УДК 004.432 DOI: 10.31673/2518-7678.2019.041426

Макаренко А.О., д.т.н.; **Гринкевич Г.О.,** к.т.н.; **Легомінова С.В.,** д.т.н.

СПЕЦІАЛЬНІ МОВИ ПРОГРАМУВАННЯ ДЛЯ СТВОРЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМ НАДАННЯ ТЕЛЕКОМУНІКАЦІЙНИХ ПОСЛУГ

Makarenko A.O., Grynkevych G.A., Lehominova S.V. Current programming language system for providing telecommunication services

In the paper the research results of modern programming languages used to develop programs for individual units of modern systems of telecommunication services. It was determined that one of the most promising and one that is booming implementations approach to the creation of modern systems of telecommunication services is technology OpenFlow Softswitch. This technology is a logical continuation of the post NGN (IP Multimedia Subsystem) and Softswitch component of such systems. Its main instrument is the OpenFlow specification, which describes the basic components of OpenFlow-networking principles and interaction of the components. Organization for Standardization specifications are ONF - Open Networking Foundation. The results of the analysis of existing software for modern systems of telecommunication services following conclusions can be made that most existing software for modern systems of telecommunication services written in high-level languages: C, C ++, Python, Java, Rybu; cross-platform are Java-only controllers FloodLight, Beacon and Maestro, others work on Linux-platforms; dominant programming languages for switching systems are C and Java. It is shown that the results of existing research software for the creation of modern systems of telecommunication services and open source software, the most promising and those that are actively developing developments controllers Beacon, FloodLight (in the language of Java), Trema and MuL (in C). Determined that based programming languages C and Java, the programming language developed special networks on the one hand represent high-level flow control methods in the network, on the other hand hide all the details on proper and correct installation rules for low-level switches. Currently the following languages are Frenetic, NetCore and Nettle. The most popular programming language among programmers, including and modern systems of telecommunication services is language Java. There are 4 main reasons: cross-platform; friendly syntax; object-oriented language; memory management.

Keywords: programming language, telecommunication services, NGN, OpenFlow Softswitch, C, C++, Python, Java, Rybu, Frenetic, NetCore i Nettle

Макаренко А.О., Гринкевич Г.О., Легомінова С.В. Сучасні мови програмування для систем надання телекомунікаційних послуг

В статті висвітлено результати досліджень сучасних мов програмування, які використовуються з метою розробки програм для окремих вузлів сучасних систем надання телекомунікаційних послуг. Визначено, що однією з найбільш перспективних підходів до створення сучасних систем надання телекомунікаційних послуг є технологія OpenFlow Softswitch. Показано, що на основі мов програмування С та Java, розроблено спеціальні мови програмування мереж - Frenetic, NetCore і Nettle. Найбільш популярною мовою програмування серед програмістів, в т.ч. і сучасних систем надання телекомунікаційних послуг, є мова Java.

Ключові слова: мова програмування, телекомунікаційні послуги, NGN, OpenFlow Softswitch, C, C++, Python, Java, Rybu, Frenetic, NetCore і Nettle

Макаренко А.А., Гринкевич А.А., Легоминова С.В. Современные языки программирования для систем предоставления телекоммуникационных услуг

В статье отражены результаты исследований современных языков программирования, используемых для разработки программ для отдельных узлов современных систем предоставления телекоммуникационных услуг. Определено, что одним из наиболее перспективных подходов к созданию современных систем предоставления телекоммуникационных услуг является технология OpenFlow Softswitch. Показано, что на основе языков программирования С и Java, разработаны специальные языки программирования сетей - Frenetic, NetCore и Nettle. Наиболее популярным языком программирования среди программистов, в т.ч. и современных систем предоставления телекоммуникационных услуг, является язык Java.

Ключевые слова: язык программирования, телекоммуникационные услуги, NGN, OpenFlow Softswitch, C, C++, Python, Java, Rybu, Frenetic, NetCore i Nettle

Вступ

Донедавна доставка телекомунікаційних послуг була вертикальною, вимагала спеціалізованої інфраструктури та орієнтації на те, що вийшло у оператора. Важливим елементом було абонентське обладнання. Сьогодні, коли абонент працює в декількох мережах одночасно, така сервісна практика стає незручною. Тому цілком логічним продовженням розвитку телефонних «інтелектуальних мереж» (IN) в епоху NGN стала архітектура IMS (IP Multimedia Subsystem), орієнтована на надання будь-якого сервісу в будь-якому місці мережі з пакетною комутацією. У той час як на насичених ринках прибутки від традиційних послуг зменшуються, завданням IMS є побудова єдиної мережі, завдяки якій абонент може отримати вказані вище послуги, включаючи конвергентні послуги від мереж фіксованого та мобільного зв'язку. Передумовою цього є зменшення в загальному сервісному пакеті операторів частки голосових сервісів [1].

IMS як концепція використовується операторами для зменшення стратегічних ризиків при виборі шляхів технічного розвитку. В цілому IMS – це інструмент, який дає оператору на тлі стрімкозростаючої конкуренції утримувати абонента за допомогою надання мультисервісу на універсальний термінал (конвергенція сервісу).

Мережі наступного покоління мають два варіанти побудови [1-7]:

- з використанням програмних комутаторів (Softswitch) і медіашлюзи (MGW);
- з використанням програмно-апаратного комплексу IMS.

Архітектури IMS і Softswitch мають рівневий розподіл (абонентських пристроїв і транспорту, управління викликами і сеансами, серверів додатків), при чому межі цих логічних рівнів проходять в обох концепціях/архітектурах практично в одних і тих же місцях. Просто в архітектурі Softswitch зазвичай зображають мережеві пристрої, а архітектура IMS визначається на рівні функцій. Абсолютно однакові також ідеї надання всіх послуг на базі IP-мережі та розподілу функцій управління викликом і комутації.

Перш за все, Softswitch - це обладнання конвергентних мереж. Функція управління шлюзами являється домінуючою. У свою чергу, IMS проектувалася в рамках мобільної спільноти 3GPP [6], повністю базується на IP. Основним її протоколом є SIP, що дозволяє встановлювати однорангові сесії між абонентами і використовувати IMS тільки як систему, що надає сервісні функції з безпеки, авторизації, доступу до послуг і т.д. Функція управління шлюзами і сам медіашлюз, в даному випадку, лише засіб для зв'язку з абонентами фіксованих мереж (телефона мережа загального користування).

Але в IMS частково згладжуються проблеми сумісності обладнання, властиві «пулу» рішень Softswitch [1-7], оскільки взаємодія функціональних модулів регулюється стандартами. Новий підхід до надання послуг виявився надзвичайно вдалим і забезпечив роумінг послуг, що повинно принести додатковий прибуток оператору. Використання у фіксованих мережах NGN [7] та в мобільних мережах 3G однакової системи IMS, власне, і відкриває перспективу конвергенції фіксованих та мобільних мереж (FMC) на операторському рівні. Оператору надаються широкі можливості по управлінню мережними ресурсами, оптимізації процесу доставки послуг та розширення клієнтської бази.

Передумовами міграції «традиційних» мереж «традиційних» операторів до IMS є подальша стратегічна безперспективність позиціонувати себе на ринку як «бітова» труба, спостерігаючи лише «втрачену користь»; постійний відтік прибутків до провайдерів VoIP і ISP; відносно високі поточні експлуатаційні витрати і висока вартість введення нових сервісів; «Історичне» тяжіння до розподіленої, відкритої та стандартизованої архітектури; зрілість технологій IMS і SIP, а також наявність відповідних стандартів.

У процесі створення мереж NGN на перше місце виходять питання використання мов програмування з метою розробки програми для окремих вузлів сучасних систем надання телекомунікаційних послуг. При цьому на теперішній час недостатньо висвітлені дослідження даного питання, виконані на такому ж рівні, як це було зроблено для додаткових послуг цифрових АТС або для послуг інтелектуальної мережі в ТМЗК, тобто на рівні достатньому для аналізу і синтезу, що і обумовлює актуальність даної статті.

Ступінь наукової розробки. Мережі NGN в даний час розглядаються у багатьох роботах, включаючи монографії, численні статті в науково-технічних журналах і доповіді на конференціях. У роботах і в документах ряду організацій, таких як ITU, IETF, ETSI, IMS Forum, 3GPP, розглянуті технічні рішення створення мереж NGN [1-7]. Наукові основи для їх реалізації містяться в роботах В.К. Стеклова, Л.Н. Беркман, М.М. Климаша, В.В. Поповського, Boxma O.J., Everitt D., Ferguson M.J., Fuhrmann S.W., Levy H., Takagi H. та інших вчених [8].

Незважаючи на велику кількість публікацій по NGN, серед них поки що практично відсутні аналітичні дослідження, спрямовані на вивчення використання мов програмування для розробки програм окремих вузлів сучасних систем надання телекомунікаційних послуг.

Мета роботи - дослідити сучасні мови програмування, які використовуються з метою розробки програм для окремих вузлів сучасних систем надання телекомунікаційних послуг.

1. Архітектура IP-мультимедійної підсистеми

Архітектура IMS дозволяє надавати новітні мультимедійні послуги на універсальному обладнанні з використанням різних способів доступу, зменшуючи ризики, пов'язані з введенням окремих послуг, і налагоджуючи внутрішні бізнес-процеси оператора. Але перш ніж будувати що-небудь, бажано зрозуміти, наскільки все це вам доступно. І для початку доцільно оцінити ризики, пов'язані з новою ініціативою.

Зокрема, підсистема IMS забезпечує надання послуг в пакетному середовищі: це може бути або вузол телефонної мережі, або телематичний вузол, або вузол мережі передачі мови в мережі передачі даних. У всіх трьох випадках чинна нормативно-правова база накладає суттєві обмеження на реалізацію FMC (наприклад, вузол телефонної мережі не може мати підключення абонентів по Ethernet). Крім того, залежно від обраної базової послуги, IMS потрапляє під обмеження відповідної служби і по взаємодії різних послуг. Правда, з певними розбіжностями IMS відповідає вимогам до обладнання відповідних служб, але в підсумку варто продумати, як мережа на базі IMS буде віддаватися в експлуатацію. Слід також мати на увазі, що сама по собі IMS не є набором сервісів. Ядро IMS лише забезпечує реалізацію базового набору внутрішньо мережевих послуг на зразок авторизації, управління з'єднаннями і мультимедійними сесіями. Найбільш цікаві додаткові послуги реалізуються сервісними платформами (SDP), які часто являють собою автономні і дешеві технічні рішення. Але їх теж треба купувати і підключати. Таким чином, важливо, що ключові переваги IMS по швидкому виведенню сервісів на ринок багато в чому визначаються властивостями не цієї підсистеми, а сервісних платформ (SDP) та інструментальних засобів (серверів додатків - AS). Сама по собі IMS не несе в собі будь-яку killer application. Відповідно, визначити економічний ефект від впровадження IMS теж складно. Адже IMS - лише зручний механізм доставки та обробки (тарифікації) сервісів. Крім того, вимога до нових послуг ринком найчастіше нижчі від передбачуваних - впливають і соціокультурні обмеження, та інертність публіки, і погане внутрішнє покриття мобільних мереж, і економічні кризи, і ті ж регуляторні обмеження.

Архітектура IMS, яка задовольняє викладеним вище вимогам, визначена в стандартах 3GPP, Європейського інституту стандартів зв'язку ETSI і форуму Parlay (рис. 1) [2, 3, 9].

Уніфікована сервісна архітектура IMS підтримує широкий спектр сервісів, заснованих на гнучкості протоколу SIP (Session Initiation Protocol), і множини серверів аплікацій, які надають як звичайні телефонні послуги, так і нові сервіси (обмін миттєвими повідомленнями, миттєвий багатоточечний зв'язок, передача відеопотоків, обмін мультимедійними повідомленнями тощо). Сервісна архітектура являє собою набір логічних функцій, які можна поділити на три рівні: рівень транспорту та абонентських пристроїв і шлюзів, рівень управління сеансами й рівень серверів аплікацій [1, 9].

Рівень транспорту та абонентських пристроїв. На цьому рівні ініціюється й здійснюється сигналізація SIP, необхідна для встановлення сеансів і надання базових послуг, таких як перетворення мови з аналогової або цифрової форми в IP-пакети з використанням

протоколу RTP (Realtime Transport Protocol). На цьому рівні функціонують медіашлюзи, які перетворюють базові потоки VoiP у телефонний формат TDM. Медіасервер надає різні медіасервіси, у тому числі конференц-зв'язок, збір тонових сигналів, розпізнавання мови тощо.

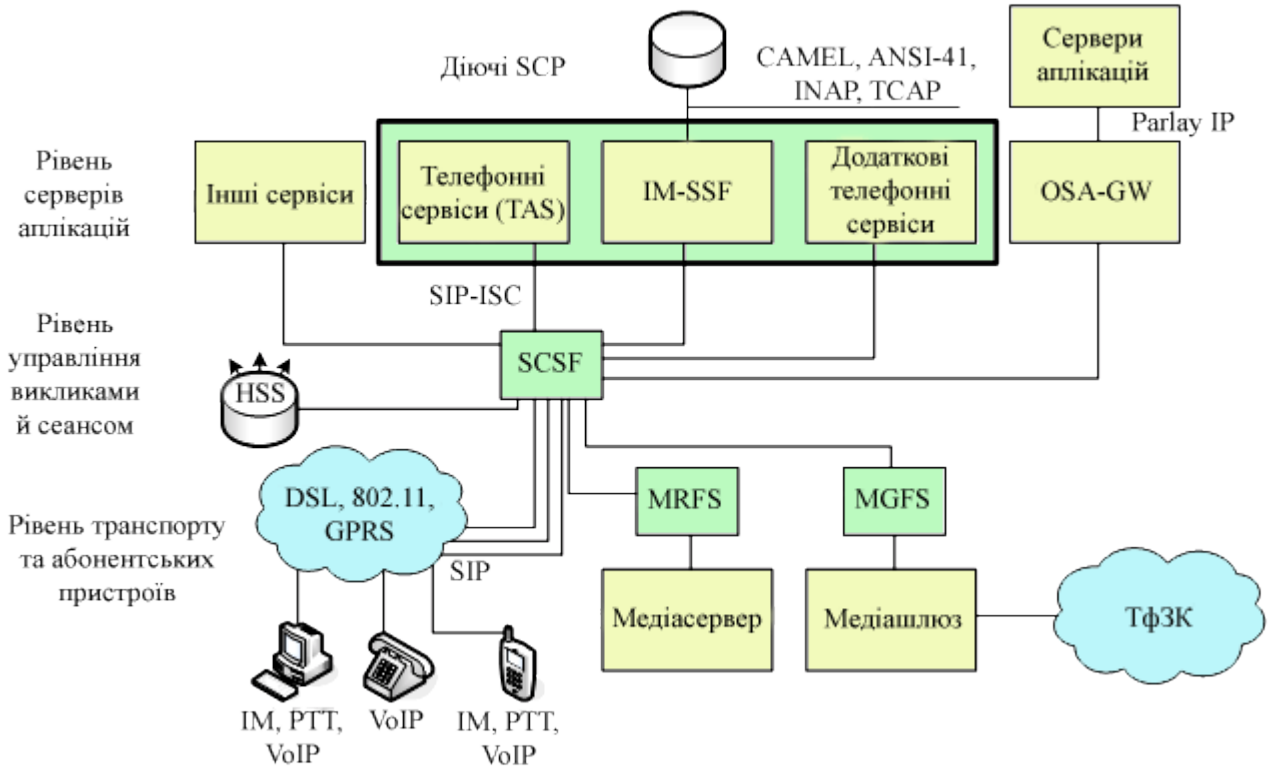

Рис. 1. Спрощений варіант архітектури IMS

Ресурси медіасерверу доступні всім аплікаціям, тобто будь-яка аплікація (голосова пошта, безкоштовний номер 800, інтерактивні VXML-сервіси тощо), якій необхідно відтворити оповіщення або отримати цифри набраного номера, може використати загальний сервер. Медіасервери також підтримують нетелефонні функції, наприклад, тиражування мовленнєвих потоків для надання сервісу миттєвого багатоточкового зв'язку (PTT). При використанні для різних сервісів загального пула медіасерверів відпадає необхідність у плануванні й інжинірингу медіаресурсів для кожної окремої аплікації.

Рівень управління викликами й сеансами. На цьому рівні реалізується функція управління викликами й сеансами CSCF (Call Session Control Function), яка реєструє абонентські пристрої й направляє сигнальні повідомлення протоколу SIP до відповідних серверів аплікацій. Функція CSCF взаємодіє з рівнем транспорту й доступу для забезпечення якості обслуговування за всіма сервісами. Рівень управління викликами й сеансами включає сервер абонентських даних HSS (Home Subscriber Server), де централізовано зберігаються унікальні сервісні профілі всіх абонентів. Профіль містить поточну реєстраційну інформацію (наприклад, IP-адресу), дані роумінгу, дані щодо телефонних послуг (наприклад, номер переадресації), дані щодо обміну миттєвими повідомленнями (список абонентів), параметри голосової пошти (наприклад, вітання) тощо.

Централізоване зберігання дозволяє різним аплікаціям використовувати ці дані для створення персональних довідників, інформації про присутність у мережі абонентів різних категорій, а також об'єднаних послуг. Централізація також істотно спрощує адміністрування користувальницьких даних і гарантує однорідне подання активних абонентів за різними сервісами. На рівні управління викликами й сеансами також реалізується функція управління медіашлюзами MGCF (Media Gateway Control Function), які забезпечують взаємодію сигналізації SIP із сигналізацією інших медіашлюзів (наприклад, H.248). Функція MGCF

управляє розподілом сеансів по множині медіашлюзів, для медіасерверів це виконується функцією MSFC (Media Server Function Control).

Рівень серверів аплікацій (програм). Цей рівень містить сервери аплікацій, які забезпечують обслуговування кінцевих користувачів. Архітектура IMS і сигналізація SIP забезпечують достатню гнучкість для підтримки різноманітних телефонних й інших серверів аплікацій.

Сервер аплікацій телефонії. Архітектура IMS підтримує множину серверів аплікацій для телефонних сервісів. Сервер телефонних аплікацій TAS (Telephony Application Server) приймає й оброблює повідомлення протоколу SIP, а також визначає, яким чином має бути ініційований вихідний виклик. Сервісна логіка TAS забезпечує базові сервіси обробки викликів, включаючи аналіз цифр, маршрутизацію, встановлення, очікування й перенаправлення викликів, конференц-зв'язок тощо. TAS також забезпечує сервісну логіку для звернення до медіасерверів за необхідності відтворення оповіщень і сигналів проходження виклику.

Якщо виклик ініційований у ТфЗК, сервер TAS забезпечує сигналізацію SIP до функції MGCF для видачі команди медіашлюзам на перетворення бітів мовленнєвого потоку TDM (ТфЗК) у потік IP RTP, і направлення його на IP-адресу відповідного IP-телефону. TAS оброблює тригерні точки виклику IN відповідно до моделі телефонного виклику. При досягненні викликом тригерної точки TAS припиняє обробку виклику й перевіряє профіль абонента (де міститься інформація про те, які мають бути задіяні сервери аплікацій) на необхідність виконання додаткових послуг. TAS формує повідомлення управління ISC (SIP IP Multimedia Service Control) і передає управління викликом відповідному серверу аплікацій. Цей механізм може бути використаний для виклику як успадкованих сервісів IN, так і нових сервісів на базі SIP. В одному повідомленні IMS можуть міститися дані про декілька TAS, які надають певні послуги різним типам абонентських пристроїв [6, 7, 9].

Наприклад, один сервер TAS надає послуги IP Centrex (часткові плани нумерації, загальні довідники, автоматичний розподіл викликів тощо), інший сервер підтримує УАТС і надає послуги VPN. Взаємодія декількох серверів аплікацій здійснюється за допомогою сигналізації SIP-I для завершення викликів між абонентськими пристроями різних класів.

Функція комутації послуг IM-SSF. Функція комутації послуг IM-SSF (IP Multimedia - Services Switching Function) забезпечує взаємодію повідомлення SIP з відповідними повідомленнями CAMEL, ANSI-41, підсистем INAP (Intelligent Network Application Protocol) або TCAP (Transaction Capabilities Application Part). Ця взаємодія дозволяє підтримуваним IMS IP-телефонам отримувати доступ до сервісів визначення імені сторони, яка викликає, безкоштовного номера 800, переносу локального номера тощо.

Додаткові сервери телефонних аплікацій. Прикладний рівень може також містити автономні незалежні сервери, які надають додаткові послуги в будь-якій стадії виклику за допомогою тригерів. До таких послуг належать набір номера, переадресація та встановлення конференц-зв'язку, послуги голосової пошти, послуги інтерактивної мовленнєвої взаємодії (IVR), VoiP VPN, попередньо оплачений білінг, блокування вхідних і вихідних викликів.

Інші сервери аплікацій. На прикладному рівні також можуть перебувати сервери аплікацій SIP, які не використовують модель телефонного виклику. Такі сервери взаємодіють із клієнтами абонентських пристроїв для надання сервісів PTT, сервісів присутності тощо. Реалізація сервісів на базі SIP (нетелефонних сервісів) у загальній архітектурі IMS дозволяє здійснювати взаємодію двох видів сервісів і створювати нові змішані послуги. Як приклад можна навести відображення на дисплеї списку абонентів із вказівкою статусу присутності в мережі. Інший приклад - використання одного попередньо оплаченого рахунку для оплати послуг телефонії й відео за запитом.

Шлюз відкритого сервісного доступу OSA-GW. Гнучкість архітектури IMS дозволяє сервіс-провайдерам додавати сервіси до мережі VoIP шляхом взаємодії з діючими аплікаціями, або ж шляхом інтеграції з власних або розроблених третіми фірмами серверів аплікацій на базі SIP. Окрім того, сервіс-провайдери можуть надати можливість своїм

клієнтам розробляти й впроваджувати сервіси, які задіють ресурси мережі VoiP. Наприклад, підприємство може реалізувати сервіс автоматичної генерації мовленнєвого або миттєвого повідомлення щодо доставки замовлення.

Однак найчастіше працюючі на таких підприємствах виробники незнайомі із протоколами телефонної сигналізації (SS7, ANSI41, CAMEL, SIP, ISDN тощо), хоча й мають освіту в галузі інформаційних технологій. Для розв'язання цієї проблеми Форум Parlay у тісному співробітництві з 3GPP й ETSI розробив прикладний програмний інтерфейс Parlay API для організації взаємодії з телефонними мережами. Взаємодія SIP і Parlay API здійснюється за допомогою шлюзу OSA-GW (Open Services Access, Gateway), який входить до прикладного рівня архітектури 3GPP IMS. Інші прикладні сервери, як згадувалося вище, забезпечують взаємодію між SIP і протоколами телефонії (ANSI-41, CAMEL, INAP, TCAP, ISUP тощо). Шлюз OSA-GW дозволяє корпоративним аплікаціям на базі Parlay отримати доступ до інформації щодо присутності і стану виклику, встановлювати й розривати сеанси зв'язку, незалежно управляти сегментами виклику. Шлюз OSA-GW реалізує інтерфейс Parlay Framework, який дозволяє корпоративним серверам аплікацій реєструватися в мережі й управляє доступом до мережних ресурсів.

Розвиток архітектури IMS. Більшість із описаних вище сервісів були вузькосмуговими сервісами передачі мовлення й даних. Однак сигналізація SIP й архітектура IMS підтримують і широкосмугові мультимедійні сервіси, такі як віщальне телебачення із багатоадресними (IP multicast) відеопотоками, відео за запитом, відеоспостереження, відеотелефонія, відеоконференц-зв'язок, віртуальні лекційні зали й багато чого іншого. Для реалізації таких сервісів у мережі мають бути встановлені додаткові мультимедійні сервери аплікацій і абонентські пристрої (рис. 2) [2, 7, 9].

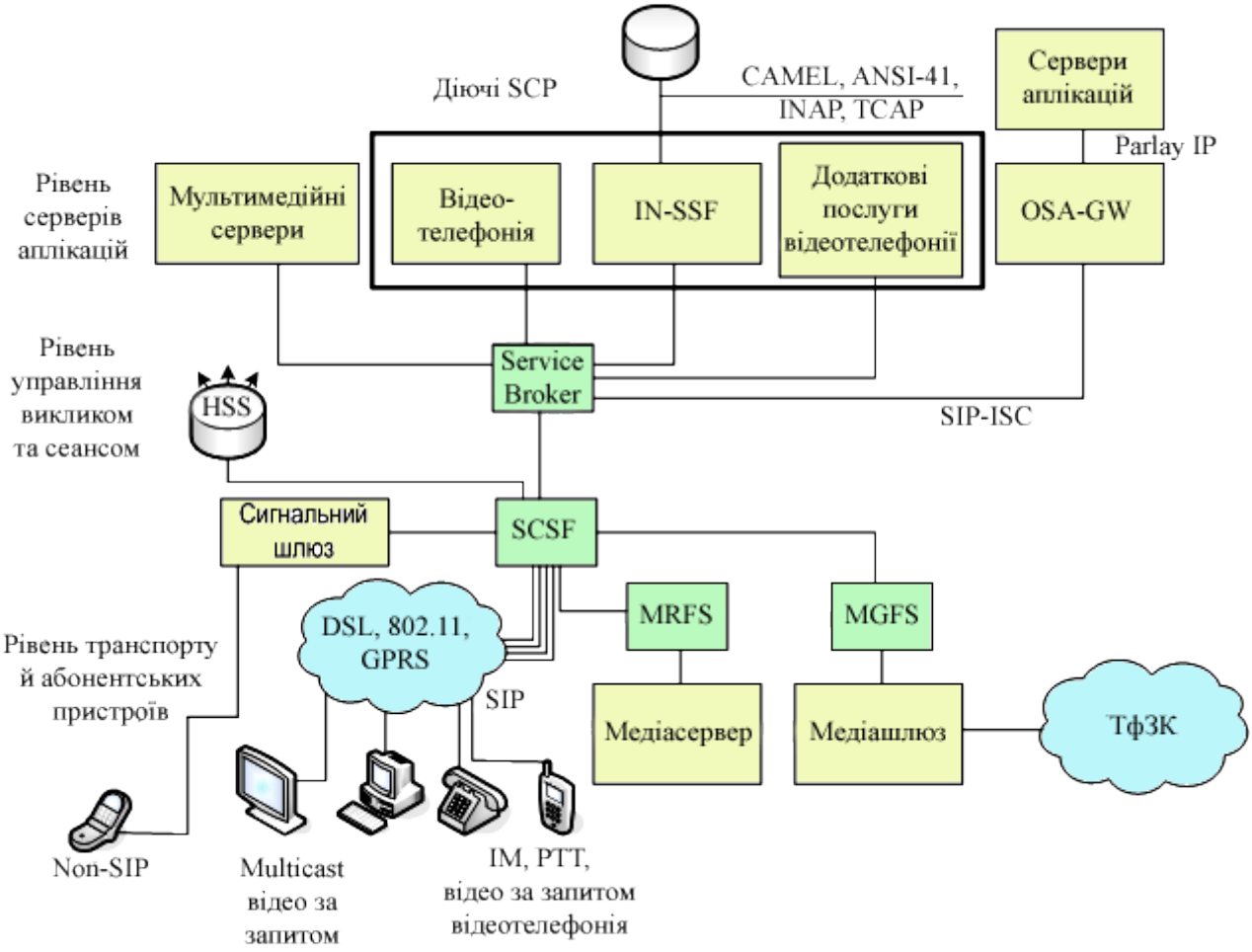

Рис. 2. Додаткові послуги IMS

З розширенням сфери застосування мультимедійних послуг виникла необхідність переходу від використовуваних сьогодні базових механізмів забезпечення якості обслуговування на більш ефективні механізми. Крім моніторингу доступної пропускної здатності необхідно контролювати кількість активних сеансів зв'язку реального часу. В архітектурі IMS абонентські пристрої й сервери аплікацій VoiP і широкосмугових мультимедійних послуг надсилають запити на ініціювання сеансу через загальний елемент CSCF. Функція CSCF визначає рівні трафіка, взаємодіючи з мережею транспорту й доступу, і може відмовити у встановленні додаткових сеансів. Відповідно, для підтримки цих поширених абонентських пристроїв у мережі IMS необхідно забезпечити взаємодію підтримуваних ними стандартів сигналізації й протоколу SIP. Із цією метою вже запропоновані нові граничні сигнальні шлюзи.

Основна перевага IMS в тому, що послуга одноманітно зможе працювати на мережі всіх операторів. Якщо мережа буде реалізована - уніфікований сервіс стане доступний в будь-якій точці присутності абонента, як це відбувається сьогодні в мережі Інтернет. Це, власне, і є кінцевою метою всіх операторів, які впроваджують IMS.

2. Новітня технологія OpenFlow Softswitch

Однією з найбільш перспективних і такої, що бурхливо розвивається реалізацій підходу до створення сучасних систем надання телекомунікаційних послуг є технологія OpenFlow Softswitch. Дана технологія є логічним продовженням систем пост NGN (IP Multimedia Subsystem) та складової таких систем, як Softswitch. Основним її документом є специфікація OpenFlow [10], в якій описуються основні компоненти OpenFlow-мереж, принципи роботи і взаємодії компонентів. Організацією стандартизації для специфікації є ONF - Open Networking Foundation (www.opennetworking.org). До засновників ONF входять всі без виключення міжнародні виробники телекомунікаційного обладнання. Варто відзначити найвідоміших засновників - Nokia, Cisco, Huawei, Facebook, Google.

OpenFlow - протокол управління процесом обробки даних, що передаються по мережі передачі даних маршрутизаторами і комутаторами, який реалізує технологію програмно-конфігурується мережі.

Протокол використовується для управління мережевими комутаторами і маршрутизаторами з центрального пристрою - контролера мережі (наприклад, з сервера або навіть персонального комп'ютера). Це управління замінює або доповнює працюючу на комутаторі (маршрутизаторі) вбудовану програму, яка здійснює побудова маршруту, створення карти комутації і т. д. Контролер використовується для управління таблицями потоків комутаторів, на підставі яких приймається рішення про передачу прийнятого пакета на конкретний порт комутатора. Таким чином в мережі формуються прямі мережеві з'єднання з мінімальними затримками передачі даних і необхідними параметрами.

Згідно специфікації основним елементом системи створення сучасних систем надання телекомунікаційних послуг є OpenFlow Softswitch комутатор та контролер (рис. 3).

Порівняльна таблиця переваг технології OpenFlow Softswitch показана нижче (табл. 1) [11].

Більшість існуючих ПЗ для створення сучасних систем надання телекомунікаційних послуг написана на мовах високого рівня: C, C++, Python, Java, Rybu [12, 13].

Кросплатформенними являються тільки Java-контролери FloodLight, Beacon і Maestro, інші працюють на Linux-платформах.

Усі розглянуті ПЗ для створення сучасних систем надання телекомунікаційних послуг є багатопотоковими, окрім POX.

Усі ПЗ підтримують протокол OpenFlow версії 1.0 і, отже, працюють з усіма існуючими програмними і апаратними комутаторами.

Усі контролери розроблялися з абсолютно різними цілями, тому немає універсального контролера, який дозволяє вирішувати усі завдання в різних середовищах (WAN, центри

обробки даних). Кожен контролер має свої переваги і недоліки. Наприклад, контролер POX дозволяє швидко розробляти додатки для нього в порівнянні з іншими. Проте C, C++, Java - контролери, що вимагають більше трудомісткого написання додатків, істотно виграють по продуктивності. Розподілені контролери мають складнішу архітектуру і вимагають складніших алгоритмів, проте дозволяють підвищити продуктивність, масштабованість і надійність рівня управління IMS.

Таблиця 1. Переваги технології OpenFlow Softswitch

Softswitch	OpenFlow Softswitch
Підтримка багатьох протоколів: STP, RIP, OSPF, BGP...	Всі обчислення і логіка виконуються контролером та OpenFlow Softswitch
Продавець надає спеціальні інтерфейси	Загальний програмний інтерфейс
Softswitch для комутації L2; Маршрутизатор для маршрутизації L3	Один пристрій OpenFlow Softswitch для L2-L4

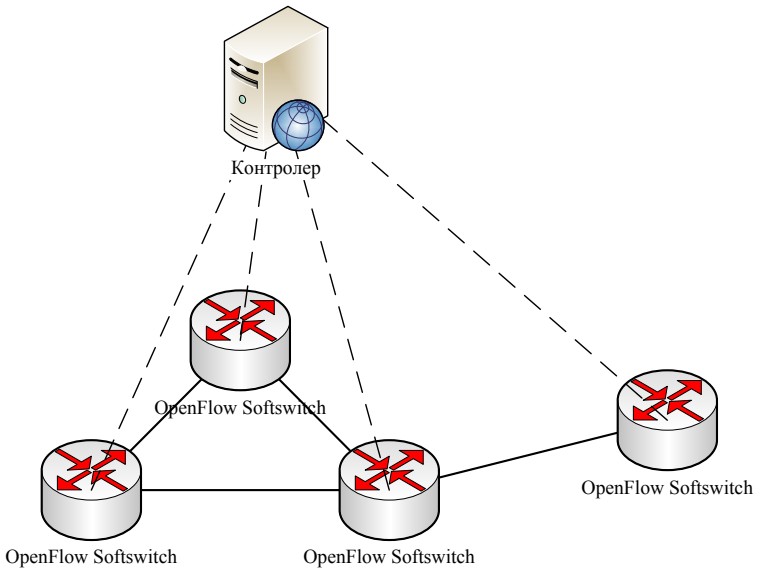

Рис. 3. Основні компоненти OpenFlow Softswitch мережі

З точки зору архітектури усі контролери використовують модульний принцип, що дозволяє нарощувати мережеві сервіси для додатків усередині ПЗ для систем комутації, розширюючи функціональні можливості контролера. Усі контролери мають дуже схожий базовий набір сервісів: сервіси для обробки OpenFlow повідомлень, підтримки з'єднань з комутаторами, механізми підписки/генерації подій для модулів і додатків, механізми управління мережевими пристроями і побудови топології мережі.

Для усіх контролерів на даний момент створені репозиторії найрізноманітніших додатків, більшість з яких носить дослідницький або учбовий характер. Одним з таких, що розвиваються і має багато додатків є репозиторій контролера FloodLight.

З точки зору надійності, існуючі контролери не здатні протистояти загрозам, пов'язаним з відмовами контролерів і їх додатків, оскільки такі механізми в них не закладені. Виключенням є розподілені контролери: Onix, які здатний передавати управління з вузла контролера, що відмовив, на справний, і Kandoo, в якому у разі відмови локального контролера управління передається на верхній рівень - кореневому контролеру [12, 13].

Про продуктивність і масштабованість контролерів по доступних матеріалах і інтернет

ресурсам судити украй складно. Це пов'язано з наступними проблемами: старіння досліджень і їх відтворюваність - проекти розвиваються, і характеристики контролерів змінюються, відмінності в методиках досліджень, необ'єктивність результатів.

На основі сформульованих вимог до контролерів і проведеного аналізу отримані результати, представлені в табл. 2 [12, 13]. У цій таблиці курсивом виділені контролери, що представляють найбільший інтерес по результатах порівняння.

За підсумками дослідження існуючих ПЗ для систем комутації з відкритим початковим кодом, найбільш перспективними і такими, що активно розвиваються розробниками являються: контролери Beacon, FloodLight (на мові Java), Trema і MuL (на мові C).

3. Спеціальні мови програмування для створення програмного забезпечення систем надання телекомунікаційних послуг

Надивлячись на широке застосування мов високого рівня C, C++, Python, Java, Rybu на основі мов програмування C, Python та Java, розроблено спеціальні мови програмування мереж, які, з одного боку, представляють високорівневі способи управління потоками в мережі, а з іншого боку, приховують усі деталі по правильній і коректній установці низькорівневих правил на комутатори. На даний час такими мовами є Frenetic, NetCore і Nettle.

Таблиця 2. Порівняння ПЗ з відкритим початковим кодом

	Назва ПЗ	Відкритість	Розвиток	OF≥1.0	Платформа Linux	Багато-поточність
1	NOX Classic	+	-	+	+	-
2	NOX	+	-	+	+	+
3	POX	+	+	+	+	-
4	SNAC	+	-	+	+	-
5	*Beacon*	+	+	+	+	+
6	Maestro	+	-	+	+	+
7	*Floodlight*	+	+	+	+	+
8	*Trema*	+	+	+	+	+
9	*MUL*	+	+	+	+	+
10	ONIX	-	-	+	+	+
11	Kandoo	-	-	+	+	+

Frenetic - це одна з перших мов програмування мереж [14]. Вбудований в мову загального програмування Python. Складається з двох частин:

- обмежена, але високорівнева декларативна мова запитів до мережі - network query language*;*
- функціональна мова загального призначення для реактивного управління політиками маршрутизації в мережах - network policy management library.

NetCore (Network Core Programming) [15] є розвитком мови Frenetic. NetCore пропонує більш високорівневу мову за завданням політик маршрутизації:

1) Можна використовувати довільні функції по обробці пакетів, а не тільки заздалегідь задані.

2) Можна використовувати групові символи (wildcards), а не тільки правила з точним

зіставленням заголовка пакетів.

3) Можна задавати правила проактивно (до того моменту, як вони знадобляться).

Усе це є першим кроком до можливості завдання динамічних правил маршрутизації, які враховуватимуть поточні стани мережі. Крім того, NetCore додатково надає можливість аналізу історії трафіку.

Nettle [16] є попередником Frenetic. Це був перший підхід, який використовував функціональну реактивну мову (FRP) для програмування OpenFlow комутаторів. Nettle приймає на вхід потік OpenFlow подій (наприклад, switch Join, port_change, packet_in*)* і видає потік OpenFlow повідомлень (install, uninstall, query_stats). Nettle представляє нижчий рівень в розробці мережевих додатків. З іншого боку Frenetic - це надбудова над контролером NOX, тоді як Nettle є повноцінним рішенням для розробки не лише окремих додатків, але і контролера в цілому.

Існуючі мови програмування політик комутації полегшують створення нових додатків. Кожна з мов має свої переваги.

Мова NetCore вводить поняття стану контролера, як історію усіх пакетів, які коли-небудь проходили через комутатори мережі. Мова NetCore дозволяє динамічні політики, вимагаючи при цьому явної вказівки параметрів заголовка, від яких залежить політика. Останні обмеження хоч і не є критично важливими - компілятор мови зможе синтезувати необхідні правила OpenFlow і без них, проте відмова від використання таких вказівок призводить до серйозного погіршення показників продуктивності системи.

Описи великої кількості політик на мовах Frenetic і NetCore потребують попередньої компіляції, в результаті якої виходить додаток, що управляє, відповідає усім вибраним політикам.

Система Nettle є сукупністю інтерпретатора відповідної мови і контролера, яка самостійно вирішує можливі конфлікти між додатками, що управляють, безпосередньо в процесі їх роботи.

Середовище виконання Frenetic автоматично обробляє «другі» пакети, переслані комутатором вже після того, як додаток здійснив обробку відповідного потоку по його першому пакету.

Проте, щоб побудувати по-справжньому надійну мережу передачі даних, необхідно мати можливість перевіряти що синтезуються додатком правила на суперечність специфікаціям. При цьому існує декілька альтернативних варіантів реалізації такої перевірки. Один з них - безпосередня перевірка, що синтезуються додатком правил OpenFlow через засоби динамічної перевірки специфікації.

Перспективним же є спосіб перевірки специфікацій через формальну верифікацію програм. Здійснити формальну верифікацію додатка, що управляє, написаного на універсальній мові програмування, є не можливим через надмірну обчислювальну складність. Проте, верифікація додатка написаного на спеціальній обмеженій підмножині такої мови стає цілком можливою. Тому актуальним стає завдання створення такої мови, виразної потужності якої вистачає для опису нескладних додатків, що управляють, а властивості дозволяють проводити верифікацію простих властивостей за прийнятний час.

Згідно з даними Oracle, понад 3 мільярди приладів у світі працюють на Java. Так чому ж Java така популярна? Можна виділити 4 основні причини:

1. Написане один раз працює скрізь (кросплатформність):

Мова Java хороша тим, що один і той же написаний код буде працювати, наприклад:

- і на Windows;
- і на Linux;
- і на MacOs.

Тоді, як на інших мовах програмування потрібно написати не 1, а відразу 3 різних коди - під Windows, під Linux і під MacOs. Таку особливість Java, що “написане один раз працює скрізь”, називають кросплатформністю (рис. 4).

Коли потрібно писати програми мовою Java, вони завжди будуть зберігатися окремими

файлами. При чому ці файли завжди будуть мати розширення .java. Наприклад, Program.java. Якщо, наприклад, колеги захочуть ознайомитися з таким файлом, вони легко зможуть прочитати написаний у ньому код, щось переписати або дописати в файлі, якщо буде потрібно. Тому що цей код людиночитабельний.

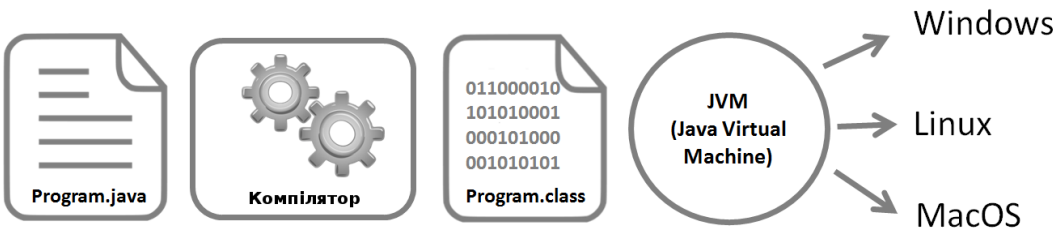

Рис. 4. Кросплатформеність у Java

Далі, коли запускається компілятор (якщо бути зовсім точними, то це називається компілятор javac), код з людиночитабельного перетворюється в так званий байт-код (тобто у вигляді різних комбінацій 0 і 1) і стає винятково машиночитабельним. Після цього з'явиться ще один файл, який завжди матиме розширення .class. У нашому прикладі – Program.class. Потім JVM (Java Virtual Machine) виконує байт-код.

2. Дружній синтаксис:

Розробники мови Java не стали винаходити велосипед з нуля, а, грубо кажучи:

- взяли все найкраще від мови програмування С – і його прямого спадкоємця – мови програмування С ++;
- видалили все, що вважали зайвим і не особливо вдалим у С і С ++;
- внесли нововведення в нову мову програмування Java.

І виграли від такого підходу. Оскільки між Java, С і С ++ є багато спільного, програмістам було набагато легше переходити на нову мову. Адже не треба абсолютно все вчити з нуля, багато конструкцій були вже зрозумілими. Що також сприяло швидкому росту популярності Java серед програмістів.

3. Об'єктно-орієнтована мова – це програмування за допомогою класів і об'єктів.

4. Керування пам'яттю.

Згідно рейтингу української спільноти програмістів dou.ua Java в 2019 р. є, як і раніше, найпоширеніша мова програмування (рис. 5).

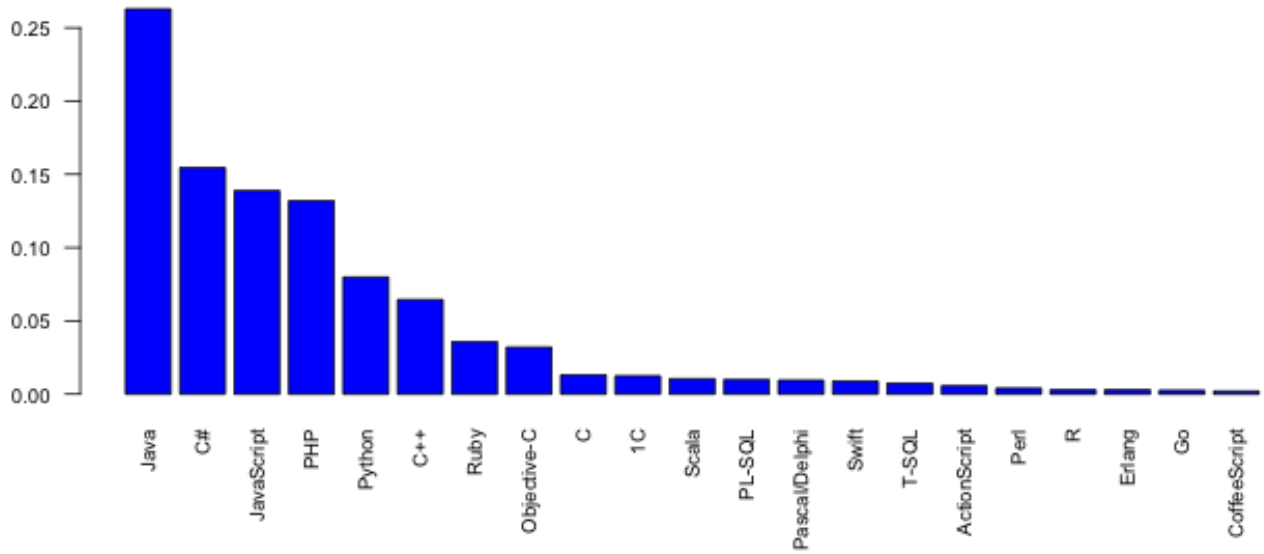

Рис. 5. Використання в 2019 р. мов програмування

Висновки

При дослідженні поставлених завдань отримані результати, на підставі яких можна зробити наступні висновки:

1. Однією з найбільш перспективних і такої, що бурхливо розвивається реалізацій підходу до створення сучасних систем надання телекомунікаційних послуг є технологія OpenFlow Softswitch. Дана технологія є логічним продовженням систем пост NGN (IP Multimedia Subsystem) та складової таких систем Softswitch. Основним її документом є специфікація OpenFlow, в якій описуються основні компоненти OpenFlow-мереж, принципи роботи і взаємодії компонентів. Організацією стандартизації для специфікації є ONF - Open Networking Foundation (www.opennetworking.org). Варто відзначити найвідоміших засновників - Nokia, Cisco, Huawei, Facebook, Google.

2. За результатами проведеного аналізу існуючого ПЗ для сучасних систем надання телекомунікаційних послуг можна зробити наступні висновки:

- більшість існуючого ПЗ для сучасних систем надання телекомунікаційних послуг написана на мовах високого рівня: C, C++, Python, Java, Rybu.
- кроссплатформенними являються тільки Java-контролери FloodLight, Beacon і Maestro, інші працюють на Linux-платформах.
- переважаючими мовами програмування для систем комутації є C та Java.

За підсумками дослідження існуючого ПЗ для створення сучасних систем надання телекомунікаційних послуг з відкритим початковим кодом, найбільш перспективними і такими, що активно розвиваються розробниками являються: контролери Beacon, FloodLight (на мові Java), Trema і MuL (на мові C).

3. На основі мов програмування C та Java, розроблено спеціальні мови програмування мереж, які з одного боку представляють високорівневі способи управління потоками в мережі, а з іншого боку приховують усі деталі по правильній і коректній установці низькорівневих правил на комутатори. На даний час такими мовами є Frenetic, NetCore і Nettle.

Усі існуючі мови програмування OpenFlow мереж надають розробникові мережевих додатків такі конструкції мови, в термінах яких зручно описувати правила управління потоками, абстрагуючись тим самим від рівня команд комутатора OpenFlow Softswitch.

Найбільш популярною мовою програмування серед програмістів, в т.ч. і сучасних систем надання телекомунікаційних послуг, є мова Java. Можна виділити 4 основні причини: кросплатформеність; дружній синтаксис; об'єктно-орієнтована мова; керування пам'яттю.

Список використаної літератури

1. Голышко А. IMS: полезные рекомендации / А. Голышко // Connect. – 2009. – № 5. – С. 2-6.

2. Гольдштейн А. Б., Соколов Н. А. Подводная часть айсберга по имени NGN // Технологии и средства связи. – 2006. - №2. – С. 12 - 21.

3. Беркман Л.Н., Терещенко Н.М. NGN: архітектура і реалізація // Зв'язок. – 2006. - №6. - С. 49 - 51.

4. Ложковский А. Г., Вербанов О. В. Проблемы перехода к сетям нового поколения NGN // Наукові праці ОНАЗ ім. О. С. Попова. – 2007. - №1. - С.161-163.

5. Балькін Г.Ф. Міграція мереж зв'язку до NGN // Вісник Українського науково-дослідного інституту зв'язку. – 2008. - №3. – С. 32 – 41.

6. Overview of 3GPP Release 5. Summary of all Release 5 Features // ETSI Mobile Competence Centre. - Finland, 2003. – 42 p.

7. Y.2001. Next Generation Networks – Frameworks and functional architecture models // ITU-T Recommendation. – Geneva, 2005. – 18 p.

8. Атцик А.А. Модели и методы управления медиа-шлюзами в сетях NGN: автореф. дис. на здобуття ступеня канд. техн. наук; спец. 05.13.13. - Модели и методы управления медиа-шлюзами в сетях NGN / А.А. Атцик: - Санкт-Петербург, 2009. - 19 с.

9. Поповський В.В. та ін. Телекомунікаційні системи та мережі. Структура й основні функції. Том 1 [Електронний ресурс] / В.В. Поповський та ін. // Компанія СМІТ – Режим доступу: http://www.znanius.com/3533.html

10. OpenFlow Switch Specification, Version 1.3.1 (Wire Protocol 0x04) [Electronic resource] // Open Networking Foundation. - Mode of access: ttps://www.opennetworking.org/images/stories/ downloads/sdn-resources/onf-specifications/ openflow/openflow-spec-v 1.3.1.pdf

11. The OpenFlow Soft Switch [Електронний ресурс] Krzysztof Rutka // Erlang Factory – Режим доступу: http://www.erlang-factory.com/upload/presentations/635/ openflow.soft.switch-krzysztof.rutka.pdf

12. Смелянский Р.Л. и др. Отчет о научно-иследовательской работе [Електронний ресурс] / Р.Л. Смелянский и др. // ВМК МГУ – Режим доступу: https://cs.msu.ru/sites/cmc/files/scproj/ 4047_otchet_nir1_0.pdf

13. SDN Series Part Eight: Comparison Of Open Source SDN Controllers [Електронний ресурс] Sridhar Rao // Thenewstack – Режим доступу: http://thenewstack.io/sdn-series-part-eight-comparison-of-open-source-sdn-controllers/

14. N. Foster, R. Harrison, M.J. Freedman, C. Monsanto, J. Rexford, A. Story, D. Walker Frenetic: a network programming language // Proceedings of the 16th ACM SIGPLAN international conference on Functional programming: ICFP '11. — New York, NY, USA: ACM, 2011. - P. 279–291.

15. A. Courtney, H. Nilsson, J. Peterson The Yampa arcade // Proceedings of the 2003 ACM SIGPLAN workshop on Haskell: Haskell '13. - New York, NY, USA: ACM, 2013. - P. 7–18.

16. C. Monsanto, N. Foster, R. Harrison, D. Walker A compiler and run-time system for network programming languages // Proceedings of the 39th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages: POPL '12. - New York, NY, USA: ACM, 2012. - P. 217–230.

Автори статті

Макаренко Анатолій Олександрович – доктор технічних наук, доцент, професор кафедри Мобільних та відеоінформаційних технологій, Державний університет телекомунікацій, Київ, Україна.

Гринкевич Ганна Олександрівна - кандидат технічних наук, доцент, доцент кафедри Телекомунікаційних систем, Державний університет телекомунікацій, Київ, Україна.

Легомінова Світлана Володимирівна - доктор економічних наук, доцент, завідувач кафедрою управління інформаційною та кібернетичною безпекою, Державний університет телекомунікацій, м. Київ, Україна.

Authors of the article

Makarenko Anatoliy Oleksandrovych – doctor of Science (technic), associate professor, professor of Department of Information and communications technology, State University of Telecommunications, Kyiv, Ukraine.

Grynkevych Ganna Aleksandrovna – candidate of Science (technic), assistant professor, assistant professor of Department of Telecommunication systems and networks, State University of Telecommunications, Kyiv, Ukraine.

Lehominova Svitlana Volodymyrivna - doctor of Science (economic), associate professor, professor of Department of Information and communications technology, State University of Telecommunications, Kyiv, Ukraine.

Дата надходження в редакцію: 25.10.2019 р. Рецензент: д.т.н., доцент А.П. Бондарчук