УДК 004.5(045)

І.В. Гученко, асп.

МАТЕМАТИЧНА МОДЕЛЬ ОЦІНКИ ПРАКТИЧНОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Національний авіаційний університет
E-mail: inna.zaporozhets@livenau.net

Досліджено практичність (зручність застосування) програмного забезпечення. Побудовано математичну модель оцінки заданого рівня практичності. Використано методологію структурного аналізу, методи багатокритеріальної оптимізації та теорії прийняття рішень, метод згортки, загальнонаукові методи аналізу та аналогій. Запропоновано модель оцінки зручності застосування програмного забезпечення, що дозволяє не лише оцінювати поточний рівень практичності під час кожної ітерації гнучкої розробки, але й керувати зручністю використання створюваних програмних продуктів. Результати дослідження можна використати для побудови автоматизованих систем підтримки керування зручністю використання програмного забезпечення.

Ключові слова: забезпечення оптимальної практичності, зручність використання, модель оцінки, практичність, програмне забезпечення.

Постановка проблеми

В умовах постійно зростаючої конкуренції між розробниками все більше виробників програмного забезпечення вважають зручність використання їхніх продуктів стратегічною ціллю [1].

Зручність використання розглядається у багатовимірному просторі, який вивчається в різних аспектах.

У літературі термін «usability» має різний зміст [2].

Єдиний підхід до зручності використання є спірним, оскільки на неї впливають різні фактори.

Найчастіше зручність розглядається як фактор якості з певними технічними аспектами [2]. Область людино-машинної взаємодії забезпечує теоретичні основи та пропонує методи для створення якісного інтерфейсу користувача.

Зручність використання можна інтерпретувати як корисність та легкість у застосуванні системи [2]. Проектування інтерфейсу та функціональності, дані та метадані, комп'ютерні системи та мережі являють інтерес для проектувальників програмного забезпечення, розробників і користувачів для отримання зручної системи. Однак на практиці у процесі розроблення некоординовані дії в покращенні зручності мають малу ефективність.

Зважаючи на поширення гнучких (agile) методів розробки в інженерії програмного забезпечення, багато експертів із програмної інженерії та проектування інтерфейсу користувача розвинули так звану гнучку людиноорієнтовану інженерію програмного забезпечення – agile human-centered software engineering [3]. Вона наголошує на орієнтованому на користування підході до розроблення програмного забезпечення (usage-centered design), підкреслюючи недоліки надмірного фокусування на досвіді та побажаннях користувача (user-centered design) [3].

В обох випадках споживачі є джерелом інформації та підтвердження зручності програмного забезпечення.

Agile методам і досі бракує інтеграції зручності використання і agile процесів.

Розроблення невеликими кроками дає коректні результати і майже не фокусується на проблемі зручності використання.

Адже зміни в архітектурі програмного забезпечення зазвичай не впливають на те, що бачить та з чим взаємодіє користувач, зі зручністю використання – навпаки. Тому в agile розробці бракує орієнтації на користувача, що суттєво впливає на якість кінцевого продукту [3].

Зазначене підкреслює важливість обізнаності інженерів-практиків програмного забезпечення щодо різних методів оцінки зручності використання та застосуванні їх відповідно до особливостей проекту.

Малодослідженими залишаються задачі оцінки поточного та забезпечення бажаного рівня зручності використання програмної системи на основі оцінок користувачів.

Аналіз досліджень і публікацій

Існуючі дослідження з інтеграції ітераційних методів та орієнтованого на користувача підходу до розробки програмного забезпечення виділяють такі найпоширеніші методи досягнення бажаної зручності використання [4]:

– низькорівневе прототипування;
– концептуальний дизайн;
– спостереження за користувачами;
– оцінка практичності експертами;
– дослідження в реальних умовах;
– швидке ітеративне тестування;
– анкетування;
– перевірка зручності використання в лабораторних умовах;
– аналіз потреб;
– автоматизована оцінка практичності.

Оцінка зручності використання може бути досить дорогою, зважаючи на час та людські ресурси, тому автоматизація знижає витрати.

Автоматизована оцінка практичності фокусується на розробленні методів, що забезпечують прискорення обрахунків, ширшу аудиторію для тестування зручності використання, та створенні засобів, які мають вбудовані аналітичні можливості.

Одним із відомих чотирьох підходів до автоматизації, яких є так звана автоматизована критика (Automatic Critic), що містить методи, які не тільки виявляють недоліки, але й пропонують покращення [5].

Залежно від джерела даних, яким можуть бути користувачі, експерти або моделі, наприклад, GOMS, EPIC, методи оцінки зручності використання поділяються на три групи [6]:

– оцінка, зосереджена на користувачі;
– експертнка оцінка;
– оцінка на основі моделей.

В оцінці за допомогою користувачів зазвичай використовують метрики ефективності, економічності та задоволеності [7].

Аналіз джерел показав, що переважна більшість методів стосується, як правило, оцінки Web-інтерфейсів та графічних інтерфейсів користувача [8].

Проведені дослідження, свідчать, що поняття зручності використання програмного забезпечення є більш багатогранним.

Розроблений метод автоматизованої оцінки зручності використання програмної системи при agile-розробці для врахування та прогнозування вимог користувачів дозволяє [9]:

– оцінювати поточний рівень зручності застосування програмних систем;
– керувати якістю процесів їхнього створення на основі математичних оптимізаційних моделей.

Мета роботи – побудова математичної моделі забезпечення та оцінки практичності.

Модель практичності програмного забезпечення

У рамках дослідження розглядається найбільш загальне трактування практичності програмного забезпечення як відповідності певній ієрархічній моделі [10; 11].

Зручність застосування декомпозується на атрибути, кожен з яких – на відповідні показники, які в свою чергу – на метрики.

Можна розглядати поняття «практичність програмного забезпечення» як сукупність складових, поступово деталізуючи її доти, доки не буде досягнуто рівня абстракції, прийнятного для формування кількісних значень $m_{ij1},\ldots, m_{ijk}$ (k – кількість метрик, що відповідають j-му показнику i-го атрибуту).

Формальну модель перетворень побудуємо, спираючись на зведення значень окремих показників атрибутів в інтегральну оцінку Q.

Складові зручності використання об'єднаємо в систему з трьох рівнів. Кожен вищий рівень містить критерії нижчих рівнів. Допускається введення додаткових показників на кожному з рівнів (див. рисунок).

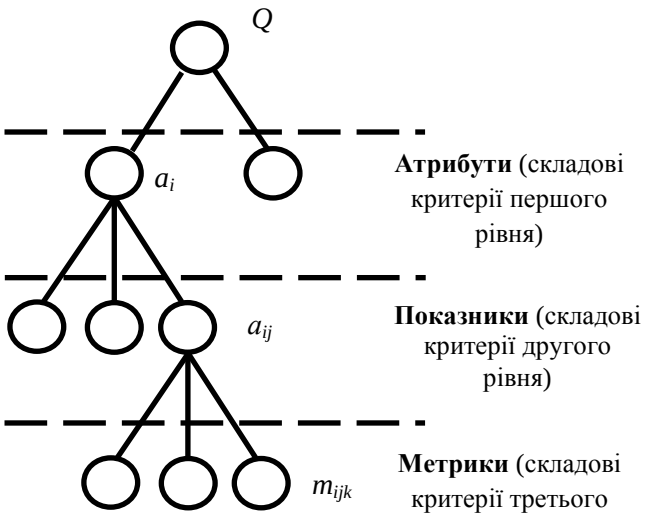

Рис. 1. Графічне зображення ієрархії складових критеріїв

Кожний критерій першого і другого рівня ієрархії характеризується двома числовими параметрами:

– кількісним значенням;

– ваговим коефіцієнтом.

Числові значення показників зручності використання програмного забезпечення розраховуються на основі відповідних метрик, значення яких обчислюється як середнє арифметичне оцінок користувачів.

На першому рівні ієрархії розглядаються такі атрибути:

– економічність;

– ефективність;

– задоволеність;

– продуктивність;

– можливість навчання;

– безпека;

– довіра;

– доступність;

– універсальність;

– корисність;

– якість документації програмного забезпечення.

Для вирішення задачі отримання інтегральної оцінки найчастіше використовують такі методи [12]:

– зваженої суми;

– адитивної згортки;

– мультиплікативної згортки (при багатокритеріальних моделях).

Розглядатимемо багатоцільові моделі, в яких кожна альтернатива оцінюється множиною критеріїв. Найбільш відомими багатокритеріальними моделями є:

– багатовимірні функції корисності;

– моделі багатовимірного шкалювання;

– модель (метод) аналізу ієрархій (Сааті).

Найчастіше використовують адитивні та мультиплікативні функції корисності [12].

Адитивна функція малочутлива до зміни показників з малою вагою, мультиплікативна, навпаки, сильно залежить від зміни показників з малими значеннями оцінок корисності.

У теорії прийняття рішень доведено, що функція корисності має [12]:

– адитивний вигляд, якщо фактори, які входять у модель, лінійно незалежні;

– мультиплікативний вигляд, якщо фактори взаємонезалежні за корисністю, тобто модель містить взаємодію факторів різних порядків.

Оскільки показники практичності не є лінійно залежними, то зупинимось на використанні адитивної функції та адитивної згортки.

Узагальнена формула адитивної функції корисності має вигляд:

$$U(x) = \sum_{i=1}^{n} p_i \hat{U}_i(x),$$

де U – функція корисності варіанта x;

p_i – вага фактора (субфактора) i;

$\hat{U}_i(x)$ – оцінка корисності варіанта x за критерієм i.

Метод згортки полягає в заміні локальних критеріїв одним загальним критерієм Q.

Метод доцільно застосовувати, адже за умови, що показники практичності можна розташувати за спаданням важливості таким чином, що важливість кожної пари сусідніх показників відрізняється несуттєво.

Адитивна згортка для побудованої ієрархії визначається формулою

$$Q(x) = \sum_{i=1}^{n} p_i a_i(x) = \sum_{i=1}^{n} p_i \sum_{j=1}^{m} q_j a_{ij}(x) =$$

$$= \sum_{i=1}^{n} p_i \sum_{j=1}^{m} q_j \sum_{k=1}^{l} r_k m_{ijk}(x),$$

де $Q(x)$ – загальний критерій для оцінок користувачів $x \in X$;

n, m, l – кількість критеріїв на рівнях;

p_i, q_j, r_k – вага складових критеріїв a_i, a_{ij}, m_{ijk};

$\{a_i(x)\}_1^n$, $\{a_{ij}(x)\}_1^m$, $\{m_{ijk}(x)\}_1^l$ – набори складових критеріїв відповідних рівнів ієрархії.

Для важливості (ваги) виконується умова нормування:

$$\sum_{i=1}^{n} p_i = \sum_{j=1}^{m} q_j = \sum_{k=1}^{l} r_k = 1.$$

Для критеріїв зручності використання на всіх рівнях застосовують єдину шкалу оцінки від 0 до 1.

Виходячи з наведеного, можемо оцінити загальний критерій:

$0 \le Q(x) \le 1$.

Достатність рівня зручності використання програмного забезпечення визначається шляхом порівнянням отриманої загальної оцінки та розрахованих значень показників з відповідними аналогами, що приймаються за еталонний зразок.

Аналогами обирають реально існуюче програмне забезпечення того самого функціонального призначення, з такими ж основними параметрами, подібної структури та умовами експлуатації.

Уведемо функцію оцінки трудомісткості забезпечення заданого рівня практичності, що дозволить звести обчислення до розв'язання задачі оптимального керування.

Функція оцінки трудовитрат розглядається в загальному вигляді, оскільки її вигляд залежить від специфіки предметної області програмного забезпечення, розміру проекту, факторів масштабу.

Визнаною методикою такої оцінки є COCOMO II, головною особливістю якої є те, що для розрахунку трудовитрат у людино-місяцях необхідно знати розмір програмного продукту або його компонентів у тисячах рядків вихідного коду (KSLOC) [13]. Цей розмір може бути оцінений експертами з використанням метода PERT. Якщо ж проводиться аналіз програмного забезпечення методом функціональних точок, то його розмір можна розраховувати з використанням власних статистичних даних або галузевої статистики.

Формула оцінки трудомісткості в людино-місяцях має вигляд:

$$PM = A \times SIZE^E \times \prod_{i=1}^{n} EM_i;$$

$$A = 2{,}94;$$

$$E = B + 0{,}01 \times \sum_{j=1}^{5} SF_j;$$

$$B = 0{,}91,$$

де SIZE – розмір продукту в KSLOC;

EM_i – множники трудомісткості;

SF_j – фактори масштабу.

Для попередньої оцінки n = 7, для детальної оцінки n = 17.

Покращення показників зручності використання програмного забезпечення безпосередньо пов'язано зі зміною існуючого коду, написанням додаткових рядків. У загальному вигляді функція оцінки трудомісткості забезпечення заданого рівня практичності залежить від зміни поточних значень показників практичності, а отже, кількості рядків коду:

$$H(\Delta a_{1j}, \ldots, \Delta a_{nj}) = H(KSLOC_{1j}, \ldots, KSLOC_{nj}).$$

Сумарна трудомісткість зміни декількох показників практичності не дорівнює простій сумі трудовитрат на покращення кожного з показників:

$$H \ne \sum_{k=1}^{n} H_{kj}.$$

Проста сума не враховує трудовитрат на інтеграцію всіх змін. Модель COCOMO II описує послідовність визначення трудомісткості в ході багатокомпонентної розробки [8].

Таким чином, на основі проведеного аналізу пропонується формальна постановка задачі забезпечення практичності програмної оцінки системи.

Нехай n – кількість показників зручності застосування, k – значення, яке обмежує кількість показників, за якими одночасно ведеться пошук, Q' – необхідний рівень практичності, $Q(a_{1j},\ldots,a_{nj})$ – функція оцінки практичності, $H(\Delta a_{1j},\ldots,\Delta a_{nj})$ – функція оцінки трудомісткості.

Задача забезпечення необхідного рівня практичності зводиться до оптимізаційної, для якої цільовою функцією виступатиме трудомісткість, а область допустимих рішень та обмеження визначатимуться так:

$$a_{ij},\ (\Delta a_{ij}+a_{ij})\in[0;1];$$

$$\Delta a_{ij}\ge 0,\ \ i=\overline{1,n},\ j=\overline{1,m};$$

$$Q\left(a_{1j}+\delta_{1j}(1-a_{1j}),\ldots,a_{nj}+\delta_{nj}(1-a_{nj})\right)\ge Q'; \quad (1)$$

$$\sum_{j=1}^{m}\delta_{ij}\le k,\ \ \delta_{ij}\in\{0,1\}. \quad (2)$$

Оскільки функція трудомісткості розглядається в загальному вигляді і ніяких припущень щодо її властивостей немає, то використовуватимемо метод багатовимірної оптимізації нульового порядку, який не потребує диференційованості функції, а саме – метод покоординатного спуску (прямого перебору) Гауса–Зейделя [14].

Критерієм зупинки алгоритму є умова різниці між «оптимальними» значеннями однойменних змінних на двох останніх кроках, яка дорівнює наперед заданій похибці:

$$\left|a_{ij}^{onmt}-a_{ij}^{onm(t-1)}\right|=\Delta_t\le\varepsilon_{ij},\ i=\overline{1,n}, \quad (3)$$

де t – номер кроку покоординатного спуску.

Математична постановка оптимізаційної задачі при мінімальній трудомісткості матиме вигляд

$$Q\left(a_{1j}+\delta_{1j}\Delta a_{1j},\ldots,a_{nj}+\delta_{nj}\Delta a_{nj}\right)\ge Q';$$

$$H(\delta_{1j}\Delta a_{1j},\ldots,\delta_{nj}\Delta a_{nj})\to\min;$$

$$\sum_{j=1}^{m}\delta_{ij}\le k,\ \ \delta_{ij}\in\{0,1\}.$$

Указана задача має розв'язки згідно з умовами (1) та (2), остання з яких штучно обмежує кількість показників, за якими одночасно ведеться пошук, для зосередження зусиль на головних напрямах покращення показників практичності програмного забезпечення.

Умова (3) забезпечує використання методу прямого перебору – покоординатного спуску, який зводить багатовимірну задачу до послідовності одновимірних.

У ході оцінки практичності програмного забезпечення виникає декілька альтернатив a зміни різних показників. Найкращій розв'язок визначається виразом

$$a^{*}=\arg\max_{a\in A}Q(a),$$

тобто найкращим вважається розв'язок, якому відповідає максимум загального критерія на множині альтернатив.

Остаточний вибір альтернативи, яка має оптимальне співвідношення отриманого рівня зручності використання та трудомісткості, виконується експертами.

Розв'язком поставленої задачі буде набір показників a_{ij}, де $\delta_{ij}=1$, на яких досягається мінімальне або одне з мінімальних значень функції трудомісткості при забезпеченні рівня практичності, не нижче заданого. Також надається інформація про значення Δa_{ij}, на які необхідно покращити показники.

Згідно з заданими умовами як модель забезпечення практичності виступає n-вимірний простір. Значення показників змінюються від поточного до абсолютного (одиниці) з кроком Δ, який забезпечує ε похибку.

У вершинах задано:

– рівень зручності застосування в заданий момент часу відповідно до ієрархічної моделі оцінки практичності Q;

– значення трудомісткості забезпечення даного рівня у вершині H.

Задача зводиться до відшукання вершини у просторі, для якої виконується:

$$Q \geq Q';\ \ \forall (Q', H') : H' < H \Rightarrow Q < Q'.$$

У разі застосування описаної моделі виникає природне запитання щодо впливу покращення певних показників зручності використання на інші її показники.

Уникнути погіршення останніх можна, встановивши існування залежностей між показниками за допомогою кореляційно-регресійного аналізу.

Визначення ваги показників

Визначення ваги показників практичностіабо цілей при багатоцільовій оптимізації з наступним обчисленням вагових коефіцієнтів – одна з головних задач ефективного використання описаної моделі.

Існуючі методики експертного опитування можно умовно розділити на дві групи, що ґрунтуються порівнянні характеристик за важливістю:

– безпосередньому;

– опосередкованому.

Найбільш розповсюдженими методами експертного опитування є [15]:

– безпосередня оцінка;

– ранжування;

– повне парне порівняння.

Виробленню колективного рішення має передувати дослідження результатів експертного опитування для виявлення узгодженості суджень усіх експертів.

Оцінки експертів часто вимірюють у порядковій шкалі. Типовими є задачі ранжування та класифікації об'єктів.

Як показали численні дослідження, людина більш вірно і з меншими неточностями відповідає на питання якісного, порівняльного характеру, ніж кількісного [15].

Специфічними є підходи до перевірки узгодженості, що використовуються при оцінці об'єктів методом ранжування. Передбачається впорядкування характеристик відповідно до зменшення їх відносної важливості.

При цьому кожній характеристиці приписується ранг – натуральне число, що характеризує її порядковий номер. Якщо неможливо розрізнити за важливістю дві або декілька характеристик, їм привласнюється один і той самий ранг, значення якого є середнім суми місць, поділених цими характеристиками.

Результатом роботи експерта є ранжування, що являє собою послідовність рангів (для експерта j): $x_{1j}, x_{2j}, \ldots, x_{nj}$.

В аналізі експертної бази на узгодженість застосовують найчастіше розрахунок коефіцієнта конкордації Кендела [15]. Це не дає можливості якісного аналізу ситуації, оскільки не вказує на експертів-«дисидентів», що є причиною неузгодженості, а також не дає інформації про об'єкти, що мають найбільші розбіжності при ранжуванні експертами. До того ж у процесі обчислення коефіцієнта конкордації перевіряється нульова гіпотеза, згідно з якою ранжування незалежні та рівномірно розподілені на множині всіх ранжувань. Якщо ця гіпотеза приймається, то не можна говорити про узгодженість думок експертів. У випадку відхилення гіпотези узгодженості також немає, оскільки, наприклад, може бути два (або більше) центри, навколо яких групуються думки.

Отже, точність коефіцієнта конкордації напряму залежить від однорідності об'єктів.

Скупчення в розподілі можуть внести похибки в розрахунки.

У роботі [16] запропоновано використання методів та інструментів Data Mining, зокрема кластерного аналізу методом карт Кохонена, для виявлення структури експертних ранжувань. У сукупності з обчисленням коефіцієнта конкордації вказаний метод дає повну картину узгодженості експертної бази.

Якщо думки експертів виявляються узгодженими, то наступним кроком є визначення загального підсумкового ранжування. Оскільки відповіді експертів виміряні в порядковій шкалі, то для них неправомірно проводити усереднення методом середніх арифметичних.

Необхідно використовувати метод медіан [15]: відповіді експертів для кожного з об'єктів (характеристик) розташовують в порядку неспадання і обчислюють медіану як середнє арифметичне центральних членів варіаційного ряду. Отримане підсумкове ранжування виражає загальну думку експертів.

Виходячи з підсумкового ранжування, можемо визначити ваговий коефіцієнт для кожного показника практичності програмного забезпечення, використовуючи шкалу Фішберна [15]:

$$p_i = \frac{2(n-1+i)}{n(n+1)},$$

де i – ранг поточного показника.

Висновки

Багато стандартів та концептуальних моделей розглядають зручність використання як один з аспектів якості програмного забезпечення [2]. Більшість із них подають її як набір певних атрибутів і показників та пов'язаних з ними специфічних метрик. Це дозволяє розглядати зручність використання у вигляді ієрархічної системи з трьох рівнів, які містять локальні критерії. Запропонована модель оцінки (із застосуванням методу адитивної згортки) та забезпечення зручності застосування програмного забезпечення дозволяє не лише оцінювати поточний рівень практичності під час кожної ітерації гнучкої розробки, але й керувати зручністю використання створюваних програмних продуктів. Останнє зводиться до вирішення оптимізаційної задачі з цільовою функцією трудомісткості. Описана модель вносить достатній формалізм, надаючи можливість отримати математичний розв'язок за умови ефективної технічної реалізації.

Література

1. *Nebe K.* Integrating Software Engineering and Usability Engineering / K. Nebe, D. Zimmermann, V. Paelke // Advances in Human-Computer Interaction. – 2008. – P. 331–351.

2. *Analytical* Roadmap to Usability Definitions and Decompositions / Kumar Dubey Ajay Rana / International Journal of Engineering Science and Technology. – 2010. – Vol. 2(9). –P. 4723–4729.

3. *Memmel T.* Agile Human-Centered Software Engineering / *T.* Memmel, F. Gundelsweiler, H. Reiterer // Proceedings of the 21st British CHI Group Annual Conference on HCI. – 2007. – Vol.1. – P. 167–175.

4. *A survey* of user-centered design practice in China / *R. Zhou, S. Huang, X. Qin, J. Huang* // IEEE International Conference on Systems, Man and Cybernetics, SMC, 2008. – P. 1885–1889.

5. *Balbo S.* Automatic evaluation of user interface usability: Dream or reality / S. Balbo // Proceedings of the Queensland Computer-Human Interaction Symposium. Bond University, 1995. – P. 478–487

6. *Scholtz J.* Usability evaluation / J. Scholtz // IAD National Institute of Standards and Technology, Encyclopedia of Computer-Human Interaction. – 2004. – P. 115-122.

7. *Chang E.* A Usability-Evaluation Metric Based on a Soft-Computing Approach / E. Chang, T.S. Dillon // IEEE transactions on systems, man and cybernetics – Part A: systems and humans. – 2006. – Vol. 36, No. 2. – P. 356–372.

8. *Ivory M.Y.* The State of the Art in Automating Usability Evaluation of User Interfaces / M.Y. Ivory, M.A. Hearst // University of California, Berkeley ACM Computing Surveys. – 2001. – Vol. 33, No. 4. – P. 470–516.

9. *Сидоров М.О.* Оцінка зручності застосування програмного забезпечення в контексті Agile-розробки / М.О. Сидоров, І.В. Гученко // Наукоємні технології. – К.: Вид-во «НАУ-друк», 2009. – № 4(4). – С. 64–68.

10. *International* Organization for Standardization / International Electrotechnical Commission, 2001. ISO/IEC 9126-1 Standard, Software Engineering, Product Quality, Part 1: Quality Model, Geneva.

11. *Padda Harkirat.* QUIM: A Model for Usability/Quality in use Measurement / Harkirat. – Padda / LAP Lambert Academic Publishing. – 2009. – 120 p.

12. *Романов В.Н.* Системній анализ для инженеров / В.Н. Романов. – СПб.: СЗГЗТУ – 2006. – 186 с.

13. *Software* cost estimation with COCOMO II / Barry W. Boehm, Chris Abts, A. Winston Brown. – Englewood Cliffs, NJ:Prentice-Hall, 2000.

14. *Сергиенко И.В.* Математические модели и методы решения задач дискретной оптимизации / И.В. Сергиенко. – 2-е изд., доп. и перераб. – К.: Наук. думка, 1988. – 472 с.

15. *Горский В.Г.* Метод согласования кластеризованных ранжировок / В.Г. Горский, А.И. Орлов, А.А. Гриценко // Автоматика и телемеханика. – 2000. – Вып. 3. – С. 159–167.

16. *Гученко І.В.* Аналіз експертної бази на узгодженість методами DATA MINING / І.В. Гученко // Теоретичні та прикладні аспекти побудови програмних систем: тези доп. Сьомої Міжнар. конф. TAAPSD'2010. – С. 201–209.

Стаття надійшла до редакції 15.03.2011.