СИСТЕМЫ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ

УДК 681.39

В.В. Александров, Н.П. Макаров, А.С. Шустов

АВТОМАТИЗИРОВАННЫЙ АНАЛИЗ И ОЦЕНКА СТАТЕЙ КОЛЛЕКТИВНЫХ ДОГОВОРОВ

Предложена реализация метода автоматизированного анализа и оценки эффективности коллективных договоров – правовых актов, реализующих договорную форму регулирования трудовых отношений. Метод базируется на использовании количественного показателя эффективности, который рассчитывается непосредственно по тексту договора путем выявления и суммирования статей, реализующих его правовые функции. Приведен пример реализации метода средствами информационных технологий. Для сокращения общего времени анализа договоров предлагается автоматизировать процесс оценки правового качества отбираемых экспертом статей за счет использования метода обработки форм слов и числительных естественных языков. Разработанный метод анализа коллективных договоров внедрен в ведомственной лаборатории автоматизированного анализа коллективно-договорных актов при РГРТУ.

Ключевые слова: *эффективность коллективного договора, автоматическая обработка текста, обработка количественных числительных.*

Введение. Коллективный договор – правовой акт, обеспечивающий реализацию договорной формы регулирования трудовых отношений (она дополняет государственное регулирование, которое осуществляется посредством трудового законодательства) (ч. 1 ст. 9 Трудового кодекса РФ – далее ТК РФ). Содержание коллективного договора (КД) определяют его стороны (ч. 1 ст. 37 ТК РФ), которыми являются работодатель и работники предприятия (организации) в лице своих представителей. С учетом финансово-экономического положения на предприятии (в организации) стороны могут устанавливать в КД льготы и преимущества для работников, условия труда (оплата, охрана труда, социальные гарантии и др.), более благоприятные по сравнению с установленными законами, иными нормативными правовыми актами, соглашениями (ст. 41 ТК РФ). Чем выше будет дополнительный уровень трудовых прав и гарантий, предоставляемый работникам через КД, тем выше будет его эффективность. Однако анализ практики заключения КД, проводимый Минтрудом России [1], показывает, что они имеют низкую эффективность и не обеспечивают реализации в полной мере возможности договорного регулирования.

Эффективность КД определяется качеством содержащихся в нем договоренностей сторон, оформленных в виде нормативных и обязательственных положений (статей) по тем или иным вопросам социально-трудовых отношений. Поэтому одним из основных этапов решения проблемы повышения эффективности договорного регулирования является анализ содержания множества заключенных КД с целью определения их количественных, а не качественных, как это делается сейчас, показателей эффективности $Эф_i$, $i = 1, 2, \ldots, n$. Количественный показатель позволит объективно сравнивать договорные акты между собой, отбирать лучшие и на этой основе вырабатывать рекомендации сторонам, делать заключение о качестве договорного регулирования в том или ином множестве из **n** организаций.

Целью работы является разработка метода автоматизированного анализа и количественной оценки эффективности содержания КД, позволяющего упростить работу экспертов и опера-

тивно получать оценки эффективности КД, заключаемых в организациях отрасли или региона – субъекта РФ.

Методика количественной оценки эффективности содержания коллективных договоров. Методика базируется на системном анализе коллективно-договорного регулирования трудовых отношений, проведенном одним из авторов в работе [2] и показавшем, что в качестве количественной оценки эффективности КД следует использовать обеспечиваемую им величину приращения уровня трудовых прав и гарантий для работников организации. Оценка этой величины характеризуется значением безразмерного показателя $Э_Ф$, который в отличие от известных показателей, например экономического [3], непосредственно зависит от степени реализации в договорном акте его правовых функций и всегда доступен для измерения и контроля.

Доступность объясняется тем, что значения показателя эффективности $Э_Ф$ рассчитываются непосредственно по тексту КД путем выявления и определения количества статей (или пунктов), в которых реализуются его правовые функции:

$$Э_Ф = N_{\mathbf{Вы}}+N_{\mathbf{До}}+N_{\mathbf{Св}},$$

где

$N_{\mathbf{Вы}}$ – число статей договора, которые по**Вы**шают уровень трудовых прав и гарантий работников (реализация основной функции договора, определенной в ст. 41, часть 3 ТК РФ);

$N_{\mathbf{До}}$ – число статей, в которых реализуется первичное правовое регулирование (**До**зволение решать в организации) вопросов социально-трудовых отношений (реализация второй функции договора);

$N_{\mathbf{Св}}$ – число статей, в которых решаются **Св**ои, характерные для предприятия (организации) проблемы, то есть реализация третьей правовой функции договора – ликвидация пробелов в праве.

Поскольку в договоры обычно включаются договоренности по четырем составляющим социально-трудовых отношений сторон, то предлагается показатель эффективности $Э_Ф$ представить во взаимосвязи с количественными показателями, отдельно характеризующими эффективность решения вопросов этих отношений. Для представления достаточно сгруппировать пункты КД с соответствующими договоренностями. В результате получаем следующую взаимосвязь показателя эффективности КД с показателями эффективности его разделов, представляемых безразмерными коэффициентами:

$$Э_Ф = К_{эТР}+К_{эСЦ}+К_{эГДП}+К_{эПСП}, \quad (1)$$

где

$К_{эТР} = N_{\mathbf{Вы}/ТР}+N_{\mathbf{До}/ТР}+N_{\mathbf{Св}/ТР}$ – коэффициент эффективности решения в КД вопросов трудовых отношений; эти решения являются статьями (договоренностями) разделов: трудовой договор, занятость, рабочее время и время отдыха, оплата труда, охрана труда, которые занимают, примерно, 60 % содержания КД;

$К_{эСЦ} = N_{\mathbf{Вы}/СЦ}+N_{\mathbf{До}/СЦ}+N_{\mathbf{Св}/СЦ}$ – коэффициент эффективности решения в договоре вопросов социальных отношений – предоставление работникам социально-бытовых услуг;

$К_{эГДП}=N_{\mathbf{До}/ГДП}+N_{\mathbf{Вы}/ГДП}+N_{\mathbf{Св}/ГДП}$ – коэффициент эффективности решения вопросов обеспечения работодателем условий и гарантий деятельности профкома как представительного органа работников;

$К_{эПСП} = N_{\mathbf{До}/ПСП}+N_{\mathbf{Вы}/ПСП}+N_{\mathbf{Св}/ПСП}$ – коэффициент эффективности решения вопросов взаимоотношения сторон (их представителей) как социальных партнеров в процессах заключения и выполнения договора, при обеспечении в организации социального мира;

$N_{F/O}$ – означает число статей, в формулах всех коэффициентов N с индексами F/O означает число статей, реализующих правовые функции F (F∈Вы, До, Св) при регулировании тех или иных отношений О (О∈Тр, Сц, ГДП, ПСП).

Коэффициенты $К_{эТР}$, $К_{эСЦ}$, $К_{эГДП}$, $К_{эПСП}$ в своей совокупности более полно характеризуют эффективность содержания КД. Чем больше будут их значения, тем выше будет эффективность соответствующего им КД.

Порядок оценки эффективности коллективных договоров. Рассмотренная методика оценки эффективности КД была реализована средствами информационных технологий в виде автоматизированной системы [4]. В ней процесс анализа и оценки КД разбит на этапы. Начало каждого этапа инициируется пользователем – экспертом. Вначале текст анализируемого договора вводится в систему как текст-копия и ему присваиваются атрибуты принадлежности к организации, в которой он заключен, и даты заключения. На этапе анализа: 1) эксперт последовательно выделяет фрагменты текста КД, представляющие собой статью или часть пункта договорного акта, которые обеспечивают возможность извлечения суждения, касающегося одного и только одного социально-трудового параметра или одной правовой нормы; 2) идентифицирует социально-трудовые параметры или правовые нормы и присваивает им значения классификационных характеристики «Раздел», «Вопрос»; 3) соотносит содержание фрагмента с содержанием соответствующей статьи трудового зако-

нодательства и определяет значение характеристики «Качество» фрагмента (Вы, До, Св); 4) после определения всех классификационных характеристик фрагмент заносится в базу данных системы (БД) и вычеркивается из текста-копии. После вычеркивания из текста-копии КД последнего фрагмента система рассчитывает значения коэффициентов $К_{эТР}$, $К_{эСЦ}$, $К_{эГДП}$, $К_{эПСП}$ и анализ КД прекращается. Проанализированный КД сохраняется в БД системы в виде проиндексированных фрагментов. В дальнейшем они могут использоваться для обобщения опыта решения тех или иных вопросов в договорном порядке и для автоматизации определения содержания вновь анализируемых фрагментов.

При определении значения атрибута «Качество» эксперт имеет возможность получить справочную информацию по значениям характеристик «Раздел», «Вопрос». Справка выдается либо в виде необходимых фрагментов статей и положений из законодательных и нормативных правовых актов, либо похожих фрагментов из ранее проанализированных КД. Таким образом, время оценки качества анализируемого фрагмента существенно сокращается по сравнению с ручным анализом.

Однако в условиях постоянно возрастающего потока вновь заключаемых КД, применение справочной системы не дает желаемого сокращения времени анализа, что приводит к увеличению очереди КД на анализ и нагрузки на эксперта. В качестве первого шага к разрешению этой проблемы предлагается использовать различные методы автоматической обработки текста.

Автоматизированный анализ и оценка статей. Предлагается использовать методы обработки форм слов и числительных [5-7] для автоматизации процесса оценки правового качества статьи КД, анализируемой экспертом. Это позволяет еще в большей степени сократить общее время анализа КД. Предпосылками для этого являются следующие моменты.

1. В БД содержится достаточное количество фрагментов ранее анализируемых КД, которые являются примерами статей (договоренностей сторон) с теми или иными правовыми качествами.

2. Статьи вновь заключаемого договора в организации обычно формируются следующим образом: сохраняются, то есть переносятся из предыдущей редакции КД, получаются из соответствующих статей предыдущей редакции КД путем их частичной переработки, статьи отображают новые договоренности сторон и являются полностью новыми. Практика показывает, что число новых статей во вновь заключаемых КД организаций составляет 15-30 % от общего числа статей. Предполагается, что эту часть статей и останется проанализировать экспертам.

Известные алгоритмы сравнения текстов (выявления дубликатов) для решения задачи сравнения статей КД неприменимы. В работе [8] сделан такой вывод об этих алгоритмах: «Лучшие результаты по точности были у алгоритмов, базирующихся на использовании более длинных фрагментов текста». Статьи КД являются короткими текстами и содержат не более 15-60 слов. Также помимо расчета степени сходства необходимо показать эксперту отличия статей.

Изложенные способы формирования статей приводят к следующим параметрам сравнения статей [9].

1. Количество одинаковых словоформ в сравниваемых статьях F.

2. Количество одинаковых слов (лексем) в сравниваемых статьях S. При подсчете параметра используется словарь синонимов (например, «*работодатель*» – «*университет*»). Основы синонимов считаются одинаковыми.

3. Количество общих последовательностей словоформ P. Общей последовательностью считаются два и более слов подряд, встречающихся в обоих статьях.

Коэффициент сходства учитывает данные параметры и рассчитывается по следующей формуле:

$$K = c_1 \cdot F/N + c_2 \cdot S/N + c_3 \cdot 1/P \cdot W/N,$$

где c_1, c_2, c_3 – весовые коэффициенты (экспериментально подобраны значения 0,4; 0,5; 0,1 соответственно); N – количество слов в анализируемой статье; W – количество слов в последовательностях.

Если последовательности в статье не выявлены, значение $1/P \cdot W/N$ принимается равным 0.

Рассмотрим примеры аналогов и расчета коэффициента K для статей реальных КД.

Пример 1. (Почти полное совпадение). Для статьи (подчеркнуты общие последовательности)

«*Работодатель обязуется при наличии письменных заявлений работников и обучающихся ежемесячно и бесплатно перечислять на счет профсоюзной организации профсоюзные взносы.*»

найден аналог

«*Университет обязуется при наличии письменных заявлений работников ежемесячно и бесплатно перечислять на счет профсоюзной организации профсоюзные взносы.*»

при коэффициенте сходства K = 0,87.

Пример 2. (Частичное совпадение). Для статьи

«- расследование и учет в установленном порядке несчастных случаев на производстве и профессиональных заболеваний;»
найден аналог

«*Обеспечивает участие профкома в расследовании несчастных случаев на производстве и профессиональных заболеваний.*»,
а коэффициент сходства составляет K = 0,72.

Пример 3. (Случайное совпадение.). Для статьи

«*Ректор обязуется предоставлять профкому бесплатную, достоверную и полную информацию по любому вопросу, связанному с использованием труда и социальным положением работников.*»
найден аналог

«- *предоставлять представителям работников полную и достоверную информацию, необходимую для заключения коллективного договора, соглашения и контроля за их выполнением;*»
с коэффициентом сходства K = 0,27.

В статьях КД могут использоваться числительные, записанные в цифровой и словесной форме, например:

«*Выдачу заработной платы штатным работникам осуществлять два раза в месяц не позднее 10 и 25 числа каждого месяца.*»

Перед сравнением в тексте статьи количественные числительные в словесной форме заменяются числительными в цифровой форме с помощью метода обработки числительных [7].

Коэффициент сходства используется при автоматизированном анализе и оценки статей. Алгоритм анализа и оценки статей КД состоит из следующих шагов.

1. Эксперт выбирает источник для поиска аналога статей (например, поиск по всей БД или по определенной редакции анализируемого КД).

2. Эксперт выделяет статью анализируемого договора и передает системе на анализ.

3. Из БД происходит выборка фрагментов из выбранного экспертом источника.

4. Анализируемая статья сравнивается с другими статьями и для каждой пары вычисляется коэффициент сходства.

5. Отобранные статьи-аналоги ранжируются по убыванию значения коэффициента сходства. Определяется статья с максимальным коэффициентом.

6. Если максимальное значение коэффициента превышает значение 0,85, то статья считается аналогом, анализируется автоматически, получая оценки аналога. Если максимальное значение коэффициента лежит в пределах от 0,6 до 0,85, то статья выдается эксперту вместе с аналогами и эксперт принимает решение, нужно ли переносить оценки аналога анализируемой статье или нет. В остальных случаях эксперт анализирует статью и выставляет оценки самостоятельно.

Пороговые значения установлены в результате анализа оценок более 1200 статей КД и решений, принятых экспертами, за последние 2 года. Направлением дальнейшего развития является автоматизация выбора пороговых значений на основе данных решений.

Разработка и исследование алгоритма анализа и оценки статей КД с использованием коэффициента сходства выполнены под руководством Александра Викторовича Пруцкова.

Заключение. В работе получены следующие результаты:

1) разработан метод автоматизированного анализа и оценки эффективности содержания КД, характеризуемой коэффициентами $К_{эТР}$, $К_{эСЦ}$, $К_{эГДП}$, $К_{эПСП}$, которые рассчитываются непосредственно по тексту договора путем выявления и суммирования статей по формуле (1);

2) метод реализован средствами информационных технологий;

3) для сокращения времени выявления статей, реализующих правовые функции КД и необходимых для расчетов коэффициентов эффективности $К_{эТР}$, $К_{эСЦ}$, $К_{эГДП}$, $К_{эПСП}$ по формуле (1), разработан алгоритм автоматизированного анализа и оценки статей КД, использующий коэффициент сходства; данный алгоритм упрощает работу эксперта, сокращает время анализа КД более чем на 30 %, что способствует оперативному получению оценок эффективности КД, заключаемых в организациях отрасли или региона РФ.

Метод автоматизированного анализа и оценки эффективности КД внедрен в ведомственной лаборатории анализа коллективно-договорных актов образования, созданной по предложению Центрального Совета профсоюза работников народного образования и науки РФ и Министерства образования РФ.

Библиографический список

1. *Николаева Д.* Коллективный договор пролетариату не оружие // [Электронный ресурс]. URL: http://www.kommersant.ru/doc/1620660.

2. *Александров В.В.* Системный анализ коллективно-договорного регулирования трудовых отношений // Труд и социальные отношения. Соискатель. – М.: АТиСО, 2002. – № 1. – С. 73-85.

3. *Бородин И.И.* Об эффективности коллективного договора // Труд и социальные отношения. Спецвыпуск науч. журнала. – М.: АТиСО, 2001. – С. 134-143.

4. *Макаров Н.П., Александров В.В.* Использование информационной системы для оценки правовой

эффективности коллективно-договорных актов // «Информационное общество и актуальные проблемы экономических, гуманитарных, правовых и естественных наук»: материалы VI межвуз. науч.-практ. конф. – Рязань: Рязанский филиал МЭСИ, 2010. – С. 122-124.

5. *Пруцков А.В.* Морфологический анализ и синтез текстов посредством преобразований форм слов // Вестник Рязанской государственной радиотехнической академии. – 2004. – № 15. – С. 70-75.

6. *Пруцков А.В.* Методы поиска решений в лингвистических автоматизированных обучающих системах // Научно-техническая информация. Серия 2: Информационные процессы и системы. – 2006. – № 4. – С. 15-18.

7. *Пруцков А.В.* Обработка числительных естественных языков с помощью формальных грамматик и нормальных алгоритмов Маркова // Вестник Рязанского государственного радиотехнического университета. – 2009. – № 28. – С. 49-55.

8. *Захаров В.Н., Хорошилов А.А.* Автоматическая оценка подобия тематического содержания текстов на основе сравнения их формализованных смысловых описаний // Труды 14-й Всерос. науч. конф. «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» RCDL-2012. – Переславль-Залесский, 2012. – С. 143-149.

9. *Пруцков А.В., Розанов А.К.* Программное обеспечение методов обработки форм слов и числительных // Вестник Рязанского государственного радиотехнического университета. – 2011. – № 38. – С. 78-82.

УДК 332.14+352

И.Ю. Каширин, Н.С. Курдюков

SIR – АЛГОРИТМ АВТОМАТИЧЕСКОГО ПОСТРОЕНИЯ ИНТЕРФЕЙСОВ ВЗАИМОДЕЙСТВИЯ ВЕБ-СЕРВИСОВ

Предложен алгоритм авто-построения интерфейсов взаимодействия веб-сервисов, рассмотрены результаты работы алгоритма.

Ключевые слова: *онтология, веб-сервис, SOA, дескриптивная логика.*

Введение. Рост популярности за последние годы в сети Internet таких технологий как облачные вычисления (cloud computing), семантические сети [8] и Semantic web демонстрирует актуальность задачи организации взаимодействия Internet-ресурсов между собой. Обеспечить такое взаимодействие позволяет сервис-ориентированная архитектура(SOA) разработки программного обеспечения. На данный момент существует множество систем для создания веб-приложений с SOA архитектурой: IBM Lotus Notes, Microsoft .NET, Oracle SOA Suite. В то же время не существует эффективных систем, способных автоматически строить интерфейсы и превращать обычные сайты старого поколения в веб-сервисы. Необходимость таких систем обусловливается проблемами реализации EAI (Enterprise Application Integration) [1]. Одной из этих проблем является фактор постоянного развития сети Internet. При этом постоянное изменение архитектуры и разработка заново уже существующих интерфейсов требует непрерывной работы специалистов.

Рассматриваемая задача авторами настоящей статьи решена впервые с помощью применения парадигмы основанного на онтологиях доступа к данным(OBDA) [2]. Этот подход позволил использовать преимущества онтологий и технологий дескриптивных логик. Упростилась работа по изменению структуры SOA-системы по сравнению с IBM Lotus Notes и Oracle SOA Suite. Стало достаточно изменить онтологию системы, что не составляет труда для человека без специального образования с помощью редактора онтологий. При этом открывается возможность построения интерфейсов на базе семантической сервис-ориентированной архитектуры(SSOA), чем обеспечивается дальнейшее эффективное развитие технологии Semantic web.

Цель работы – *описание нового алгоритма, способного автоматически строить интерфейсы взаимодействия веб-сервисов посредством логического анализа.*

Теоретические исследования. В качестве основной теоретической концепции используется сервис-ориентированная архитектура(SOA), являющаяся парадигмой разработки программного обеспечения посредством использования распределённых слабо связанных компонентов[3]. При разработке системы для сети Internet основным объектом концепции SOA выступает веб-сервис – идентифицируемая веб-адресом программная система, оснащённая стандартизированными интерфейсами для взаимодействия по стандартизированным протоколам. Основой работы веб-сервисов может послужить предос-

тавление актуальной информации для заранее созданной базы данных. В том случае, если сервис предлагает услуги по обработке данных, он может запрашивать данные у другого сервиса и обрабатывать эти данные. Для построения интерфейса между двумя веб-сервисами необходимо иметь следующую информацию в виде триплета значений:

$$P = <C, Q, S>,$$

где C – адрес веб-сервиса, запрашивающего информацию (клиент); Q – запрашиваемая информация; S – адрес веб-сервиса, содержащего запрашиваемую информацию (сервер).

Для работы системы авто-построения интерфейсов веб-сервисов необходимо описать схемы взаимодействия веб-сервисов. Далее требуется разработать интеллектуальную систему, способную автоматически строить интерфейсы по запросу пользователя на основе описанных схем. Такую систему можно построить с помощью технологии OBDA. Основанным на онтологиях доступом к данным (Ontology-Based Data Access, OBDA) – называется доступ к данным через высокоуровневый интерфейс онтологии. Этот подход использует преимущества дескриптивных логик для работы с онтологиями для извлечения запросов к базам данных [2]. Основной принцип работы систем OBDA заключается в переписывании запросов к онтологии в соответствующие запросы к базам данных [4, 5]. Базы данных в технологии OBDA являются структурой набора утверждений (Abox) онтологии.

Для функционирования системы автопостроения интерфейсов взаимодействия вебсервисов авторами настоящей статьи получен алгоритм, использующий концепцию OBDA. Этот алгоритм именуется – SIR (Service Interfaces Rendering). Алгоритм SIR посредством логического анализа к БЗ автоматически строит интерфейсы между веб-сервисами на основе входных параметров, состоящих из запросов к базе знаний.

Далее представлен пример онтологии взаимодействия веб-сервисов, с которой алгоритм SIR гарантирует свою работоспособность.

Онтология государственных и муниципальных учреждений SAMAO (State And Municipal Agencies Ontology). Набор аксиом (Tbox) онтологии SAMAO содержит данные о ведомствах и их территориальном расположении. Abox онтологии SAMAO содержит данные об адресах вебсервисов и данные, предоставляемые веб-сервисами.

Входными параметрами алгоритма запросов к онтологии являются Tbox и триплет *P* (SIR-триплет). Выходными данными алгоритма является объединение SIR-триплетов с запросами к базам данных. Далее запросы автоматически оборачиваются в стандартный код интерфейса веб-сервисов. Полученные интерфейсы размещаются на веб-сервисах, исходя из содержания выходных данных алгоритма SIR.

Далее приведены примеры входных параметров алгоритма SIR и соответствующих выходных данных.

Параметры без зависимых условий. Требуется построить интерфейсы (Клиент C, Данные Q, Сервер S), где C – это сайт образовательного портала (EducationPortal) города Рязани (RyazanCity). В веб-сервис C будут отправляться данные из веб-сервисов S, содержащие данные Q о телефонах учреждений (Phone). Веб-сервисы S, в свою очередь, являются образовательными сайтами учебных заведений (Education) города Рязани (RyazanCity):

C(x1, x2) ← EducationPortal(x1), RyazanCity(x2)
Q(x3) ← Phone(x3)
S(x4, x5) ← Education(x4), RyazanArea(x5)

Приведем один из SIR-триплетов, полученных в качестве выходных данных. Этот триплет содержит информацию для создания триплета следующего содержания. Образовательный портал (EducationPortal) города Рязани (RyazanCity) запрашивает данные о телефонах (Phone) у сайтов средних общеобразовательных школ (HighSchool) Рязанского района (RyazanDistrict).

C(x1, x2) ← EducationPortal(x1), RyazanCity(x2)
Q(x3) ← Phone(x3)
S(x4, x5) ← HighSchool(x4), RyazanDistrict(x5)

Параметры с зависимыми условиями. Запрос адреса сервера и запросы адреса клиента могут содержать одинаковые элементы условий. Такие параметры называются параметрами с зависимыми условиями.

Цель алгоритма – построить интерфейсы (Клиент C, Данные Q, Сервер S), где на всех сайтах образовательных учреждений (EducationPortal) каждого города (City) центрального региона (CentralRegion) отобразить данные о персонале (Personal) из веб-сервисов S. Веб-сервисы S, в свою очередь, являются образовательными сайтами города (City), указанного в C. Зависимые условия обозначаются одинаковыми переменными элементов условия запроса. В следующем примере – это переменная x3.

C(x1, x2, x3) ← EducationPortal(x1), CentralRegion(x2), City(x3)
Q(x4) ← Personal(x4)
S(x5, x3) ← Education(x5), City(x3)

Далее приведен один из SIR-триплетов, полученных в качестве выходных данных. Этот триплет содержит информацию для создания

триплета следующего содержания. Образовательный портал (EducationPortal) города Рязани (RyazanCity) запрашивает данные о профессорах (Professor) у сайтов университетов (University) города Рязани (RyazanCity):

C(x1, x2, x3) ← EducationPortal(x1), RyazanCity(x3)

Q(x4) ← Professor(x4)

S(x5, x3) ← RyazanCity(x5), University(x3)

Представим описание алгоритма SIR с помощью псевдокода и информацию о принципе его работы.

Алгоритм SIR (C, S, Q, T)

Входные параметры: конъюнктивные запросы: C, S, Q; TBox T.

Выходные данные: коллекция SIR-триплетов; SIRset.

```
uC := Rewriting(C);
uQ := Rewriting(Q);
DTDic := DTRew(C, uC, S, T);
if DTDic = 0 do
then
    uS := Rewriting(S);
    foreach  uC' in uC do
        SIRSet :=  SIRSet ^ SIRTrip (to := uC',
        query := uQ', from := uS');
else
    foreach  uC' in uC do
        uC'S := Rewriting(DTDic[uC']);
        foreach uC'S' in uC'S do
            SIRSet := SIRSet ^ SIRtrip (to :=
            uC', query := uQ, from := uC'S');
return SIRSet;
```

Принцип работы алгоритма SIR. Входными параметрами являются SIR-триплет (C, Q, S) и Tbox(T) онтологии. Выходными данными является объединение (SIRSet) SIR-триплетов (SIRTrip). За переписывание запросов отвечает процедура Rewriting, содержащая алгоритм переписывания. Алгоритм состоит из следующих действий.

1. Посредством алгоритма переписывания запросов переписать C и поместить объединение запросов в uC.

2. Переписать запрос Q и поместить объединение запросов в uQ.

Если входные параметры не содержат зависимых элементов, то произведутся следующие действия.

1. Переписать запрос S и поместить объединение запросов в uS.

2. Для каждого uC' из uC, создать SIR-триплет где клиент = uC', сообщение = uQ' из uQ, сервис = uS' из uS.

3. Поместить триплет в SIRSet.

Если входные параметры содержат зависимые элементы, то произведутся следующие действия. Для обработки параметров с зависимыми элементами будет использоваться функция DTRew. Принцип работы функции DTRew заключается в следующем.

1. Если существуют переменные (varqC) элементов условий запроса C, идентичные переменным (varqS) условий запроса S, добавить эти переменные в коллекцию переменных зависимых элементов (DTSet).

2. Поместить S в C'S. Если в объединении переписанных запросов uC есть переменные элементов условий запроса uC', входящие в состав коллекции DTSet, то дописать этот элемент в C'S.

3. Поместить C'S в ассоциативный массив C'SDict[uC'].

4. После завершения работы функции для каждого uC' из uC переписать запрос DTDic[uC'] и поместить результат в uC'S.

5. Для каждого uC'S' из uC'S создать SIR-триплет где клиент = uC', сообщение = uQ' из uQ, сервис = uC'S'.

Далее представлено описание функции DTRew с помощью псевдокода.

Функция DTRew (C, uC, S, T)

Входные параметры: конъюнктивные запросы: C, S; объединение переписанных запросов uC; TBox T;

Выходные данные: ассоциативный массив C'SDict.

```
foreach C' in C do
foreach S' in S do
        foreach varqC in C' do
                foreach varqS in S' do
                        if (varqC = varqS) and
                        (not(C' in DTSet))
                        then C' apply DTSet;
if DTSet < > 0
then
foreach  uC' in uC do
        C'S := S
        foreach C'  in uC' do
                foreach varqC in C' do
                        if varqC in DTSet
                                then C'S :=
                                C'S ^ C';
                C'SDict[uC'] := C'S;
return C'SDict;
```

Экспериментальные исследования. Программный код алгоритма написан на языке программирования Java. Программа тестировалась на ОС Windows 8, 3 гб ОЗУ, AMD Phenom(tm) II N850 Triple-Core Processor 2,20 Гц. Для тестиро-

вания использовалась онтология SAMAO. Эта онтология была написана с помощью редактора онтологий Protégé на языке веб онтологий OWL 2 QL. Данные из Abox в этой реализации хранятся в СУБД MySQL. Для описания интерфейса веб-сервисов использованы стандарты SOAP и WSDL. Для сравнительного анализа использовались 2 алгоритма переписывания запросов: PerfectRef [4] и Requiem [5].

Время переписывания запросов для параметров без зависимых условий рассчитывалось по следующей формуле:

$$t(x) = t(q) + t(s) + t(c), \qquad (1)$$

где $t(q)$, $t(s)$, $t(c)$ – это время срабатывания алгоритма переписывания соответствующих запросов.

Время переписывания запросов для параметров с зависимыми условиями рассчитывалось по следующей формуле:

$$t(x) = t(q) + t(s) + t(d) + \sum_{k=0}^{n} t(u_k), \qquad (2)$$

где $t(q)$, $t(s)$ – это время срабатывания алгоритма переписывания соответствующих запросов, $t(u_k)$ – время срабатывания алгоритма переписывания для массива, порожденного функцией DTRew, $t(d)$ – время работы функции DTRew.

При работе алгоритма порождаются новые запросы к базам данных. Порожденные запросы сохраняются в текстовом формате. Для вычисления занимаемой памяти результатов работы алгоритма следует подсчитать количество порожденных запросов.

Количество порожденных запросов было подсчитано по формуле, идентичной формуле расчета времени работы алгоритма, с той лишь разницей, что вместо функции подсчета времени ($t(x)$) использовалась функция подсчета количества порожденных запросов.

Результаты тестирования приведены в таблице.

Тестирование алгоритма SIR

	Количество порожденных запросов				
Тип входных параметров	Кол-во запросов DTrew	Кол-во запросов PerfectRef	Кол-во запросов Requiem	Время DTRew, мс	Время PerfectRef, мс
Без зависимых условий	-	75	72	-	19
С зависимыми условиями	3	147	121	1	33

Заключение. Небольшое количество времени, требуемое для работы SIR алгоритма, и приемлемое количество порожденных запросов доказывают эффективность алгоритма системы авто-построения интерфейсов взаимодействия веб-сервисов. Кроме того, результаты экспериментов демонстрируют существенное уменьшение времени разработки Web-ресурсов при использовании алгоритма SIR в сравнении с ручным построением сервисов.

Для реализации алгоритма рекомендуется использовать языки веб-онтологий OWL [7] и OWL 2. В частности, профиль OWL 2 QL. Для представления выходных данных рекомендуется использовать SQL-подобные языки запросов.

Полученный алгоритм может использоваться в основе систем автоматического построения интерфейсов взаимодействия веб-сервисов. Структура онтологии, необходимой для работы алгоритма, имеет OWL-разметку стандарта Semantic web. Этот фактор открывает возможность перехода системы на базу семантической сервис-ориентированной архитектуры (SSOA), ориентированной на Semantic web сервисы [6].

Апробация алгоритма проводилась на данных веб-портала образовательных учреждений г. Рязани. В результате серии экспериментов зафиксировано среднее уменьшение трудоемкости создания веб-сервисов на базе веб-сайтов примерно на 25 %.

Библиографический список

1. *Gian Trotta.* Dancing Around EAI «Bear Traps». [Electronic resource] – http://www.ebizq.net. Retrieved 2006. – P. 3.

2. *Курдюков Н.С.* Основанный на онтологиях доступ к данным формата Semantic web // Общество, современная наука и образование: проблемы и перспективы: сборник научных трудов по материалам Международной научно-практической конференции. Часть 5. Тамбов. Бизнес-Наука-Общество, 2012. – С. 67 − 68.

3. *Michael Bell.* SOA Modeling Patterns for Service-Oriented Discovery and Analysis // Wiley & Sons, 2010. – P. 390.

4. *Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini,Riccardo Rosati.* Tractable Reasoning and Efficient Query Answering in Description Logics: The DL-Lite Family // Journal of Automated Reasoning, 2007. Vol. – 39(3). – P. 385 – 429.

5. *H. Pérez−Urbina, I. Horrocks, B. Motik.* Practical Considerations for Query Answering in OWL 2 // In Proc. of the OWL: Experiences and Directions Workshop. Chantilly, 2009. – P. 1 – 16.

6. *Armin Haller, Juan Miguel Gomez, Christoph Bussler.* Exposing Semantic Web Service principles in SOA to solve EAI scenarios // Workshop at 14th International World Wide Web Conference Chiba, Japan, 2005. – P. 1 – 13.

7. *Каширин И.Ю., Пылькин А.Н.* Структуризация и унификация онтологических описаний на языке OWL в задачах информационного поиска // Известия высших учебных заведений. Научно-технический журнал. Проблемы полиграфии и издательского дела №4. Москва 2008. – С. 45 – 57.

8. *Каширин И.Ю., Крошилин А.В., Крошилина С.В.* Автоматизированный анализ деятельности предприятия с использованием семантических сетей // Горячая линия – Телеком. Москва. 2011. – 140 с.

УДК 62.681.5.004

О.В. Фалеев

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ МОДУЛЯ АГРЕГАТНОГО КОМПЛЕКСА ТЕХНИЧЕСКИХ И ПРОГРАММНЫХ СРЕДСТВ

В статье предложена математическая модель модуля агрегатного комплекса технических и программных средств при создании систем автоматизации испытаний изделий ракетно-космической техники.

Ключевые слова: *модуль агрегатного комплекса, модель.*

Введение. Исследования в области автоматизации испытаний ракетно-космической техники показали, что, несмотря на все кажущееся многообразие решаемых системами автоматизации испытаний (САИ) задач, функции этих систем достаточно сильно пересекаются. Поэтому внедрение в процесс проектирования САИ политики унифицированных решений позволит повысить эффективность этого процесса, но при этом возникает вопрос об уровне унифицированных решений.

Среди разработчиков сложных систем широкое распространение получил метод их модульного построения [1]. Модульный принцип построения подразумевает сочетание функционального и конструктивного принципов разбиения. В основе этого принципа лежит многоуровневое разбиение на компоненты как по функциональному, так и по конструктивному исполнению. В области автоматизации испытаний ракетно-космической техники можно указать на такие специфические задачи, как выделение телеметрических параметров из потока телеметрических данных, согласование интенсивностей обработки этих параметров с выходным потоком телеметрической системы, задание различных по своей физической природе воздействий на испытуемое изделие и т.д.

С формальной точки зрения модуль можно рассматривать как математический объект, описываемый функциональными пространствами состояний входных и выходных сигналов и аналитических зависимостей критериальных показателей от параметров этого модуля [2].

В целом проблему создания семейства систем автоматизации испытаний изделий ракетно-космической техники необходимо рассматривать как совокупность таких задач, как определение оптимального состава комплекса модулей, получившего название агрегатного, из которых проектируется конкретная система, и методов синтеза конкретной системы из этого набора модулей. Для решения этих задач необходимо предварительно формализовать описание модуля.

Теоретическая часть. Математическая модель модуля должна связывать такие его характеристики, как реализуемые функции, межмодульные интерфейсы, область существования входных и выходных переменных с множеством возможных вариантов каждой функции, множеством состояний объекта, реализующего определенный вариант функции, и критериальными показателями.

Пусть $F=\left\{f_h, h=\overline{1,H}\right\}$ – множество задач обработки, реализуемых семейством САИ. При этом задача f_h считается решаемой семейством систем, если существует преобразование Π^h множества значений вектора $\overrightarrow{X^h}$ входных переменных во множество значений вектора $\overrightarrow{Y^h}$ выходных переменных при условии, что объект, реализующий преобразование Π^h, находится в состоянии, характеризуемом вектором $\overrightarrow{Z^h}$, т.е.

$$\overrightarrow{Y^h}=\Pi^h\left(\overrightarrow{X^h},\overrightarrow{Z^h}\right), \qquad (1)$$

где $\overrightarrow{X^h}=\left(x_1^h,...,x_i^h,...,x_I^h\right)$ – вектор входных переменных задачи f_h;

$\overrightarrow{Y^h}=\left(y_1^h,...,y_j^h,...,y_J^h\right)$ – вектор выходных переменных задачи f_h;

$\overrightarrow{Z^h}=\left(z_1^h,...,z_k^h,...,z_K^h\right)$ – вектор состояния объекта, реализующего преобразование Π^h.

Каждое преобразование Π^h реализуется соответствующим множеством процедур $P^h=\left\{p_\nu,\ \nu=\overline{1,\eta^h}\right\}$, а каждая процедура p_ν преобразует вектор $\overrightarrow{X^\nu}$ при наличии вектора $\overrightarrow{Z^\nu}$ в вектор $\overrightarrow{Y^\nu}$

$$\overrightarrow{Y^\nu}=p_\nu\left(\overrightarrow{X^\nu},\overrightarrow{Z^\nu}\right), \qquad (2)$$

при котором значению входной переменной $\overrightarrow{X^\nu}$ из области ее определения D^ν соответствует определенное значение выходной переменной $\overrightarrow{Y^\nu}$ из области ее изменения E^ν при условии, что значение переменной состояния $\overrightarrow{Z^\nu}$ принадлежит области возможных состояний M^ν и соответствует данной процедуре,

$$D^\nu \overset{p_\nu,M^\nu}{\Longrightarrow} E^\nu. \qquad (3)$$

Область определения D^h, область изменения E^h и область возможных состояний M^h преобразования Π^h определяются следующим образом:

$$D^h=D_1^h\times...\times D_i^h\times...\times D_I^h; \qquad (4)$$

$$E^h=E_1^h\times...\times E_j^h\times...\times E_J^h; \qquad (5)$$

$$M^h=M_1^h\times...\times M_k^h\times...\times M_K^h. \qquad (6)$$

Здесь:

D_i^h – область определения переменной x_i^h вектора $\overrightarrow{X^h}$;

E_j^h – область изменения переменной y_j^h вектора $\overrightarrow{Y^h}$;

M_k^h – область возможных состояний переменной z_k^h вектора $\overrightarrow{Z^h}$;

$i=1,...,I;\ \ j=1,...,J;\ \ k=1,...,K;$

Каждой процедуре $p_\nu\in P^h$ соответствует вектор входных переменных $\overrightarrow{X^\nu}=(x_1^\nu,...,x_i^\nu,...,x_I^\nu)$, вектор выходных переменных $\overrightarrow{Y^\nu}=(y_1^\nu,...,y_j^\nu,...,y_J^\nu)$ и вектор возможных состояний $\overrightarrow{Z^\nu}=\left(z_1^\nu,...,z_k^\nu,...,z_K^\nu\right)$, а

$D^\nu=D_1^\nu\times...\times D_i^\nu\times...\times D_I^\nu$ – область определения вектора входных переменных $\overrightarrow{X^\nu}$,

$E^\nu=E_1^\nu\times...\times E_j^\nu\times...\times E_J^\nu$ – область изменения вектора выходных переменных $\overrightarrow{Y^\nu}$,

$M^\nu=M_1^\nu\times...\times M_k^\nu\times...\times M_K^\nu$ – область возможных состояний вектора состояния объекта $\overrightarrow{Z^\nu}$.

Введем ориентированный мультиграф $\Gamma^h\left(P^h,L^h\right)$ в качестве модели преобразования Π^h. Вершинами $P^h=\left\{p_{\nu s},\ s=\overline{1,\varphi s^h}\right\}$ являются варианты реализации процедур, а дуги – общие для соответствующих процедур переменные, причем пропускная способность этих дуг равна критериальным показателям реализации данной процедуры. Здесь $s=\overline{1,\varphi^\nu}$ – множество индексов различных вариантов процедуры p_ν. Каждая процедура p_ν преобразует множество входных либо промежуточных переменных задачи во множество промежуточных либо выходных переменных той же задачи. Объединение совокупности входных и выходных переменных вариантов реализации процедур $p_{\nu s}$ по множеству P^h, реализуемых при решении задачи f_h, является множеством входных, промежуточных и выходных переменных той же задачи, т.е.

$$L^h=\bigcup_\nu\bigcup_s\left\{\overrightarrow{X_{\nu s}^h}\cup\overrightarrow{Y_{\nu s}^h}\right\}. \qquad (7)$$

Разобьем множество P^h на ряд подмножеств $M_\nu^h,\nu=\overline{1,2^\nu-1}$ таким образом, чтобы каждое подмножество $M_\nu^h=\left\{p_{\nu s}^{h\nu}\right\}$ представляло собой возможное сочетание вариантов реализации процедур $p_{\nu s}$. Множество подмножеств M_ν^h обозначим через $M^h\left(P^h\right)$, элементы которого M_ν^h удовлетворяют условию:

$$\bigcup_v M_v^h = P^h, \quad M_v^h \in M_1\left(P^h\right). \qquad (8)$$

Объединение мультиграфов $\Gamma^h\left(P^h, L^h\right)$ по множеству задач f_h дает мультиграф $\Gamma\left(P, L\right)$ множества задач, реализуемы семейством систем

$$\Gamma\left(P, L\right) = \bigcup_h \Gamma\left(P^h, L^h\right) = \Gamma\left(\bigcup_h P^h, \bigcup_h L^h\right). \qquad (9)$$

Подмножеству $M_1^h\left(P^h\right)$ поставим в соответствие агрегированный граф $T^h = \left(\Gamma_v^h, R^h\right)$. Вершинами графа T^h являются подграфы $\Gamma_v^h = \left(M_v^h, L_v^h\right)$, у которых множество вершин отображает $M_v^h \in M_1\left(P^h\right)$, L_v^h – множество дуг, инцидентных вершинам $\left\{p_{vs}^{hv}\right\}$. Множество дуг R^h графа T^h представляет собой множество переменных, связывающих процедуры подграфов $\Gamma_v^h, \left(v = \overline{1, 2^v - 1}\right)$ между собой

$$R^h = \bigcup_v \left\{ L_v^h \bigcap_{v \neq v^1} L_{v^1}^h \right\}. \qquad (10)$$

Аналогичные действия проведем и в отношении множества процедур P, реализуемых при решении множества задач F семейством систем. В этом случае множество P разбивается на ряд подмножеств M_v, где $M_v = \left\{p_{vs}^v\right\}$ и $M_v = \bigcup_h M_v^h$. Множество подмножеств M_v обозначим через $M\left(P\right)$. Выделим подмножество множества $M_1\left(P\right)$, элементы которого удовлетворяют условию:

$$\bigcup_v M_v = P, \quad M_v \in M_1\left(P\right). \qquad (11)$$

Пусть подмножеству $M_1\left(P\right)$ соответствует также агрегированный граф $T = \left(\Gamma_v, R\right)$, у которого множество вершин представляет множество подграфов $\Gamma_v = \left(M_v, L_v\right)$, каждый из которых имеет множество вершин, соответствующих подмножеству процедур $M_v \in M_1\left(P\right)$ и множеству дуг $L_v = \bigcup_h L_v^y$, инцидентных вершинам $\left\{p_{vs}^v\right\} = \bigcup_h \left\{p_{vs}^{vh}\right\}$, а множество дуг R графа T отображает множество L_v, соответствующее переменным, связывающим процедуры подграфов Γ_v между собой,

$$R = \bigcup_v \left\{ L_v \bigcap_{v \neq v^1} L_{v^1} \right\} = \bigcup_v \left\{ \left(\bigcup_h L_v^h\right) \bigcap_{v \neq v^1} \left(\bigcup_h L_{v^1}^h\right) \right\}. \qquad (12)$$

Подмножество $M_1^h\left(P^h\right)$ и его модель в виде графа T^h реализует конкретное преобразование Π^h вектора $\overrightarrow{X^h}$ в вектор $\overrightarrow{Y^h}$. Подграфы $\Gamma_v^h = \left(M_v^h, L_v^h\right)$ соответствуют преобразованию вектора $\overrightarrow{X^v}$ в вектор $\overrightarrow{Y^v}$ графа обработки данных Γ^h. Подграфы $\Gamma_v^h = \left(M_v^h, L_v^h\right)$ получили название модулей графа $\Gamma^h = \left(M^h, L^h\right)$, соответствующего преобразованию Π^h.

Внешним интерфейсом множества процедур преобразования Π^h назовем множество входных и выходных переменных этого преобразования L^h. Множество дуг R графа T образует межмодульный интерфейс системы модуля графа Γ. Для выражения интерфейса R через входные и выходные переменные введем множество $K_{vv^1} = i\left(v\right) \times j\left(v^1\right)$ упорядоченных пар индексов вход-выход для модулей M_v и M_{v^1} и множество $\overline{K}_{vv^1} = i\left(v\right) \times j\left(v^1\right)$ упорядоченных пар выход-вход для тех же модулей. При этом $\overrightarrow{X^{v^1}}$ и $\overrightarrow{Y^{v^1}}$ – векторы входных и выходных переменных модуля M_{v^1}, а $\overrightarrow{X^v}$ и $\overrightarrow{Y^v}$ – аналогичные векторы переменных модуля M_v.

Пусть i-я входная переменная модуля M_v является j-й выходной переменной M_{v^1}:

$$x_i^v = y_j^{v^1}, \forall\left(i, j\right) \subseteq K_{vv^1} \qquad (13)$$

и/или наоборот:

$$y_j^v = x_i^{v^1}, \forall\left(i, j\right) \subseteq \overline{K_{vv^1}}. \qquad (14)$$

Тогда в соответствии с (11):

$$R = \bigcup_{v < v^1} \left\{ \left\{ x_i^v \bigcap_{v \neq v^1} y_j^{v^1} \right\} \cup \left\{ x_i^{v^1} \bigcap_{v \neq v^1} y_j^v \right\} \right\}. \qquad (15)$$

Интерфейсом модуля M_v является сово-

купность переменных $\left(\overrightarrow{X^v} \cup \overrightarrow{Y^v}\right)$, а множество L_v – внутримодульным интерфейсом.

Свойства модулей:

1. Для любой пары модулей $M_v, M_{v^1} \subseteq M$ модуль M_v эквивалентен модулю M_{v^1} $(M_v = M_{v^1})$, если:

$$\begin{cases} \overrightarrow{X^v} = \overrightarrow{X^{v^1}} \\ \overrightarrow{Y^v} = \overrightarrow{Y^{v^1}} \\ \overrightarrow{Z^v} = \overrightarrow{Z^{v^1}} \\ D^v \equiv D^{v^1} \\ E^v \equiv E^{v^1} \\ M^v \equiv M^{v^1} \\ \overrightarrow{N^v} \equiv \overrightarrow{N^{v^1}} \end{cases} \quad (16)$$

где N^v, N^{v^1} – векторы системных характеристик (параметров) модуля, например, быстродействие, объем памяти и т.д.

2. Модуль M_v включает модуль M_{v^1} $(M_v \subset M_{v^1})$ только тогда, когда

$$\begin{cases} \overrightarrow{X^{v^1}} \subseteq \overrightarrow{X^v} \\ \overrightarrow{Y^{v^1}} \subseteq \overrightarrow{Y^v} \\ \overrightarrow{Z^{v^1}} \subseteq \overrightarrow{Z^v} \\ D_i^{v^1} \subseteq D^v_i, i = \overline{1,I} \\ E_j^{v^1} \subseteq E^v_j, j = \overline{1,J} \\ M_k^{v^1} \subseteq M^v_k, k = \overline{1,K} \end{cases} \quad (17)$$

3. Модуль M_v является вариантом модуля M_{v^1}, если выполняются условия (16) кроме условия $\overrightarrow{N^v} \equiv \overrightarrow{N^{v^1}}$.

4. Модули M_v и M_{v^1} являются информационно совместимыми, если

$$x_i^v = y_j^{v^1} \quad (18)$$

и/или

$$x_i^{v^1} = y_j^v \quad (19)$$

хотя бы для одной пары (i, j).

5. Модули M_v и M_{v^1} являются параметрически совместимыми, если

$$n_i^v = n_j^{v^1} \quad (20)$$

и/или

$$n_i^{v^1} = n_j^v \quad (21)$$

хотя бы для одной пары (i, j), где n_i^v, n_j^v – компоненты вектора $\overrightarrow{N^v}$, а $n_i^{v^1}, n_j^{v^1}$ – компоненты вектора $\overrightarrow{N^{v^1}}$, причем $n_i^v, n_i^{v^1}$ – входные параметры по входу i, а $n_j^v, n_j^{v^1}$ – выходные параметры этих модуле по выходу j.

Таким образом, если множество преобразований $\Pi = \left\{\Pi^h, h = \overline{1,H}\right\}$ обладает свойством модульности, то оно может быть представлено графовой моделью, вершины которой отображают модули обработки, а дуги – информационные связи между ними с пропускными способностями, соответствующими критериальным показателям реализации вариантов процедур, формирующих выходные переменные, причем связи определяются переменными задачи f_h. Эта модель позволит в дальнейшем свести задачу синтеза набора модулей к одной из оптимизационных задач на графах.

Заключение. В статье предложена математическая модель модуля агрегатного комплекса технических и программных средств при создании систем автоматизации испытаний изделий ракетно-космической техники, описаны свойства модуля, что позволяет в дальнейшем свести задачу синтеза набора модулей к одной из оптимизационных задач на графах.

Библиографический список

1. *Бусленко Н.П.* Моделирование сложных систем. – М.:Наука, 1968. – 357 с.

2. *Зарубин В.С.* Математическое моделирование в технике. – М.:МГТУ им. Н.Э. Баумана, 2010. - 496 с.