*УДК 004.052.42: 004.053: 004.94*

Шкарупило В. В.[1], Кудерметов Р. К.[2], Польська О. В.[3]

*[1]Канд. техн. наук, старший викладач кафедри комп'ютерних систем та мереж Запорізького національного технічного університету, Запоріжжя, Україна*
*[2]Канд. техн. наук, доцент, завідувач кафедри комп'ютерних систем та мереж Запорізького національного технічного університету, Запоріжжя, Україна*
*[3]Старший викладач кафедри комп'ютерних систем та мереж Запорізького національного технічного університету, Запоріжжя, Україна*

# DEVS-ОРІЄНТОВАНА МЕТОДИКА ВАЛІДАЦІЇ КОМПОЗИТНИХ ВЕБ-СЕРВІСІВ

Запропоновано методику валідації композитних веб-сервісів за рахунок синтезу імітаційних дискретно-подійних моделей на основі формалізму DEVS. Це дозволяє виконувати автоматизовану перевірку придатності таких систем до цільового використання при проектуванні шляхом імітаційного моделювання. В якості вхідних даних використано формальну специфікацію на основі темпоральної логіки TLA, що дозволяє математично строго представляти функціональні характеристики композитних сервісів у форматі обчислювальних процесів. За аналітичну модель в основі TLA-специфікації взято структуру Кріпке. У межах методики запропоновано правила синтезу із вихідної TLA-специфікації DEVS-моделі композитного сервісу, призначеної бути засобом валідації. Результуюча DEVS-модель складається із моделей атомарних сервісів, моделі клієнта композитного сервісу та моделі координатора атомарних сервісів, що функціонує згідно специфікації WS-BPEL.

Для перевірки методики проведено експериментальні дослідження, що підтвердили адекватність результуючої DEVS-моделі. Перевірку здійснено згідно запропонованого підходу, що полягає у порівнянні результатів валідації шляхом імітаційного моделювання із результатами валідації шляхом тестування. За результатами проведених досліджень обґрунтовано доцільність використання запропонованої методики при проектуванні композитних сервісів, що базується на зменшенні часових витрат на валідацію. Наголошено на доречності використання методики при ітераційній розробці.

**Ключові слова:** SOA, WS-BPEL, композитний веб-сервіс, специфікація, верифікація, валідація, TLA, DEVS.

**НОМЕНКЛАТУРА**

AWS – Atomic Web Service;
CPU – Central Processing Unit;
CWS – Composite Web Service;
DEVS – Discrete Event System Specification;
JAX-WS – технологія створення атомарних веб-сервісів;
ODE – Orchestration Director Engine;
RAM – Random-access Memory;
SOA – Service-oriented Architecture;
TA – Timed Automata;
TLA – Temporal Logic of Actions;
UML – Unified Modeling Language;
WS-BPEL – Web Services Business Process Execution Language;
ОП – обчислювальний процес;
*activate* – повідомлення активації моделі *MG*;
*AP* – множина атомарних висловлювань (літералів);
*Atomics* – множина атомарних моделей у складі результуючої моделі CWS;
*break* – функція видалення міжпортових зв'язків DEVS-моделі CWS;
*Calc* – умовне позначення TLA-специфікації окремого ОП;
*Cnd* – умовне позначення специфікації розмітки $L(s)$;
$D$ – множина допустимих значень змінних станів;
$e$ – час, що пройшов від останнього переходу $(s,s')\in R$;
*ev* – повідомлення активації моделі $MA_i$;
$F$ – статистичний критерій Фішера;
$f_1(x)$ – функція залежності часових витрат на моделювання від затримок AWS;
$f_2(x)$ – функція залежності оціночних значень показника $\xi_{nf}^{s}$ від затримок AWS;
$f_3(x)$ – функція залежності фактичних значень показника $\xi_{nf}^{t}$ від затримок AWS;
*Init* – умовне позначення TLA-специфікації розмітки $L(s_0)$ структури Кріпке;
*IP* – множина вхідних портів складеної (результуючої) DEVS-моделі CWS;
$job_j$ – $j$-а заявка на обслуговування;
$L$ – функція розмітки станів системи переходів;
$L(s)$ – розмітка поточного стану системи переходів;
$L(s')$ – розмітка наступного стану системи переходів;
$m$ – кількість AWS у складі CWS;
$m'$ – фактична кількість AWS у складі CWS, необхідна для окремого ОП;
$MA_i$ – базова DEVS-модель $i$-го AWS;
*MC* – базова модель координатора AWS;
*MG* – базова модель клієнта (генератора заявок на обслуговування);
*OP* – множина вихідних портів складеної (результуючої) DEVS-моделі CWS;
$R$ – множина переходів між станами;
$res_4$ – шуканий результат експериментальних обчислень;
*rs* – повідомлення-результат роботи моделі $MA_i$;
$S$ – множина станів системи переходів;
$s_0$ – початковий стан системи переходів;


DOI 10.15588/1607-3274-2015-4-12

$s$ – поточний стан системи переходів;

$s'$ – наступний стан системи переходів;

*set* – функція встановлення міжпортових зв'язків DEVS-моделі CWS;

*ST* – множина міток станів атомарних DEVS-моделей;

$t$ – статистичний критерій Ст'юдента;

*ta* – функція просування модельного часу;

*UNCHANGED* – ключове слово для визначення псевдоподій у специфікації;

$V$ – множина змінних станів системи переходів;

$X$ – темпоральний оператор часового зсуву;

$\delta_{ext}^{MA_i}$ – зовнішня функція переходів DEVS-моделі $MA_i$;

$\delta_{ext}^{MC}$ – зовнішня функція переходів DEVS-моделі $MC$;

$\delta_{ext}^{MG}$ – зовнішня функція переходів DEVS-моделі $MG$;

$\delta_{\text{int}}^{MA_i}$ – внутрішня функція переходів DEVS-моделі $MA_i$;

$\delta_{\text{int}}^{MC}$ – внутрішня функція переходів DEVS-моделі $MC$;

$\delta_{\text{int}}^{MG}$ – внутрішня функція переходів DEVS-моделі $MG$;

$\lambda^{MA_i}$ – функція одержання результату роботи DEVS-моделі $MA_i$;

$\lambda^{MC}$ – функція одержання результату роботи DEVS-моделі $MC$;

$\lambda^{MG}$ – функція одержання результату роботи DEVS-моделі $MG$;

$\xi_f^s$ – показник функціональної характеристики CWS при моделюванні;

$\xi_{nf}^s$ – показник нефункціональної характеристики CWS при моделюванні;

$\xi_f^t$ – показник функціональної характеристики CWS при тестуванні;

$\xi_{nf}^t$ – показник нефункціональної характеристики CWS при тестуванні.

**ВСТУП**

На сьогодні при організації різноманітних бізнес-процесів (у загальному випадку – обчислювальних процесів) інтенсивно використовуються технології створення розподілених програмних систем на основі архітектури SOA. Відповідно до принципів SOA, функціонування таких систем може бути реалізоване шляхом централізованого координування основних компонентів (надалі – сервісів або атомарних сервісів) [1]. Безпосередньо системи прийнято називати композитними сервісами (CWS). Розповсюдженим засобом створення сервісів є технологія JAX-WS [2].

Для перевірки коректності функціонування названих компонентів може бути проведене модульне тестування [3]. Перевірка узгодженості взаємодії сервісів при здійсненні допустимих сценаріїв ОП можлива за рахунок формальної верифікації специфікацій. Зазначені перевірки спрямовані на встановлення відповідності проектованої SOA-системи заданим вимогам. Підтвердження цього шляхом сумісного проведення модульних тестувань та формальної верифікації, однак, не дозволяє формулювати судження відносно задовільності функціонування системи в реальних умовах. Така потреба виникає у зв'язку із доцільністю врахування при проектуванні також і нефункціональних характеристик сервісів: вартості, часу відгуку, імовірності безвідмовної роботи тощо.

Врахування названих характеристик при проектуванні можливе за рахунок проведення валідації, реалізованої шляхом імітаційного моделювання. Для цього створено багато математичних апаратів. Серед них – формалізм DEVS Б. Зейглера (Bernard P. Zeigler) [4], теорія часових автоматів (ТА) [5], ланцюги Маркова, мережі Петрі та ін. При цьому актуальним, на нашу думку, є залучення одного з таких формалізмів до процесу проектування CWS для зменшення супутніх розробці часових витрат. У зв'язку із цим поняття верифікації та валідації будемо розглядати в роботі як взаємодоповнюючі [6].

Мета роботи – розробка методики валідації CWS, яка дозволяла би здійснювати перевірку придатності такої системи до цільового використання при проектуванні після проведення формальної верифікації.

**1 ПОСТАНОВКА ЗАДАЧІ**

Для досягнення сформульованої мети в роботі вирішується задача синтезу імітаційної DEVS-моделі CWS з вихідної формальної TLA-специфікації функціональних характеристик.

Нехай маємо формальну TLA-специфікацію композитного сервісу, який складається із $m$ атомарних сервісів [7]. Представимо таку специфікацію аналітично структурою Кріпке на множині атомарних висловлювань $AP$ [8]:

$$\langle S, \{s_0\}, R, L \rangle, \tag{1}$$

де $s_0 \in S$, $R \subseteq S^2$, $L : S \to 2^{AP}$.

Множину $AP$ при цьому сформуємо із елементів множини $V \times D$, де $V = \{v_i | i = 1,2,...,m\}$, а $D = \{0,1,2\}$: $0 \in D$ – AWS простоює (обчислення ще не почато), $1 \in D$ – AWS функціонує (обчислення виконується), $2 \in D$ – AWS простоює (обчислення завершено).

Тоді вирішувана задача полягає в одержанні на основі структури (1) структури

$$\langle IP, OP, Atomics, set, break \rangle, \tag{2}$$

яка є представленням результуючої складеної DEVS-моделі CWS як засобу валідації [9].

Вирішення поставленої задачі вважатимемо успішним, якщо синтезована DEVS-модель композитного сервісу буде адекватною, а результати валідації на основі цієї моделі – достовірними. За показник достовірності візьмемо близькість результатів валідації шляхом імітаційного моделювання та шляхом тестування. При цьому валідація шляхом моделювання має бути менш витратною за часом.

**2 ОГЛЯД ЛІТЕРАТУРИ**

Темпоральну логіку TLA Л. Лемпорта (Leslie Lamport) обрано у якості формалізму з позицій достатності виразних можливостей та зручності внесення змін до специфікації CWS [10]. Тобто кожний визначений сценарій ОП, що зумовлює відповідну функціональну характеристику CWS, можна подати на основі TLA окремою темпоральною формулою [11].

Вже існують певні спроби сполучення TLA- та DEVS-формалізмів [12]. Вони базуються на твердженні про активну адаптацію науковою спільнотою можливостей TLA з метою верифікації. Акцентується увага, що DEVS-моделі є малопоширеними, проте більш гнучкими та наочними. При цьому обґрунтовується важливість синтезу TLA-специфікацій для DEVS-моделей.

У порівнянні із цим, DEVS-модель в нашій роботі розглядається саме як засіб валідації, похідний від TLA-специфікації – засобу верифікації. Обґрунтування доцільності використання названого засобу валідації було здійснене нами раніше [13].

## 3 МАТЕРІАЛИ ТА МЕТОДИ

Визначимо поняття події, що ініціює зміну стану системи переходів (1), тобто спричиняє деякий перехід $(s, s') \in R$, у двох інтерпретаціях: з позиції вихідної формальної TLA-специфікації функціональних характеристик CWS та з позиції результуючої синтезованої DEVS-моделі. У першому випадку – це координуючий вплив на атомарний сервіс у складі CWS з метою просування ОП згідно певного сценарію. У другому випадку – це надходження повідомлення активації моделі AWS на відповідний вхідний порт з метою виконання окремої обчислювальної процедури у межах ОП.

Композитний сервіс розглядатимемо як систему, а атомарний сервіс у його складі – як компонент системи. Розгляд AWS як компонента (а не елемента) зумовлено наявністю в нього інтерфейсу (множини методів або функцій), з якими можна взаємодіяти безпосередньо. Зауважимо, що при організації окремого ОП ми асоціюватимемо із AWS тільки один елемент його інтерфейсу.

Визначення події у двох інтерпретаціях (TLA- та DEVS-контекстах) необхідне для встановлення співвідношення (один до одного) між вхідними та результуючими концептами, відповідно. Визначення події в TLA-контексті розкриває рівень деталізації вихідної специфікації. Це означає, що при верифікації нас не цікавить питання виявлення помилок, пов'язаних із специфікою роботи окремого AWS самого по собі (виокремлено від системи). Це питання, згідно нашого підходу до проектування, має бути вирішене раніше – шляхом модульного тестування, передуючи здійсненню верифікації (рис. 1). Тобто при верифікації вже оперуємо припущенням, що компоненти системи функціонують безвідмовно та безпомилково.

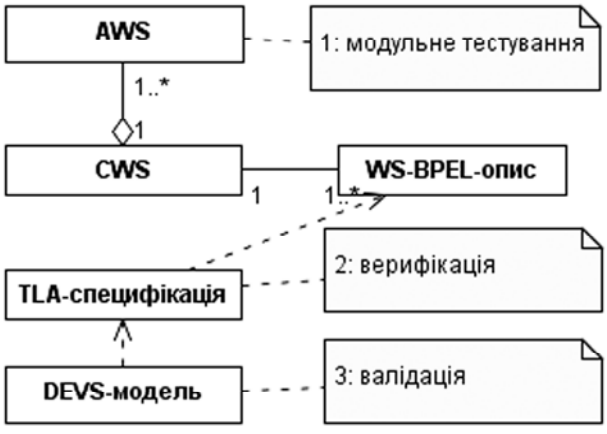


Рисунок 1 – Запропонований підхід до проектування композитних сервісів

На рис. 1 для встановлення зв'язку між сутностями AWS та CWS нами використано відношення агрегації (а не композиції) мови UML. Мета цього – підкреслити автономність атомарного сервісу, який у загальному випадку може входити до складу й інших композитних сервісів. Передумовою до здійснення верифікації є успішність модульного тестування кожного із компонентів системи. Під модулем при цьому розуміється окремий елемент інтерфейсу атомарного сервісу. Певний набір із елементів різних AWS (по одному від кожного) із встановленим порядком виклику методів задається згідно стандарту WS-BPEL [14]. Названий стандарт регламентує взаємодію компонентів композитного сервісу згідно моделі централізованого координування – моделі оркестровки.

Отже, розглядатимемо композитний сервіс як поліморфну сутність, функціональні характеристики якої визначаються наступними факторами – складом задіяних атомарних сервісів, інтерфейсами останніх, а також порядком виклику (координування) елементів інтерфейсів. Зазначені фактори враховуються у TLA-специфікації. Специфікація, у свою чергу, формується з одного або декількох WS-BPEL-описів (рис. 1). Кожний такий опис задає сценарій ОП для одержання відповідної функціональної характеристики системи. Це означає, що TLA-специфікація є формальною моделлю в загальному випадку декількох сценаріїв функціонування CWS [15]. Кожний із сценаріїв характеризується сукупністю темпоральних ознак. Для встановлення цих ознак нами використано темпоральний оператор $X$. Ознаки представлено у якості подій, що ініціюють продовження функціонування CWS згідно певного сценарію. Події формалізуємо в імплікативній формі, модифікованій оператором [7]:

$$\neg(v_i = d) \vee X(v_i = d+1),\ d = 0,1;\ d \in D. \qquad (3)$$

Враховуючи специфіку виразних можливостей TLA, події запропонуємо класифікувати на значимі події, псевдоподії та неприпустимі події. Значимі події представлено виразом (3). Вони є засобами формалізації переходів $(s, s') \in R$ структури (1).

Значиму подію охарактеризуємо симетричною різницею

$$L(s) \Delta L(s') = \{(v_i = d), (v_i = d+1)\} \subseteq AP, \qquad (4)$$

де розмітки станів $L(s)$ та $L(s')$ є, відповідно, перед- та постумовами виникнення події.

Потреба фіксації в специфікації також і псевдоподій зумовлена потребою просування модельного часу для DEVS-моделей атомарних сервісів, які при валідації продовжують або функціонувати, або простоювати:

$$\neg(v_i = d) \vee X(v_i = d),\ d = 0,1,2. \qquad (5)$$

Вираз (5) названо нами шаблоном псевдоподії, бо, попри імплікативну форму, він вже не характеризується множиною (4). В (3) $v_i \in V$ змінює своє значення в наступний момент модельного часу, а в (5) – ні.

Наприклад, якщо розглядати сценарій послідовних координуючих впливів на $m$ компонентів CWS (тобто

$|V| = m$), то для просування ОП в формальній моделі кожна значима подія має супроводжуватися $m-1$ псевдоподією.

Ґрунтуючись на вищезазначеному, запропонуємо формат представлення концептів значимої події та псевдоподії в TLA-специфікації:

$$\left(IF\ Cnd\ THEN\ \left(v_i' := v_i + 1\right) ELSE\ \left(v_i' := v_i\right)\right) \wedge$$
$$\wedge UNCHANGED << v_1, v_2, ..., v_{i-1}, v_{i+1}, ..., v_m >>. \quad (6)$$

Верифікація специфікацій на основі конструкцій (6) полягає у виявленні недопустимих подій – подій, що ініціюють недопустимі переходи структури (1):

$$\neg\left(v_i = d\right) \vee X\left(v_i = d-1\right),\ d = 1, 2. \quad (7)$$

Варто відзначити, що події (7) також характеризуються множинами (4). Відмінність їх від подій (3) полягає в наступному: передумови для (3) є постумовами для (7) і навпаки.

За умови, якщо формальну верифікацію специфікації на основі конструкцій (6) здійснено успішно – встановлено, що атомарні сервіси функціонують узгоджено та несуперечливо (коректно), переходимо до валідації (рис. 1). При цьому шляхом імітаційного DEVS-моделювання здійснимо оцінювання показників функціональних характеристик CWS, а також агрегованих значень показників нефункціональних характеристик. Для цього синтезуємо імітаційну DEVS-модель (2), архітектура та поведінка якої визначаються змістом вихідної TLA-специфікації. З цією метою запропонуємо правила синтезу названої моделі:

1) правила синтезу базових моделей AWS (табл. 1);

2) правила синтезу допоміжних базових моделей – моделі клієнта CWS (генератора заявок на обслуговування) та моделі координатора атомарних сервісів у складі CWS – для слідування моделі оркестровки згідно стандарту WS-BPEL (табл. 2);

3) правила синтезу результуючої складеної моделі (2) CWS із залученням базових та допоміжних базових моделей (табл. 3).

Результатом використання правил 1) мають бути $m$ структур

$$MA_i = \left\langle \{ev\}, \{rs\}, ST, \delta_{ext}^{MA_i}, \delta_{\text{int}}^{MA_i}, \lambda^{MA_i}, ta \right\rangle. \quad (8)$$

Елементи множини $ST = \{st_0, st_1, st_2\}$ структури (8) є представленнями елементів множини $D$ допустимих значень змінних станів, з яких формуються елементи множини літералів $AP$ структури (1). Це означає, що мітка $st_0 \in ST$ – представлення $0 \in D$, $st_1 \in ST$ – $1 \in D$, а $st_2 \in ST - 2 \in D$. Базові моделі на основі структур (8) реалізуємо як двопортові:

– вхідний порт – для одержання повідомлення $ev = event_i$, що ініціює однойменну подію активації, від моделі координатора;

– вихідний порт – для відправлення повідомлення-результату обчислень $rs = res_i$ до моделі координатора.

Результатами використання правил 2) мають бути структура

$$MG = \left\langle \{activate\}, \{job_j\}, ST, \delta_{ext}^{MG}, \delta_{\text{int}}^{MG}, \lambda^{MG}, ta \right\rangle \quad (9)$$

та структура

$$MC = \left\langle \{job_j\} \cup \{res_i\}, \{event_i\}, ST, \delta_{ext}^{MC}, \delta_{\text{int}}^{MC}, \lambda^{MC}, ta \right\rangle. \quad (10)$$

Результатом використання правил табл. 3 має бути шукана структура (2).

Отже, результуюча DEVS-модель CWS включає $m$ моделей атомарних сервісів, модель клієнта та модель координатора (рис. 2).

Таблиця 1 – Правила синтезу DEVS-моделей атомарних сервісів

| Мета | | Вихідні TLA-конструкції | Результуючі дії над DEVS-елементами |
|---|---|---|---|
| Фіксація | значимих подій | на основі (3) | виклик функцій $ta: ST \to R_{0,\infty}^{+}$ та:<br>– при $d = 0$ – $\delta_{ext}^{MA_i}: (event_i, st_0, e) \mapsto st_1$;<br>– при $d = 1$ – $\delta_{\text{int}}^{MA_i}: st_1 \mapsto st_2$; $\lambda^{MA_i}: st_2 \mapsto res_i$. |
| | псевдоподій | на основі (5) | виклик функцій $ta: ST \to R_{0,\infty}^{+}$. |

Таблиця 2 – Правила синтезу DEVS-моделей клієнта та координатора

| Мета | Вихідні TLA-конструкції | Результуючі дії над DEVS-елементами |
|---|---|---|
| Фіксація початкового стану | *Init* | $\delta_{ext}^{MG}: (activate, st_0, e) \mapsto st_1$. |
| Фіксація ОП | *Calc* | $\delta_{ext}^{MC}: (job_j, st_0, e) \mapsto st_1,\ j = 1,2,...,n$. |

Таблиця 3 – Правила одержання елементів результуючої DEVS-моделі CWS

| Мета | Вихідні TLA-конструкції | Результуючі дії над DEVS-елементами |
|---|---|---|
| Фіксація значимих подій | згідно (3) при $d = 0$ | $set: (MC, event_i) \mapsto (MA_i, event_i)$. |
| | згідно (3) при $d = 1$ | $set: (MA_i, res_i) \mapsto (MC, res_i)$. |
| Фіксація ОП | *Calc* | $set: (MG, job_j) \mapsto (MC, job_j)$. |

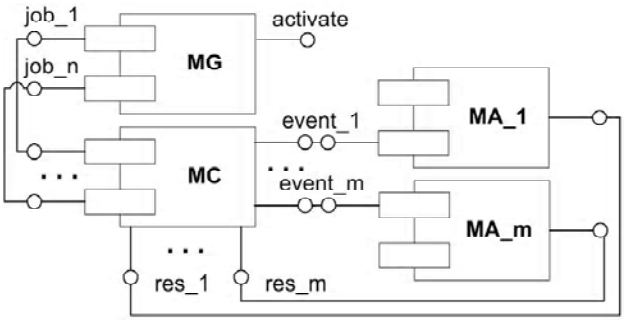


Рисунок 2 – Архітектура синтезованого засобу валідації

Модель, представлена на рис. 2, функціонує наступним чином:

1) на вхідний порт моделі (9) надходить повідомлення активації, що ініціює генерування заявок $job_j$. Кожна $j$-а заявка представляє специфікацію окремої функціональної характеристики CWS. Заявки генеруються із заданим законом розподілу;

2) на $j$-й вхідний порт моделі (10) надходить заявка $job_j$, що зумовлює $m' \le m$ повідомлень $event_i$ (оскільки певні AWS за окремого ОП можуть бути зайвими);

3) кожна із $m$ моделей (8) виконує власну частину обчислень згідно $j$-го ОП. Результати обчислень відправляються до моделі (10) у вигляді повідомлень $res_i$.

## 4 ЕКСПЕРИМЕНТИ

Експериментальні дослідження запропонованої методики здійснено на апаратній платформі наступної конфігурації: частота CPU 3 ГГц, обсяг RAM – 2 ГБ.

Щоб стверджувати відносно достовірності результатів валідації згідно запропонованої методики, а саме – відносно адекватності синтезованої результуючої DEVS-моделі CWS, проведемо експерименти за наступними сценаріями:

1) валідація шляхом тестування – AWS реалізуємо за технологією JAX-WS. Координування здійснимо централізовано – згідно моделі оркестровки. Таку перевірку розглядатимемо як один із завершальних кроків розробки CWS;

2) валідація шляхом моделювання – використаємо результуючу DEVS-модель CWS, синтезовану згідно запропонованих правил, у якості засобу валідації. Така перевірка є завершальним кроком проектування (рис.1).

У першому випадку за вихідні дані візьмемо WS-BPEL-опис CWS, у другому – відповідну формальну TLA-специфікацію. Порівняємо результати валідацій зазначеними шляхами (рис. 3).

На рис. 3 process.bpel – вихідний WS-BPEL-опис; spec.tla – відповідна TLA-специфікація; TLA-to-DEVS – програмна реалізація запропонованих правил синтезу (табл. 1–3); model.java – синтезована результуюча DEVS-модель CWS як засіб валідації; BPEL Engine – програмний засіб (sun-bpel-engine) централізованого координування AWS при валідації шляхом тестування.

Якщо, згідно рис. 3, $\xi_f^t = \xi_f^s$, а $\xi_{nf}^t \approx \xi_{nf}^s$, стверджуватимемо, що результуюча модель (2) є адекватною, а запропонована методика дозволяє одержувати достовірні результати валідації при проектуванні CWS. При цьому вважатимемо використання запропонованої методики при проектуванні доцільним, якщо часові витрати на валідацію за сценарієм 2) будуть істотно меншими за часові витрати за сценарієм 1).

У моделі (9) реалізуємо нормальний закон розподілу заявок на обслуговування, що генеруються. DEVS-моделі реалізуємо на мові програмування Java, а моделювання здійснюватимемо у середовищі DEVS Suite.

Вихідні значення для розрахунку часових витрат одержуватимемо як середнє арифметичне $10^2$ замірів. У якості нефункціональної характеристики розглядатимемо час відгуку сервісу. Для AWS такі затримки визначимо послідовністю $0, 1\cdot10^{-1}, 2\cdot10^{-1},..., 5\cdot10^{-1}$ с: 0 – компоненти розгорнуто локально, $1\cdot10^{-1}$ – вони є рівновіддаленими (значно), $2\cdot10^{-1},..., 5\cdot10^{-1}$ – окрім територіальної віддаленості моделюються також додаткові затримки, зумовлені інтенсивними обчисленнями. Затримки реалізуємо на основі Java-методу sleep.

Для перевірки методики розглянемо синтетичний сценарій обчислення значення $\pi$ через арктангенси [16]: $\pi = 48\cdot\mathrm{arctg}\left(\frac{1}{18}\right) + 32\cdot\mathrm{arctg}\left(\frac{1}{57}\right) - 20\cdot\mathrm{arctg}\left(\frac{1}{239}\right)$. Цей вираз розглядатимемо як функціональну характеристику CWS, а значення-результат його обчислення – як шукане значення показника такої характеристики. Вираз також розглядатимемо як аналітичне представлення ОП. Для реалізації характеристики залучимо чотири AWS: три з них – для обчислення арктангенсів, четвертий – для одержання результуючого значення (рис. 4).

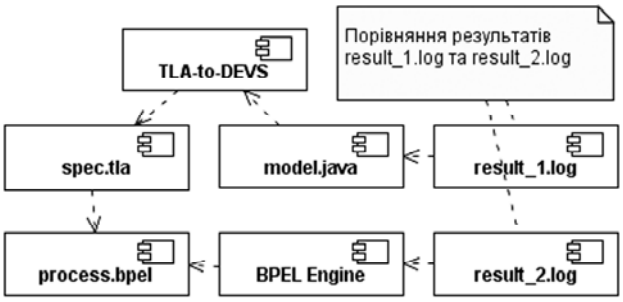


Рисунок 3 – Схема підходу до перевірки методики

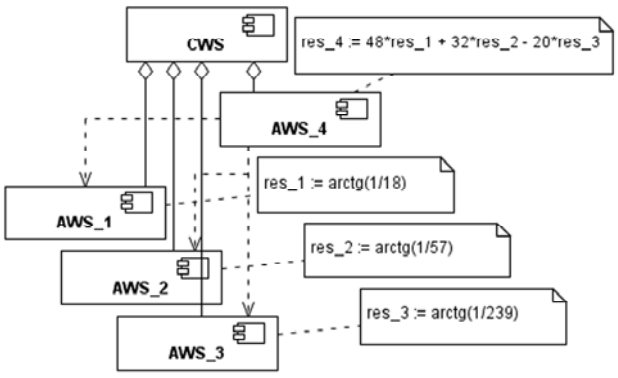


Рисунок 4 – Архітектура композитного сервісу

На рис. 4 $res_1$, $res_2$ та $res_3$ – часткові результати – артефакти протікання ОП – контейнери показників функціональних характеристик атомарних сервісів $AWS_1$, $AWS_2$ та $AWS_3$, відповідно; $res_4$ – контейнер показника функціональної характеристики атомарного сервісу $AWS_4$, який також є показником названої характеристики CWS – контейнером шуканого результату, що залежить від значень $res_1$, $res_2$ та $res_3$.

## 5 РЕЗУЛЬТАТИ

В результаті проведення експериментальних досліджень встановлено, що на визначеному діапазоні вхідних даних часові витрати на валідацію шляхом моделювання згідно запропонованої методики в середньому у 3,4 рази менші за витрати, пов'язані із валідацією шляхом тестування (рис. 5). Таке зменшення часових витрат нами охарактеризоване як суттєве.

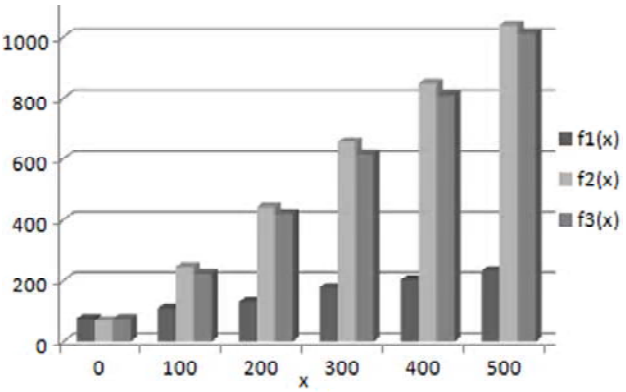


Рисунок 5 – Результати проведених експериментальних досліджень

На рис. 5 значення наведено в мс: $x$ – заданий час відгуку кожного із чотирьох AWS у складі CWS; $f_1(x)$ – часові витрати на моделювання на основі результуючої моделі (2); $f_2(x)$ – оціночні значення показника $\xi_{nf}^s$; $f_3(x)$ – фактичні значення показника $\xi_{nf}^t$. При цьому перевірено рівність $\xi_f^t = \xi_f^s$, що підтвердило коректність запропонованих правил синтезу. Перевірку адекватності результуючої DEVS-моделі CWS здійснено на основі статистичних критеріїв $t$ та $F$ для довірчої імовірності 0,95.

## 6 ОБГОВОРЕННЯ

Попри те, що валідація шляхом моделювання згідно запропонованої методики виглядає більш пріоритетно за валідацію шляхом тестування за показником супутніх часових витрат, не менш важливим вбачається питання корисності такої перевірки. Корисність, у свою чергу, варто, на нашу думку, розглядати як певний компроміс за трьома наступними критеріями: адекватність результуючої DEVS-моделі композитному сервісу, достовірність результатів валідації та зменшення часових витрат. Останній критерій набуває особливої значимості у контексті актуальної на сьогодні ітераційної розробки. При цьому нами розглядається доцільність збільшення саме повноти відповідності TLA-специфікацій WS-BPEL-описові сервісу, що, зокрема, підвищить достовірність результатів формальної верифікації. Це, проте, ускладнить як саму запропоновану методику, так і інтеграцію її до процесу проектування.

З позиції ітераційної розробки перспективним може бути використання комбінованого підходу до валідації, за якого, наприклад, на першій та заключній ітераціях здійснюється перевірка шляхом тестування, а на інших – шляхом моделювання. Моделювання дозволить зменшити загальні часові витрати, а тестування – одержати більш достовірні вихідні дані показників нефункціональних характеристик AWS та підтвердити придатність розробленого CWS до цільового використання.

## ВИСНОВКИ

Таким чином, в роботі вирішено актуальну науково-прикладну задачу синтезу імітаційної DEVS-моделі CWS з вихідної формальної TLA-специфікації функціональних характеристик.

Наукова новизна отриманих в роботі результатів полягає в наступному: було запропоновано DEVS-орієнтовану методику валідації CWS, призначену до використання при проектуванні. Це дозволило зменшити супутні валідації часові витрати, що було підтверджено результатами проведених експериментальних досліджень. Вони показали, що на тестовому діапазоні значень нефункціональних характеристик компонентів CWS синтезована результуюча DEVS-модель є адекватною, а правила її одержання – коректними: у порівнянні із валідацією шляхом тестування, валідація шляхом імітаційного моделювання згідно запропонованої методики супроводжувалася зменшенням часових витрат в середньому у 3,4 рази. Таке зменшення охарактеризоване нами як суттєве. Отже, поставлену задачу можна вважати вирішеною.

Практичне значення отриманих результатів полягає в автоматизації процедури валідації CWS при проектуванні згідно запропонованої методики за рахунок програмної реалізації запропонованих правил синтезу DEVS-моделей.

Перспективи подальших досліджень полягають у адаптації запропонованої методики до більш деталізованих вихідних формальних TLA-специфікацій CWS, що, за рахунок поліпшення повноти таких специфікацій, дозволить одержувати більш достовірні результати валідації.

## ПОДЯКИ

Роботу виконано в рамках науково-дослідної роботи «Дослідження і розробка методів підвищення ефективності комп'ютерних систем та мереж, пошук шляхів удосконалення навчального процесу» кафедри комп'ютерних систем та мереж Запорізького національного технічного університету.

## СПИСОК ЛІТЕРАТУРИ

1 Service-Oriented Computing: State of the Art and Research Challenges / [M. P. Papazoglou, P. Traverso, S. Dustdar, F. Leymann] // IEEE Computer. – 2007. – Vol. 40, No. 11. – P. 38–45. DOI: 10.1109/MC.2007.400.

2. Vohra D. Java 7 JAX-WS Web Services: A practical, focused mini book for creating Web Services in Java 7 / D. Vohra. – Birmingham-Mumbai: Packt Publishing Ltd., 2012. – 64 p.

3. Freeman S. Growing Object-Oriented Software, Guided by Tests / S. Freeman, N. Pryce. – New York : Addison-Wesley, 2010. – 384 p.

4. Wainer G. A. Discrete-Event Modeling and Simulation: Theory and Applications / G. A. Wainer, P. J. Mosterman. – New York : CRC Press, 2010. – 534 p.

5. Kindermann R. Beyond Lassos: Complete SMT-Based Bounded Model Checking for Timed Automata / R. Kindermann, T. Junttila, I. Niemela // Lecture Notes in Computer Science. Formal Techniques for Distributed Systems. – 2012. – Vol. 7273. – P. 84–100. DOI: 10.1007/978-3-642-30793-5_6.
6. Системи управління якістю. Основні положення та словник термінів (ISO 9000:2005, IDT) : ДСТУ ISO 9000:2007. – [Чинний від 2008-01-01]. – К.: Держспоживстандарт України. – 29 с. – (Національний стандарт України).
7. Шкарупило В. В. Модель TLA-спецификации композитного веб-сервиса с множеством динамик / В. В. Шкарупило // Радіоелектроніка, інформатика, управління. –2013. – № 1. – С. 94–100. DOI: 10.15588/1607-3274-2013-1-15.
8. Карпов Ю. Г. MODEL CHECKING. Верификация параллельных и распределенных программных систем / Ю. Г. Карпов. – СПб. : БХВ-Петербург, 2010. – 560 с.
9. Шкарупило В. В. DEVS-модель как средство валидации композитных веб-сервисов распределенной системы / В. В. Шкарупило, С. Ю. Скрупський, Р. К. Кудерметов // Комп'ютерно-інтегровані технології: освіта, наука, виробництво. – 2011. – № 7. – С. 61–67.
10. Lamport L. Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers / L. Lamport. – Boston: Addison-Wesley, 2002. – 364 p.
11. Шкарупило В. В. WS-BPEL-модификация метода TLC-верификации / В. В. Шкарупило // Східно-європейський журнал передових технологій. – Харків: НВП «Технологічний центр», 2013. – № 4/2 (64). – С. 23–28.
12. Cristia M. A TLA+ Encoding of DEVS Models / M. Cristia // Proc. Int. Modeling and Simulation Multiconference (Buenos Aires, Argentina, February 8–10, 2007). – P. 17–22.
13. Шкарупило В. В. Сравнительный анализ подходов к реализации процесса автоматизированного синтеза композитных веб-сервисов / В. В. Шкарупило, Р. К. Кудерметов // Науковий вісник Чернівецького національного університету імені Юрія Федьковича. Серія: комп'ютерні системи та компоненти. – Чернівці: ЧНУ, 2011. – Т. 2, № 4. – С. 80–85.
14. Web Services Business Process Execution Language Version 2.0 [Electronic resource] : OASIS Standard, April 11, 2007. – Access mode: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf. – Title from screen.
15. Шкарупило В. В. Концептуальная модель процесса автоматизированного синтеза композитных веб-сервисов / В. В. Шкарупило, Р. К. Кудерметов, Т. А. Паромова // Наукові праці Донецького національного технічного університету. Серія: Інформатика, кібернетика та обчислювальна техніка. – Донецьк : ДонНТУ, 2012. – № 15 (203). – С. 231–238.
16. Шкарупило В. В. Методика автоматизированного синтеза композитных веб-сервисов / В. В. Шкарупило, Р. К. Кудерметов / / Інформатика і комп'ютерні технології : VII Міжнар. наук.-техн. конф. студентів, аспірантів та молодих учених, 22–23 листопада 2011 р. : тези доп. – Донецьк : ДонНТУ, 2011. – Т. 1. – С. 382–384.

Стаття надійшла до редакції 11.02.2014.
Після доробки 23.02.2015.

Шкарупило В. В.[1], Кудерметов Р. К.[2], Польская О. В.[3]

[1]Канд. техн. наук, старший преподаватель кафедры компьютерных систем и сетей Запорожского национального технического университета, Запорожье, Украина

[2]Канд. техн. наук, доцент, заведующий кафедрой компьютерных систем и сетей Запорожского национального технического университета, Запорожье, Украина

[3]Старший преподаватель кафедры компьютерных систем и сетей Запорожского национального технического университета, Запорожье, Украина

**DEVS-ОРИЕНТИРОВАННАЯ МЕТОДИКА ВАЛИДАЦИИ КОМПОЗИТНЫХ ВЕБ-СЕРВИСОВ**

Предложена методика валидации композитных веб-сервисов путем синтеза имитационных дискретно-событийных моделей на основе формализма DEVS. Это позволяет осуществлять автоматизированную проверку пригодности таких систем к целевому использованию при проектировании путем имитационного моделирования. В качестве входных данных использована формальная спецификация на основе темпоральной логики TLA, что позволяет математически строго представлять функциональные характеристики композитных сервисов в формате вычислительных процессов. В качестве аналитической модели в основе TLA-спецификации взята структура Крипке. В рамках методики предложены правила синтеза из исходной TLA-спецификации DEVS-модели композитного сервиса, предназначенной быть средством валидации. Результирующая DEVS-модель состоит из моделей атомарных сервисов, модели клиента композитного сервиса и модели координатора атомарных сервисов, функционирующего согласно спецификации WS-BPEL.

Для проверки методики проведены экспериментальные исследования, подтвердившие адекватность результирующей DEVS-модели. Проверка выполнена согласно предложенному подходу, состоящему в сравнении результатов валидации путем имитационного моделирования с результатами валидации путем тестирования. По результатам проведенных исследований обоснована целесообразность использования предложенной методики при проектировании композитных сервисов, которая заключается в уменьшении временных затрат на валидацию. Акцентировано внимание на уместности использования методики при итерационной разработке.

**Ключевые слова:** SOA, WS-BPEL, композитный веб-сервис, спецификация, верификация, валидация, TLA, DEVS.

Shkarupylo V. V.[1], Kudermetov R. K.[2], Polska O. V.[3]

[1]PhD, Senior Lecturer of Computer Systems and Networks department, Zaporizhzhya National Technical University, Zaporizhzhya, Ukraine

[2]PhD, Associate Professor, Head of Computer Systems and Networks department, Zaporizhzhya National Technical University, Zaporizhzhya, Ukraine

[3]Senior Lecturer of Computer Systems and Networks department, Zaporizhzhya National Technical University, Zaporizhzhya, Ukraine

**DEVS-ORIENTED TECHNIQUE FOR COMPOSITE WEB SERVICES VALIDITY CHECKING**

A technique for Composite Web Services validity checking has been proposed. It is based on discrete-event DEVS-models synthesis, which provides the ability to conduct the automated validation by way of simulation during the design process. Temporal Logic of Actions has been chosen as the basis for input data – formal specification of Composite Web Service. It allows to specify the functional properties of such systems mathematically strictly. Functional properties has been represented as computational processes. The Kripke structure has been used as TLA-specification analytical model. Our technique leans on the proposed rules, aimed at simulation DEVS-model synthesis from given TLA-specification. The resulting coupled Composite Web Service DEVS-model consists of atomic web services models, model of client, simulated as job-requests generator, and coordinator model. Coordinator represents the WS-BPEL-engine, functioning in accordance with centralized orchestration model.

A case study has been conducted to verify the proposed technique. Its artifacts confirmed the adequacy of resulting DEVS-model. The technique verification is based on the proposed approach: simulation-driven validation results are compared with the ones, obtained with test-driven validation. Technique expediency has been grounded by Composite Web Services validity checking time costs reduction.

**Keywords:** SOA, WS-BPEL, Composite Web Service, Specification, Verification, Validation, TLA, DEVS.

## REFERENCES

1. Papazoglou M. P., Traverso P., Dustdar S., Leymann F. Service-Oriented Computing: State of the Art and Research Challenges. *IEEE Computer*, 2007, Vol. 40, No. 11, P. 38–45. DOI: 10.1109/MC.2007.400.
2. Vohra D. Java 7 JAX-WS Web Services: A practical, focused mini book for creating Web Services in Java 7. Birmingham-Mumbai, Packt Publishing Ltd., 2012, 64 p.
3. Freeman S., Pryce N. Growing Object-Oriented Software, Guided by Tests. New York, Addison-Wesley, 2010, 384 p.
4. Wainer G. A., Mosterman P. J. Discrete-Event Modeling and Simulation: Theory and Applications. New York, CRC Press, 2010, 534 p.
5. Kindermann R., Junttila T., Niemela I. Beyond Lassos: Complete SMT-Based Bounded Model Checking for Timed Automata, *Lecture Notes in Computer Science. Formal Techniques for Distributed Systems*, 2012, Vol. 7273, pp. 84–100. DOI: 10.1007/978-3-642-30793-5_6.
6. Systemy upravlinnia yakistiu. Osnovni polozhennia ta slovnyk terminiv (ISO 9000:2005, IDT) : DSTU ISO 9000:2007. [Chynnyi vid 2008-01-01]. Kiev, Derzhspozhyvstandart Ukrainy, 29 p. (Natsionalnyi standart Ukrainy).
7. Shkarupylo V. V. Model' TLA-specifikacii kompozitnogo veb-servisa s mnozhestvom dinamik, *Radioelektronika, informatyka, upravlinnia,* 2013, No. 1, pp. 94–100. DOI: 10.15588/1607-3274-2013-1-15.
8. Karpov Yu. G. MODEL CHECKING. Verifikaciya parallel'nyx i raspredelennyx programmnyx sistem. Sankt-Peterburg, BXV-Peterburg, 2010, 560 p.
9. Shkarupylo V. V., Skrupskyi S. Iu., Kudermetov R. K. DEVS-model' kak sredstvo validacii kompozitnyx veb-servisov raspredelennoj sistemy, *Kompiuterno-intehrovani tekhnolohii: osvita, nauka, vyrobnytstvo*, 2011, No. 7, pp. 61–67.
10. Lamport L. Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers. Boston, Addison-Wesley, 2002, 364 p.
11. Shkarupylo V. V. WS-BPEL-modifikaciya metoda TLC-verifikacii, *Eastern-European Journal of Enterprise Technologies*. Kharkiv, NVP «Tekhnolohichnyi tsentr», 2013, No. 4/2 (64), pp. 23–28.
12. Cristia M. A TLA+ Encoding of DEVS Models, *Proc. Int. Modeling and Simulation Multiconference (Buenos Aires, Argentina, February 8–10, 2007),* pp. 17–22.
13. Shkarupylo V. V., Kudermetov R. K. Sravnitel'nyj analiz podxodov k realizacii processa avtomatizirovannogo sinteza kompozitnyx veb-servisov, *Naukovyi visnyk Chernivetskoho natsionalnoho universytetu imeni Yuriia Fedkovycha. Seriia: kompiuterni systemy ta komponenty*. Chernivtsi, ChNU, 2011, Vol. 2, No. 4, pp. 80–85.
14. Web Services Business Process Execution Language Version 2.0 [Electronic resource] : OASIS Standard, April 11, 2007. Access mode: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf. Title from screen.
15. Shkarupylo V. V., Kudermetov R. K., Paromova T. A. Konceptual'naya model' processa avtomatizirovannogo sinteza kompozitnyx veb-servisov, *Naukovi pratsi Donetskoho natsionalnoho tekhnichnoho universytetu. Seriia: Informatyka, kibernetyka ta obchysliuvalna tekhnika*. Donetsk, DonNTU, 2012, No. 15 (203), pp. 231–238.
16. Shkarupylo V. V., Kudermetov R. K. Metodika avtomatizirovannogo sinteza kompozitnyx veb-servisov, *Informatyka i kompiuterni tekhnolohii : VII Mizhnar. nauk.-tekhn. konf. studentiv, aspirantiv ta molodykh uchenykh, 22–23 lystopada 2011 r. : tezy dop.* Donetsk, DonNTU, 2011, Vol. 1, pp. 382–384.