АВТОМАТИКА, КОМП'ЮТЕРН І ТА ТЕЛЕКОМУНІКАЦІЙНІ ТЕХНОЛОГІЇ

УДК 621.391:681.5

Моделювання процесів управління наданням інтелектуальних сервісів в NGN

***С. В. Шестопалов*[1], *Т. В. Кунуп*[2], *Б. Л. Пустовий*[3]**
Одеська національна академія харчових технологій, вул. Канатна 112, Одеса, 65082, Україна
✉ e-mail: [1] sshestopalov1984@gmail.com; [2] ttvv@ukr.net; [3] b.pustoviy@gmail.com,
ORCID: [1] http://orcid.org/0000-0001-8941-4610; [3] https://orcid.org/0000-0002-3181-8062

Стаття присвячена розробці імітаційних моделей процесів управління наданням інтелектуальних сервісів в NGN. Показано, що для визначення ефективності управління наданням інтелектуальних сервісів різними авторами запропонована низка аналітичних моделей інтелектуальних надбудов як без врахування, так і з врахуванням самоподібності потоку заявок на інтелектуальні сервіси. При цьому зазначається, що створення аналітичних моделей при суттєвому ускладненні інтелектуальних надбудов стає практично неможливим. В такому випадку пропонується використовувати імітаційні моделі. Запропоновано інтелектуальну надбудову представити у вигляді системи масового обслуговування. Для моделювання інтелектуальних надбудов NGN у вигляді систем масового обслуговування без врахування самоподібності потоку заявок на інтелектуальні сервіси було прийнято рішення скористатися системою GPSS. Для моделювання інтелектуальних надбудов з урахуванням самоподібності потоку заявок пропонується використати систему «Network Simulator-2» (NS-2). В роботі представлено алгоритми функціонування та імітаційні моделі інтелектуальної надбудови, що надає один клас заявок без урахування самоподібності трафіку, інтелектуальної надбудови, що надає три класи заявок з відносними пріоритетами та без урахування самоподібності трафіку, і інтелектуальної надбудови з урахуванням самоподібності потоку заявок на інтелектуальні сервіси. Використовуючи імітаційні моделі без урахування та з урахуванням самоподібності трафіку для інтелектуальної надбудови, що обслуговує один клас заявок на інтелектуальні сервіси, визначені показники якості надаваних інтелектуальних сервісів, що надає можливість отримати оцінку ефективності управління наданням інтелектуальних сервісів. Запропоновані імітаційні моделі дозволять ще на етапі проектування NGN визначити ефективність управління наданням інтелектуальних сервісів інтелектуальною надбудовою.

Ключові слова: *NGN; інтелектуальний сервіс; ефективність управління; показники якості; самоподібність; GPSS; NS-2; імітаційна модель.*

© The Author(s) 2018. This article is an open access publication
This work is licensed under the Creative Commons Attribution 4.0 International License (CC BY)
http://creativecommons.org/licenses/by/4.0/

1 Вступ

Розвиток телекомунікацій в світі забезпечив перехід до мультисервісних мереж, таких як *NGN* (*Next Generation Network* – мережа наступного покоління), *FN* (*Future Network* – мережа майбутнього) та нова специфікація передачі мультимедійного вмісту в електрозв'язку на основі протоколу *IP – IMS* (*IP Multimedia Subsystem*). Виникає поняття потрійної послуги – *Triple-Play Services* та інтелектуальних сервісів (ІС) – сервісів, котрі спроможна надавати інтелектуальна надбудова *NGN*. *NGN* – концепція побудови мереж зв'язку, що забезпечує надання необмеженого набору сервісів з гнучкими можливостями по їх управлінню, персоналізації і створенню нових сервісів за рахунок уніфікації мережних рішень [1].

Попит на сучасні ІС постійно зростає у всіх сферах людської діяльності. При цьому істотно підвищуються вимоги до якості сервісів. Особливе значення це завдання набуває в різних сферах промисловості, де необхідно здійснювати постійний контроль та моніторинг за технологічними процесами в режимі реального часу. Що стосується енергетики, в цій сфері необхідно забезпечити передачу з заданим рівнем якості даних випробувань, телеметричної інформації від автономних пристроїв, інформації про стан обладнання, в тому числі тривимірних зображень, забезпечити можливість мережної взаємодії і контролю за станом навколишнього середовища для негайного прийняття рішень. Тому виникає необхідність в оцінюванні ефективності управління наданням ІС.

Питання стосовно оцінювання ефективності управління наданням сервісів та розрахунку мережних показників якості обслуговування зазвичай базуються на теорії телетрафіку. Значний внесок в розвиток зазначеного напрямку зробили такі вчені: Г. Башарін, Б. Лівщиц, В. Саморєзов, Д. Потапов, В. Вишнєвський, С. Самохвалов, В. Крилов, У. Айверсен, Л. Клейнрок,

П. Кюн, А. Росляков [2], Д. Агеєв [3, 4], Б. Цибаков [5], Н. Князєва [7, 8], M. Taqqu, B. Willinger, D. Wilson та інші.

Для визначення ефективності управління наданням ІС в роботах [2-8] запропоновані аналітичні моделі інтелектуальних надбудов як без врахування, так і з врахуванням самоподібності потоку заявок на ІС. При цьому зазначається, що створення аналітичних моделей при суттєвому ускладненні інтелектуальних надбудов стає практично неможливим.

В такому випадку набуває актуальності питання використання імітаційного моделювання інтелектуальних надбудов для визначення ефективності управління наданням ІС.

2 Розробка імітаційних моделей

У даній роботі надані основні підходи для організації імітаційного моделювання процесів управління наданням ІС інтелектуальними надбудовами *NGN* та наведено результати, що отримані при імітаційному моделюванні.

Існує можливість представлення інтелектуальної надбудови у вигляді системи масового обслуговування (СМО) [8]. Аналіз існуючих інструментів імітаційного моделювання показав, що для моделювання інтелектуальних надбудов у вигляді СМО доцільно скористатися *GPSS, AnyLogic* або *NS*2.

GPSS (*General Purpose Simulation System*) – загально-цільова система імітаційного моделювання, призначена для розробки моделей складних систем з дискретним і безперервним характером функціонування.

NS (*network simulator*) – назва ряду мережних симуляторів дискретних подій, зокрема, *NS*-1, *NS*-2 та *NS*-3. Всі вони є дискретно-подієвими комп'ютерними мережними тренажерами.

AnyLogic – програмне забезпечення для імітаційного моделювання бізнес-процесів. Інструмент забезпечений сучасним графічним інтерфейсом та дозволяє використовувати мову програмування *Java* для розробки моделей.

При моделюванні інтелектуальних надбудов (ІН) без врахування самоподібності потоку заявок на інтелектуальні сервіси було прийнято рішення скористатися системою *GPSS*, оскільки ця система досить проста, процес моделювання не трудомісткий, дає точні результати.

Для моделювання ІН з урахуванням самоподібності потоку заявок на ІС використано систему «*Network Simulator*-2» (*NS*-2). У порівнянні з іншими системами імітаційного моделювання – *Cisco Packet Tracer, Riverbed Modeler* (у минулому *OPNET*), *OMNe*T++, *QualNe*t, *SimPy*, *AnyLogic*, *GPS*S та іншими – *NS*-2 вигідно відрізняється вдалою комбінацією низки характеристик: безкоштовністю, широкими можливостями, відкритою архітектурою, наявністю засобів анімації модельних процесів тощо.

Розробка імітаційної моделі інтелектуальної надбудови з одним класом заявок без урахування самоподібності трафіку

Представимо таку ІН у вигляді одноканальної СМО з накопичувачем обмеженою ємкості (*M*/*M*/1/*r*).

1. Опис системи

1.1. Система (рисунок 1) містить один обслуговуючий прилад (П), тобто є одноканальною.

1.2. Потік заявок, що надходять у систему, однорідний.

1.3. Тривалість обслуговування заявок в приладі – величина випадкова.

1.4. Перед приладом є r місць для заявок, які очікують обслуговування і утворюють чергу, тобто в системі є накопичувач обмеженої ємкості: $r < \infty$.

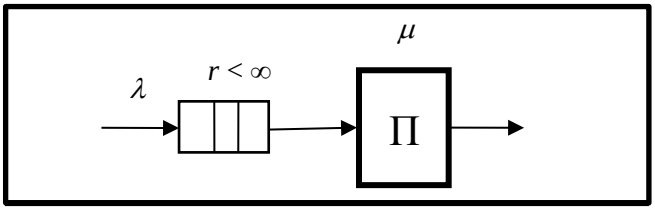

Рисунок 1 – СМО з накопичувачем обмеженої ємкості

2. Припущення і допущення

2.1. Заявки, що надходять в систему, утворюють найпростіший потік з інтенсивністю λ.

2.2. Тривалість обслуговування заявок в приладі розподілена за експоненційним законом з інтенсивністю $\mu = 1/b$, де b – середня тривалість обслуговування заявок в приладі.

2.3. Дисципліна буферизації – з втратами: заявка, що надійшла в систему і застала накопичувач заповненим, втрачається.

2.4. Дисципліна обслуговування – в порядку надходження за правилом «першим прийшов – першим обслужений» (*FIFO*).

У СМО з накопичувачем обмеженою ємкості завжди існує стаціонарний режим, оскільки довжина черги не буде зростати до нескінченності навіть при великих значеннях навантаження.

Алгоритм функціонування ІН з заданими параметрами представлено на рисунку 2.

Реалізація процесу моделювання полягає в переміщенні в моделі деяких рухомих об'єктів, так званих транзактів. В нашому випадку в якості транзактів виступають заявки на ІС. Транзакти послідовно переміщаються від блоку до блоку в заданій алгоритмом моделювання послідовності. Транзакти створюються і знищуються в моделі за допомогою операторів (блоків): *GENERATE* і *TERMINATE*. На початку моделювання в *GPSS*-моделі немає жодного транзакта. У процесі моделювання транзакти формуються в моделі в певні моменти часу відповідно до умов, заданих за допомогою блоку *GENERATE*.

Призначення оператора *GENERATE*: генерування транзактів відповідно до заданих правилом формування інтервалів між транзактами.

Формат оператора [9, 10]:

GENERATE [*A*][*B*][*C*][*D*][*E*]

Значення операндів:

A – середній інтервал часу між транзактами, що генеруються або імовірнісний розподіл інтервалу з вбу-

дованої бібліотеки процедур, поміщений в круглі дужки; [за умовчанням – нуль];

B – величина напівінтервалу рівномірно розподіленого інтервалу або модифікатор таблично заданої функції; [за умовчанням – нуль];

C – зміщення – момент формування першого транзакта; [за умовчанням – нуль];

D – обмежувач – число транзактів, що генеруються даним оператором; [за умовчанням – не обмежено];

E – рівень пріоритету від 0 до 127 (чим більше номер, тим вище пріоритет); [за умовчанням – нуль].

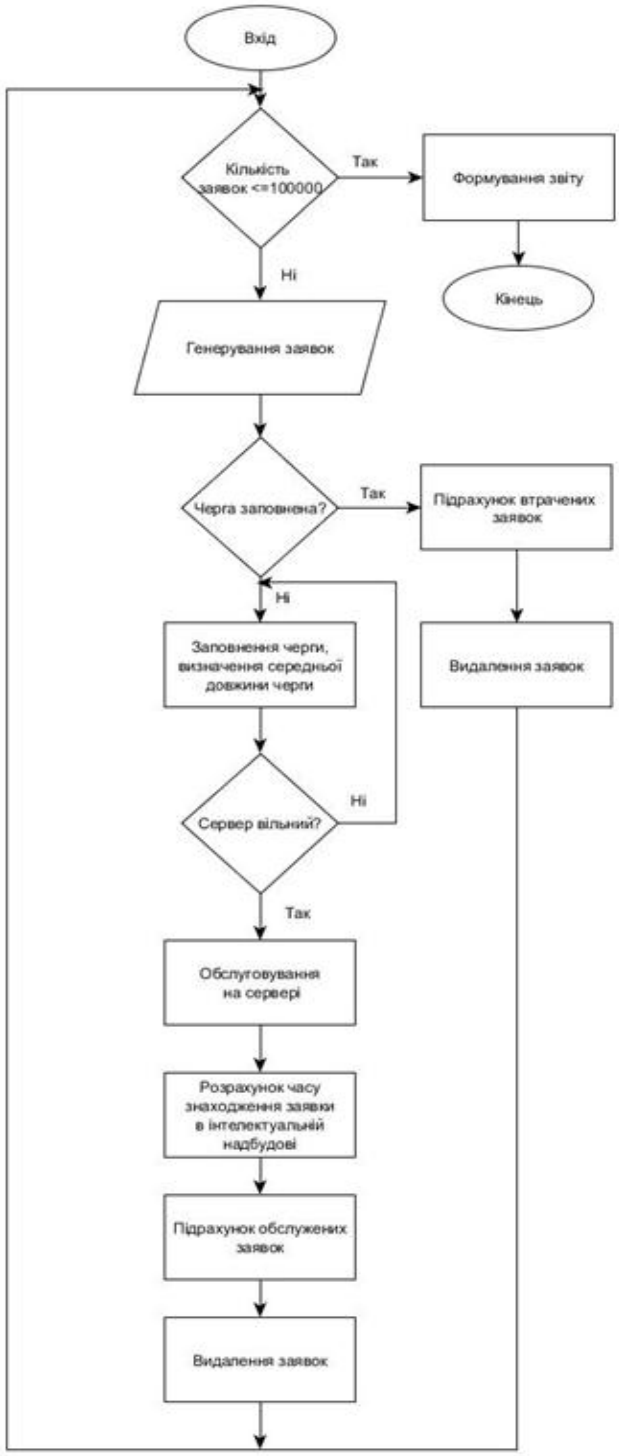

Рисунок 2 – Алгоритм функціонування інтелектуальної надбудови

Оскільки час надходження заявок розподілений за експоненційним законом, то в якості першого операнда слід використати бібліотечну функцію *EXPONENTIAL* (*G,Min,Mean*). Тут *G* – номер генератора рівномірно розподілених випадкових чисел (від 1 до 999) – використовується як аргумент для формування випадкових величин із заданим законом розподілу, *Mean* – математичне очікування (середнє значення) випадкової величини, розподіленої за експоненційним законом, *Min* – зміщення розподілу щодо нуля (мінімальне значення випадкової величини) [10, 11].

Тоді для генерування заявок оператор буде мати вигляд *GENERATE* (*Exponential*(1,0,1/λ)).

Наступним елементом системи, до якого потрапляють заявки, є черга. Черга – об'єкт, котрий використовується для накопичення транзактів, що знаходяться в стані очікування якоїсь події. В нашому випадку при моделюванні ІН черга використовується для імітації процесів очікування перед обслуговуванням на сервері. Насправді транзакти, чекаючи звільнення серверу, заносяться в список затримки відповідного серверу (приладу), при цьому формування списків затримки, тобто занесення в чергу і видалення з черги відбувається автоматично, незалежно від наявності операторів *QUEUE* і *DEPART*. Останні використовуються тільки з метою збору статистики по чергам шляхом фіксації моментів надходження транзакта в чергу і видалення його з черги [9, 10]. Для перевірки вільного місця в черзі доцільно скористатися оператором:

TEST X A,B,[*C*]

Призначення оператора: перевірка значення і передача активного транзакта в блок, відмінний від подальшого, якщо вказана умова не виконується.

Значення операндів:

A – значення, що перевіряється;

B – контрольне значення;

C – ім'я (мітка) блоку призначення *C*; [за умовчанням – режим відмови];

X – оператор відношення, що визначає умову перевірки операнда *A* з операндом *B*. Значення оператора *X* представлені в таблиці 1 [9, 10].

Таблиця 1 – Значення оператора *X*

Значення *X*	**Інтерпретація в сенсі блоку *TEST***
G	*A* більше *B*?
GE	*A* більше або рівне *B*?
E	*A* рівне *B*?
NE	*A* не рівне *B*?
LE	A менше або рівне B?
L	*A* менше *B*?

В нашому випадку існує єдина черга з назвою *1*. За довжину черги відповідає стандартний системний числовий атрибут *Q*. Як вже зазначалося, довжина черги *r* = 4. Тоді оператор для перевірки вільного місця в черзі для заявок матиме вигляд: *TEST L Q*1,4,*M*. У разі заповнення черги заявка передається до мітки *M* для підрахунку необслужених заявок та їх видалення.

При існуючих припущеннях та обмеженнях сервер необхідно моделювати за допомогою приладу. Прилади (одноканальні пристрої) – об'єкти, які можуть знаходитися в одному з двох станів: вільному або зайнятому. При моделюванні систем масового обслуговування використовуються для імітації процесів заняття і звільнення приладу, тобто для моделювання одноканальних СМО; заняття і звільнення приладу транзактом виконується за допомогою операторів *SEIZE* і *RELEASE* [9, 10].

Для імітації часу обслуговування заявки сервером доцільно скористатися оператором *ADVANCE*.

Призначення оператора: затримка транзакта на заданий час.

Формат оператора:

ADVANCE [*A*][*B*]

Значення операндів:

A – середній час затримки або імовірнісний розподіл з вбудованої бібліотеки процедур, поміщеної в круглі дужки; [за умовчанням – нуль];

B – величина напівінтервалу рівномірно розподіленого інтервалу затримки або модифікатор таблично заданої функції; [за умовчанням – нуль] [9, 10].

Середній час затримки розподілений також за експоненційним законом. Тоді для моделювання часу обслуговування заявки на сервері оператор матиме вигляд *ADVANCE* (*Exponential*(1,0,*b*)).

Для визначення загального часу обслуговування заявки (часу перебування заявки в системі) необхідно скористатися оператором *TABULATE*. З його допомогою в таблицю буде заноситися стандартний системний числовий атрибут *M*1 – резидентний час транзакта (абсолютний модельний час мінус час появи транзакта в моделі).

Транзакти покидають модель (знищуються), потрапляючи у блок *TERMINATE*. Функціонування моделі припиняється після надходження до неї необхідної кількості заявок, котра визначається користувачем.

По завершенню моделювання результати формуються автоматично у вигляді стандартного звіту, котрий містить значення середньої довжини черги, часу знаходження заявки в системі. Ймовірність блокування заявки можна розрахувати як відношення кількості необслужених заявок до кількості обслужених, використовуючи для підрахунку та видалення кожної з них окремі блоки *TERMINATE*.

Розробка імітаційної моделі інтелектуальної надбудови з трьома класами заявок з відносними пріоритетами та без урахування самоподібності трафіку

Представимо ІН у вигляді системи масового обслуговування *M*/*M*/1/3.

Опис системи (рис. 3):

1. Система – *одноканальна*.
2. Вхідний потік заявок – *неоднорідний:* при цьому у систему надходить обмежена кількість класів (три класи) заявок.
3. Накопичувач для заявок – *обмеженою ємкістю* $m = 3$.
4. Дисципліна буферизації – *без витіснення* заявок: якщо при надходженні в систему заявки будь-якого класу накопичувач заповнений до кінця, то заявка втрачається.
5. Дисципліна обслуговування – *з відносними пріоритетами*.

Припущення:

1. Заявки трьох класів, що надходять в систему, утворюють *найпростіші* потоки з інтенсивностями λ_1, λ_2 та λ_3, відповідно.
2. Тривалість обслуговування заявок кожного класу розподілена по *експоненційному* закону з інтенсивностями $\mu_1 = 1/b_1$, $\mu_2 = 1/b_2$, $\mu_3 = 1/b_3$, де b_1, b_2 і b_3 – середня тривалість обслуговування заявок класу 1, 2 і 3, відповідно.

У СМО завжди існує стаціонарний режим, оскільки не може бути нескінченних черг [9].

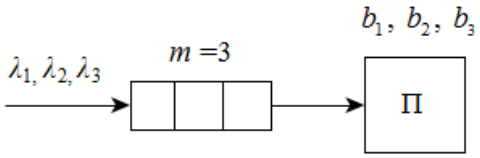

Рисунок 3 – Представлення ІН у вигляді СМО виду *M*/*M*/1/3 з пріоритетами

Для побудови імітаційної моделі також скористаємося середовищем *GPSS*. Представимо алгоритм (правила) функціонування ІН (рисунок 4). Для досягнення необхідної точності результатів модель повинна працювати протягом певного часу або до неї має надійти відповідна кількість заявок. Спочатку відбувається генерування заявок трьох класів з різними пріоритетами, які надходять до системи. Система перевіряє, чи є місце в черзі. Якщо місця немає, то заявка втрачається. Ведеться підрахунок втрачених заявок першого, другого та третього класів, що є необхідним для розрахунку ймовірності блокування. Якщо ж місце в черзі є, то заявка потрапляє до черги і перевіряється, чи вільний сервер. Якщо сервер вільний, то заявка, не чекаючи в черзі, обслуговується. Інакше поки що залишається в черзі на першому місці доти, поки не надійде заявка з вищим пріоритетом. Паралельно відбувається підрахунок середньої довжини черги та часу знаходження у черзі заявки кожного класу. Після обслуговування заявки відбувається розрахунок часу знаходження заявки в системі, визначення кількості обслужених заявок та їх видалення.

Імітаційна модель буде містити три модулі – окремий модуль для кожного класу. Оскільки час надходження заявок розподілений за експоненційним законом, то в якості першого операнда в кожному з модулів слід використати бібліотечну функцію *EXPONENTIAL*. Тоді для генерування заявок першого класу оператор буде мати вигляд *GENERATE* (*Exponential*(1,0,1/λ_1)),,,,5, для другого класу – *GENERATE* (*Exponential*(1,0,1/λ_2)),,,,3, для третього класу – *GENERATE* (*Exponential*(1,0,1/λ_3)).

В нашому випадку існує єдина черга з назвою *1*. За довжину черги відповідає стандартний системний числовий атрибут *Q*. Як вже зазначалося, довжина черги *r* = 3. Тоді оператор для перевірки вільного місця в черзі для заявок матиме вигляд: *TEST L Q*1,3,*M*. У разі за-

повнення черги заявка передається до мітки *M* для підрахунку необслужених заявок та їх видалення. Оператор *TEST* необхідно розмістити у кожному з трьох модулів.

Імітація обслуговування на сервері в кожному модулі описується операторами *SEIZE*, *RELEASE* та *ADVANCE*.

Для визначення часу знаходження в черзі та загального часу обслуговування заявки (часу перебування заявки в системі) необхідно скористатися операторами *TABULATE*.

По завершенню моделювання результати формуються автоматично у вигляді стандартного звіту, котрий містить значення середньої довжини черги, часу знаходження заявки в системі для кожного класу. Ймовірність блокування заявки можна розрахувати як відношення кількості необслужених заявок до кількості обслужених, використовуючи для підрахунку видалення кожної з них окремі блоки *TERMINATE*. Моделювання будь-якої ІН з іншими параметрами відбувається подібним чином.

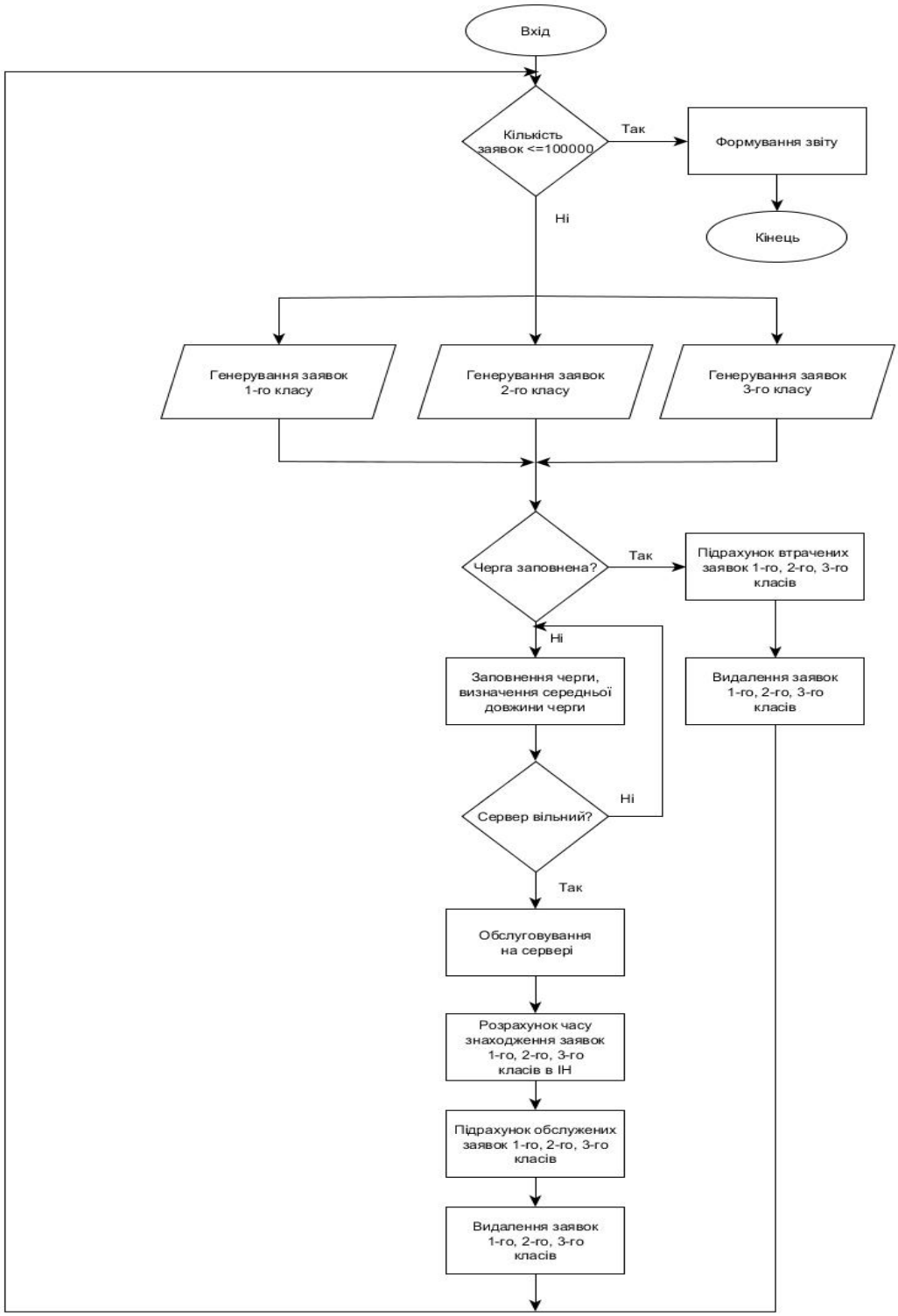

Рисунок 4 – Алгоритм функціонування ІН

Розробка імітаційної моделі інтелектуальної надбудови з урахуванням самоподібності потоку заявок на інтелектуальні сервіси

Для розробка імітаційної моделі ІН з урахуванням самоподібності потоку заявок на інтелектуальні сервіси скористаємося *NS*-2.

При генеруванні заявок на ІС необхідно вказати розмір пакетів даних (*packetSize_*), інтервал їх відправки (*rate_*) і розмір вікна *TCP* (*window_*).

Відправка заявок відбувається через протокол *TCP*. В імітаційній моделі допущено, що мережа має необмежену пропускну здатність. Абоненти генерують самоподібний трафік.

Для генерації трафіку в системі *NS*-2 використовуються об'єкти типу *Traffic*. Вони створюються методами *Traffic/type*, де, в нашому випадку, *type* – *Pareto*.

Об'єкт *Traffic/Pareto* – *ON/OFF* генератор трафіку згідно розподілу Парето. Проста *ON/OFF* модель передбачає, що джерела перемикаються між двома станами: *ON* – стан, в якому джерела генерують трафік з постійною швидкістю, *OFF* – стан, в котрому вони простоюють. При постійному розмірі пакетів *ON* і *OFF* періоди розподілені згідно закону Парето.

Розподіл Парето має функцію розподілу[8]:

$$F(x) = 1 - \left(\frac{\beta}{x}\right)^{\alpha}, \qquad (1)$$

де α – параметр форми, β – мінімальне значення випадкової величини x.

Щільність розподілу Парето задається функцією:

$$f(x) = \frac{\alpha}{\beta}\left(\frac{\beta}{x}\right)^{\alpha+1}, \text{ коли } x > \beta \text{ и } \alpha > 0,$$
$$f(x) = F(x) = 0, \text{ коли } x \le \beta. \qquad (2)$$

Середнє значення випадкової величини визначається з формули:

$$m(x) = \frac{\alpha\beta}{\alpha - 1}. \qquad (3)$$

Параметр α визначає середнє і дисперсію випадкової величини:

- якщо $\alpha \le 1$, розподіл має нескінченне середнє;
- якщо $1 \le \alpha \le 2$, розподіл має кінцеве середнє і нескінченну дисперсію;
- якщо $\alpha \le 2$, розподіл має нескінченну дисперсію.

Існує відношення між параметром α і параметром Хьорста (*H*):

$$H = \frac{3 - \alpha}{2}. \qquad (4)$$

Тривалість *ON* і *OFF* періодів є випадковими величинами з розподілом Парето, які задаються середніми значеннями.

Сервер обслуговування здатний обробляти лише одну заявку в одиницю часу. Час обслуговування заявки змінюється в залежності від її розміру. Якщо сервер обслуговування зайнятий, заявки стають в чергу. Максимальна довжина черги задається параметром *queue-limit*; якщо довжина черги перевищується, відбувається втрата заявок.

Отриманий в результаті моделювання *trace*-файл дає змогу розрахувати час знаходження заявки в мережі, середню довжину черги на обслуговування, поточну довжину черги, ймовірність блокування заявки в мережі, а також кількість втрачених і доставлених пакетів.

Розроблена імітаційна модель ІН, що містить 100 абонентських вузлів 0 – 99, які генерують мережний трафік, один *Softswitch*, а також сервер, який складається з основного вузла і допоміжного вузла, введеного для реалізації певного алгоритму обробки заявок.

Імітаційна модель реалізована в системі *NS*-2 у вигляді сценарію мовою *TCL*. Блок-схема алгоритму цього сценарію наведена на рис. 5. Далі описані основні блоки наведеного алгоритму моделювання.

Рисунок 5 – Загальний алгоритм сценарію моделювання ІН в системі *NS*-2

Екземпляр симулятора створюється стандартною командою:

set ns [new Simulator]

Трасування результатів роботи моделі здійснюється в файл *trace.tr* завдяки командам:

set tracefd [open trace.tr w]

$ns trace-all $tracefd

Вузли абонентів (*sp_#*), *Softswitch* (*ss_#*), сервера і вузла обслуговування (*serv*1*_n#*, *serv*1*_ob*1*_n#*) створюються командами:

set sp_($i) [$ns node]
set ss_($j) [$ns node]
set serv1_n($m) [$ns node]
set serv1_ob1_n($k) [$ns node]

В описаних командах:

$c – кількість абонентів;
$j – номер вузла *Softswitch*;
$m – номер вузла сервера обслуговування;
$k – номер вузла обслуговування.

Зв'язки між вузлами-абонентами, *Softswitch* та сервером обслуговування моделюються об'єктами *duplex-link*. Зв'язки створюються і налаштовуються командами:

$ns duplex-link $sp_($i) $ss_($j) $L $d DropTail
$ns duplex-link $ss_($j) $serv_n($c) $L $d DropTail

де *$i* – номер вузла абонента;
$j – номер вузла *Softswitch*;
$c – номер вузла сервера обслуговування;
$L – пропускна здатність каналу зв'язку;
$d – затримка між вузлами.

Параметр *DropTail* вказує, що черга між вузлами будуватиметься за структурою *FIFO*.

Для налаштування вузла обслуговування задіюються об'єкти *duplex-link* і *queue-limit*:

$ns duplex-link $serv_($i) $obsl_($j) $L 0ms DropTail
$ns queue-limit $serv_($i) $obsl_($j) $queueLimit

де *$i* – номер вузла сервера обслуговування;
$j – номер вузла обслуговування;
$L – час обслуговування, який розраховується як добуток необхідного часу обслуговування і розміру відправленого пакету;
$queueLimit – максимальна довжина черги на обслуговування.

Налаштування генераторів трафіку в загальному випадку виконується такими командами:

set tcp_($i) [new Agent/TCP/Newreno]
$ns attach-agent $sp_($i) $tcp_($i)
set sink_($j) [new Agent/TCPSink/DelAck]
$sink_($j) set fid_ $j
$ns attach-agent $serv_($c) $sink_($j)
$ns connect $tcp_($i) $sink_($j)
$tcp_($i) set fid_ $i
$tcp_($i) set window_ $windowSize
$tcp_($i) set segsize_ $packetSize
set traffic_($m) [new Application/Traffic/$type]
$traffic_($m) set burst_time_ $burstTime
$traffic_($m) set idle_time_ $idleTime
$traffic_($m) set rate_ $rate
$traffic_($m) set shape_ $shape
$traffic_($m) set packetSize_ $packetSize
$traffic_($m) attach-agent $tcp_($i)

В наведеному коді використовуються такі параметри:

$i – номер агента з'єднання;
$j – номер агента відповіді від сервера про доставлені пакети;
$c – номер сервера;
*windowSize_* – розмір вікна *TCP* з'єднання;
*packetSize_*– розмір пакетів, що будуть відправлятися;
$m – номер генератора трафіку;
$type – вид генерованого трафіку (*Exponential*, *Pareto*);
$burstTime – середній час *ON* періоду роботи генератора;
$idleTime – середній час *OF*F періоду роботи системи;
$rate – бітова швидкість, з якою агент буде генерувати пакети;
$shape – параметр, що визначає середнє значення і дисперсію випадкової величини, які обумовлюють самоподібні властивості генерованого трафіку.

Розклад роботи агентів здійснюється командами:

$ns at $timeStart "$traffic_($i) start"
$ns at $timeEnd "$traffic_($i) stop"
$ns at 5.0 "finish"

де *$i* – номер генератора трафіку;
$timeStart – час початку роботи генератора;
$timeEnd – час закінчення роботи генератора.

3 Есперимент

Використовуючи імітаційні моделі без урахування та з урахуванням самоподібності трафіку для ІН, що обслуговує один клас заявок на ІС отримана оцінка ефективності управління наданням ІС. На рисунку 6 представлено графіки залежності ймовірності відмови в наданні ІС $P_в$ з врахуванням та без врахування самоподібності трафіку від інтенсивності надходження заявок на ІС. Як видно з графіків, ймовірність відмови в наданні сервісу $P_в$ при врахуванні самоподібності потоку на ІС завжди більша ніж без врахування. Максимальна різниця, виходячи з отриманих даних, становить порядку 80% при значенні λ = 15– 30 од./год. Тобто в реальній мережі буде втрачатися значно більше заявок, ніж при визначенні без врахування самоподібності трафіку.

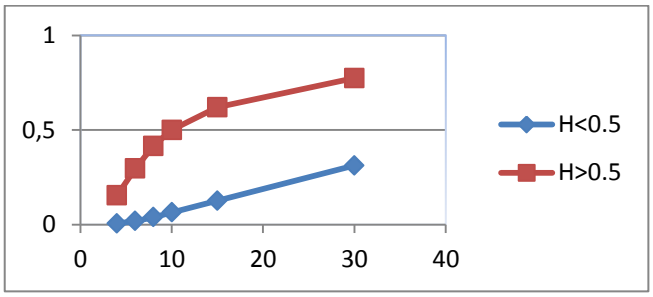

Рисунок 6 – Графіки залежності ймовірності відмови в наданні ІС від інтенсивності надходження заявок

На рисунку 7 представлено графіки залежності довжини черги *L* при сервері з врахуванням та без врахування самоподібності трафіку від інтенсивності надходження заявок на ІС. Як видно з графіків, при при значенні λ < 15 од./год. довжина черги *L* для самоподібного трафіку більша, ніж для експоненційного (максимальне відхилення складає порядку 65% при значенні λ = 5 од./год., але починаючи зі значення λ = 15 од./год. і при подальшому збільшенні значення λ довжина черги *L* для самоподібного трафіку поступово стає меншою, ніж для експоненційного. Це пов'язано з характером трафіку (фрактальність), а також з тим, що істотно збільшується показник – ймовірність втрати заявки $P_в$ для самоподібного трафіку (рисунок 6).

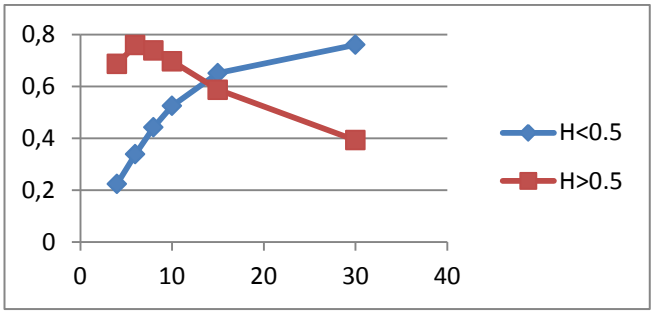

Рисунок 7 – Графіки залежності довжина черги при сервері від інтенсивності надходження заявок

На рисунку 8 представлено графіки залежності часу обслуговування заявки в ІН T з врахуванням та без врахування самоподібності трафіку від інтенсивності надходження заявок на ІС. Як видно з графіків, час обслуговування спочатку буде майже однаковим. При збільшенні інтенсивності надходження заявок на ІС суттєво зростає ймовірність втрати заявки $P_в$ при врахуванні самоподібності трафіку і, відповідно, зменшується середній час обслуговування.

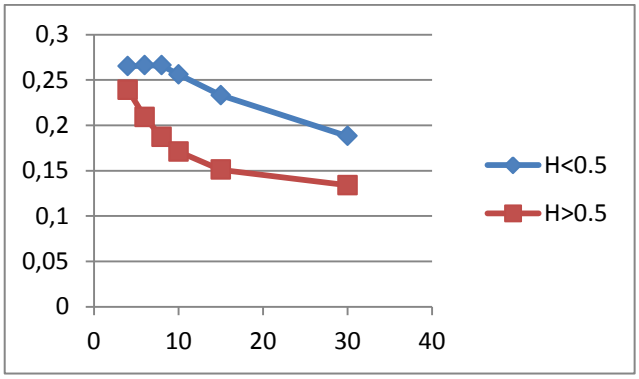

Рисунок 8 – Графіки залежності часу обслуговування заявки в ІН від інтенсивності надходження заявок

Висновки

На підставі аналізу алгоритмів функціонування ІН запропоновано її імітаційні моделі. Для імітаційного моделювання використано GPSS (*General Purpose Simulation System*) – загально-цільову систему імітаційного моделювання у тих випадках, коли не враховується самоподібність потоку заявок на ІС та *Network Simulator*-2 (*NS*-2) – у тих випадках, коли враховується самоподібність потоку заявок на ІС. Визначено способи моделювання окремих елементів ІН.

Запропоновано імітаційні моделі ІН з одним класом заявок без урахування самоподібності трафіку, з трьома класами заявок з відносними пріоритетами без урахування самоподібності трафіку, з одним класом заявок з урахуванням самоподібності трафіку. Зазначені моделі дозволяють визначити показники якості управління наданням ІС на основі заданих вхідних параметрів – інтенсивності надходження заявок на ІС, інтенсивності обслуговування заявок сервером, кількості класів заявок та їх пріоритетів.

Використовуючи імітаційні моделі без урахування та з урахуванням самоподібності трафіку для ІН, що обслуговує один клас заявок на ІС, визначені показники якості надаваних ІС, що надає можливість отримати оцінку ефективності управління наданням ІС. В результаті аналізу отриманих результатів встановлено, що ймовірність відмови в наданні сервісу при врахуванні самоподібності потоку на ІС завжди більша ніж без врахування. Максимальна різниця, виходячи з отриманих даних, становить порядку 80%. Максимальна різниця довжини черги для самоподібного трафіку і для експоненціального складає порядку 65%. Запропоновані імітаційні моделі дозволять ще на етапі проектування *NGN* визначити ефективність управління наданням інтелектуальних сервісів інтелектуальною надбудовою.

Література

1. **Бакланов И.Г.** NGN. Принципы построения и организации / И.Г. Бакланов. – Эко-Трендз, 2008. – 400 с.
2. **Росляков А.В., Кашин М.М.** Исследование свойств сигнального трафика протокола SIP / А. В. Росляков, М. М. Кашин //TComm, №5, 2009, с. 26-29.
3. **Агеев Д.В.** Методика определения параметров потоков на разных участках мультисервисной телекоммуникационной сети с учетом эффекта самоподобия / Д.В. Агеев, А.А. Игнатенко, А.Н. Копылев // – Електронне наукове фахове видання - журнал «Проблеми телекомунікацій» – № 3 (5) – 2011 – С. 18 – 35
4. **Агеев Д.В.** Параметрический синтез мультисервисных телекоммуникационных систем при передаче группового трафика с эффектом самоподобия / Д.В. Агеев. // Проблемы телекоммуникаций. – Харьков, 2013. – №1(10). – С. 46-65.
5. **Tsybakov B., Georganas N.D.** On self-similar traffic in ATM queues: definition, overflow probability bound, and cell delay distribution // IEEE/ACM Trans. on Networking. – 1997. – Vol. 5, No 3. – P. 397-408.
6. **Князєва Н.О.** Теорія проектування комп'ютерних систем і мереж. Ч.2. Методи аналізу і синтезу комп'ютерних систем і мереж [Текст]: навч. посібник / Н. О. Князєва. – Одеса: СПД, 2012. – 240 с.
7. **Князєва Н.О.** Підвищення якості управління послугами при застосуванні децентралізованої системи управління / Н.О. Князєва, С.В. Шестопалов // Вісник ДУІКТ. – т.8. – №1 – К.: ДУІКТ, 2010. – С. 21-28.
8. **Князєва Н.О.** Управління інтелектуальними сервісами в мережах наступного покоління / Н.О. Князєва, С.В. Шестопалов. – Одеса: ТОВ Плутон, 2017. – 268 с.
9. **Алиев Т.И.** Основы моделирования дискретных систем / Алиев Т.И. – СПб: СПбГУ ИТМО, 2009. – 363 с.
10. **Кудрявцев Е.М.** GPSS World. Основы имитационного моделирования различных систем / Е.М. Кудрявцев. – М: ДМК Пресс, 2004. – 320 с.

Отримана в редакції 12.02.2018, прийнята до друку 20.03.2018

Imitation Modeling of Control Providing Processes of Intelligent Services in NGN

S. V. Shestopalov, T. V. Kunup, B. L. Pustovyi
Odessa National Academy of Food Technologies, 112 Kanatnaya St., Odessa, 65039, Ukraine

The article is devoted to the development of imitation models of control providing processes of intelligent services in NGN. It is shown that in order to determine the effectiveness of control of providing intelligent services by various authors, a number of analytical models of intelligent superstructures are proposed both without consideration and taking into account the self-similarity of the flow of applications to intelligent services. It is noted that the creation of analytical models with a significant complication of intelligent superstructures becomes virtually impossible. In this case, it is suggested to use imitation models. It is proposed to present the intelligent superstructure as a queuing system. To model the intelligent superstructure of NGN in the form of queuing system without taking into account the self-similarity of the flow of requests for intelligent services, it was decided to use the GPSS system. To model the c superstructure taking into account the self-similarity of the flow of applications, it is proposed to use the system "Network Simulator-2" (NS-2). The algorithms of functioning and imitation models of intelligent superstructure are presented, which provides one class of applications without taking into account the self-similarity of traffic intelligent superstructure, which provides three classes of applications with relative priorities and without taking into account the self-similarity of traffic, and the intelligent superstructure, taking into account the self-similarity of the flow of applications for intelligent services. By using imitation models without account and taking into account the self-similarity of traffic for intelligent superstructure that serves one class of applications for intelligent services, the quality criterions provided by intelligent services are determined, which provides an opportunity to gauge the efficiency of control of provision of intelligent services. The proposed imitation models will allow the NGN design stage to determine the effectiveness of control the provision of intelligent services with an intelligent superstructure.

Keywords*: NGN; Intelligent Service; Control Efficiency; Quality Criterions; Self-similarity; GPSS; NS-2; Imitation Model.*

References

1. **Baklanov, I.G.** (2008) *NGN. Printsipy postroieniia i organizatsii*. Eko-Trendz, 400 p.
2. **Roslyakov, A.V., Kashin, M.M.** (2009) *Issledovanie svoistv signalnogo trafika protokola SIP*. TComm, No.5, 26-29.
3. **Ageev, D.V., Ignatenko, A.A., Kopyilev, A.N.** (2011) Metodika opredeleniia parametrov potokov na raznykh uchastkakh multiservisnoi telekommunikatsionnoi seti s uchetom effekta samopodobiia. *Problemy telekomunikatsiy*, No. 3(5), 18-35.
4. **Ageev, D.V.** (2013) Parametricheskii sintez multiservisnyikh telekommunikatsionnykh sistem pri peredache gruppovogo trafika s effektom samopodobika. *Problemy telekomunikatsiy*, No. 1(10), 46-65.
5. **Tsybakov, B., Georganas, N.D.** (1997) On self-similar traffic in ATM queues: definition, overflow probability bound, and cell delay distribution. *IEEE/ACM Trans. on Networking*, 5(3), 397-408.
6. **Kniazieva, N.O.** (2012) *Teoriia proektuvannia kompiuternikh sistem i merezh. Ch.2. Metody analizu i syntezu kompiuternikh system i merezh*. Odesa: SPD, 240 p.
7. **Kniazieva, N.O., Shestopalov, S.V.** (2010) Pidvyschennia iakosti upravlinnia poslugamy pry zastosuvanni detsentralizovanoi systemy upravlinnia. *VIsnik DUIKT*, 8(1), 21-28.
8. **Kniazieva, N.O., Shestopalov, S.V.** (2017) *Upravlinnia intelektualnymi servisami v merezhakh nastupnogo pokolinnia*. Odesa: TOV Pluton, 268 p.
9. **Aliev T.I.** (2009) *Osnovy modelirovaniia diskretnykh sistem.* SPb: SPbGU ITMO, 363 p.
10. **Kudryavtsev E.M.** (2004) *GPSS World. Osnovyi imitatsionnogo modelirovaniya razlichnyih sistem*. M: DMK Press, 320 p.

Received 12 February 2018
Approved 20 March 2018