УДК 004.434

ПРЕДМЕТНО-ОРІЄНТОВАНЕ МОДЕЛЮВАННЯ ТА КЕРУВАННЯ ПРОЦЕСАМИ СИСТЕМНОЇ ІНЖЕНЕРІЇ

В. І. Межуєв
Бердянський державний педагогічний університет
вул. Шмідта, 4, м. Бердянськ, 71100, Україна. E–mail: mejuev@ukr.net

Доведена застосовність інформаційної технології предметно-орієнтованого математичного моделювання (Domain Specific Mathematical Modelling, DSMM) до створення моделей різноманітних процесів системної інженерії (специфікації, проектування, розробки, тестування та ін.). Сутність інформаційної технології полягає у визначені онтологічної метамоделі процесу та додаванні до кожного концепту онтології множини предметних атрибутів, що визначають зміну стану моделі. Модель процесу розглянута як скінчений автомат, що визначає порядок переходу станів онтології деякої системи. На основі машини Мілі наведене формальне визначення граматики метамоделі процесів, що включає вхідний алфавіт Σ, вихідний алфавіт Γ, початковий стан S_0, функцію зміни стану δ та вихідну функцію ω. Для реалізації управління виконання процесами запропонований підхід, що пов'язує декларативні умови граматики з функціями, які визначають дозволи на дії користувачів засобу DSMM. Такий підхід є програмною реалізацією поведінки скінченого автомату.

Ключові слова: предметно-орієнтоване математичне моделювання, метамодель, модель процесу, скінчений автомат.

ПРЕДМЕТНО-ОРИЕНТИРОВАННЫХ МОДЕЛИРОВАНИЕ И УПРАВЛЕНИЕ ПРОЦЕССАМИ СИСТЕМНОЙ ИНЖЕНЕРИИ

В. И. Межуев
Бердянский государственный педагогический университет
ул. Шмидта, 4, г. Бердянск, 71100, Украина. E–mail: mejuev@ukr.net

Доказана применимость информационной технологии предметно-ориентированного математического моделирования (Domain Specific Mathematical Modelling, DSMM) для создания моделей различных процессов системной инженерии (спецификации, проектирования, разработки, тестирования и др.). Сущность информационной технологии заключается в определении онтологической метамодели процесса и добавлении к каждому концепту онтологии множества предметных атрибутов, определяющих изменение состояния модели. Модель процесса рассмотрена как конечный автомат, определяющий порядок перехода состояний онтологии некоторой системы. На основе автомата Мили приведено формальное определение грамматики метамодели процессов, которое включает входной алфавит Σ, выходной алфавит Γ, исходное состояние S_0, функцию изменения состояния δ и исходную функцию ω. Для реализации управления выполнения процессов предложен подход, связывающий декларативные условия грамматики с функциями, которые определяют разрешения на действия пользователей средства DSMM. Такой подход является программной реализацией поведения конечного автомата.

Ключевые слова: предметно-ориентированное математическое моделирование, метамодель, модель процесса, конечный автомат.

АКТУАЛЬНІСТЬ РОБОТИ. Сутність інформаційної технології (ІТ) предметно-орієнтованого математичного моделювання (англ. Domain-Specific Mathematical Modelling, DSMM) полягає у побудові моделей предметних областей (ПрО) шляхом розробки та застосування специфічних для ПрО мов моделювання (англ. Domain-Specific Language, DSL) [1]. При цьому розробка DSL здійснюється у рамках певної метамоделі. У роботах [2–4] нами була запропонована онтологічна метамодель, що дозволяє охопити різні аспекти системної інженерії, включаючи формулювання вимог і специфікацій, архітектурне моделювання, створення робочого плану тощо.

У той же час ІТ DSMM надає можливість створення не лише статичних структурних моделей предметних областей, але і моделей процесів, що відповідають специфіці ПрО (методології, проекту, підприємства, технології та ін.). Крім того, разом із моделюванням специфічних для ПрО процесів, засоби DSMM дозволяють організовувати діяльність користувачів у відповідності до створеної моделі з метою досягнення мети процесу.

Запропонований підхід має спільні риси з інженерією бізнес-процесів BPE (Business Process Engineering) [5], а також комп'ютерно-орієнтованою інженерією методів CAME (Computer Aided Method Engineering) [6], але є більш гнучким як завдяки можливості розробки довільної метамоделі процесу, так і керування користувачами засобів DSMM з метою досягнення цілі процесу. Наприклад, технічний інженер може використати засіб DSMM для моделювання різних сценаріїв функціонування системи; комерційний директор може створювати бізнес-плани, моделюючи бізнес-процеси з використанням фінансової термінології й т. і.

Створення метамоделі в ІТ DSMM здійснюється шляхом визначення онтології процесу на основі граматики, що зумовлює порядок виконання процесу. Якщо онтологічна метамодель для специфікації систем будувалася на основі теорії графів, метамодель процесів створюється на засадах скінчених

автоматів. У цьому контексті модель процесу розглядається як автомат, де правила граматки метамоделі визначають порядок переходу станів у процесі проектування системи.

Мета роботи – предметно-орієнтоване моделювання та керування процесами системної інженерії.

МАТЕРІАЛ І РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ. З метою розкриття застосовності IT DSMM, розглянемо кілька прикладів моделей процесів. В галузі BPE існує багато семантичних моделей, наприклад, PDCA (Plan, Do, Check, Act – Планування, Виконання, Перевірка, Дія), AIDA (Attention, Interest, Desire, Action - Увага, Інтерес, Бажання, Дія), OODA (Observe, Orient, Decide, Act – Спостереження, Орієнтація, Рішення, Дія) та ін. Всі ці моделі визначають процес, як послідовність кроків, що маркіровані поняттями (як то «Планування», «Виконання», «Перевірка», «Дія»). Взагалі кажучи, абстрагування від природи реального процесу та його подання як послідовності кроків є найпоширенішим способом моделювання процесу. Зазначимо, що типовою семантикою кроку процесу у системотехніці є дія користувача, технічна операція, зміна ресурсу (наприклад, версії документу) та ін.

Для визначення структури та візуалізації моделей процесів також можна використати метамодель на основі орієнтованого графу, вузли яких мають семантику кроків процесу, а ребра визначають напрямок його виконання. Вузли графу також можуть включати логічні операції для визначення умов виконання процесу (рис. 1).

Для того, щоб наступний крок процесу залежав не лише від результату виконання попереднього, визначимо граматику метамоделі на основі скінченого автомату (Finite State Machine, FSM), а саме, машини Мілі [7]. FSM визначає модель процесу як кінцеву множину станів деякої системи і правил переходів між цими станами. Поняття стану визначимо як множину значень атрибутів всіх концептів (екземплярів типів T^{X} онтологічної метамоделі), що складають модель ПрО.

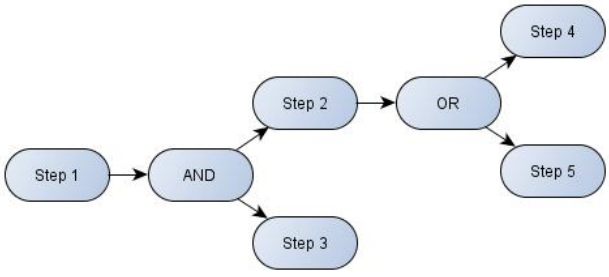

Рисунок 1 – Графова форма подання моделі процесу

Наприклад, елементи T^{X} можуть бути визначені наступним чином:

T^{X} = {«Вимога», «Специфікація», «Архітектурна сутність», «Задача розробки», «Задача валідації», «Задача верифікації»}.

Розширимо онтологічну метамодель шляхом додавання до кожного концепту T^{X} множини предметних атрибутів А, що визначають зміну стану системи. Такий підхід дозволяє перетворити онтологію ПрО в онтологію процесу ПрО.

Визначимо граматику метамоделі для моделювання процесів. Формально, FSM (машина Мілі) визначається як кортеж S:

$$S \prec \langle S, \Sigma, \Gamma, S_o, \delta, \omega \rangle, \qquad (1)$$

де S – скінчена, непуста множина станів; Σ – вхідний алфавіт (скінчена, непуста множина символів); Γ – вихідний алфавіт (скінчена, непуста множина символів); S_0 – початковий стан, $S_0 < S$; δ – функція зміни стану, δ: S × Σ→ S; ω – вихідна функція, ω: S × Σ→ Γ.

З визначення (1) випливає можливість застосування FSM для специфікації граматики (регулярної) предметно-орієнтованої мови моделювання. Для розробки вхідного алфавіту машини Мілі використаємо множину понять онтології T^{X}. Ці поняття розширимо атрибутами А, що визначають можливі стани моделі системи. Символ вхідного алфавіту Σ, що приймається скінченим автоматом S, приводить до символу, що належить вихідному алфавіту Γ.

Визначимо вхідний алфавіт Σ як декартовий добуток множини понять онтології (типів концептуальної метамоделі) ПрО T^{X} й атрибутів А.

$$\Sigma = T^{\mathrm{X}} \times \mathrm{A}. \qquad (2)$$

Наведемо приклад атрибуту А, що визначає зміну стану моделі процесу з розробки системи:

$$\mathrm{A} = \{\text{«Робота почата», «Робота зроблена»}\}. \qquad (3)$$

Зазначимо, що у загальному випадку множина атрибутів А позв'язує онтологію ПрО із процесом ПрО і дозволяє визначити етапи розв'язання специфічних для ПрО задач (у контексті системотехніки, специфікації, проектування, планування, розробки, випробування, стандартизації й т. і.).

З визначення (1) та з використанням множини понять T^{X}, будемо мати наступний вхідний алфавіт Σ скінченого автомату:

Σ = {{Вимога, Робота почата},
{Вимога, Робота зроблена},
{Специфікація, Робота почата},
{Специфікація, Робота зроблена}
...
{Валідація, Робота почата},
{Валідація, Робота зроблена}} (4)

Вихідний алфавіт у простішому випадку визначимо як булеву множину Γ={«так», «ні»}, тобто будемо розглядати скінчений автомат S як акцепторну машину.

Наведемо визначення початкового стану автомату S

$$S_0 = \{ I^X \mid a \in I^X \in a = \text{"Робота почата"} \}, \quad (5)$$

де I^X є екземпляром $(T^X \times A)$, а ∈ А.

S_0 є множиною екземплярів I^X, елементи якої мають значення атрибуту а ∈ А "Робота почата".

Визначення функції зміни стану δ скінченого автомату залежить від семантики ПрО. У нашому прикладі вона визначає етапи процесу проектування системи та зумовлює порядок визначення атрибутів понять I^X користувачами засобу DSMM (що й є способом керування їх діяльністю).

Визначимо вихідну функції ω автомату S:

$$\omega(i,\varsigma) = \begin{cases} \gamma \leftarrow так, i \leftarrow \varsigma : \forall a \in I^X : \\ a = "Робота\ зроблена" \\ \gamma \leftarrow ні, інакше \end{cases}, \quad (6)$$

де $\gamma \in \Gamma$.

Результат ω є функцією двох змінних: екземпляру $i \in I^X$, атрибут а якого треба встановити, і вхідного символу $\varsigma \in \Sigma$.

I^X є множиною екземплярів $(T^X \times A)$, $i \in I^X$,

γ {"так", "ні"}.

Функція ω встановлює символ γ вихідного алфавіту у "так" та екземпляр $i \in I^X$ у ς, якщо всі раніше встановлені атрибути, а у всіх екземплярах I^X мають значення "Робота зроблена". Інакше, символ γ вихідного алфавіту встановлюється в "ні". Така вихідна функція ω є правилом граматики метамоделі процесу специфікації системи та дозволяє керувати його виконанням. Тобто користувач засобу DSMM не може здійснити дії, що не відповідає встановленому на рівні метамоделі правилу (6).

Здійснення керування процесом специфікації системи.

Визначена у [2–4] концептуальна метамодель для специфікації систем була розширена граматикою (6), що визначає правила задання та зміни атрибутів екземплярів понять «вимога», «специфікація», «задача розробки», «задача тестування», «задача валідації», «задача верифікації» та ін. Процес специфікації систем розглядається як інстанціація загальних понять конкретними твердженнями, що описують властивості й поведінку майбутньої системи.

Правила виду (6) дозволяють організовувати процеси діяльності, зокрема проектування, планування, розробку, тестування, стандартизацію й т. і. Модель процесу є скінченим автоматом, який будується шляхом накладання обмежень на переходи між його різними етапами. Наприклад, у контексті проектування систем, етап архітектурного моделювання може мати місце тільки після узгодження всіх специфікацій архітектури системи.

Порядок переходів станів скінченого автомату моделі процесу проектування системи визначається логічними умовами, покладеними на величини атрибутів екземплярів типів метамоделі. Наприклад, визначмо атрибут статус із набором можливих значень {«схвалено», «не застосовне»}. У даному випадку зв'язок між екземплярами вимоги і специфікації є логічною імплікацією:

$$\forall((вимога.статус = "схвалено") \vee (вимога.статус = "не\ застосовне")) \rightarrow специфікація.статус = "схвалено", \quad (7)$$

тобто специфікація може бути схвалена, якщо відповідні вимоги були схвалені або ж не є застосовними. Таким чином, перехід до наступного рівня визначення системи (рівня специфікацій) можливий лише після схвалення попереднього (рівня вимог).

Логічні умови, покладені на відношення типових понять метамоделі проектування систем, визначають порядок зміни станів моделі системи між різними етапами її розробки. Ці зміни станів і визначають процес розробки системи. На рисунку 2 зображені дозволені переходи між різними етапами визначення системи. Зазначимо, що ці етапи є водночас поняттями метамоделі проектування систем (вимога, специфікація, сутність та ін.).

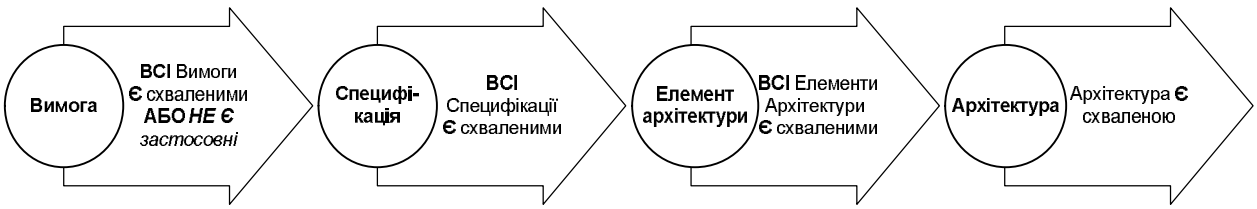

Рисунок 2 – Переходи станів онтологічної моделі майбутньої системи

Розглянемо програмні засоби керування діяльністю користувачів засобів DSMM.

Відповідно даному у [1] визначенню метамоделі, її структура включає множину програмних методів, що зокрема використовується для формулювання правил граматики мови моделювання. З метою практичної реалізації таких правил в засобі DSMM пропонується програмний інтерфейс користувача (Application Programming Interface, API).

API засобів DSMM дозволяє зв'язати декларативні умови граматики з функціями, що, зокрема, визначають дозволи на дії користувачів (*англ.* permissions). Якщо логічна формула, що визначає правило граматики, є істинною, викликається відповідній умові метод, що, наприклад, у розглянутому прикладі (7), дозволяє користувачу засобу встановити значення атрибута "статус" у "схвалено". Зазначимо, що API засобів DSMM визначає не лише до-

зволи на зміну атрибутів елементів моделі ПрО, але і можливість створення (знищення) екземплярів, тобто модифікацію структури онтології ПрО.

Наведемо приклад визначення метамоделі процесу розробки системи на основі графової мета-метамоделі, що включає наступні етапи:

Визначення алфавіту (типів) метамоделі - вимога (Requirement), специфікація (Specification).

Node Requirement, Specification;

де Node (вузол), Edge (ребро) є метатипами мета-метамоделі, основаної на теорії графів.

Визначення граматики.

Edge Derive(Requirement, Specification);

де Edge є ребром графу метамоделі, що визначає простіше правило граматики - встановлення дозволу на можливість поєднання екземплярів вузлів Requirement (вимога) та Specification (специфікація) ребром Derive (породжує).

Атрибути типів.

Типи Requirement, Specification мають такі атрибути як: ім'я, автор, дата створення, текстовий опис та ін. Для організації процесу найбільш важливим є атрибут статус (Status), що має набір можливих значень {"In Work", "Approved", "Not Applicable"}. Додавання атрибутів здійснюється за допомогою функції AddAttribute, наприклад:

AddAttribute(Requirement, Status,
{"In Work", "Approved", "Not Applicable"});
AddAttribute (Specification, Status,
{"In Work", "Approved", "Not Applicable"})

Значення атрибуту Status визначає можливість схвалення користувачами засобів екземплярів типів вимог та специфікацій. Цей принцип формулюється як правило граматики метамоделі за допомогою функції ConditionalAction, що викликає дію (Action) у випадку істинності умови (Condition):

ConditionalAction ((Subset(Requirement, Status, "In Work") == NULL), AllowToChangeValue (Specification, Status));

Функція Subset засобу DSMM виділяє підмножину екземплярів вимог (Requirement), статус (Status) яких має значення "у роботі" ("In Work"). Якщо ця підмножина є порожньою (NULL), встановлюється дозвіл на зміну значення статусу (Status) специфікації (Specification) за допомогою функції AllowToChangeValue.

Зазначимо, що наведене правило граматики не бере до уваги поєднання елементів; у дійсності, специфікація може бути схваленою, якщо статуси пов'язаних з нею вимог є схваленими або не застосовними. Для формулювання цього правила граматики необхідно спочатку виділити підмножину всіх вимог, пов'язаних зі специфікацією:

ForAll (Specs)
Reqs = Subset (Specs, "LinkTo", "Requirements")
ConditionalAction ((Subset(Reqs, Status, "In Work") == NULL), AllowToChangeValue (Specs, Status));

Зазначимо, що визначення наведеного методу здійснюється на рівні метамоделі. На рівні побудови моделі ПрО здійснюється використання цього методу, що зводиться до виклику методу Approve (Spec);

Таким чином, засоби DSMM дозволяють не лише моделювати різні процеси системотехніки (специфікацію, планування, стандартизацію й т.ін.), але також й керувати діяльністю користувачів відповідно до визначеної моделі процесу. Взагалі кажучи, інструменти, що реалізують таку технологію, відносяться до класу інструментів керування потоками робіт WMS (англ. Workflow Management System) [8]. Їх головна мета полягає в тому, щоб гарантувати, що користувач виконав всі завдання у визначений термін.

Але порівняно з WMS засоби DSMM мають низку переваг:

– можливість моделювання процесів шляхом визначення послідовності кроків і порядку їх виконання;

– моделювання процесів, виконання яких визначається не лише попереднім кроком, але минулим станом системи;

– можливість організації дій користувачів відповідно до створеної моделі процесу;

– гарантування виконання процесу користувачем (наприклад, його відповідність до стандарту, технології, алгоритму і т. і.);

– можливість формальної перевірки моделі у просторі її станів;

– здійснення частково автоматизованої обробки концептуальної моделі (наприклад, стандартизації);

– можливість застосування математичних алгоритмів (наприклад, для оптимізації послідовності кроків процесу, розподілу ресурсів, виявлення тупиків процесу й т. і.).

ВИСНОВКИ. Розглянута застосовність IT DSMM до предметно-орієнтованого моделювання та керування процесами системної інженерії. Доцільність IT DSMM полягає в теоретичній і практичній можливості створення моделей процесів, що відповідають специфіці ПрО (підприємству, методології, проекту, технології й ін.).

З метою побудови моделей процесів, основана на теорії графів онтологічна метамодель була розширена положеннями теорії скінчених автоматів. До кожного концепту метамоделі була додана множина предметних атрибутів, що визначають зміну стану моделі проектування системи. У цьому контексті, модель процесу розглядається як скінчений автомат, що визначає порядок переходу станів процесу.

На основі скінченого автомату Мілі наведене формальне визначення граматики метамоделі процесів, що включає вхідний алфавіт Σ, вихідний алфавіт Γ, початковий стан S_0, функцію зміни стану δ та вихідну функцію ω.

Для реалізації правил граматики метамоделі процесів запропоноване API засобів DSMM, що дозволяє зв'язати декларативні умови граматики з функціями, що визначають дозволи на дії користувачів. Такий підхід є програмною реалізацією поведінки скінченого автомату.

ЛІТЕРАТУРА

1. Межуєв В.І. Інформаційна технологія розробки комплексних інструментальних засобів предметно-орієнтованого математичного моделювання // Автореф. дис. докт. техн наук. – Одеса: ОНПУ, 2012. – 36 с.

2. Межуев В.И. Онтологические модели систем и процесса системной инженерии // Искусственный интеллект. – 2010. – № 4. – С. 606–616.

3. Interacting Entities Modelling Methodology for Robust Systems Design / V. Mezhuyev, B. Sputh, E. Verhulst // Advances in System Testing and Validation Lifecycle. – CPS publishing. – 2010. – PP. 75–80.

4. OpenCookbook: unified and formalised environment for systems engineering / V. Mezhuyev, E. Verhulst // Computer systems and components. – Vol. 429. – 2009. – PP. 57–65.

5. Hansen Gregory. Automating Business Process Reengineering. – Prentice Hall, 1993. – 321 p.

6. Dahanayake Ajantha. Computer-Aided Method Engineering: Designing CASE Repositories for the 21st Century. – IGI Global, 2001. – 252 p.

7. Automata and Languages: Theory and Applications / Alexander Meduna. – Springer, 2000. – 920 p.

8. The design and implementation of a workflow analysis tool / V. Curcin, M. Ghanem, Y. Guo // Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences. – 2010. – PP. 4193–4208.

DOMAIN-SPECIFIC MODELLING AND CONTROL OF PROCESSES OF SYSTEM ENGINEERING

V. Mezhuyev
Berdyansk State Pedagogical University
vul. Schmidt, 4, Berdyansk, 71100, Ukraine. E–mail: mejuev@ukr.net

The applicability of the Information Technology (IT) of Domain Specific Mathematical Modelling (DSMM) for the modelling different processes of the systems engineering (specification, design, development, testing etc.) is proved. The essence of IT is the definition of the ontological metamodel of a process and adding to each concept of the ontology the set of domain attributes that define the state transitions of the model. The model of the process is considered as a finite automata that determines the order of the state transition of the ontology of a system. Based on the Mealy machine the formal definition of the metamodel grammar of process that includes input alphabet Σ, output alphabet Γ, the initial state S_0, the changes function δ and output function ω is given. To implement the rules of the metamodel grammar of processes the approach proposed that connects the declarative conditions of grammar with functions that define permissions for user actions of DSMM tools. This approach is the software support of behaviour of finite state machine.

Key words: domain-specific mathematical modelling, metamodel, model of a process, finite state machine.

REFERENCES

1. Mezhuyev V. *Information technology of development of multipurpose tools for domain specific mathematical modelling* // Thesis abstract… Doc. Sc., Engineering. – Odessa: ONPU, 2012. – 36 p. [in Ukrainian]

2. Mezhuyev V.I. Ontological models of systems and of system engineering process // *Artifitial intelegence*. – № 4. – 2010. – PP. 606–616. [in Ukrainian]

3. Interacting Entities Modelling Methodology for Robust Systems Design / V. Mezhuyev, B. Sputh, E. Verhulst // *Advances in System Testing and Validation Lifecycle*. – CPS publishing. – 2010. – PP. 75–80.

4. OpenCookbook: unified and formalised environment for systems engineering / V. Mezhuyev, E. Verhulst // *Computer systems and components*. – Vol. 429. – 2009. – PP. 57–65.

5. Hansen Gregory. *Automating Business Process Reengineering*. – Prentice Hall, 1993. – 321 p.

6. Dahanayake Ajantha. *Computer-Aided Method Engineering: Designing CASE Repositories for the 21st Century*. – IGI Global, 2001. – 252 p.

7. *Automata and Languages: Theory and Applications* / Alexander Meduna. – Springer, 2000. – 920 p.

8. The design and implementation of a workflow analysis tool / V. Curcin, M. Ghanem, Y. Guo // *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*. – 2010. – PP. 4193–4208.

Стаття надійшла 22.01.2013.