

ТЕЛЕНИК С.Ф.,
РОЛІК О.І.,
БУКАСОВ М.М.,
ЛАБУНСЬКИЙ А.Ю.

МОДЕЛІ УПРАВЛІННЯ ВІРТУАЛЬНИМИ МАШИНАМИ ПРИ СЕРВЕРНІЙ ВІРТУАЛІЗАЦІЇ

В статті проводиться аналіз методів управління розподілом ресурсів при серверній віртуалізації. Розроблені моделі та методи управління ресурсами для випадку живої міграції віртуальних машин. Для здійснення управління реалізовано метод гілок і меж. Наводяться опис та результати експериментальних досліджень.

In the article the analysis of methods of resource management for server virtualization is realized. The models and methods of resource management for case of virtual machines live migration are developed. Implementation of the branch and bound algorithm for realization of management is proposed. Description and results of experiments are described.

Вступ

Останнім часом при створенні інформаційних систем спостерігається тенденція підвищення вимог до якості і надійності функціонування сервісів інформаційно-телекомунікаційної інфраструктури, як основи ведення бізнесу. Найпростішим вирішенням цього питання протягом тривалого часу було розміщення найбільш критичних застосувань на виділених серверах чи кластерах [1]. Таке рішення на порядок підвищувало надійність функціонування інформаційної системи, але в той же час значно збільшувало сукупну вартість володіння серверною складовою інформаційної інфраструктури за рахунок вартості додаткових серверів та ліцензій на комерційне програмне забезпечення, витрат на електроживлення, кондиціонування, адміністрування, обслуговування тощо.

За статистичними даними компанії VMware, зазвичай завантаження виділених під окремі задачі серверів не перевищує 15 – 20 % від їх повної потужності. Тому в останні роки стало популярним скорочення кількості фізичних серверів для підтримки різноманітних сервісів шляхом впровадження технології віртуалізації серверів. Такий підхід за рахунок більшого завантаження серверів дозволяє не тільки зменшити їх кількість, але й скоротити витрати на їх утримання [2]. До того ж наявність функції «живої міграції» віртуальних машин (ВМ) між фізичними серверами дозволяє отримати показники го-

товності сервісів, близькі до кластерних рішень, при значно менших витратах.

Тому дуже актуальною стає задача управління розподілом ресурсів між ВМ та розташування ВМ по серверам таким чином, щоб при обмежених обчислювальних ресурсах у першу чергу забезпечити функціонування застосувань, пов'язаних з найбільш важливими бізнес-процесами, навіть за рахунок менш важливих [3], а при надлишку ресурсів розташувати ВМ таким чином, щоб вивільнити сервери з метою економії енергоспоживання. Реалізація подібних систем управління розподілом ресурсів потребує розроблення відповідних моделей та методів чи адаптації вже відомих [4] з урахуванням віртуалізації серверів та «живої міграції» ВМ.

У праці [3] запропоновано ресурсний підхід до управління ІТ-інфраструктурою підприємства. Пропонується розглядати діяльність підприємства як сукупність бізнес-процесів, що споживають визначену кількість ресурсів певних типів. Розглянуто дискретний випадок, коли окремі бізнес-процеси підтримуються в повному обсязі чи не підтримуються зовсім, та неперервний випадок, коли можлива часткова підтримка бізнес-процесів.

Цей підхід розвинений у праці [4], де крім ресурсних обмежень додавались обмеження по надійності підтримки бізнес-процесів.

У праці [5] запропоновані моделі і методи управління ІТ-інфраструктурою на основі ресурсного підходу на всіх етапах життєвого

циклу – від планування при створенні до експлуатації та подальшого розвитку.

У праці [6] розглянуто питання управління ресурсами, застосуваннями та навантаженням для випадку віртуалізації серверів, але без можливості «живої міграції» ВМ.

Питання управління ресурсами ІТ-інфраструктури розглянуті досить детально, але використання для цього «живої міграції» ВМ, можливість якої в більшості системах віртуалізації з'явилася не так давно, все ще потребує додаткових досліджень.

Мета дослідження

Метою статті є розробка моделей і методів управління розподілом ресурсів ІТ-інфраструктури з використанням «живої міграції», при якому забезпечується як підвищення доступності найважливіших сервісів в умовах нестачі обчислювальних ресурсів, так і економія енергоспоживання в умовах їх надлишку.

Моделі та методи управління ресурсами при живій міграції

Нехай є декілька фізичних серверів S_i , $i = 1, \dots, n$, на яких під управлінням гіпервізорів функціонують ВМ V_j , $j = 1, \dots, m$ (рис. 1).

Кожна з ВМ в залежності від потоку клієнтських запитів використовує певну кількість ресурсів типу R_k , $k = 1, \dots, l$ (пам'ять, процесорний час, дисковий простір, пропускна спроможність підсистеми вводу-виводу, зовнішніх каналів зв'язку тощо).

Введемо необхідні для побудови моделей позначення:

r_{ki} – кількість ресурсу типу R_k , що встановлена на сервері S_i ;

p_{kj} – потреби ВМ V_j у ресурсах типу R_k для забезпечення заданого рівня SLA;

x_{ij} – булева змінна, яка визначає, чи встановлена ВМ V_j на сервері S_i .

Оскільки кожна ВМ одночасно розташована не більше, ніж на одному сервері, має виконуватись умова:

$$\sum_{i=1}^n x_{ij} \leq 1, j = 1, \dots, m. \quad (1)$$

До того ж, для нормального функціонування ВМ повинні бути забезпечені достатнім обсягом ресурсів серверів, на яких вони розташовані:

$$\sum_{j=1}^m x_{ij} p_{kj} \leq r_{ki}, k = 1, \dots, l; i = 1, \dots, n. \quad (2)$$

Розглянемо дві задачі розташування ВМ для випадків нестачі та надлишку ресурсів.

Задача 1. У разі, якщо внаслідок збільшення клієнтських запитів потреба окремих ВМ у ресурсах певного типу збільшилась настільки, що стає неможливим забезпечити усі ВМ необхідною кількістю ресурсів, природним виходом стає задача підтримки найбільш важливих бізнес-процесів шляхом забезпечення ресурсами тих ВМ, на яких працюють пов'язані з ними сервіси, за рахунок менш важливих.

Позначимо через w_j , $j = 1, \dots, m$, важливість застосувань, встановлених на ВМ V_j .

Тоді, задачу можна сформулювати таким чином:

$$\sum_{j=1}^m \sum_{i=1}^n x_{ij} w_j \rightarrow \max, \quad (3)$$

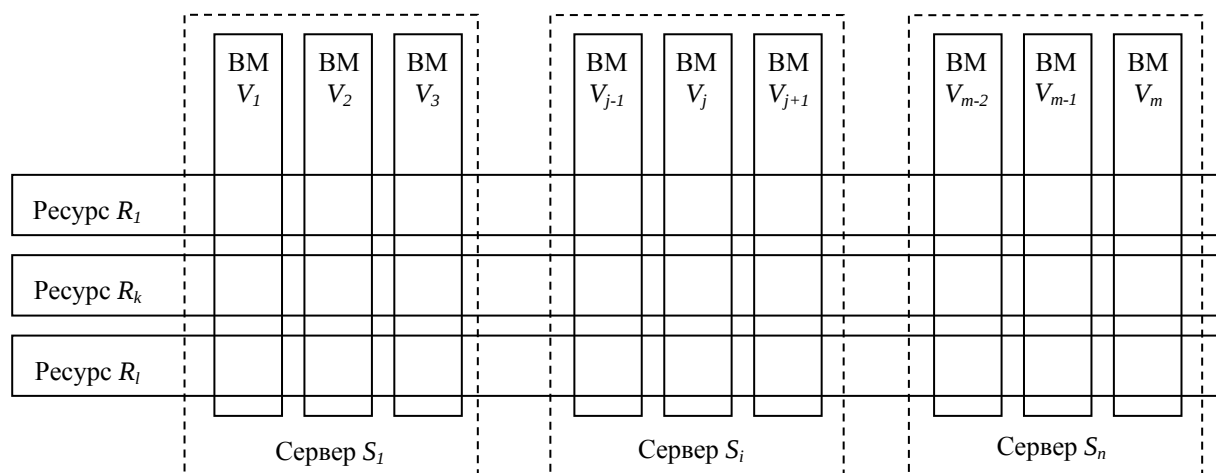


Рис. 1. Розподіл ресурсів серверів між ВМ

при обмеженнях (1), (2).

Ця задача становить собою лінійну задачу булевого програмування. Для її вирішення скористаємося методом гілок та меж. Але спочатку покажемо, як можна, враховуючи специфіку задачі, зменшити початкову кількість можливих комбінацій, яка складає 2^m .

По-перше, згідно з обмеженням (1), кожна ВМ може бути розташована не більше, ніж на одному сервері, тобто замість перебору 2^n комбінацій для кожної ВМ V_j , можна обмежитись розглядом лише $n+1$ варіантів: $y_j=0, \dots, n$, де y_j – номер сервера, на якому встановлена ВМ V_j ($y_j=0$ у випадку, якщо внаслідок нестачі ресурсів ВМ V_j не встановлена на жодному сервері або ресурси для неї виділяються по залишковому принципу).

По-друге, враховуючи те, що вимоги ВМ у ресурсах p_{ij} є невід’ємними, якщо на якомусь з етапів побудови дерева варіантів розміщення ВМ по серверам одне з обмежень (2) перестав виконуватись, продовжувати побудову цієї гілки немає сенсу, оскільки для усіх її вузлів це обмеження також не буде виконуватись.

По-третє, слід врахувати те, що дуже часто при побудові серверних ферм використовують сервери з однаковою конфігурацією. Якщо, наприклад, є два ідентичних сервера, кількість можливих варіантів розміщення ВМ можна скоротити в 2 рази, оскільки перенесення усіх ВМ з першого сервера на другий, а з другого на перший ніяк не вплине на об’єм ресурсів, які використовуються. Тобто перестановка рядків у матриці $X = \|x_{ij}\|$, які відповідають однаковим стовпцям у матриці $R = \|r_{ki}\|$, ніяк не впливає ані на виконання обмежень (1), (2), ані на значення критерію (3). Іншими словами, для серверів з однаковою конфігурацією важливе не абсолютне розміщення ВМ на цих серверах, а їх розташування одна відносно іншої. При збільшенні кількості серверів з однаковою конфігурацією кількість «зайвих» комбінацій значно зростає від $(n-1)$ для випадку розміщення усіх ВМ на одному сервері до $(n!-1)$ при розміщенні ВМ на різних серверах.

Так, наприклад, для випадку трьох ВМ і трьох ідентичних серверів з можливих 27 комбінацій розташування ВМ лише 5 є унікальними, а решта – 22 комбінації – «дзеркальними» до них. Тобто, навіть для таких незначних значень m і n початкову множину

варіантів можна скоротити більше, ніж у 5 разів.

Алгоритм перебору та його результати для цього випадку проілюстровані на рис. 2. Для наочності буквами («А», «В», «С») позначені ВМ, цифрами (1, 2, 3) – номери серверів, комбінацією букви з цифрою – розміщення ВМ на сервері. Поруч з кінцевими результатами на рисунку перелічені комбінації, які є «дзеркальними» до них, тобто призводять до таких же результатів і можуть бути виключені з перебору. Алгоритм перебору при побудові дерева розташування ВМ полягає у наступному: на черговому рівні чергової гілки кількість серверів, що розглядаються, для перебору приймається на 1 більше, ніж максимальна для попередніх рівнів цієї ж гілки, але не більше ніж n .

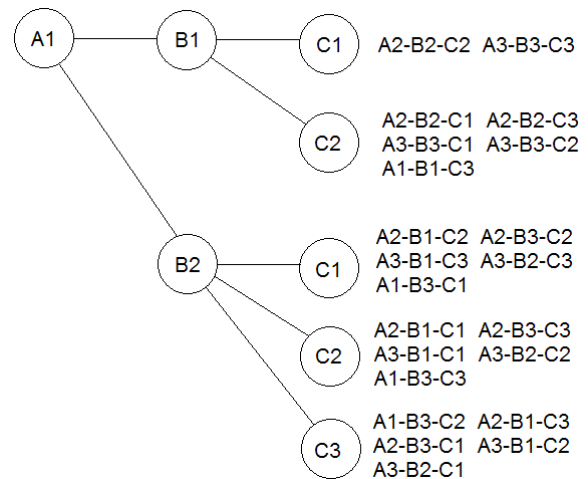


Рис. 2. Схема перебору комбінацій для ідентичних серверів

Так, на рівні «А» розглядається лише випадок розташування ВМ «А» на сервері №1, а на серверах №2 та №3 не розглядається, оскільки сервери №1-3 є повністю ідентичними і розташування ВМ «А» на них призведе до тих самих результатів. При розв’язанні задачі в реальних умовах, з метою зменшення кількості міграцій ВМ, при виборі довільного сервера обирається той, на якому ця ВМ вже встановлена. На рівні «В» розглядаються два варіанти «В1» – розміщення ВМ «В» на тому ж сервері, що і ВМ «А», та «В2» – розміщення ВМ «В» на іншому сервері, у цьому випадку №2. Варіант «В3» не розглядається, оскільки він нічим принципово не відрізняється від варіанту «В2» і призведе до тих самих результатів (аналогічно, якщо ВМ

«В» уже встановлена на сервері №3, слід використовувати саме його, а не сервер №2).

Якщо у наявності є сервери з іншою конфігурацією, до схеми перебору додаються відповідні вузли на кожному рівні. Сюди також відносяться випадки, коли внаслідок нестачі ресурсів дозволяється низькопріоритетні ВМ не розміщувати на жодному сервері, чи виділяти ресурси для них за залишковим принципом. Формально це описується шляхом розміщення таких ВМ на неіснуючому сервері №0 (це можливо у задачі 1, неможливо у задачі 2). В цьому випадку на першому рівні буде вже декілька вузлів (рис. 3).

Крім визначення процедури розгалуження метод гілок і меж передбачає визначення процедур оцінки верхньої та нижньої меж за допомогою наближених, але швидких алгоритмів.

При вирішенні задачі 1 (задача максимізації) для визначення оцінки зверху можна використати оптимістичний прогноз, який полягає в припущенні того, що на черговому кроці алгоритму усі ВМ, не розподілені між серверами, можуть бути розподілені таким чином, що жодна з них не отримує відмови в обслуговуванні (не буде розміщена на неіснуючому сервері №0) за умови виконання ресурсних обмежень (2).

Щоб одержати оцінку знизу, пропонується використати жадібний алгоритм, упорядкувавши ВМ за зменшенням вимог до того ресурсу, нестача якого стала причиною переходу від старого до нового плану розміщення ВМ.

Процедура відсікання гілок полягає у тому, що гілка, для якої оцінка верхньої межі менша, ніж оцінка нижньої межі хоча б однієї з інших гілок, розглядається як безперспективна та виключається з дерева рішень. Враховуючи те, що вимоги ВМ до ресурсів є невід'ємними, ця процедура може бути доповнена відсіканням тих гілок, для яких припиняють виконуватись обмеження (2).

У разі, якщо на черговому кроці буде знайдено рішення, при якому усі ВМ успішно розміщені, тобто жодна ВМ не розміщена на сервері №0, пошук можна припинити, оскільки глобальний оптимум знайдений і критерій (3) набуває максимального значення. Але, за умови наявності часових можливостей, продовження пошуку може допомогти знайти рішення, що забезпечить меншу

кількість міграцій ВМ при реалізації нового плану розташування, ніж уже знайдене рішення та його «дзеркальні» рішення.

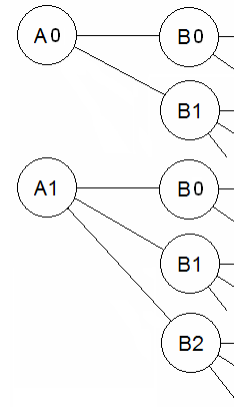


Рис. 3. Схема перебору комбінацій для неідентичних серверів

Задача 2. У разі, якщо кількість наявних обчислювальних ресурсів суттєво перевищує потреби ВМ, природною стає задача розподілити ВМ між серверами таким чином, щоб зменшити енергоспоживання за рахунок вимкнення живлення серверів, які не використовуються на цей час.

Оскільки відключати сервери для економії електроенергії має сенс лише у тому випадку, коли у наявності є вільні ресурси (випадок, коли економічний ефект від бізнес-процесів не перевищує енерговитрати на живлення серверів не розглядається), модифікуємо обмеження (1), щоб гарантувати, що кожна ВМ буде забезпечена ресурсами одного і лише одного сервера:

$$\sum_{i=1}^n x_{ij} = 1, j = 1, \dots, m. \quad (4)$$

Позначимо через e_i енергоспоживання сервера S_i коли на ньому не працює жодної ВМ.

Ознаку того, що на сервері S_i не працює жодної ВМ виразимо наступним чином:

$$d_i = \prod_{j=1}^m \overline{x_{ij}}, j = 1, \dots, m. \quad (5)$$

Тоді задачу мінімізації енергоспоживання можна сформулювати як задачу:

$$\sum_{i=1}^n d_i e_i \rightarrow \max, \quad (6)$$

при обмеженнях (2), (4), (5).

Для вирішення задачі 2 пропонується той же алгоритм, але зі спрощеною процедурою перебору, оскільки згідно з обмеженням (4)

неможливі випадки, коли якась ВМ не розміщена на жодному сервері.

Реалізація запропонованих методів

Зазначені моделі були апробовані для управління ВМ, які працювали під управлінням гіпервізора VMware ESX на двох серверах.

Схема макету наведена на рис. 4. Кожен з серверів мав два мережних інтерфейси: один – для взаємодії через мережу, другий – для передачі стану пам'яті ВМ у процесі міграції ВМ. «Жива міграція» здійснювалася за допомогою утиліти VMware vMotion, яка є складовою пакету VMware vSphere.

Управляюча програма, яка є частиною системи управління ІТ-інфраструктурою (СУІ) SmartBase ITS Control [7], була написана мовою C#.

Для взаємодії з програмним забезпеченням (ПЗ) VMware (моніторинг споживання ресурсів, управління розподілом ресурсів та міграцією ВМ) використовувалася бібліотека vSphere SDK for .NET.

На всіх ВМ була встановлена операційна система Linux. На ВМ «А» був встановлений веб-сервер Apache, для якого мовою PHP було створено ПЗ, яке генерувало динамічний контент у відповідь на запити, що надходили від клієнтів. На ВМ «В» та «С» були запущені обчислювальні процеси для створення постійного фонованого завантаження ВМ.

У результаті проведених експериментів було виявлено, що за даних умов проведення експерименту з наявних типів ресурсів (процесор, пам'ять, пропускна здатність мережі, дискова підсистема) вузьким місцем в сис-

темі виявився процесор. Тому далі на графіках зображені лише завантаження процесорів, моніторинг яких здійснювався засобами VMware, та статистика обробки клієнтських запитів, яка збиралася та оброблялася за допомогою написаного ПЗ.

Потік клієнтських запитів поступово збільшувався. На графіках (рис. 5–7) видно, що після того, як один з параметрів (у цьому експерименті – завантаження процесора на сервері 1), протягом заданого часу перевищував заданий поріг, СУІ прийняла рішення щодо необхідності перенесення ВМ «А» на сервер 2, що дозволило ліквідувати вузьке місце та відновити динаміку обробки запитів. Порогові значення та максимальна тривалість їх перевищення задається в конфігурації СУІ окремо для кожного з параметрів, виходячи з вимог, оговорених у SLA. У цьому експерименті рішення про необхідність міграції ВМ приймалося у разі, якщо середнє завантаження процесора перевищувало 90% протягом трьох хвилин.

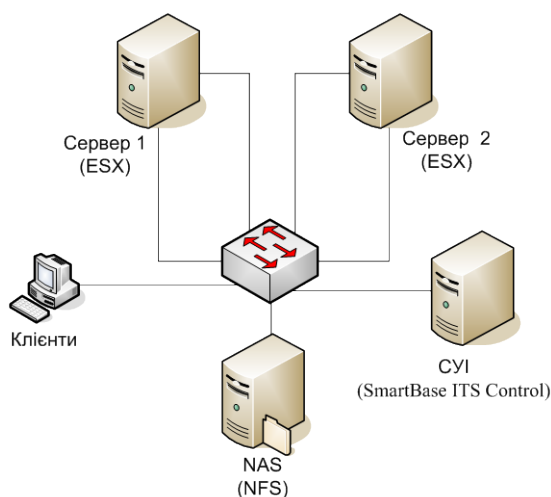


Рис. 4. Схема макету

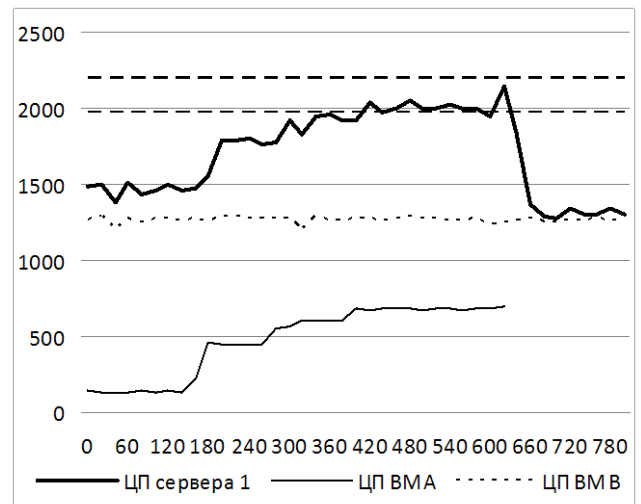


Рис. 5. Завантаження ЦП сервера 1

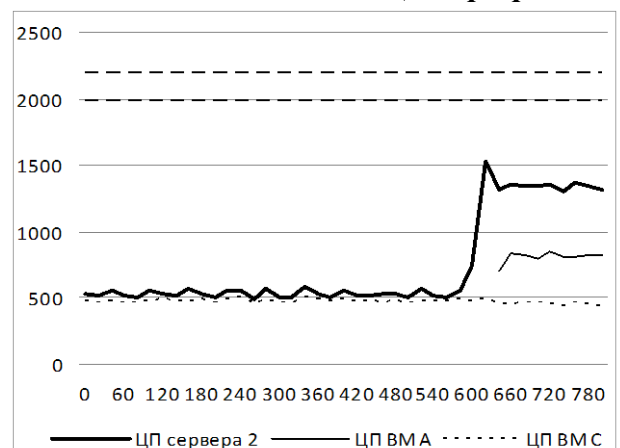
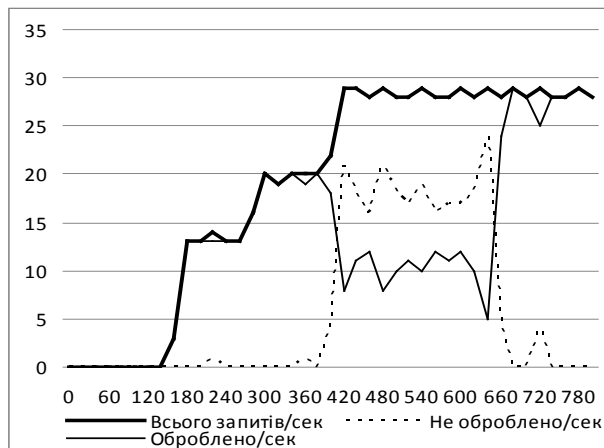


Рис. 6. Завантаження ЦП сервера 2**Рис. 7. Надходження та обробка запитів**
Висновки

У статті розглянуті питання управління розміщенням ВМ при серверній віртуалізації з можливістю здійснення «живої міграції» ВМ. Запропоновані моделі управління міг-

рацією ВМ та розподілом ресурсів між ними для випадків нестачі та надлишку обчислювальних ресурсів. Ці моделі можуть бути застосованими для вирішення задач балансування навантаження між серверами шляхом міграції ВМ, для забезпечення високого рівня доступності найбільш важливих сервісів, та для зниження експлуатаційних витрат, а саме економії електроенергії шляхом відключення електроживлення окремих серверів з попереднім перенесенням ВМ з них на інші сервери. Для вирішення зазначених задач запропонована модифікація алгоритму гілок і меж. Запропоновані моделі і методи реалізовані в СУІ SmartBase ITS Control. Наведені результати виконаних експериментів, які підтвердили працездатність та ефективність зазначених моделей і методів.

Список посилань

1. Теленик С.Ф. Управління навантаженням і ресурсами центрів оброблення даних при виділених серверах / С.Ф. Теленик, О.І. Ролік, М.М. Букасов, Р.В. Римар, К.О. Ролік // Автоматика. Автоматизація. Електротехнічні комплекси та системи. — 2009. — №2 (24). — С. 122—136.
1. David Marshall, Wade A. Reynolds, Dave McCrory. Advanced Server Virtualization: VMware and Microsoft Platforms in the Virtual Data Center. — Auerbach Publications, 2006. — 760 p.
2. Теленик С.Ф., Ролік О.І., Букасов М.М. Моделі управління розподілом обмежених ресурсів в інформаційно-телекомунікаційній мережі // Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка. — К.: Екотех. — 2006. — №44. — С. 243—246.
3. Теленик С.Ф., Ролік О.І., Терещенко П.І., Букасов М.М. Забезпечення процесів діяльності з визначеним рівнем надійності в ІТС спеціального призначення // 36. наук. праць ВІПІ НТУУ «КПІ». — №3. — К., 2007. — С. 134—138.
4. Теленик С.Ф., Ролік О.І., Букасов М.М. Технологія управління ІТ-інфраструктурою на основі ресурсного підходу // Вісник ЖДТУ. — 2008. — №4(47). — С. 180—189.
5. Теленик С.Ф., Ролік О.І., Букасов М.М. Моделі і методи розподілу ресурсів в інформаційних системах з віртуалізацією серверів// Проблеми й перспективи розвитку ІТ-індустрії: Матеріали 1-й Міжнародної науково-практичної конференції. — Харків: ХНЕУ, 2009. — С. 46—47.
6. Теленик С.Ф., Ролік О.І., Букасов М.М., Соколовський Р.Л. Система управління інформаційно-телекомунікаційною системою корпоративної АСУ// Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка. — К.: «ВЕК+», — 2006. — № 45. — С. 112—126.

Поступила в редакцію 3.12.2009