УДК 681.324

ПОРІВНЯЛЬНИЙ АНАЛІЗ ОБ'ЄКТНО-ОРІЄНТОВАНИХ МЕТОДОЛОГІЙ РОЗРОБКИ ПРОГРАМНИХ СИСТЕМ

О.І. Тимочко, С.В. Осієвський, О.С. Гурін
(Харківський університет Повітряних Сил)

Проведено аналіз об'єктно-орієнтованих методологій розробки програмних систем. Викладено рекомендації щодо доцільності їх використання.

порівняльний аналіз, об'єктно-орієнтовані методології, програмні системи

Постановка проблеми. В сучасній практиці проектування програмного забезпечення інформаційних систем широко використовуються візуальні моделі, які являють собою засоби опису, проектування, документування архітектури системи [1 – 3]. Необхідність використання таких моделей перш за все викликана високими вимогами, що пред'являються до спеціалізованих інформаційних систем та багатогранністю середовищ збереження інформації, з якою працюють користувачі. В такому випадку роль моделей інформаційних систем полягає в упереджені створення завідомо непрацездатних та неоптимальних ІС, тобто вони є основою для взаємодії учасників проекту та гарантують коректність розроблюваної архітектури. Але великий вибір програмних засобів для створення моделей ІС ставить перед проектувальником складне завдання вибору ПЗ, за умови максимального охоплення розроблюваних задач [4].

Метою даної статті є проведення всебічного аналізу функціональних можливостей найбільш поширених методологій та викладення рекомендацій щодо їх використання. Зокрема в статті розглянуті наступні об'єктно-орієнтовані методології аналізу і розробки програмних систем: OMT (Object Modeling Technique), SA/SD (Structured Analysis/Structured Design), JSD (Jackson Structured Development), OSA (Object-Oriented System Analysis).

Основний матеріал.

Методологія OMT. Методологія, що підтримує дві перші стадії життєвого циклу програмних систем [1, 2]. Середовищем реалізації даної методології є програмний продукт *OMTTool*, який дозволяє розробляти моделі проектованої програмної системи в інтерактивному режимі з використанням багатовіконного графічного редактора і інтерпретатора

наборів діаграм. Діаграми розробляються на етапах аналізу вимог до системи і її проектування. Тобто по мірі отримання повного набору діаграм проектованої програмної системи, користувач має нагоду проінтерпретувати і заздалегідь оцінити різні властивості майбутньої реалізації системи. В даний час *OMTTool* входить до складу системи Paradigm+.

Методологія SA/SD. Методологія, що містить декілька варіантів систем позначень для формальної специфікації програмних систем [1, 2]. На етапі аналізу вимог і попереднього проектування для логічного опису проектованої системи використовуються специфікації (формальні описи) процесів, словник даних, діаграми потоків даних, діаграми станів і діаграми залежностей об'єктів.

Діаграми потоків даних, складають основу методології *SA/SD*, та моделюють перетворення даних при їх проходженні через систему. Методологія *SA/SD* полягає в послідовному розгляді та деталізації процесів, що входять до складу моделі програмного середовища. Для кожного елементарного процесу (процесу нижнього рівня) складається специфікація, яка описується за допомогою псевдокоду, таблиць прийняття рішень і т.п.

Набір діаграм станів процесів відіграє ту ж роль, що і динамічна модель в методології *OMT*. Діаграми залежностей об'єктів відображають залежності між сховищами даних. Ці діаграми аналогічні об'єктній моделі методології *OMT*. Так, в методології *SA/SD* організовано етап структурного аналізу (*SA*), логічним продовженням якого є етап структурного конструювання (*SD*), в процесі якого розробляються і уточнюються всі деталі проектованої системи.

Методологія JSD. Дана методологія характеризується своєрідним стилем розробки програмних систем. Він відрізняється від стилю, прийнятого в методологіях *SA/SD* або *OMT*. Ця відмінність полягає у тому, що між етапами розробки та аналізу не існує чіткого розмежування. Обидва етапи об'єднуються в один загальний етап розробки специфікацій проектованої системи. Методологія *JSD* знайшла своє застосування при проектуванні систем реального часу [1, 2].

Як і інші методології, методологія JSD використовує систему графічних позначень, хоча ця методологія менш орієнтована на графіку, чим методології *SA/SD* і *OMT*.

Розробка системи за методологією *JSD* включає наступні шість фаз:

- розробка дій і об'єктів;
- розробка структури об'єктів;
- розробка початкової моделі;
- розробка функцій;
- розробка часових обмежень;

– реалізація системи.

Таблиця 1

Аналітичні можливості порівнюваних методологій об'єктно-орієнтованого аналізу

Аналітична можливість	OSA	OMT	SA/SD	JSD
Об'єкти (наявність індивідуального і незалежного стану та поведінки)	+	+	+	+
Класи об'єктів (визначеність властивостей, що належать об’єктам)	+	+	+	+
Множинність зв'язків	+	+	+	+
Реляційні класи об'єктів (розгляд класів, як об’єктів)	+	+	–	+
Повна інтегрованість підмоделей	+	–	–	+
Агрегація	+	+	+	+
Узагальнення/спадкоємство	+	+	-	-
Повномасштабність обмежень на потужності зв'язків	+	–	–	–
Синоніми і омоніми	+	–	–	–
Повна система тригерів	+	+	–	–
Діяльність	+	+	+	+
Недетермінованість поведінки	+	+	–	–
Міжоб'єктний паралелізм	+	+	+	+
Внутрішньооб'єктний паралелізм	+	+	–	–
Виключення (допускається виявлення і обробка помилок за умовами)	+	–	–	–
Часові обмеження (забезпеченість обмеження часу на визначену операцію)	+	–	+	+
Темпоральність умов (формулювання умов, що посилаються на події у минулому, сьогоденні і майбутньому)	+	–	–	–
Можливість створення метамоделі:	+	–	–	–
Родовий клас	–	–	–	–
Взаємодія даних і подій	+	+	+	–
Уніфікація взаємодій	+	–	–	–
Деталізація взаємодій	+	–	–	–
Безперервність взаємодій	+	–	–	–
Взаємодії по бродкастингу	+	–	–	–
Загальне число аналітичних можливостей	24	12	8	9

Детальний аналіз вказаних етапів показав, що методологія *JSD* являється умовно об'єктно-орієнтованою. Це пов’язано з тим, що у ній майже не розглядається структура об'єктів та практично не приділяється

уваги їх атрибутам. Виходячи з цього, використання методології *JSD* може бути рекомендовано для проектування і реалізації наступних типів прикладних програмних систем:

– паралельні асинхронні програмні системи, в яких процеси можуть взаємно синхронізувати один одного;

– програмні системи реального часу;

– програмні системи для паралельних комп'ютерів.

Небажано використовувати методологію *JSD* для вирішення наступних задач:

– високорівневий аналіз (методологія *JSD* не забезпечує широкого розуміння проблеми; вона неефективна для абстракції і спрощення проблем);

– розробка баз даних.

Методологія OSA. Забезпечує об'єктно-орієнтований аналіз програмних систем і не містить можливостей, пов'язаних з підтримкою етапу розробки. Дану методологію слід сприймати, як реалізаційно-орієнтовану, а не проблемно-орієнтовану. Вона забезпечує попередню розробку, а не аналіз вимог до системи. Такий висновок зроблено, виходячи з табл. 2, в якій приведено аналіз можливостей розглянутих методологій, щодо етапів розробки системи.

Таблиця 2

Функціональні можливості порівнюваних методологій об'єктно-орієнтованого аналізу, що використовуються на етапі розробки системи

Можливість	**OSA**	**OMT**	**SA/SD**	**JSD**
Значення станів (Наявність стану об’єкта, але відсутність поведінки та індивідуальності).	–	+	+	+
Атрибути (методи) (визначеність класів об'єктів в термінах атрибутів та/або методів)	–	+	+	+
Шаблони класів об'єктів (шаблони, по яким створюються екземпляри класів об'єктів, т.т. властивості екземпляра об'єкту визначають клас, а не властивості об'єкту)	–	+	–	+
Абстрактність класів	–	+	+	+
Псевдоспадкоємство	–	+	+	+
Тотожність по значеннях (множина атрибутів, що використовується тільки для визначення тотожності об'єктів)	–	+	+	–
Зміна семантики	–	+	+	–
Імперативний виклик операцій (існування виклик методу у відношенні клієнт-сервер)	–	–	–	–
Загальне число функціональних можливостей	0	7	6	6

по розробці				

Методологія *OSA* зосереджена тільки на проблемах аналізу, пропонуючи зручні методи аналізу систем. Вона забезпечує інтерпретацію моделей на комп'ютері на перших етапах аналізу системи.

Методологія *OSA*, як і інші методології, підтримує три взаємно-ортогональні представлення (моделі) проектованої системи:

– модель залежностей між об'єктами;

– модель поведінки об'єктів;

– модель взаємодії об'єктів.

Модель залежностей між об'єктами аналогічна об'єктній моделі методології *OMT*.

Висновки. Таким чином, узагальнюючи вище приведені дані можна сказати, що у методологій *SA/SD* і *OMT* багато спільного: обидві методології використовують схожі конструкції для моделювання і підтримують три взаємно-ортогональні представлення проектованої системи. Методології *SA/SD* і *OMT* можна розглядати як два способи використання інструментального засобу – *OMTTool*.

Методологія *JSD* є дещо обмеженою в роботі з високорівневими моделями та при моделюванні функціонування програмних систем в яких функціонують бази даних в той же час її доцільно використовувати при розробці простих систем реального часу.

Методологія *OSA* забезпечує досконалу інтерпретацію моделей при виконанні перших етапів аналізу системи. Але використання даної методології на етапах реалізації є недоцільним, що викликано низькою функціональністю.

ЛІТЕРАТУРА

1. *Принципы проектирования и разработки программного обеспечения. Учебный курс MCSD. – М.: «Русская редакция», 2000. – 608 с.*
2. *Вендров А.М. Проектирование программного обеспечения экономических информационных систем: Учебник. – М.: Финансы и статистика, 2000. – 352 с.*
3. *Мацяшек Л. Анализ требований и проектирование систем. Разработка информационных систем с использованием UML.: Пер. с англ.: – М.: Вильямс, 2002. – 432 с.*
4. *Ройс У. Управление проектами по созданию программного обеспечения: Пер. с англ. – М.: ЛОРИ, 2002. – 344 с.*

Надійшла 12.10.2005

Рецензент: доктор фізико-математичних наук, професор С.В. Смеляков,

Харківський університет Повітряних Сил.