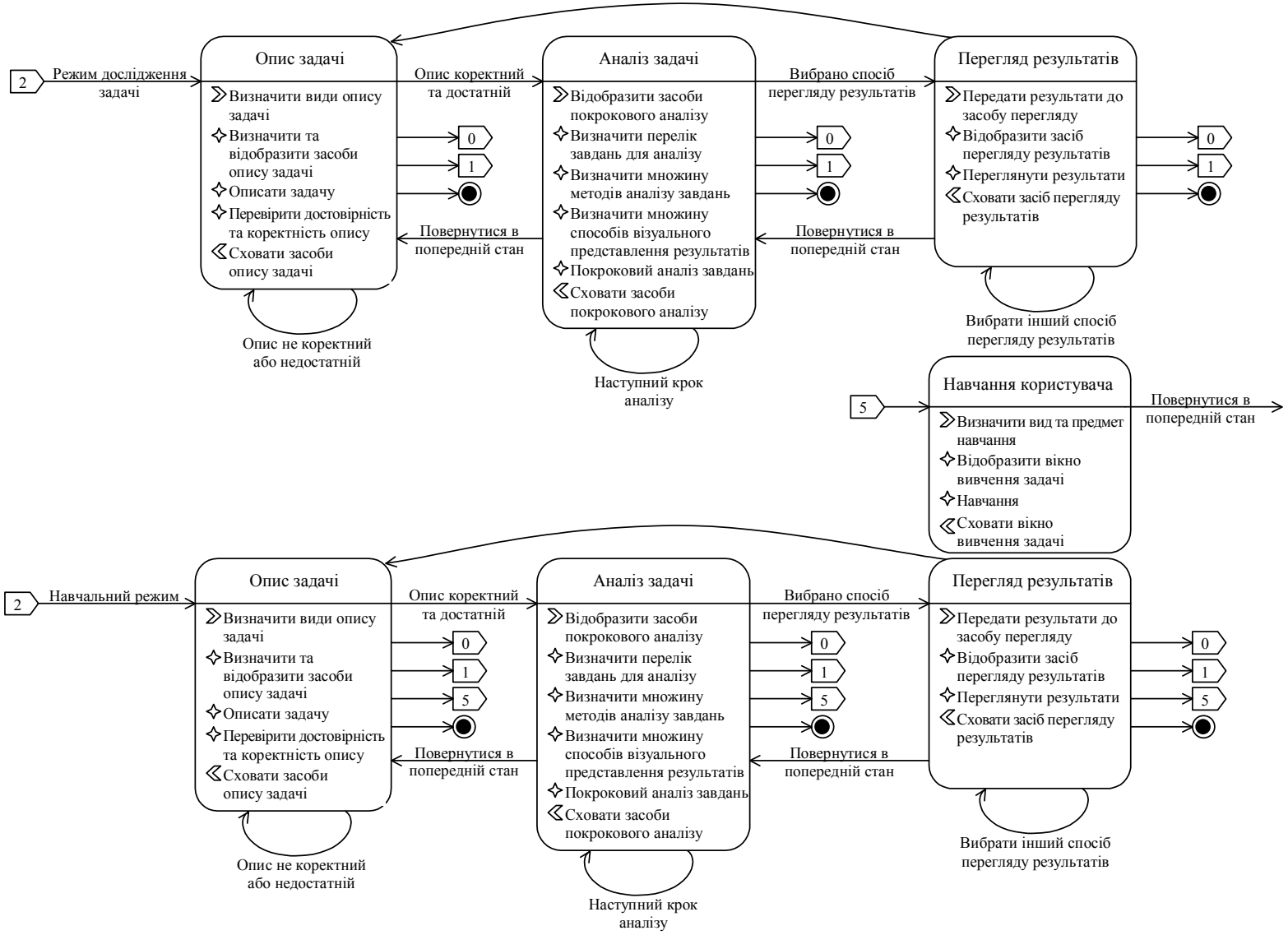

Рис. 2. Діаграма станів інтерфейсу користувача, частина 2

Серед загальних вимог до інтерфейсу користувача найважливішими є ті, що визначені орієнтованістю ПЗ на непідготовленого користувача, а саме: зрозумілість, зручність, стандартність (інтерфейс не повинен викликати труднощів навіть у початківців). Також ІК програмної системи повинен бути достатньо потужним для вирішення складних задач, мати можливості опису даних мовою, що є максимально приближеною до предметної області, представляти уніфіковані засоби для гнучкого та адаптивного управління процесом чисельного аналізу. Умова уніфікованості інтерфейсу користувача в програмній системі забезпечується використанням одного алгоритму формування інтерфейсу для різних математичних задач в рамках однієї програмної системи та скінченою кількістю програмних компонент. Адаптивність в архітектурі інтерфейсу користувача підтримується використанням програмних компонент: вид компоненти, що використовується, залежить від множин параметрів на вході моделі, а саме: від типу математичної задачі, виду чисельного аналізу, групи, до якої відноситься вид аналізу, номера етапу, методу вирішення задачі, вибраного виду представлення результатів обчислення тощо.

Створена діаграма станів ІК надасть можливість побудувати архітектуру інтерфейсу користувача ПС, на основі математичного апарату теорії агрегативних систем [11], компонентної моделі об'єкта (COM) [12, 13] та клієнт-серверної моделі взаємодії між програмними системами [14]. Відзначимо, що подальша деталізація запропонованої архітектури призводить до компонентного характеру програмних підсистем, а саме: кожен стан ІК програмної системи представляється за допомогою спеціальних компонент (програмних підсистем, що використовують клієнт-серверну модель).

Висновки

В роботі було запропоновано метод проектування адаптивного, уніфікованого та функціонально відкритого інтерфейсу програмної системи чисельного аналізу математичних задач. Створення даного інтерфейсу користувача є результатом розробки формального опису предметної області чисельного аналізу та виділення спільних етапів аналізу різних математичних задач (створення діаграми станів інтерфейсу користувача), проектування компонентної моделі інтерфейсу користувача та використання програмних компонент. Необхідність розробки процедури формування адаптивного ІК обумовлена тим, що створення окремого інтерфейсу для кожного методу дослідження математичної задачі було б незручним не тільки для розробників, але й для користувачів та взагалі не відповідало б сучасним вимогам програмної системи. Запропонована архітектура інтерфейсу користувача дозволяє модифікувати ІК програмного засобу без додаткової модифікації програмного коду, що надає можливість розширення предметної області програмної системи.

ЛІТЕРАТУРА:

1. *Додонов А.Г.*, *Клименко В.Г.*, *Полищук А.В.*, *Ярмоленко А.М.* Концепция построения диалоговой системы моделирования и ее реализация // Электронное моделирование. – 1986. – № 4. – С. 28–32.
2. *Иткин В.Д.* Смешанные вычисления и адаптация программных средств // Кибернетика. – 1990. – № 1. – С. 28–30.
3. *Молчанов И.Н.* Об одной форме организации прикладного программного обеспечения // Кибернетика. – 1990. – № 1. – С. 1–6.
4. *Вайрадян А.С.*, *Могилевич А.К.*, *Петухов М.Н.*, *Угринович Н.В.* Особенности реализации диалогового математического обеспечения в системах реального времени и вопросы повышения его надежности // Электронное моделирование, № 3. – Киев, 1981. – С. 24–28.
5. *Глушков В.М.*, *Гринченко Т.Л.*, *Дородницина А.А. и др.* АНАЛИТИК (Аналитический язык для описания вычислительных процессов с использованием аналитических преобразований) // Кибирнетика, № 3. – 1971. – С. 102–134.
6. *Глушков В.М.*, *Молчанов И.Н.*, *Брусникин Б.Н. и др.* Программное обеспечение ЭВМ МИР-1 и МИР-2, – К.: Наукова думка, 1976. – Т. 1. 280 с.
7. *Глушков В.М.*, *Молчанов И.Н.*, *Брусникин Б.Н. и др.* Программное обеспечение ЭВМ МИР-1 и МИР-2, – К.: Наукова думка, 1976. – Т. 2. 371 с.
8. *Глушков В.М.*, *Молчанов И.Н.*, *Брусникин Б.Н. и др.* Программное обеспечение ЭВМ МИР-1 и МИР-2, – К.: Наукова думка, 1976. – Т. 3. 223 с.
9. *Колодницький М.М.*, *Левицький В.Г.* Типологія архітектури інтерфейсу користувача прикладної програмної системи "DSR Open Lab 1.0". Частина II. Лінгвістичне забезпечення // Проблемы программирования. – 1999. – Вып. 3–4.
10. *Колодницький М.М.* Тривимірна компонентна архітектура прикладної програмної системи «DSR Open Lab 1.0» як втілення концепцій реінженерії // Проблемы программирования. –1998. – Вып. 4. – С. 37–45.
11. *Бусленко Н.П.*, *Калашников В.В.*, *Коваленко И.Н.* Лекции по теории сложных систем. – М.: Советское радио, 1973. – 440 с.
12. *Rogerson D.* Inside COM: Microsoft's Component Object Model, Microsoft Press, 1996. – 416 p.

13. *Redmond F*. DCOM: Microsoft Distributed Component Object Model, IDG Books Worldwide, 1997. – 384 p.
14. *Richter J*., *Clark J*. Programming Server-Side Applications for Microsoft® Windows® 2000, Microsoft Press, 2000. – 736 p.

КОВАЛЬЧУК Андрій Михайлович – асистент кафедри АІКТ Житомирського інженерно-технологічного інституту.

Наукові інтереси:
– комп'ютерні інформаційні технології;
– використання обчислювальної техніки в навчальному процесі;
– методи організації графічних інтерфейсів користувача прикладних програмних систем;
– комп'ютерна графіка.

ЛЕВИЦЬКИЙ В'ячеслав Георгійович – кандидат технічних наук**,** старший викладач кафедри ПЗОТ Житомирського інженерно-технологічного інституту.

Наукові інтереси:
– комп'ютерні інформаційні технології;
– моделювання та розв'язок задач за допомогою обчислювальної техніки;
– використання обчислювальної техніки в навчальному процесі;
– побудова компіляторів.

Подано 14.02.2002

Ковальчук А.М., Левицький В.Г. Розробка адаптивного інтерфейсу користувача програмної системи чисельного аналізу математичних задач
Ковальчук А.М., Левицкий В.Г. Разработка адаптивного интерфейса пользователя программной системы численного анализа математических задач
Kovalchuk A.M., Levitsky V.G. A development of the adaptive user interface for software of numerical analysis of mathematical tasks

УДК 681.3.06

Разработка адаптивного интерфейса пользователя программной системы численного анализа математических задач / Ковальчук А.М., Левицкий В.Г.

Рассмотрена проблема разработки адаптивного интерфейса пользователя программной системы численного анализа математических задач. Предложен метод автоматизированного формирования адаптивного интерфейса пользователя программной системы на основе формального описания предметной области численного анализа математических задач. Обоснованна необходимость использования программных компонент и компонентной архитектуры для создания программного средства.

УДК 681.3.06

The development of the adaptive user interface for software of numerical analysis of mathematical tasks / A.M. Kovalchuk, V.G. Levitsky

The development of the adaptive user interface for software of numerical analysis of mathematical tasks is briefly characterized. The method of automated formation of adaptive user interface is presented. The formal description of mathematical domain has been utilized and presented as a main feature of proposed method. The necessity of the software component architecture and its utilization and has been described.