УДК 658.512

Разработка метода анализа экземпляров потоков работ на основе временной автоматной RVTI-грамматики

Н.Н. Войт – к.т.н., доцент, зав. лабораторией инновационных виртуальных технологий проектирования и обучения департамента научных исследований и инноваций,
кафедра «Вычислительная техника», Ульяновский государственный технический университет
E-mail: n.voit@ulstu.ru
С.Ю. Кириллов – аспирант,
кафедра «Вычислительная техника», Ульяновский государственный технический университет
E-mail: kirillovsyu@gmail.com
Д.С. Канев – к.т.н., начальник научно-технического отдела ИДДО,
Ульяновский государственный технический университет
E-mail: dima.kanev@gmail.com
А.С. Степанов – мл. науч. сотрудник,
кафедра «Вычислительная техника», Ульяновский государственный технический университет
E-mail: step_al_ul@mail.ru
Р.Ф. Гайнуллин – к.т.н., программист ООО «Эквид»,
Ульяновский государственный технический университет
E-mail: r.gainullin@gmail.com

Аннотация

Постановка проблемы. Обязательным этапом при проектировании сложных технических изделий является анализ бизнес-процессов решения поставленной задачи, причем желательно автоматический (автоматизированный) с проверкой полученных процессов на ошибки. Вопросы анализа бездефектного завершения являются актуальными, поскольку сложность бизнес-процессов в виде моделей постоянно возрастает, а встроенные в среду моделирования средства проверки пока являются далеко не совершенными. Существующие методы анализа имеют с точки зрения науки нелинейные временные зависимости – экспоненциальные, полиномиальные. Многообразие диаграммных графических языков покрывает все возможные типы описаний систем, однако существуют нерешенные проблемы. Инструментальные средства поддержки графического проектирования не используют универсальные методы синтаксического анализа и являются узкоспециализированными и направленными на работу с одним-двумя графическими языками.
Цель. Предложить метод анализа различных типов диаграмм (EPC, BPMN, IDEF3, IDEF5, BPMN и SharePoint) на основе временной автоматной RVTI-грамматики.
Результаты. Проведен вычислительный эксперимент реализации предложенного метода на примере анализа EPC-диаграммы.
Практическая значимость. Приведены примеры найденных структурных (синтаксических) и семантических ошибок.

Ключевые слова
Бизнес-процессы, анализ графических языков, EPC.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 17-07-01417.
Исследование выполнено при финансовой поддержке РФФИ и Правительства Ульяновской области в рамках научного проекта № 18-47-730032.
Исследование поддержано грантом Министерства образования и науки РФ, проект № 2.1615.2017/4.6.

DOI: 10.18127/j20700814-202001-06

Введение

При проектировании автоматизированных систем активно применяются диаграмматические модели, представленные в артефактах визуальных графических языков BPMN, UML, IDEF и др. Это значительно повышает эффективность процесса проектирования и качество создаваемых систем за счет унификации языка взаимодействия участников процесса, строгого документирования проектно-архитектурных, функциональных решений и формального контроля корректности диаграмм. При этом обязательным этапом при моделировании бизнес-процессов предприятий является автоматическая (автоматизированная) проверка полученных моделей.

Вопросы анализа бездефектного завершения является актуальными, поскольку сложность моделей постоянно возрастает, а встроенные в среду моделирования средства проверки пока являются далеко не

совершенными. Многообразие диаграммных графических языков покрывает все возможные типы описаний систем, однако существуют нерешенные проблемы. Инструментальные средства поддержки графического проектирования не используют универсальные методы синтаксического анализа и являются узкоспециализированными и направленными на работу с одним-двумя графическими языками.

Ц е л ь р а б о т ы – предложить метод анализа различных типов диаграмм (EPC, BPMN, IDEF3, IDEF5, BPMN и SharePoint) на основе временной автоматной RVTI-грамматики.

Обзор графических грамматик

Плекс-грамматики. Одним из первых расширений понятия порождающей грамматики Хомского для порождения структур, отличных от цепочек символов, явились так называемые плекс-грамматики. В соответствии с названием, такие грамматики порождают структуры произвольных объектов, соединенных в заранее определенных точках связи.

В выводах языков цепочек каждый основной или вспомогательный символ при вхождении в цепочку связан с другим символом, который соседствует с ним справа или слева. Можно считать, что каждый символ имеет две «точки примыкания» – правую и левую (или верхнюю и нижнюю), при помощи которых он связывается или соединяется с другими символами. Feder J. применил такой подход к языкам с символами, имеющими произвольное число точек примыкания для связи с другими символами. Символ с N точками примыкания называется N-сочленяемой конфигурацией (N attaching-point entity) – NAPE. Структуры, образуемые взаимосвязанными NAPE, называются «плекс-структурами». Из множества плекс-структур образуются плекс-языки. Грамматика для описания плекс-языка называется плекс-грамматикой.

Плекс-грамматику можно представить как шестерку [1, 2] $G = (V_T, V_N, P, S, Q, q_0)$, где V_T – конечное непустое множество основных NAPE; V_N – конечное непустое множество вспомогательных NAPE, $V_T \cap V_N \neq \varnothing$; P – конечное непустое множество правил подстановки или правил замены; S – специальный элемент, называемый начальной NAPE; Q – конечное множество идентификаторов, $Q \cap (V_T \cup V_N) \neq \varnothing$; q_0 – специальный идентификатор, называемый пустым идентификатором.

Связь между основными элементами, вспомогательными элементами, правилами подстановки, начальной конфигурацией и соответствующими компонентами в грамматиках цепочек очевидна. Символы из Q используются как метки точек примыкания NAPE. С каждой точкой примыкания каждой NAPE связан идентификатор. У всех точек примыкания одной и той же NAPE эти идентификаторы различны. Идентификатор q_0 не соответствует никакой точке примыкания и служит для заполнения места отсутствующего идентификатора. NAPE могут соединяться только при помощи имеющихся у них точек примыкания; «мнимые» связи с использованием идентификатора q_0 не допускаются. Число необходимых грамматике идентификаторов на единицу больше, чем число точек примыкания у NAPE с наибольшим числом таких точек.

Формально контекстно-свободная плекс-грамматика имеет продукции в форме $A\Delta_A \rightarrow \beta\Gamma_\beta\Delta_\beta$, где A – имя нетерминального объекта грамматики; Δ_A –список внешних точек соединений нетерминальной конструкции A; β – список компонентов структуры, образующей A; Γ_β – список взаимосвязей подструктур правой части при образовании конструкции A; Δ_β – список соответствия, показывающий, как каждая внешняя точка связи у конструкции A соотносится с точками связи составляющих A подконструкций.

На рис. 1 показаны два NAPE с именами *ver* и *hor* из набора терминальных объектов. Связанные точки крепления и маркировка были определены вместе с формой примитивов.

С учетом NAPE *ver* и *hor*, показанных на рис. 1, рассмотрим пример создания букв *H*, *F*, *L* через грамматику:

```
G = ({Letter, Help}, {ver, hor}, P, {Letter}, {0, 1, 2,
3}, 0)
P = {
Help(1, 2, 3) → (ver, hor)(21)(10, 02, 03)
```

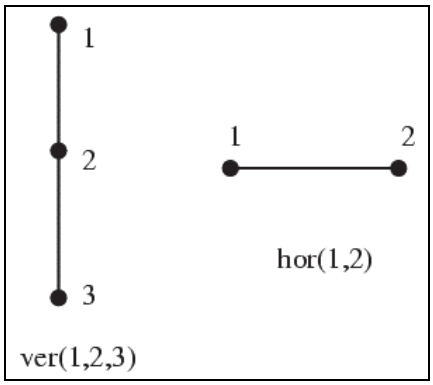

Рис. 1. Пример двух NAPE
Fig. 1. Example of two NAPE

```
Letter()→ (ver, hor)(31)()
Letter() → (Help, hor) (11)()
Letter() → (Help, ver) (22)()
}
```

Веб-грамматика. Одним из видов двухмерных грамматик являются веб-грамматики, предложенные Пфальцем и Розенфельдом [3, 4]. Предложения, порождаемые веб-грамматиками, – это ориентированные графы с символами на вершинах («вебы»). Веб-грамматикой называется выражение

$G = (V_N, V_T, P, S)$,

где V_N – множество вспомогательных символов; V_T – множество основных символов; S – множество начальных вебовl P – множество правил подстановки или правил замены.

Правило подстановки определяется как $a \to \beta, E$, где a и β – вебы; E – описание погружения β. Для того чтобы заменить подвеб a веба ω на другой подвеб β, необходимо определить, как «погружать» β в ω вместо a. Определение погружения не должно зависеть от остальной части ω, поскольку хочется иметь возможность заменять a на β в любом вебе, содержащем a как подвеб. Обычно E состоит из множества логических функций, которые определяют для всех вершин $\omega - a$ и всех вершин β, связана ли данная вершина β с данной вершиной $\omega - a$.

Заметим, что веб-грамматики ориентированы на порождение вершин или узлов в отличие от грамматик, ориентированных на порождение ребер или дуг (а именно, PDL-плекс-грамматик). Это значит, что основные символы или непроизводные элементы представляют скорее вершины графа, чем дуги.

Важным частным случаем веб-грамматик является тот, при котором алфавит основных символов состоит из единственного символа. При этом каждая точка каждого веба в языке имеет одну и ту же метку, так что метками можно пренебречь и считать, что вебы совпадают с входящими в них графами. Этот тип веб-грамматики называется «графовой грамматикой», а ее язык называется «графовым языком». Правило подстановки веб-грамматики имеет вид правила непосредственно составляющих: если в $\boldsymbol{a}$ существует точка a, такая что $\boldsymbol{a} - \{ a \}$ есть подвеб β, то все дуги между точками остальной части и точками $\boldsymbol{a} - \{ a \}$ заданы в E. В частности, правило подстановки будет бесконтекстным, если в $\boldsymbol{a}$ есть только одна точка. Таким образом, веб-грамматика называется грамматикой непосредственно составляющих (бесконтекстной), если все правила подстановки в ней имеют форму правил непосредственна составляющих (бесконтекстную).

При сравнении веб-грамматик с плекс-грамматиками можно рассматривать NAPE в плекс-грамматике как вебы, в которых одна точка помечена меткой этой NAPE, а другие – идентификаторами ее точек сочленения. Списки соединений в плекс-грамматике, описывающие, как взаимосвязаны множества NAPE, соответствуют внутренним дугам подвебов a и β в правиле подстановки, а список точек сочленения соответствует описанию погружения β в остальную часть веба E.

Позиционная грамматика. Позиционные грамматики являются прямым расширением неконтекстных строковых грамматик, где допускается больше общих отношений, чем конкатенации строк.

Контекстно-свободная позиционная грамматика имеет вид [5, 6] $PG = (N, T, S, P, POS, PE)$, где N – конечное непустое множество нетерминалов; T – конечное непустое множество терминалов с $N \cap \mathrm{T} = \varnothing$; $S \in N$ – стартовый нетерминал; P – конечный набор правил; POS – конечный набор идентификаторов бинарных отношений; PE – графический вычислитель.

Терминалы и нетерминалы являются графическими объектами.

Каждая продукция P имеет следующую форму:

$A \to x_1\, REL_1\, x_2\, REL_2 \ldots REL_{m-1}\, x_m$, где $m \geq 1$, $A \in N$, $x_i \in N \cup T$, $REL_i \in POS$.

Каждое позиционное отношение REL_i дает информацию о положении x_{i+1} относительно x_i. В то время как в строковой грамматике единственно возможным позиционным отношением является конкатенация строк, в позиционной грамматике могут быть определены другие позиционные отношения и затем использованы для описания языков.

Графический вычислитель *PE* – это функция, которая преобразует сентенциальную форму, полученную из грамматики, в соответствующую графическую форму.

В качестве примера рассмотрим позиционную грамматику, которая генерирует древовидные графы потоков данных. Терминалы и продукции грамматики изображены на рис. 2 и 3.

```
N = {DFL, IF}
T = {op, arg, test, mux}
S = DFL
POS = {JOINT}
P = {
DFL →arg
DFL → op 1_1 DFL 2^1_1 DFL'
DFL → mux 1_1 DFL 2^1_1 DFL' 2^3_1 IF
IF → test 1_1 DFL 2^1_1 DFL'
}
```

Объект *op* является оператором из набора {+, *, −, /}, объект условия *test* представлен множеством {<, >, =}. Объект *arg* является аргументом и может указывать имя константы или переменной; *mux* описывает узел мультиплексора.

Многоуровневая графовая грамматика. Контекстно-свободная грамматика (где левая сторона правил состоит из одного нетерминального узла) затрудняет определение синтаксиса большой части визуальных языков и является ограниченной.

Многоуровневая графовая грамматика является контекстно-зависимой грамматикой, в которой левая и правая стороны правил представлены графами. Однако, поскольку проблема разбора неразрешима для общих контекстно-зависимых грамматик графов, авторы ограничиваются грамматиками, в которых левая сторона правила лексикографически меньше, чем его правая сторона, что позволяет избежать циклических правил.

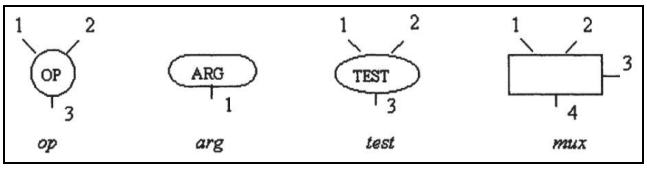

Рис. 2. Термальные символы грамматики
Fig. 2. Thermal grammar symbols

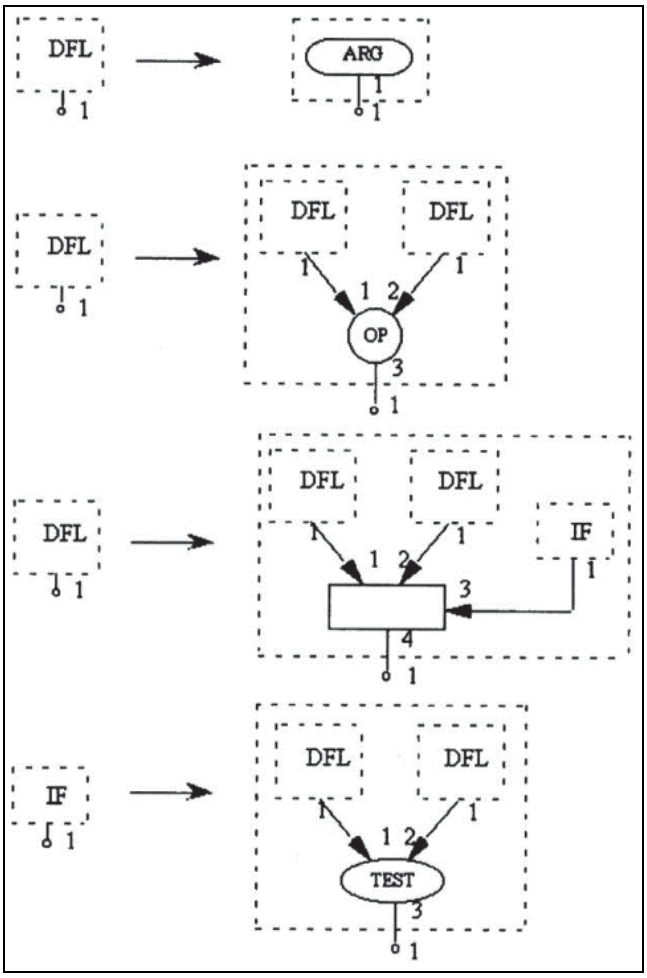

Рис. 3. Визуальное представление правил грамматики
Fig. 3. Visual representation of grammar rules

В случае линейных текстовых языков ясно, как заменить нетерминал в предложении соответствующей последовательностью (не)терминалов. Но в случае графических языков со многими возможными отношениями между языковыми элементами требуется более сложный механизм для определения отношений между окружением замененного нетерминала и его заменяющими (не)терминалами. Многоуровневые графовые грамматики расширяют левую и правую стороны правил элементами контекста, чтобы создавать ребра между новыми и старыми вершинами оригинального графа.

Графовая грамматика *gg* – это кортеж (*A*; *P*), где *A* – непустой начальный граф (аксиома); *P* – набор продукций [7].

Продукция $p := (L; R)$ является кортежем графов над одним и тем же алфавитом вершин и ребер. Его левая и его правая части могут иметь общий (контекстный) подграф. Применение правила к графу *G* означает замену подграфа *L* на подграф *R* в графе *G*.

Графовая грамматика $gg := (A; P)$ является многоуровневой графовой грамматикой, если для каждого правила $p := (L; R) \in P$ выполняются следующие условия:

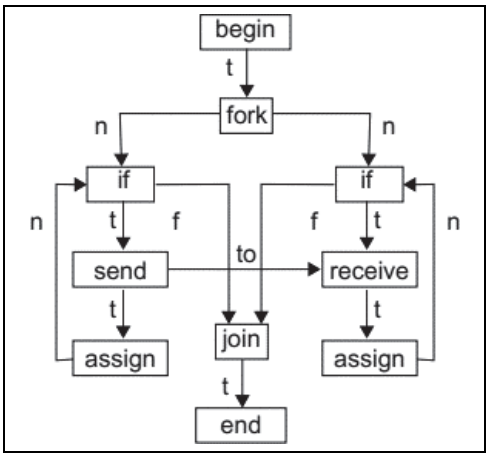

Рис. 4. Пример диаграммы PFD
Fig. 4. PFD chart example

1) правая сторона правила является связным графом;
2) левая и правая стороны правила не пустые;
3) левая сторона правила лексикографически меньше правой.

Эти ограничения гарантируют ряд свойств, а именно:
1) линейное время поиска для сопоставления с образцом правил;
2) история вывода графа всегда является связным ациклическим графом;
3) завершение алгоритма разбора.

Рассмотрим грамматику для визуального языка диаграмм процессов (PFD). Это гибрид блок-схем управления и диаграмм последовательности сообщений. Язык наследует от обоих типов диаграмм свойство наличия линейных последовательностей операторов, структуры заимствованы из блок-схемы управления. Также он позволяет определить несколько потоков управления, которые обмениваются асинхронными сообщениями так же, как диаграммы последовательности сообщений. На рис. 4 приведен пример диаграммы PFD, который содержит все вышеупомянутые элементы.

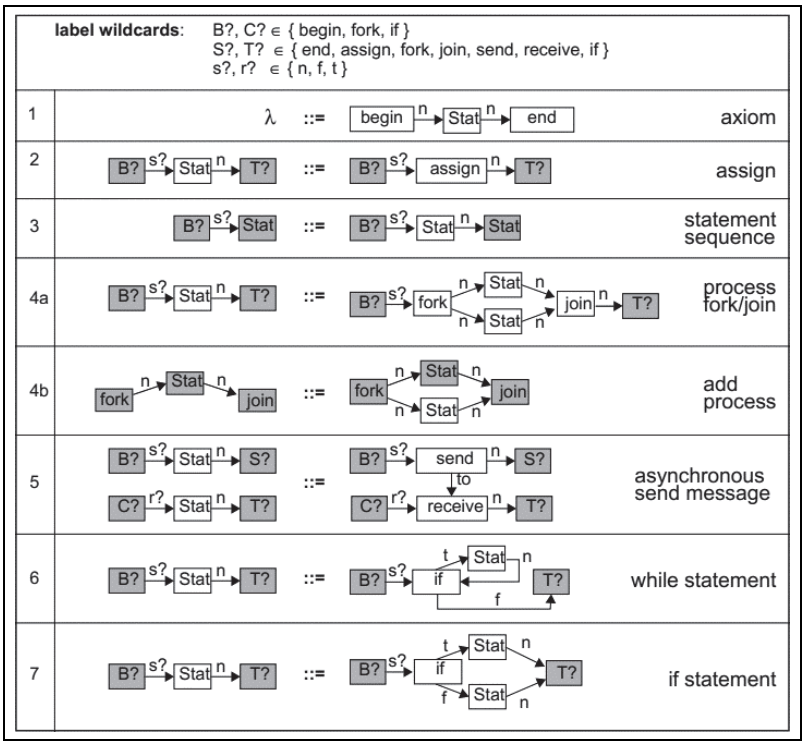

Рис. 5. Многоуровневая графовая грамматика для PFD
Fig. 5. Layered graph grammar for PFD

На рис. 5 показана многоуровневая графовая грамматика для языка PFD.

На основе проведенного обзора можно сделать вывод, что существующие средства проектирования графических спецификаций не используют универсальные методы синтаксического анализа. Разработка средств контроля для новых языков занимает значительное время, так как требует фактически создания нового анализатора с самого начала. Известные методы синтаксического анализа обладают значительными затратами по времени (экспоненциальные, полиномиальные характеристики) и памяти.

Анализ диаграмм

Рассмотрим задачу анализа диаграммы. Входным параметром является сама диаграмма, при этом модель диаграммы имеет вид $G=(V,E,TV,TE)$, где V – множество вершин; E – множество связей, $E \subset (V \times V)$; TV – множество типов вершин; TE – множество типов связей.

На выходе получим статус корректности диаграммы или сообщение об ошибке.

Запишем *алгоритм анализа диаграммы*.

Ш а г 1 . Определение функции поиска стартовых вершин. Для каждого типа диаграммы необходимо определить функцию поиска стартовых вершин $Fstart: V \to \{0,1\}$, затем с данных вершин запустить процесс обхода диаграммы для анализа.

Ш а г 2 . Определение автомата для управления процессом обхода. Для управления процессом обхода определим конечный автомат A автоматной RVTI-грамматики [8−15] $A = (S, T, S_0, C, Send, Ftrans)$, где S – множество состояний автомата; T – множество входных символов (термы диаграммы); S_0 – начальное состояние автомата; C – множество условий перехода; FA – множество функций перехода; $Send$ – множество конечных состояний; $Ftrans$ – функция перехода $S \times T \times C -> S \times FA$.

Ш а г 3 . Запуск алгоритма обхода. Алгоритм обхода имеет следующий вид.

1. Формируется множество начальных термов *START_V* для диаграммы.

2. В магазин *STACK* записываются множество начальных термов.

3. Из стека берется первый элемент T и подается на вход автомата.

4. В автомате ищется правило в упорядоченном списке правил, соответствующее текущему состоянию автомата, входному терму и условию перехода:

а) если правила нет, выполнение завершается, переход к п. 5;

б) в противном случае автомат переходит в следующее состояние. При переходе выполняются привязанные функции;

в) непроанализированные соседние термы добавляются в магазин *STACK*;

г) если следующее состояние принадлежит множеству конечных состояний, переходим к п. 5;

д) для текущего терма формируется список следующих термов и подается на вход автомата; переход к п. 5.

5. Завершение обхода.

Ш а г 4 . Проверка состояния.

Если текущее состояние автомата принадлежит множеству конечных состояний и все термы диаграммы проанализированы, то диаграмма считается корректной. В противном случае формируется сообщение об ошибке.

Анализ EPC-диаграмм

Событийная цепочка процессов (EPC) – тип диаграмм, используемых для моделирования, анализа и реорганизации бизнес-процессов (рис. 6). В то же время EPC-диаграммы могут использоваться для моделирования поведения отдельных частей системы при реализации функций и служить заменой традиционных блок-схем (поведенческого моделирования). EPC-метод был разработан Августом-Вильгельмом Шеером в начале 1990-х годов.

Рассмотрим трансляцию EPC-диаграммы. Модель для нотации EPC состоит из множества типов вершин TV = {Событие, Функция, Информация, Документ, Файл, Кластер, Набор объектов, Сообщение, Продукт, Организационная единица, Должность, Исполнитель, Местоположение, Приложение, Модуль, AND, OR, XOR, Цель, Термин} и множества типов связей TE = {Поток управления, Организационный поток, Поток ресурсов, Информационный поток, Поток информационных услуг, Поток товарно-материальных ценностей}.

Определим функцию получения стартовых вершин *Fstart* – множества вершин-событий без входящих связей.

Множество состояний автомата S = {Begin, Event, RelEvent, Function, RelFunction, Return, End, Condition, RelCondition, LineEnd}. Начальное состояние автомата S_0 = Begin. Множество конечных состояний $Send$ = {End}.

Множество условий перехода $C = \{\neg ANALYZED, ANALYZED, |STACK| > 0, DirectionalEdge\}$, где $\neg ANALYZED$ – следующий выбранный терм не проанализирован; $ANALYZED$ – следующий выбранный терм проанализирован; $|STACK| > 0$ – магазин не пустой; $DirectionalEdge$ – существует направленная непроанализированная связь, исходящая из данного терма.

Множество функций перехода $FA = \{POP\}$, где POP – вытащить терм из магазина и передать на вход автомата. Функция перехода *Ftrans* показана в виде таблицы перехода.

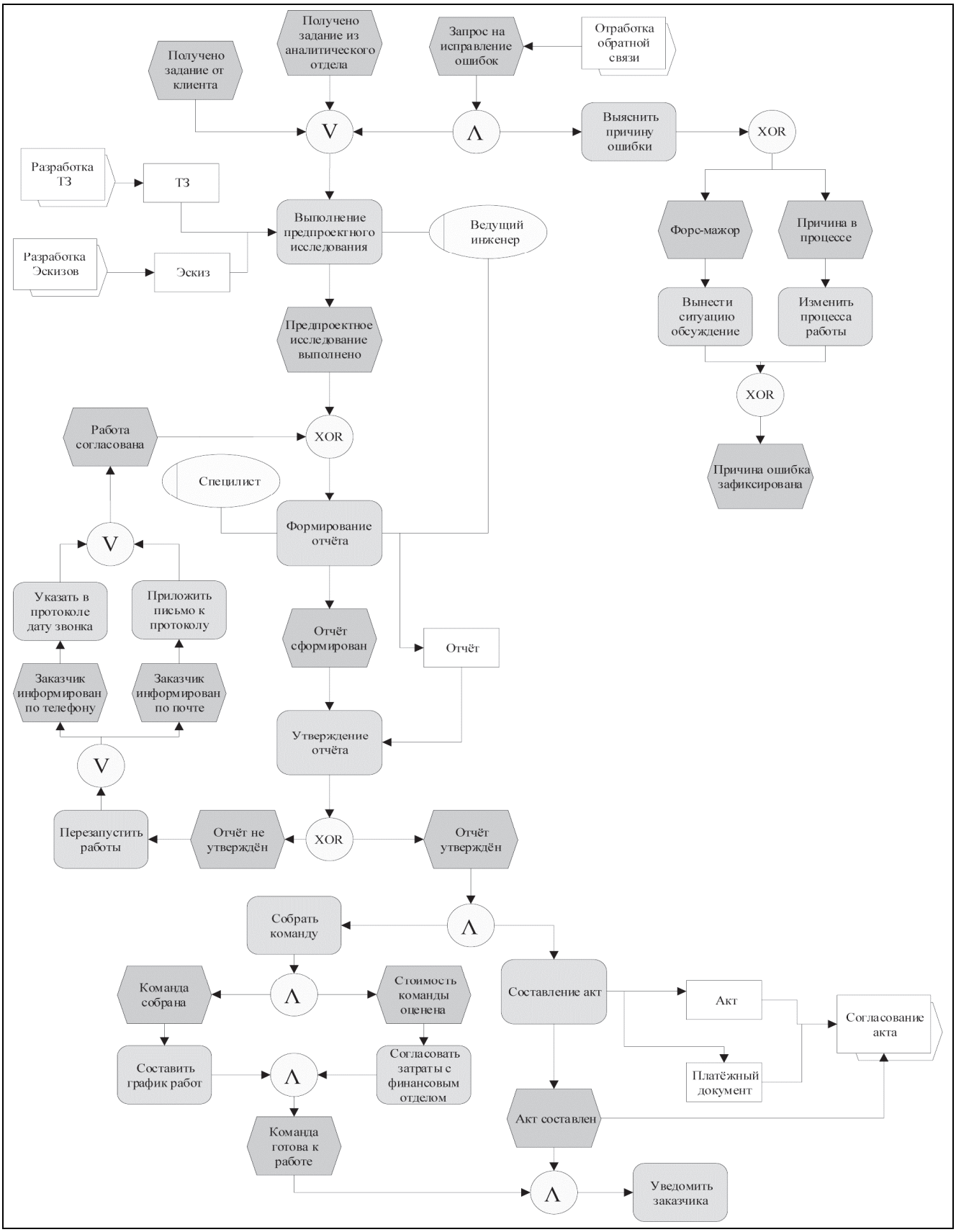

Рис. 6. Пример EPC-диаграммы
Fig. 6. EPC chart example

Таблица. ***Фрагмент таблицы переходов***

Номер	Состояние	Терм	Следующее состояние	Условие	Функция
1	Begin	Directional	RelEvent	*DirectionalEdge*	–
2	Return	–	End	not $\|STACK\| > 0$	–
3	Return	–	–	$\|STACK\| > 0$	Pop
4	Event	Directional	RelEvent	*DirectionalEdge*	–
5	Event	Directional	LineEnd	not *DirectionalEdge*	–
6	RelEvent	Function	Function	not *Analyzed*	–
7	RelEvent	OR	Condition	not *Analyzed*	–
8	RelEvent	XOR	Condition	not *Analyzed*	–
9	RelEvent	AND	Condition	not *Analyzed*	–
10	RelEvent	–	Return	–	–

Реализация

Внешний вид программы представлен на рис. 7. Программа разработана на платформе .Net Framework 4.5, для анализа используются диаграммы, построенные в Microsoft Visio 2017.

Приложение поддерживает четыре типа диаграмм для анализа: BPMN, EPC, IDEF3, IDEF5 (рис. 8).

На рис. 9 и 10 показаны примеры анализа EPC-диаграмм. В случае нахождения ошибки показывается описание ошибки и подсвечивается ошибочный элемент в Microsoft Visio 2017. Для анализа диаграммы необходимо открыть его в MS Visio и нажать кнопку «Анализ», в этом случае будет проанализирована диаграмма, находящиеся на активной странице. В случае необходимости можно проанализировать только часть диаграммы, для этого необходимо выделить нужные фигуры в MS Visio и также нажать кнопку «Анализ».

Рис. 7. Основное окно программы
Fig. 7. The main window of the program

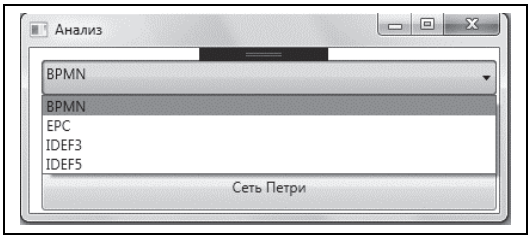

Рис. 8. Выбор типов диаграмм для анализа
Fig. 8. Choice of chart types for analysis

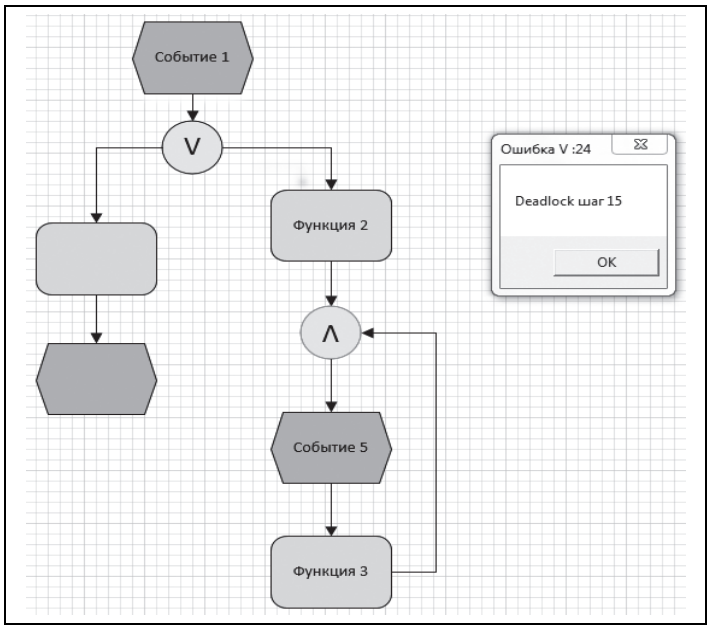

Рис. 9. Пример ошибки «Deadlock»
Fig. 9. Example of «Deadlock» error

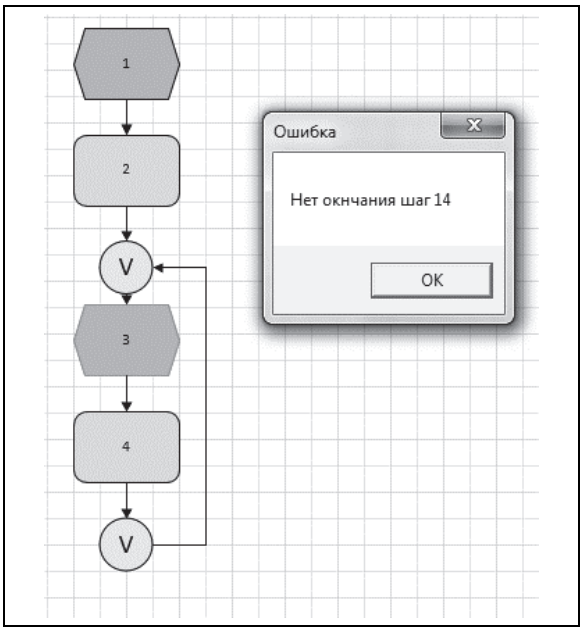

Рис. 10. Пример ошибки «бесконечный цикл»
Fig. 10. An example of an «infinite loop» error

Заключение

Исследована задача контроля и анализа динамических распределенных потоков работ бизнес-процессов решения поставленных проектных задач предприятия при создании сложных технических изделий, в том числе автоматизированных систем. Изложен метод анализа различных типов диаграмматических моделей. Также представлена модель диаграмм бизнес-процессов, разработан автомат для управления процессом обхода и алгоритм обхода.

Таким образом, разработан и исследован новый метод анализа экземпляров потоков работ на основе временной автоматной RVTI-грамматики, отличающийся от известных методов линейной временной характеристикой анализа и учетом понятия «время». Представлено применения метода (вычислительный эксперимент) для анализа EPC-диаграммы, состоящей из восьми типов элементов, приведены примеры найденных структурных (синтаксических) и семантических ошибок. Разработанный метод реализован на платформе .Net Framework 4.5 для анализа различных типов диаграмм (в том числе EPC, BPMN, IDEF3, IDEF5, UML, BPMN и SharePoint) в формате Microsoft Visio 2017.

Литература

1. *Feder J.* Plex languages // Information Sciences. 1971. V. 3. № 3. P. 225−241.
2. *Фу К.* Структурные методы в распознавании образов. М.: Мир. 1977. 319 с.
3. *Pfaltz J.L.*, *Rosenfeld A.* Web Grammar // Proc. of the International Joint Conference on Artificial Intelligence. Washington, D.C. 1969. P. 609−619.
4. *Фу К.* Структурные методы в распознавании образов. М.: Мир. 1977. 319 с.
5. *Costagliola G. et al*. Positional grammars: A formalism for LR-like parsing of visual languages // Visual Language Theory. Springer, New York, NY. 1998. P. 171−191.
6. *Costagliola G.*, *Chang S.K.* Using linear positional grammars for the LR parsing of 2-D symbolic languages // Grammars. 1999. V. 2. № 1. P. 1−34.
7. *Rekers J.*, *Schürr A.* Defining and parsing visual languages with layered graph grammars // Journal of Visual Languages & Computing. 1997. V. 8. № 1. P. 27−55.
8. *Афанасьев А.Н.*, *Войт Н.Н.*, *Уханова М.Е.*, *Ионова И.С.*, *Епифанов В.В.* Анализ конструкторско-технологических потоков работ в условиях крупного радиотехнического предприятия // Радиотехника. 2017. № 6. С. 49−58.
9. *Afanasyev A.N.*, *Voit N.N.*, *Kirillov S.Y.* Development of RYT-grammar for analysis and control dynamic workflows // 2017 International Conference on Computing Networking and Informatics (ICCNI). IEEE. 2017. P. 1−4.
10. *Афанасьев А.Н.*, *Войт Н.Н.* Грамматико-алгебраический подход к анализу и синтезу диаграмматических моделей гибридных динамических потоков проектных работ // Информационно-измерительные и управляющие системы. 2017. Т. 15. № 12. С. 69−78.
11. *Voit N.* Development of timed RT-grammars for controlling processes in cyber-physical systems // INTERACTIVE SYSTEMS: Problems of Human–Computer Interaction. Collection of scientific papers. Ulyanovsk: USTU. 2017. 290 p.
12. *Афанасьев А.Н.*, *Войт Н.Н.*, *Уханова М.Е.* Контроль и анализ денотативных и сигнификативных семантических ошибок диаграмматических моделей потоков работ в проектировании автоматизированных систем // Радиотехника. 2018. № 6. С. 84−92.
13. *Афанасьев А.Н.*, *Войт Н.Н.* Грамматико-алгебраический подход к анализу гибридных динамических потоков проектных работ // Сб. статей Всерос. научно-технич. конф. «Информационные технологии и информационная безопасность в науке, технике и образовании «ИНФОТЕХ-2017». Севастопольский государственный университет. Институт «Информационные технологии и управление в технических системах». 2017. С. 43−48.
14. *Кириллов С.Ю.*, *Войт Н.Н.*, *Молотов Р.С.*, *Степанов А.С.*, *Воеводин Е.Ю.*, *Бригаднов С.И.* Разработка и исследование методов анализа и контроля семантической целостности и согласованности диаграмматических моделей динамических распределенных потоков работ на основе временной RV-грамматики // Сб. научных трудов IX Всерос. школы-семинара аспирантов, студентов и молодых ученых «Информатика, моделирование, автоматизация проектирования». 2017. С. 135−139.
15. *Войт Н.Н.* Методы и средства автоматизации проектирования потоков работ // Информационно-измерительные и управляющие системы. 2018. Т. 16. № 11. С. 84−89. DOI: 10.18127/j20700814-201811-14.

Поступила 28 ноября 2019 г.

UDC 658.512

Development of a method for analyzing work flow instances based on a temporary automatic RVTI-grammar

N.N. Voit – Ph.D.(Eng.), Associate Professor, Head of Laboratory of Innovative Virtual Design and Training Technologies of Department of Scientific Research and Innovation, Department «Computer Engineering», Ulyanovsk State Technical University
E-mail: n.voit@ulstu.ru
S.Yu. Kirillov – Post-graduate Student, Department «Computer Engineering», Ulyanovsk State Technical University
E-mail: kirillovsyu@gmail.com
D.S. Kanev – Ph.D.(Eng.), Head of Scientific and Technical Department, Ulyanovsk State Technical University
E-mail: dima.kanev@gmail.com
A.S. Stepanov – Junior Research Scientist, Department «Computer Engineering», Ulyanovsk State Technical University
E-mail: step_al_ul@mail.ru
R.F. Gainullin – Ph.D.(Eng.), Programmer of LLC «Equid», Ulyanovsk State Technical University
E-mail: r.gainullin@gmail.com

Abstract

Methods of analysis of workflows in CAD and ASTPP can be used to study the qualitative and quantitative characteristics of design workflows. Qualitative characteristics are understood as logical and algebraic correctness of workflows formalized using graph theory, workflow networks, matching matrices, graphic modeling languages, including Unified Model Language, Business Process Management Notation, IDEF0 and eEPC, etc., as well as evolutionary approach, propositional logic, etc. Quantitative characteristics represent the effectiveness of the execution of workflows in CAD and ASTPP by parameters, for example, such as average service time, utilization rate of production capacity awns (downtime), etc. Evaluation of the effectiveness of workflows is carried out using simulation modeling (Petri nets), Markov chains and queuing theory (queuing systems), etc.

There is a modern theory of graphic languages for representing diagrammatic models of workflows that contains syntactic models in spatial and logical forms, including attributes of graphic objects (for example, a rectangle or a circle) and communication types. The spatial model has relative or absolute coordinates of graphic objects. The application of a spatial model is complex to control, analyze the structure or topology (syntax) and the attributes of diagrams. Typically, a logical model is used to describe the syntax of diagrammatic models based on graphical grammar. The use of a temporary machine in the design, specification, control and analysis of workflows in the development of complex technical systems in an industrial enterprise is a well-known practice. Temporary and hybrid automata are used to analyze and manage workflows when resolving problems of access to resources, blocking, liveness restrictions (liveness, reversibility, boundedness, reachability, dead transitions, deadlocks, home states). Examples of tasks are controlling the temperature of an atomic reactor, controlling a barrier at the intersection of railways in which temporary context-free grammars have been successfully applied, and also the task of describing the structure of ribonucleic acid (RNA). The presence of a large number of interacting complex automated systems poses the problem of formal control and analysis, which can be accomplished by various methods.

A promising approach for processing diagrammatic workflows is syntactically oriented based on formal grammars. The most famous are web grammar, positional grammar, relational grammar, multi-level graph grammar and preserving graph grammar. Positional grammars are the simplest. Developing on the basis of Plex structures, they inherited their shortcomings. These grammars do not imply the use of join areas. They cannot be used for graphic languages whose graphic objects have a dynamic variable number of inputs/outputs; they cannot be used to control the syntax of graphic languages containing parallelism. The advantage of relational grammars is the ability to handle errors, but they do not have a mechanism to neutralize such errors. Multilevel and preserving graph grammars are able to provide an analysis of graphical languages with a «deep» contextual dependence, which is necessary in languages that allow you to specify the synchronization of actions performed. Examples of such languages are the languages of flowcharts and message flow diagrams (Message Sequence Charts). Common disadvantages of the above grammars are: 1) the increase in the number of products in the construction of grammar for unstructured graphic languages, i.e. with a constant number of graphic language primitives, a significant increase in the number of products occurs, since it is necessary to determine all possible options for unstructured; 2) the complexity of constructing grammar (increasing the complexity of products and their number), and for some formalisms the impossibility of constructing grammar for graph schemes with unstructured parallelism; 3) great time complexity. Analyzers based on the considered grammars offer polynomial or exponential time for analyzing diagrams of graphic languages. The main limitation of the above methods is that they do not work if there are different types of diagrams (temporal, multilevel, etc.) at the same time, which means that in some cases the input diagrams cannot be analyzed.

Thus, the development and study of a method for analyzing workflow instances based on a temporary automatic RVTI grammar is an urgent scientific and technical task.

Keywords

Business processes, graphical language analysis, EPC.

The study was carried out with the financial support of the Russian Federal Property Fund in the framework of the scientific project № 17-07-01417.
The study was financially supported by the Russian Federal Property Fund and the Government of the Ulyanovsk Region as part of a scientific project № 18-47-730032.
The study was supported by a grant from the Ministry of Education and Science of the Russian Federation, project № 2.1615.2017/4.6.

DOI: 10.18127/j20700814-202001-06

References

1. *Feder J.* Plex languages. Information Sciences. 1971. V. 3. № 3. P. 225–241.
2. *Fu K.* Strukturnye metody v raspoznavanii obrazov. M.: Mir. 1977. 319 s. (In Russian).
3. *Pfaltz J.L.*, *Rosenfeld A.* Web Grammar. Proc. of the International Joint Conference on Artificial Intelligence. Washington, D.C. 1969. P. 609–619.
4. *Fu K.* Strukturnye metody v raspoznavanii obrazov. M.: Mir. 1977. 319 s. (In Russian).
5. *Costagliola G. et al.* Positional grammars: A formalism for LR-like parsing of visual languages. Visual Language Theory. Springer, New York, NY. 1998. P. 171–191.
6. *Costagliola G.*, *Chang S.K.* Using linear positional grammars for the LR parsing of 2-D symbolic languages. Grammars. 1999. V. 2. № 1. P. 1–34.
7. *Rekers J.*, *Schürr A.* Defining and parsing visual languages with layered graph grammars. Journal of Visual Languages & Computing. 1997. V. 8. № 1. P. 27–55.
8. *Afanasev A.N.*, *Voit N.N.*, *Ukhanova M.E.*, *Ionova I.S.*, *Epifanov V.V.* Analiz konstruktorsko-tekhnologicheskikh potokov rabot v usloviyakh krupnogo radiotekhnicheskogo predpriyatiya. Radiotekhnika. 2017. № 6. S. 49–58. (In Russian).
9. *Afanasyev A.N.*, *Voit N.N.*, *Kirillov S.Y.* Development of RYT-grammar for analysis and control dynamic workflows. 2017 International Conference on Computing Networking and Informatics (ICCNI). IEEE. 2017. P. 1–4.
10. *Afanasev A.N.*, *Voit N.N.* Grammatiko-algebraicheskii podkhod k analizu i sintezu diagrammaticheskikh modelei gibridnykh dinamicheskikh potokov proektnykh rabot. Informatsionno-izmeritelnye i upravlyayushchie sistemy. 2017. T. 15. № 12. S. 69–78. (In Russian).
11. *Voit N.* Development of timed RT-grammars for controlling processes in cyber-physical systems. INTERACTIVE SYSTEMS: Problems of Human–Computer Interaction. Collection of scientific papers. Ulyanovsk: USTU. 2017. 290 p.
12. *Afanasev A.N.*, *Voit N.N.*, *Ukhanova M.E.* Kontrol i analiz denotativnykh i signifikativnykh semanticheskikh oshibok diagrammaticheskikh modelei potokov rabot v proektirovanii avtomatizirovannykh sistem. Radiotekhnika. 2018. № 6. S. 84–92. (In Russian).
13. *Afanasev A.N.*, *Voit N.N.* Grammatiko-algebraicheskii podkhod k analizu gibridnykh dinamicheskikh potokov proektnykh rabot. Sb. statei Vseros. nauchno-tekhnich. konf. «Informatsionnye tekhnologii i informatsionnaya bezopasnost v nauke, tekhnike i obrazovanii «INFOTEKH-2017». Sevastopolskii gosudarstvennyi universitet. Institut «Informatsionnye tekhnologii i upravlenie v tekhnicheskikh sistemakh». 2017. S. 43–48. (In Russian).
14. *Kirillov S.Yu.*, *Voit N.N.*, *Molotov R.S.*, *Stepanov A.S.*, *Voevodin E.Yu.*, *Brigadnov S.I.* Razrabotka i issledovanie metodov analiza i kontrolya semanticheskoi tselostnosti i soglasovannosti diagrammaticheskikh modelei dinamicheskikh raspredelennykh potokov rabot na osnove vremennoi RV-grammatiki. Sb. nauchnykh trudov IX Vseros. shkoly-seminara aspirantov, studentov i molodykh uchenykh «Informatika, modelirovanie, avtomatizatsiya proektirovaniya». 2017. S. 135–139. (In Russian).
15. *Voit N.N.* Metody i sredstva avtomatizatsii proektirovaniya potokov rabot. Informatsionno-izmeritelnye i upravlyayushchie sistemy. 2018. T. 16. № 11. S. 84–89. DOI: 10.18127/j20700814-201811-14. (In Russian).