УДК 004.413

МУЛЬТИАГЕНТНЫЙ АНАЛИЗ ПРАВИЛЬНОСТИ СПЕЦИФИКАЦИЙ ПРОТОКОЛОВ ИНИЦИИРОВАНИЯ СЕАНСОВ

В.В. Девятков, **Мьё Тхет Наунг**

МГТУ им. Н.Э. Баумана, Москва, Российская Федерация
e-mail: deviatkov@ bmstu.ru; komyothetnaung@gmail.com

Выполнена автоматизация проверки правильности спецификаций протоколов инициирования сеансов как мультиагентной системы, представляемой процессными моделями и описанием требований их правильности на языке временной модальной логики. Автоматизацию проверки правильности спецификаций предложено осуществлять логическими программами, получаемыми с помощью предлагаемой методики перехода от процессной модели описания спецификации и требований правильности на языке модальной логики к логической программе проверки правильности на языке логического программирования ПРОЛОГ. Развиты принципы перехода от процессных моделей SIP-спецификаций к логической программе до детальной методики получения всех необходимых разделов логической программы. Методика проиллюстрирована примером логической программы для случая двух взаимодействующих агентов: пользовательского и сервисного.

***Ключевые слова*:** последовательностный процесс, пользовательский агент со стороны клиента, пользовательский агент со стороны сервера, протокол инициирования сеанса, язык логического программирования VISUAL PROLOG.

MULTIAGENT ANALYSIS OF SPECIFICATION ACCURACY OF SESSION INITIALIZATION PROTOCOLS

V.V. Devyatkov, **Myo Thet Naung**

Bauman Moscow State Technical University, Moscow, Russian Federation
e-mail: deviatkov@ bmstu.ru; komyothetnaung@gmail.com

Automation of validating the specification protocols for initialization of sessions as a multiagent system, is performed. Multiagent system is presented by the process models, as well as by description of their correctness in terms of temporal modal logic. Automation of the specifications' validating is proposed to be carried out by means of logic programs obtained through the use of proposed principle of moving from the process model of the specification description and from the correctness requirements in terms of modal logic to a logic program of checking the correctness in terms of logic programming language PROLOG. Principles of moving from process models of SIP-specifications to a logic program presented are developed to the extent of a detailed procedure obtaining all necessary program sections. Proposed method is illustrated by a logic program example for the case of two interacting agents — client-side agent and server-side one.

***Keywords*:** sequential process, user agent elient, user agent server, session initiating protocol, logic programming language VISUAL PROLOG.

Протокол инициирования сеансов (Session Initiation Protocol (SIP)) — это протокол, разработанный IETF (Internet Engineering Task Force) для голосового интернет-протокола VOIP (Voice over Internet Protocol), а также для передачи текста или мультимедиа данных, например, для систем обмена мгновенными сообщениями, видео обмена, игр в реальном времени и др. Стандарт SIP описан во многих источниках [1–4]. В модели взаимодействия открытых систем SIP является сетевым протоколом прикладного уровня. Однако документация, имеющаяся по протоколу, плохо структурирована, неформальна и неполна, что затрудняет ее использование как средства проверки правильности спецификаций для конкретных приложений SIP.

В статье [2] для проверки SIP-спецификаций предлагается использовать модель последовательностных взаимодействующих агентов, однако для описания спецификаций, в отличие от работы [5], предлагается применять значительно более выразительный, хорошо структурированный и теоретически, как формальная система, более развитый вариант языка, основанный на моделях взаимодействующих последовательностных процессов (π-исчислений [6]).

Настоящая статья также посвящена проверке правильности SIP-спецификаций по тем же принципам, которые изложены в статье [2]. Однако в отличие от статьи [2] в настоящей статье упор делается на изложение детальной методики перехода от процессных моделей описания SIP-спецификаций и условий их правильности, формулируемых на языке модальной временно́й логики, к конкретным логическим программам, реализующим проверку правильности SIP-спецификаций.

Принципы описания SIP-спецификаций графами переходов процессов. В настоящей статье, как и в работе [2], SIP-спецификации создаются как множество пользовательских агентов на стороне клиентов *UAC* (*User Agent Client*) и множество пользовательских агентов на стороне серверов *UAS* (*User Agent Server*). Взаимодействие агентов осуществляется с помощью обмена сообщениями. Каждый агент является последовательностным процессом, параллельно функционирующим с другими. Последовательные процессы в рамках настоящей статьи для наглядности будем представлять графами переходов процессов или процессными графами переходов, хотя, как показано в статье [2], в случае процессов (агентов) большой размерности графовое представление становится громоздким и вместо него используется адекватное представление в виде процессных выражений.

В процессных графах переходов, описывающих SIP-спецификации, каждое состояние графа изображается кружком, внутрь которого помещается символьное обозначение состояния.

Для иллюстрации описания поведения агентов процессными графами воспользуемся парой простых агентов *UAC* и *UAS*, взятой из

работы [2]. Опишем их поведение на языке процессных графов. Агенты *UAC* и *UAS* взаимодействуют по 10 каналам, имена которых *ackc*, *brpc*, *irpc*, *reqc*, *sakc*, *acks*, *brps*, *irps*, *reqs*, *saks*. Каналы *ackc*, *brpc*, *irpc*, *reqc, sakc* являются входными для процесса *UAC* и выходными для процесса *UAS*, а каналы *acks*, *brps, irps, reqs, saks* наоборот — выходными для процесса *UAC* и входными для процесса *UAS*. Каждый из агентов *UAC* и *UAS* имеет вешний канал запуска и окончания соответственно *uac* и *uas*. Кроме того, каждый из агентов имеет внешние каналы для сообщения об ошибках взаимодействия *invalidc* и *invalids*. Полагаем, что каждый из процессов может выполнять только шесть типов действий (помимо действий запуска и окончания): *invite*, *invsucc* (любой из 2nn ответов на действие *invite*), *invfail* (любой из 3nn–6nn ответов на действие *invite*), *ack*, *bye*, *byeRsp* (любой из 200 OK ответов на действие *buy*). Если какое-либо действие *a* берется из канала с именем *c*, то оно является восприятием и записывается как $c?a$, если же оно выдается в канал *c*, то оно является внешней реакцией и записывается как $c!a$.

Пара $c_1?a$ и $c_2!a$, записанная в виде $c_1?a.c_2!a$ или $c_2!a.c_1?a$, называется условием перехода.

Примеры процессных графов переходов агента *UAC* представлен на рис. 1, а агента *UAS* — на рис. 2. До начала работы процессный граф переходов агента *UAC* находится в начальном состоянии, которое обозначено *Uac*. Из начального состояния *Uac* процессный граф переходов переходит в состояние *Inviting* в результате выдачи реакции

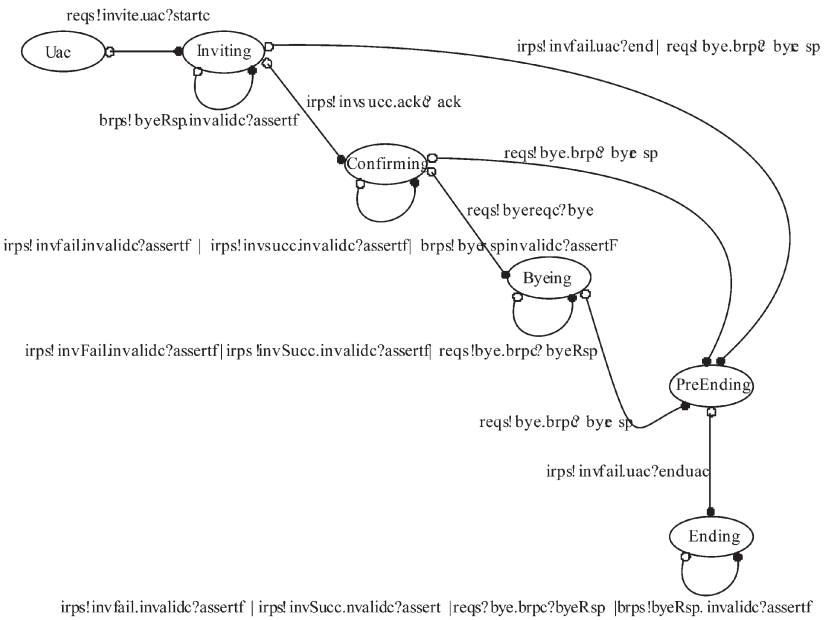

Рис. 1. Процессный граф переходов процесса *UAC*

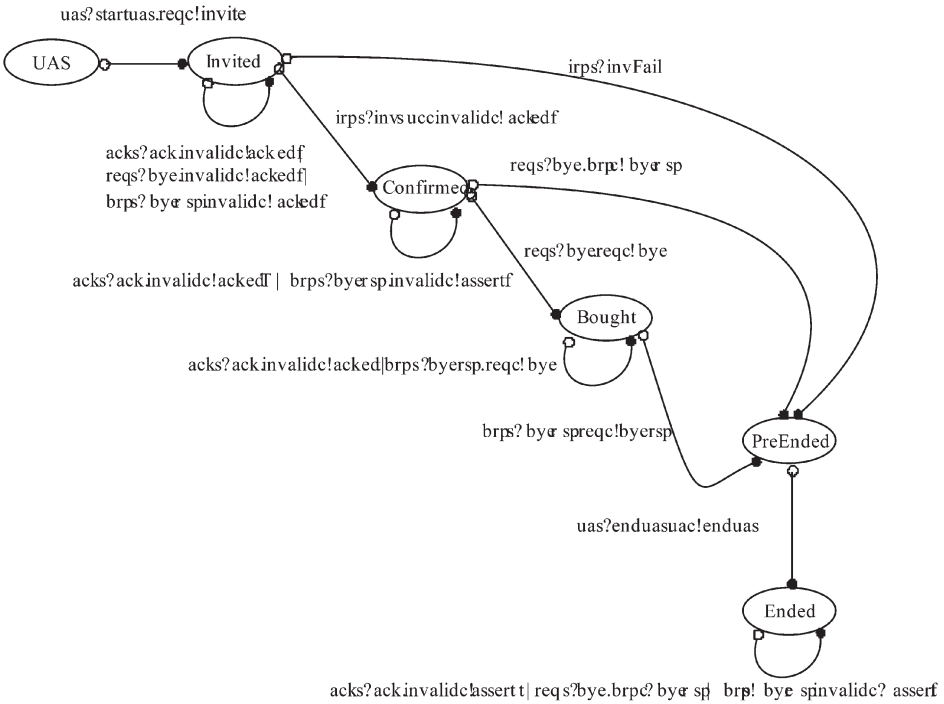

Рис. 2. Процессный граф переходов процесса *UAS*

reqs!invite и восприятия *uac?start* или, иными словами, переход совершается в результате выполнения условия *reqs!invite.uac?start*. Если процессный граф *UAC* находится в состоянии *Inviting*, то при выполнении условия *brps!byersp.invalidc?assertf* он остается в том же состоянии, при выполнении условия *irps!invsucc.ackc?ack* переходит в состояние *Confirming*, а при выполнении условий *irps!invfail* или *reqs!bye.brpc?byersp* переходит в состояние *PreEnding*. Описания переходов из остальных состояний графа переходов на рис. 1 и 2 аналогичны приведенному описанию переходов для состояний *Uac* и *Inviting*.

Условия правильного поведения агентов, представленных процессными графами переходов. В работе [2] для проверки правильности SIP-спецификаций, представленных агентами, предлагается сначала задать их процессными графами переходов или процессными выражениями, а затем сначала сформулировать условия правильности SIP-спецификаций как высказывания на языке временно́й модальной логики, а затем — для реализации проверки правильности перейти к логической программе на языке VISUAL PROLOG. В настоящей статье упор делается на принципы и процедуру перехода от процессных графов переходов и полученных указанным образом условий правильности к логическим программам на языке VISUAL PROLOG, позволяющим осуществлять проверку правильности. Всю эту процедуру перехода к логической программе в настоящей статье изложим на

примере проверки единственного условия правильности, называемого условием правильного окончания. Для остальных условий проверки правильности процедуры перехода, за исключением несущественных отличий, аналогичны.

На языке модальной логики условие правильного окончания выглядит следующим образом:

$$\begin{aligned}\Box[&\varphi_{uac}(x_1 s!y_1.uac?startuac) \supset \Diamond\xi_{uac}(v_1 s!w_1.uac?enduac)\wedge,\\ &\varphi_{uas}(x_2 s?y_2.uas?startuas) \supset \Diamond\xi_{uas}(v_2 s?w_1.uas!enduas)\wedge,\\ &\varphi_{uac}(p_1 s!q_1.\, r_1 c?t_1) \supset\ \varphi_{uas}(p_1 s?q_1.\, r_2 c!t_2)\wedge,\\ &\varphi_{uas}(r_2 c?t_2.u_1 s!s_1) \supset\ \varphi_{uac}(u_1 s?s_1.\, r_3 c?t_3)].\end{aligned}$$

Здесь

$$\begin{aligned}&\varphi_{uac}(x_1 s!y_1.uac?startuac), \xi_{uac}(v_1 s!w_1.uac?enduac),\\ &\varphi_{uas}(x_2 s?y_2.uas?startuas), \xi_{uas}(v_2 s?w_1.uas!enduas),\\ &\varphi_{uac}(p_1 s!q_1.\, r_1 c?t_1),\ \varphi_{uas}(p_1 s?q_1.\, r_2 c!t_2)\,,\\ &\varphi_{uas}(r_2 c?t_2.u_1 s!s_1),\ \varphi_{uac}(u_1 s?s_1.\, r_3 c?t_3)\end{aligned}$$

— предикаты, истинные, если в каналы $startuac$, $enduac$, $startuas$, $enduas$ поступают соответствующие восприятия или реакции и каналы *reqc, reqs, irps,brps,brpc* правильно взаимодействуют.

Далее на примере условия правильного окончания рассмотрим, как формируется на языке VISUAL PROLOG конкретная логическая программа его проверки.

Переходом из состояния b_i в состояние b_j в результате выполнения условия перехода $c_1?a.c_2!a$ называется тройка $(b_i, c_1?a.c_2!a, b_j)$. Путем на графе, ведущем из состояния b_i в состояние b_j, называется последовательность переходов, ведущих из состояния b_i в состояние b_j. Если из состояния b_i ведет какой-либо путь в состояние b_j, то будем говорить, что состояние b_j достижимо из состояния b_i этим путем. Состояние b_i, с которого начинается выполнение процесса, представленного тем или иным процессным графом переходов, называется начальным состоянием графа, а состояние b_j, в котором выполнение процесса завершается, называется конечным состоянием.

Переход от процессного графа переходов и условий правильности SIP-спецификаций к логической программе в языке VISUAL PROLOG. Анализ правильности SIP-спецификаций описания протоколов с помощью логических программ на языке логического программирования VISUAL PROLOG сводится к поиску свойств процессных графов переходов, наличие которых гарантирует правильность SIP-спецификаций. Например, это может быть наличие путей, которые должны иметь свойства определенного рода, исключающие некорректности.

В синтаксисе VISUAL PROLOG каждое состояние b_i представляется константой bi. Переменные, областью значений которых являются

состояния, обозначаются прописными символами *B* и индексируются, если это необходимо, например, как *Bi*. Условия перехода $c_1?a_1.c_2!a_2$ или $c_2!a_2.c_1?a_1$ представляются функторами. Пока соответствующие функторы будем обозначать как $f(x)$, где x — это $c_1?a_1.c_2!a_2$ или $c_2!a_2.c_1?a_1$. Используя введенные обозначения для того, чтобы ввести структуру используемых логических программ на языке VISUAL PROLOG, введем следующие предикаты и правила:

- *initial* (*Bi*) — одноместный предикат, истинный, когда автомат находится в начальном состоянии *Bi*;
- *final* (*Bj*) — одноместный предикат, истинный, когда автомат находится в конечном состоянии *Bj*;
- *transition* (*Bi, f* (*x*)*, Bj*) — трехместные предикаты, называемые переходами и истинные, если на процессном графе переходов имеются переходы из состояния *Bi* в состояние *Bj*, условиями перехода которых, являются соответственно *x*;
- *accessible* (*Bi*, [*X*]*, Bj*)) — двуместный предикат, истинный, если состояние *Bj* достижимо из состояния *Bi* в результате последовательного выполнения условий перехода из списка [*X*] (наличия на процессном графе переходов пути, ведущем из состояния *Bi* в состояние *Bj*, дуги которого помечены условиями перехода из списка [*X*]);
- *accessible* (*Bi*, *X, Bj*): — *transition* (*Bi*, *X, Bj*) – нерекурсивное правило достижимости состояния *Bj* из состояния *Bi*;
- *accessible* (*Bi*, [*X*|*Rest*]*, Bj*): — *transition* (*Bi*, *X, Bk*), *accessible* (*Bk*, [*Rest*], *Bj*) — рекурсивное правило достижимости состояния *Bj* из состояния *Bi*.

Логическая схема программы (псевдокод) на языке VISUAL PROLOG 5.0 для проверки правильного окончания для пары графов UAC и UAS будет содержать следующие разделы:

- *domains*, описывающий структуру функторов $f = f(x)$, представляющих переходы *x* и структуру списка из этих переходов $list = f^*$;
- *predicates*, описывающий предикаты перехода *transition* (*symbol, list, symbol*) между состояниями графов и предикаты достижимости *accessible* (*symbol, list, symbol*) одних состояний из других;
- *clauses,* содержащий описание всех переходов (фактов) и правил достижимости;
- *goal,* который ставит задачу проверки условий правильного поведения агентов.

Логическая программа проверки правильности SIP-спецификаций (на примере проверки условия правильного окончания). Рассмотрим в качестве примера простые графы, показанные на рис. 3 и 4. Для выполнения условия правильного окончания для каждого

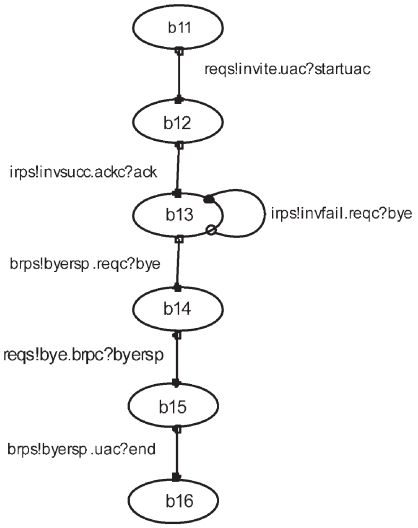

Рис. 3. Простой граф переходов UAC

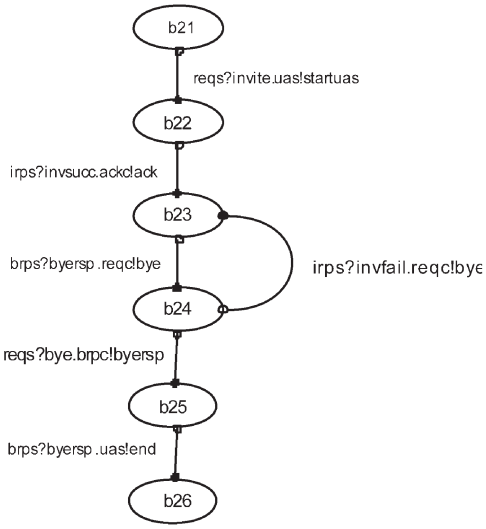

Рис. 4. Простой граф переходов UAS

пути графа UAC (см. рис. 3), ведущего из начального состояния в его финальное состояние, на графе UAS (см. рис. 4) — должен существовать путь, ведущий из его начального состояния в его финальное состояние. Каждый переход этой пары путей на графе (см. рис. 3) должен иметь вид $(b_{1i}, xs!y.vc?w, b_{1j})$, а на графе (см. рис. 4) — вид $(b_{2k}, xs?w.vc!y, b_{2l})$. Это означает, что переход $(b_{1i}, xs!y.vc?w, b_{1j})$ из любого состояния b_{1i} на графе (см. рис. 3) осуществляется в результате выдачи в выходной канал xs агента UAC реакции $!y$ и последующего получения во входном канае vc восприятия $?w$. В свою очередь, переход $(b_{2k}, xs?w.vc!y, b_{2l})$ из любого состояния b_{1i} на графе (см. рис. 4) осуществляется в результате получения в его входном канале xs агента UAS воспрятия $?w$ и последующей выдачи в его выходной канал vc реакции $!y$. Единственным отличием от этих условий являются переходы из начальных состояний и переходы в финальные состояния. Здесь восприятия являются стартовыми и поступают в специальные каналы *uac* и *uas*, соответствующих агентов.

В соответствии с предыдущим разделом будем иметь следующую логическую программу на языке VISUAL PROLOG, позволяющую проверять простейшее условие правильного окончания для агентов UAC и UAS, представленных графами на рис. 3 и 4.

```
domains
f =f(symbol, symbol, symbol, symbol)
 list = f*
predicates
 nondeterm transition(symbol, list, symbol)
 nondeterm accessible(symbol, list, symbol)
```

```
clauses
transition(b11, [f(reqs,invite,uac,startuac)], b12).
transition(b12, [f(irps,invsucc,ackc,ack)], b13).
transition(b13, [f(brps,byersp,reqc,bye)], b14).
transition(b13, [f(irps,invfail,reqc,bye)], b13).
transition(b14, [f(reqs,bye,brpc,byersp)], b15).
transition(b15, [f(brps,byersp,uac,end)], b16).
transition(b21, [f(reqs,invite,uas,startuas)], b22).
transition(b22, [f(irps,invsucc,ackc,ack)], b23).
transition(b23, [f(brps,byersp,reqc,bye)], b24).
transition(b24, [f(irps,invfail,reqc,bye)], b23).
transition(b24, [f(reqs,bye,brpc,byersp)], b25).
transition(b25, [f(brps,byersp,uas,end)], b26).

accessible(B1, [X], B2) : – transition(B1, [X], B2).
accessible(B1, [X|Rest], B2) : – transition(B1, [X], B3), accessible(B3, Rest, B2).

goal
accessible(b11,[ f(_,invite,_,_), f(_,invsucc,_,_) , f(_,byersp,_,_), f(_,bye,_,_),
f(_,byersp,_,_)], b16),
accessible(b21,[ f(_,invite,_,_), f(_,invsucc,_,_) , f(_,byersp,_,_), f(_,bye,_,_),
f(_,byersp,_,_)], b26) .
```

Раздел *goal* содержит задачу нахождения двух путей, ведущих из начального состояния *b11* в финальное состояние *b16* графа на рис. 3 и из начального состояния *b21* в финальное состояние *b26* графа на рис. 4, содержащих одинаковые последовательности реакций и восприятий. Результатом работы данной программы является ответ “yes”, означающий наличие на графе (см. рис. 3) пути *!invite, !invsucc, !byersp, !bye, !byersp*, а на графе (см. рис. 4) пути *?invite, ?invsucc, ?byersp, ?bye, ?byersp*, ведущих из начального состояния в конечное на соответствующих графах.

Заключение. 1. Рассмотрена проверка правильности SIP-спецификаций, представленных моделями последовательностных взаимодействующих агентов. В отличие от других в настоящей работе для описания агентов используется хорошо структурированный и развитый вариант языка, основанный на моделях взаимодействующих последовательностных процессов (π-исчислении), язык временно́й модальной логики для описания требований к спецификациям и язык логического программирования VISUAL PROLOG для проверки правильности спецификаций.

2. Изложены принципы перехода от процессных моделей SIP-спецификаций к логической программе и от требований на языке модальной логики к формулировке цели на языке VISUAL PROLOG.

3. Приведен пример логической программы и результаты ее работы.

4. В дальнейшем предполагается создать библиотеку требований правильности спецификаций на языке модальной логики и автоматизированную систему проверки правильности SIP-спецификаций, позволяющую непосредственно по процессным моделям и описанию требований на языке модальной логики автоматически переходить к логическим программам и осуществлять проверки правильности.

ЛИТЕРАТУРА

1. *Rosenberg J.*, *Schulzrinne H.*, *Camarillo G.*, *Johnston A.* Session Initiation Protocol (SIP). IETF Network Working Group Request. for Comments 3261, 2002.
2. *Девятков В.В.*, *Мьё Т.Н.* Формальный логический анализ корректности спецификаций сетевых SIP-протоколов // Инженерный журнал: наука и инновации, 2013. Вып. 11.
3. *Rosenberg J.*, *Schulzrinne H.* Reliability of provisional responses in Session Initiation Protocol (SIP). IETF Network Working Group Request. for Comments 3262, 2002.
4. *Основы* программирования на языке Visual Prolog. URL: http://www.intuit.ru/stuchies/courses/12333/1180/info
5. *Zave P.* Understanding SIP Through Model-Checking. Proc. of the 2nd International Conference of Principles, Systems and Applications of IP Telecommunications. Springer-Verlag, 2008. Vol. 5310. P. 256–279.
6. *Девятков В.В.*, *Сидякин И.М.* Мультиагентная система анализа телеметрической информации // Вестник МГТУ им. Н.Э. Баумана, Сер. Приборостроение, 2005. № 4 (61). С. 56–85.

REFERENCES

[1] Rosenberg J., Schulzrinne H., Camarillo G., Johnston A. Session Initiation Protocol (SIP). *IETF Network Working Group Request. for Comments* 3261, 2002.

[2] Devyatkov V.V., M'e T.N. Formal logical analysis of the correctness of the specifications of network protocol. *Jelektr. nauchno-tehn. Izd. "Inzhenernyj zhurnal: nauka i innovacii" MGTU im. Baumana* [El. Sc.-Techn. Publ. "Eng. J.: Science and Innovation" of Bauman MSTU], 2013, vyp. 11 (in Russ.). URL: http://engjournal.ru/catalog/it/network/999.html

[3] Rosenberg J., Schulzrinne H. Reliability of provisional responses in Session Initiation Protocol (SIP) // IETF Network Working Group Request. for Comments 3262, 2002.

[4] Basics of Visual Prolog language programming. URL: http://www.intuit.ru/studies/courses/12333/1180/info

[5] Zave P. Understanding SIP Through Model-Checking. *Proc. of the 2nd International Conference of Principles, Systems and Applications of IP Telecommunications*. Springer–Verlag, 2008, vol. 5310, pp. 256–279.

[6] Devyatkov V.V., Sidyakin I.M. Multi-agent System of Telemetry Data Analysis. *Vestn. Mosk. Gos. Tekh. Univ. im. N.E. Baumana, Priborostr.* [Herald of the Bauman Moscow State Tech. Univ., Instrum. Eng.], 2005, no. 4 (61), pp. 56–85 (in Russ.).

Статья поступила в редакцию 27.10.2014

Девятков Владимир Валентинович — д-р техн. наук, профессор, заведующий кафедрой "Информационные системы и телекоммуникации" МГТУ им. Н.Э. Баумана. Автор более 120 научных работ (в том числе трех монографий) в области искусственного интеллекта, распознавания образов, принятия решений, логических исчислений, представления знаний, теории конечных автоматов, логического синтеза и анализа дискретных устройств и систем.

МГТУ им. Н.Э. Баумана, Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5.

Devyatkov V.V. — Dr. Sci. (Eng.), professor, head of “Information systems and telecommunications” department of the Bauman Moscow State Technical University. Author of more than 120 publications (including three monographs) in the field of artificial intelligence, pattern recognition, decision making, logic calculi, knowledge representation, theory of finite state machines, logical synthesis and analysis of discrete devices and systems.
Bauman Moscow State Technical University, 2-ya Baumanskaya ul. 5, Moscow, 105005 Russian Federation.

Мьё Тхет Наунг — аспирант кафедры “Информационные системы и телекоммуникации” МГТУ им. Н.Э. Баумана.
МГТУ им. Н.Э. Баумана, Российская Федерация, 105005, Москва, 2-я Бауманская ул., д. 5.

Myo Thet Naung — post-graduate of “Information systems and telecommunications” department of the Bauman Moscow State Technical University.
Bauman Moscow State Technical University, 2-ya Baumanskaya ul. 5, Moscow, 105005 Russian Federation.