ІДЕАЛІЗОВАНІ МОДЕЛІ РЕІНЖИНІРИНГУ ПРОГРАМНИХ СИСТЕМ

Великодний С. С., канд. техн. наук, доцент
Коледж зв'язку та інформатизації

Одеської національної академії зв'язку ім. О. С. Попова

Програмні системи (ПС) застосовуються у різноманітних галузях життя й діяльності людини, але найбільше поширення вони отримали у галузях, де необхідною є робота із багатьма рутинними операціями, великим обсягом інформації, яку необхідно одночасно оброблювати, змінювати та доповнювати. Це стосується промисловості, виробництва, транспорту, навчання та інфокомунікацій.

Експлуатація ПС, у кожній окремій галузі застосування (будівництво, телекомунікації та зв'язок, сфера послуг, освіта та ін.), має свої принципові відмінності. Як самостійний приклад, можна навести ергатичні системи, а саме: системи моніторингу та дистанційного управління (SCADA-системи). Їх призначено для спостерігання та керування віддаленим об'єктом, який, до речі, може бути небезпечним для здоров'я оператора; крім того, за допомогою SCADA-систем виконується аналіз, накопичення та необхідне сортування робочих даних.

Спільної рисою для усіх ПС залишається те, що під впливом часу та інших невід'ємних факторів інформатизації (оновлення: операційних систем, мов програмування, принципів дії розподілених систем обробки даних тощо) відбувається еволюційне старіння видів забезпечення ПС. Така тенденція призводе до погіршення швидкісних, інформаційно-комунікаційних, графічних, часових та інших характеристик, аж до повної відмови ПС.

З цього випливає, що ПС повинна бути такою, що розвивається. За сучасними світовими тенденціями ПС спирається на життєвий цикл у 3 – 4 роки [1]. Звісно, що при оновленні об'єкту – оновлюється й ПС, за допомогою якої об'єкт обслуговується. На цьому етапі виникає питання: що робити, коли ПС жорстко прив'язано до об'єкту експлуатації? Наприклад: суднова система моніторингу та дистанційного управління, що являє собою ПС, яку жорстко прив'язано до суднової енергетичної установки та до вимірювальних каналів, при цьому життєвий цикл судна складає 12 – 15 років.

Відповідь на це питання – одна: необхідно застосовувати реінжиніринг щодо ПС. Реінжиніринг містить у собі процеси реорганізації і реструктуризації ПС, переведення окремих компонентів системи в іншу, сучаснішу мову програмування, а також процеси модифікації або модернізації структури і системи даних. [2]. При цьому архітектура системи може залишатися незмінною.

Проблему реінжинірингу ПС було розглянуто автором у [3]. Методологічні засади реінжинірингу було закладено у [4]. Згідно з цими засадами, постає необхідність розробки ідеалізованих моделей реінжинірингу (ІМР) кожного з видів забезпечення (технічне, математичне, інформаційне, програмне, лінгвістичне, методичне, організаційне, ергономічне тощо) ПС.

Початкова ідея створення ІМР була спрямована до побудови моделі у полярній системі координат. Побудова запропонованої ІМР відбувалася наступним чином: візьмемо у просторі довільну точку *O* (полюс), яку назвемо полюсом РІ та побудуємо проміні *OX* та *OY* – це будуть вісі витрат.

Нехай *M* – довільна точка площини, M_x та M_y – її прямокутні координати (проекції на вісі *OX* та *OY*). Вектор *OM* – радіус-вектор витрат (модуль якого збільшується). У подальшому позначимо цей радіус-вектор як ρ. Приймемо будь-який додатній відрізок за одиницю витрат *OA*. Кут повороту φ – це час, протягом якого відбувається реінжиніринг (зростає за годинниковою стрілкою та, надалі, збільшується кількість повних обертів).

Таким чином, ρ та φ – полярні координати. Тоді:

$$M_x = \rho \cos \varphi ; \quad (1)$$

$$M_y = \rho \ \sin \varphi ; \quad (2)$$

а радіус-вектор витрат:

$$\rho = \sqrt{(M_x)^2 + \left(M_y\right)^2}. \quad (3)$$

Проте у полярній моделі було знайдено деякі недоліки, що значною мірою ускладнюють її практичне застосування менеджерами проекту:

а) жодної з формул (1) – (3) недостатньо для розрахунку часу реінжинірингу φ, що потім було математично проаналізовано та уточнено [5, стор. 111], у зв’язку із чим час було фіксовано за фактом;

б) відсутня можливість обліку кількості ідентифікованих програмних компонентів (фізичних модулів коду), що необхідно наводити у лінійному масштабі, або кількості верифікованих рядків програмного коду – у логарифмічному масштабі;

в) подання IMP тільки у проекції часу та витрат й неможливість реалізувати модель у декількох інших проекціях, що цікавлять системного архітектора, а саме:

1) у ізометричній проекції програмних компонентів;

2) у логарифмічній проекції рядків програмного коду.

З аналізу цих недоліків було зроблено один з висновків, що полярна система координат не відповідає висунутим вимогам подання проекцій, а значить необхідно шукати шляхи побудови IMP у інших системах координат.

Методологія [6, 7] та дослідження [8], призвели до думки об'єднати ідеї побудови IMP у вигляді спіралі Архімеда та перенести її до циліндричної системи координат. В основу моделі закладено спіральний принцип організації відліку.

Початок побудови спіральної IMP дещо схожий із вище розглянутою початковою IMP. Однак, після завдання нульової точки реінжинірингу наступають нові етапи:

– вісь OZ – кількість ідентифікованих програмних компонентів (i) у лінійному масштабі або кількість верифікованих рядків програмного коду (j) у логарифмічному масштабі;

– побудована послідовність точок ($M_{i\vee j}$), які, власне, складають криву спіралі – це компоненти програмного коду (для зручності, нижче, $M_{i\vee j}$ будемо записувати як M_i);

– побудуємо вектор витрат (OM_i) – він з'єднує полюс реінжинірингу (O) з поточною точкою спіралі M_i та перевизначимо його як ρ:

$$\rho = OM_{i\vee j}. \quad (4)$$

Кут оберту φ між полюсом PI та поточною точкою M_i спіралі (кут повороту вектору OM_i) – це час, який необхідно затратити на виконання PI, причому, чим далі точка M_i від O, тим більше значення φ. Тобто кожний новий виток спіралі (n) додає 2π часу, а час тоді дорівнює:

$$t_i = \varphi_i(n+1), \text{[ум. од. часу]}. \quad (5)$$

Проекція представлення ІМР може бути різна, наприклад, якщо подати проекцію уздовж вісі ідентифікованих програмних компонентів (*OZ*), то ІМР буде являти собою Архімедову спіраль, що, у нашому випадку, описується рівнянням:

$$\frac{\rho}{M_{i\vee j}} = \frac{t_{i\vee j}}{2\pi} \tag{6}$$

або

$$\rho = \xi t_{i\vee j}, \tag{7}$$

де

$$\xi = \frac{M_{i\vee j}}{2\pi}. \tag{8}$$

Загалом, конфігурація спіралі може бути різна та залежить від багатьох факторів, що закладено у математичній моделі:

$$\Phi(r,\ \varphi,\ Z)\begin{Bmatrix} r(i\ \vee\ j) = \delta \times (i\ \vee j),\ \delta\ \forall\ [0\ \ldots\ \infty] \\ \varphi(i\ \vee\ j) = \theta \times (i\ \vee j),\ \theta\ \forall\ [0\ \ldots\ \infty] \\ Z(i\ \vee\ j) = \varepsilon \times (i\ \vee j),\ \varepsilon\ \forall\ [0\ \ldots\ \infty] \end{Bmatrix} \wedge (i \vee j)\ \forall\ [1\ \ldots\ \infty], \tag{9}$$

де δ – коефіцієнт автоматизації РІ;

θ – коефіцієнт схожості компонентів;

ε – верхня межа граничних витрат.

З наведених графічних побудов функцій $\Phi(r,\ \varphi,\ Z)$ – можна зробити висновки, що OM_i є *утворюючою.* Поверхня ІМР, для якої функція $\Phi(r,\ \varphi,\ Z)$ є *направляючою*, являє собою *лінійчату конічну поверхню*, що може бути зворотньовідновлена рухом утворюючої OM_i.

Таким чином, з наведених матеріалів, зрозуміло, що реінжиніринг дозволяє виконати еволюціонування ПС, шляхом внесення позитивних змін до її структури, з метою підвищення зручності її ж експлуатації та технічного супроводу.

Запропоновані ІМР видів забезпечення ПС являють собою еволюційні спіралі, які побудовані у циліндричній системі координат. Операції з ІМР можуть відбуватися у наступних проекціях: у проекції часу та витрат; у ізометричній проекції програмних компонентів; у логарифмічній проекції рядків програмного коду.

Перспективи подальших досліджень полягають:

– у аналітичному визначенні коефіцієнту автоматизації РІ (δ), що базується на декількох складових процесу РІ (при відомому коефіцієнті автоматизації РІ – ІМР набуде валідної конфігурації та може бути застосована з високою точністю до реальних завдань);

– у визначенні та завданні коефіцієнту схожості компонентів (θ);

– у математичному встановлені у моделі верхньої межі граничних витрат (ε).

Прогнозується, що реінжиніринг, який буде виконано за допомогою розроблених ІМР дозволить не тільки скоротити витрати на перепроектування ПС, але й підвищити ефективність технічного супроводу, збільшити життєвий цикл ПС, що вже знаходяться у експлуатації та подолати протиріччя між швидкими темпами розвитку науки, техніки і процесів проектування нових ПС.

Література

1. Тимченко А.А. Основи системного проектування та системного аналізу складних об'єктів. Кн. 1. Основи САПР та системного проектування складних об’єктів. К.: Либідь, 2003. 272 с.
2. Re-Engineering [Online]. Available: https://de.wikipedia.org/wiki/Re-Engineering.
3. Великодный С. С. Проблема реинжиниринга видов обеспечения систем автоматизированного проектирования. Междун. науч. журн. «Управляющие системы и машины». 2014. № 1. С. 57–61, 76.
4. Великодный С. С. Методологические основы реинжиниринга систем автоматизированного проектирования. Междун. науч. журн. «Управляющие системы и машины». 2014. № 2. С. 39–43.
5. Выгодский М.Я. Справочник по высшей математике. М.: ГИТТЛ, 1957. 784 с.
6. Boehm B. Spiral Development: Experience, Principles and Refinements. Special Report. CMU/SEI-2000-SR-008, 2000. 37 p.
7. Boehm B. A Spiral Model of Software Development and Enhancement. ACM SIGSOFT Software Engineering Notes. ACM. 1986. Vol. 11. Iss. 4. p. 14–24.
8. Richard W. Selby. Software Engineering: Barry W. Boehm's Lifetime Contributions to Software Development, Management and Research. Hoboken, New Jersey: John Wiley & Sons, 2007. 818 c.