ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Издается с сентября 2010 г.

Журнал издается при поддержке Отделения математических наук РАН, Отделения нанотехнологий и информационных технологий РАН, МГУ имени М.В. Ломоносова, МГТУ имени Н.Э. Баумана, ОАО "Концерн “Сириус"

СОДЕРЖАНИЕ

Журнал зарегистрирован в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций. Свидетельство о регистрации ПИ № ФС77-38590 от 24 декабря 2009 г.

Журнал распространяется по подписке, которую можно оформить в любом почтовом отделении (индексы: по каталогу агентства "Роспечать" — **22765**, по Объединенному каталогу "Пресса России" — **39795**) или непосредственно в редакции.
Тел.: (499) 269-53-97. Факс: (499) 269-55-10.
Http://novtex.ru/pi.html E-mail: prin@novtex.ru
Журнал включен в систему Российского индекса научного цитирования.
Журнал входит в Перечень научных журналов, в которых по рекомендации ВАК РФ должны быть опубликованы научные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

THEORETICAL AND APPLIED SCIENTIFIC AND TECHNICAL JOURNAL

SOFTWARE ENGINEERING

PROGRAMMNAYA INGENERIA

N. 1
January
2015

Published since September 2010

CONTENTS

Information about the jornal is available online at:
http://novtex.ru/pi.html, e-mail:prin@novtex.ru

УДК 004.4

В. В. Липаев, д-р техн. наук, проф., гл. науч. сотр., e-mail: vlip28@mail.ru,
Институт системного программирования РАН, г. Москва

Человеческий фактор в экономике производства заказных программных продуктов

Рассмотрены и проанализированы основные составляющие человеческого фактора, влияющие на показатели экономической эффективности коллективного производства крупных программных продуктов. Представлены таблицы для их численного оценивания на основе использования модели COCOMO II. Рассмотрено влияние отдельных специалистов, участвующих в создании продукта, а также особенностей организации коллективов разработчиков на экономические характеристики производства программных продуктов.

Ключевые слова: *экономические характеристики, факторы производства, программные продукты, модель COCOMO II*

Введение

Важнейшим фактором, определяющим экономическую эффективность производства и качество созданных заказных программных продуктов, являются ***люди — специалисты*** с различным уровнем их профессиональной квалификации, а также с многообразием характеров, знаний, опыта, наличием стимулов и потребностей [1—4]. Быстрый рост сложности и повышения ***ответственности за качество*** программных продуктов привели к появлению новых требований к квалификации и дифференциации специалистов в области программной инженерии. В современных условиях недостаточно навыков традиционного программирования небольших компонентов на одном или нескольких языках, необходимы глубокие знания в области математики, системотехники, технологии и стандартов проектирования, производства программных продуктов в определенной области их применения. Специалисты должны владеть ***новыми интеллектуальными профессиями***, обеспечивающими гарантии высокого качества заказных программных продуктов путем их верификации, проведения тестовых испытаний, позволяющих оценить реальные показатели качества на каждом этапе разработки и совершенствования комплексов программ. В работе [4], подготовленной на основе модели COCOMO II [3], уже отмечалось большое влияние характеристик коллективов специалистов на экономику производства программных продуктов. Ниже этот фактор рассмотрен более подробно. При этом описываемые составляющие человеческого фактора и их рейтинги не следует рассматривать как рецепты для расчета экономических характеристик. Их нужно учитывать только как ориентиры при планировании разработок заказных программных продуктов.

Человеческий фактор составляют личная профессиональная квалификация, психологические характеристики специалистов и организация работы крупного коллектива. В основном они определяют показатели трудоемкости и качества при производстве программных продуктов. Разработка сложных программных комплексов, особенно на начальных и завершающих этапах, характеризуется наиболее высокой долей творческого труда. Трудоемкость и длительность отдельных операций и частных работ существенно зависят от индивидуальных психологических особенностей их исполнителей, от функционального назначения и характеристик конкретного проекта. При сокращении суммарных затрат на разработку программных компонентов за счет автоматизации нетворческого труда, в большей степени определяющей для производства становится доля затрат на творческий труд. Как следствие, ***возрастают требования к творческим профессиональным способностям конкретных специалистов***.

В процессе разработки программных продуктов ***творческое начало в действиях руководителей проекта и специалистов-исполнителей*** заключается в поиске методов, альтернативных решений и способов реализации требований, заданных заказчиком. Такие требования, как правило, подлежат декомпозиции, дополнительному анализу, уточнению и формальному описанию. Все это сопряжено со значительными человеческими затратами. Индустриальный подход к разработке программных продуктов позволяет автоматизировать многие нетворческие, технические и рутинные операции, этапы, а также облегчает творческие процессы за счет селекции, обработки и отображения информации, необходимой для принятия творческих решений. Следствием этого является возможность значительного сокращения доли затрат на творческую составляющую в общих затратах специ-

алистов-исполнителей работ на проектирование и производство комплексов программ.

Специалистам-исполнителям проектов необходимо хорошо владеть важными ***для программной индустрии методами***, связанными с созданием крупных программных продуктов [1]. К их числу относятся методы:

— организации регламентированного режима работы больших профессиональных коллективов специалистов над целостным программным продуктом;

— распределения сотрудников разной профессиональной специализации по производственным этапам, компонентам и видам работ в жизненном цикле комплексов программ;

— планирования работы коллектива исполнителей в условиях ограниченных ресурсов, по графикам в реальном времени с заданными сроками, контролем качества и документированием результатов;

— тестирования, проведения испытаний и получения гарантии качества, надежности отдельных компонентов и программного продукта в целом.

Полученные профессиональные составляющие человеческого фактора способствуют формированию, совершенствованию и применению ***современной методологии и регламентированной инженерной дисциплины*** обеспечения процессов производства сложных программных продуктов для различных предметных областей. Анализ и учет ***человеческого фактора*** в экономике программной инженерии ***связаны с большими трудностями,*** характерными для новых разделов науки и техники, появляющихся ***на стыке различных сфер деятельности человека и областей знаний***. В данном случае такие трудности обусловлены тем, что ***менеджеры и разработчики*** комплексов программ, как правило, не знают даже основ психологии и экономики, которые важны для организации производства сложной интеллектуальной продукции. В то же время ***психологи и экономисты,*** как правило, не имеют должного представления об особенностях и свойствах объектов разработки — программных продуктов, не могут оценить особенности технологических процессов их производства и применения. Создание программных средств как результатов производственной деятельности существенно повысило ***актуальность обоснования, прогнозирования и оценивания роли человеческого фактора*** для получения адекватных оценок экономической эффективности производственного процесса и качества производимого продукта. Получение таких оценок объективно осложнено трудностью учета психологических характеристик и профессиональной квалификации руководителей и специалистов, участвующих в создании таких объектов. Широкий спектр количественных и качественных показателей, которые с разных сторон характеризуют содержание компонентов и комплексов программ, а также невысокая достоверность оценки их свойств создают для создателей продукта значительную неопределенность при попытке описать и оценить экономические характеристики процесса производства сложных программных продуктов.

Технологии и экономические характеристики регламентированного проектирования и производства крупных программных продуктов большими коллективами специалистов-исполнителей таких проектов ***принципиально отличаются от технологий и характеристик при индивидуальной разработке небольших программ или комплексов программ*** [6—8]:

- руководители больших коллективов специалистов должны быть способны выполнять ***роль лидеров — главных конструкторов***, объединяющих и координирующих знания, навыки и деятельность, направленную на производство программного продукта специалистов с разной профессиональной квалификацией и психологическими характеристиками;
- взаимодействие специалистов, ***творческая и психологическая совместимость*** в коллективе должны обеспечивать планируемое производство целостного программного продукта требуемого качества в заданные сроки;
- при формировании коллектива и выполнении совместных групповых работ необходимо учитывать и рационально использовать ***особенности каждого специалиста*** в его составе, ориентируясь на конечный результат;
- как правило, сложно выделить ***персональное авторство*** отдельных функций и фрагменты, определяющие характеристики, качество, дефекты и риски программного продукта в целом;
- экономические характеристики и качество поставляемого программного продукта зависят от того, насколько качественно выполняет работу ***каждый специалист*** и от его персональной квалификации, однако не всегда можно точно выделить конкретного специалиста, ***ответственного*** за выявленные критические дефекты, ошибки в программном продукте и соответствующие риски неблагоприятных событий.

Накопленный в России опыт создания крупных комплексов программ и острый дефицит необходимых для такой работы специалистов привели к необходимости ***принципиально изменить программы их обучения и воспитания***. Крупномасштабное производство программных продуктов различных классов, разделение труда специалистов по профессиональной квалификации при их разработке, структура и организация больших коллективов, а также ***экономика таких производств*** стали важнейшей частью решения задачи ***выбора, обучения и стимулирования специалистов*** для выполнения работ на отдельных этапах жизненного цикла сложных заказных программных продуктов.

Влияние характеристик специалистов на экономические показатели производства программных продуктов

Структура крупного коллектива разработчиков в значительной степени коррелируется со структурой разрабатываемого ими комплекса программ. Особенно это заметно при создании продуктов размером порядка $10^5...10^7$ строк текста. Для таких продуктов

распределение программ, реализующих наиболее важные функции, локализуется по соответствующим группам специалистов в целях минимизации связей и взаимодействия, как между этими группами, так и между создаваемыми ими компонентами.

Для числовых оценок влияния специалистов на экономические показатели производства ориентиром могут служить данные, приведенные в работе [3] для модели COCOMO II. В модели COCOMO II изложены экспертные оценки для учета влияния квалификации и других характеристик специалистов на экономические показатели производства программных продуктов. Аналогичные оценки особенностей специалистов и их влияния на экономические показатели базируются на отечественном опыте создания крупных программных продуктов [5]. Эти характеристики сводятся в основном к длительности и опыту работы в определенной области, и к экспертной оценке квалификации специалистов. Выбор и использование их значений при прогнозировании экономических характеристик требует высокой квалификации экспертов-руководителей и детального знания особенностей среды производства конкретного реального продукта. Для обеспечения возможности их использования при прогнозировании трудоемкости и длительности разработки сложных комплексов программ в модели COCOMO II представлены соответствующие относительные рейтинги категорий специалистов. При применении этих рейтингов следует учитывать, что некоторые из них взаимосвязаны. В связи с этим целесообразно анализировать их корреляцию, возможность объединения или исключения при оценке экономических характеристик реальных проектов.

Среди большого числа ***характеристик коллектива разработчиков*** наибольшее влияние на трудоемкость могут оказывать тематическая и технологическая квалификация специалистов, которые в табл. 1 и 2 представлены совокупностью из шести факторов (***M9—M14***). Совместно эти факторы могут изменять трудоемкость производства программных продуктов на 30...40 %. Кроме того, в модели COCOMO II выделен и обобщен на основе пяти характеристик коэффициент — ***комплексная коллективность*** участников проекта (см. ниже табл. 3). Влияние этого коэффициента может пятикратно изменять трудоемкость.

Разработчики должны иметь в своем составе квалифицированных, ***проблемно-ориентированных аналитиков и системных архитекторов*** (***M9***), способных переводить функциональные требования

Таблица 1

Характеристики разработчиков программного продукта

Характеристика	Значение характеристики				
	Очень низкая	Низкая	Номинальная	Высокая	Очень высокая
Квалификация аналитиков ***M9***, %	15	35	55	75	90
Квалификация программистов ***M10***, %	15	35	55	75	90
Тематический опыт ***M12***	2 мес	6 мес	1 год	3 года	6 лет
Инструментальный опыт ***M13***	2 мес	6 мес	1 год	3 года	6 лет
Опыт работы с языками ***M14***	2 мес	6 мес	1 год	3 года	6 лет
Стабильность коллектива ***M11***, %	48	24	12	6	3

Таблица 2

Рейтинги характеристик разработчиков программного продукта

Характеристика	Рейтинги характеристик				
	Очень низкий	Низкий	Номинальный	Высокий	Очень высокий
Квалификация аналитиков ***M9***	1,42	1,19	1,00	0,85	0,71
Квалификация программистов ***M10***	1,34	1,15	1,00	0,88	0,76
Тематический опыт ***M12***	1,22	1,10	1,00	0,88	0,81
Инструментальный опыт ***M13***	1,19	1,09	1,00	0,91	0,85
Опыт работы с языками ***M14***	1,20	1,09	1,00	0,91	0,84
Стабильность коллектива ***M11***	1,29	1,12	1,00	0,90	0,81

и общие пожелания заказчика в конкретные, максимально формализованные спецификации и технические требования к комплексу программ и его компонентам. Уровень квалификации аналитиков в COCOMO II предлагается оценивать в процентах от высшей квалификации, что может снизить трудоемкость производства почти на 30 % от номинальной, которой соответствует средний рейтинг 1,00 (также как для всех других факторов).

Трудоемкость творческой части затрат на производство программных продуктов в ***основном определяет человеческий фактор*** — квалификация и организация коллектива специалистов. Как следствие — большой разброс трудоемкости, производительности труда и длительности создания аналогичных программных продуктов разными коллективами. Коллективы с наилучшими экономическими характеристиками могут служить ***ориентирами потенциально*** достижимых в ближайшие годы значений экономических характеристик для соответствующих классов программных продуктов. Однако неуклонно повышается объем и сложность комплексов программ, что приводит к возрастанию затрат творческого труда на условную единицу размера разрабатываемого продукта.

Затраты и труд при реализации крупного продукта в ***экономике производства*** традиционно принято распределять по ***двум категориям специалистов*** — разрабатывающим компоненты и комплекс в целом и обеспечивающим технологию производственного процесса и качество конечного программного продукта. В организационном плане разделение специалистов, осуществляющих ***производство программного продукта*** (первая категория), и ***специалистов-технологов***, контролирующих и управляющих его качеством в процессе производства (вторая категория), должно обеспечивать эффективное достижение заданных характеристик, а также независимый, достоверный контроль экономических характеристик и качества результатов производства.

Специалисты первой категории непосредственно создают компоненты и комплекс программ в целом с заданными показателями качества (см. табл. 1 — ***M10, M13, M14***). В процессе производства их функции заключаются в строгом соблюдении принятой на предприятии технологии производства и в формировании всех предписанных его руководством исходных и отчетных документов. При этом предполагается, что выбранная технология способна обеспечить необходимые значения показателей качества компонентов и комплекса в целом. Достижение заданных функциональных характеристик определяется ***тематической квалификацией*** (***M12***) соответствующих специалистов и регулярным контролем этих характеристик в процессе производства. Система стандартизированного документирования отдельных работ должна обеспечить объективное отражение качества компонентов и процессов их создания на всех этапах жизненного цикла комплекса программ [1, 3].

Разделение труда специалистов этой категории в крупных проектных коллективах необходимо проводить ***по следующим группам, соответствующим квалификации и областям деятельности***:

- спецификаторы и аналитики (***M9***), подготавливающие требования и описания функций соответствующих компонентов с уровнем детализации, достаточным для корректной разработки программистами текстов программных компонентов и их интерфейсов;
- разработчики программных компонентов — программисты (***M10***), создающие компоненты, удовлетворяющие спецификациям требований, реализующие возможности компонентов продукта, отслеживающие и исправляющие ошибки при производстве сложных программных компонентов, что требует детального знания высокоуровневых языков программирования, визуального программирования, сетевых технологий и проектирования баз данных (***M13, M14***);
- системные интеграторы сложных проблемно-ориентированных комплексов программ, работающие над продуктом в значительной степени отличными от программистов методами, на разных языках производства, использующие различные средства автоматизации производства и имеющие на выходе различные результаты — крупные функциональные компоненты и комплексы программ;
- тестировщики, обеспечивающие проверку функциональных спецификаций требований, пользовательских интерфейсов, разрабатывающие стратегию, планы и выполняющие тестирование на соответствие требованиям для выявления и устранения дефектов и ошибок каждого компонента и всего комплекса программ;
- управляющие сопровождением и конфигурацией, отвечающие за снижение затрат на модификацию и сопровождение программного продукта, обеспечение максимальной эффективности взаимодействия компонентов и производство версий программного продукта, принимающие участие в обсуждениях интерфейсов и архитектуры программного продукта;
- документаторы объектов и процессов жизненного цикла комплекса программ, обеспечивающие подготовку и издание сводных технологических и эксплуатационных документов на программный продукт в соответствии с требованиями стандартов [6—8].

Следует однако отметить, что в модели COCOMO II в составе команды специалистов для производства сложного заказного программного продукта ***не учитываются такие важнейшие квалификации***, как тестировщики компонентов и комплексов программ; системные интеграторы крупных компонентов слож-

ных комплексов программ; управляющие сопровождением и конфигурацией комплексов программ.

Успех и качество при разработке сложных программных продуктов во все большей степени зависят от слаженности работы и профессионализма коллектива специалистов на всех этапах и уровнях производства продуктов — ***от стабильности коллектива*** (***M11*** в табл. 1 и 2). Особенно важна не столько индивидуальная характеристика каждого специалиста, сколько ***интегральный показатель квалификации команды***, реализующей некоторую, достаточно крупную задачу, выполняющую те или иные функции или весь проект комплекса программ [3, 8].

Тематическая квалификация и опыт (***M12***) специалистов в конкретной прикладной области, для которой разрабатывается программный продукт, приближенно может оцениваться продолжительностью работы по данной тематике. При низкой тематической квалификации допускаются наиболее грубые системные ошибки, требующие больших затрат для доработки программ. Имеются примеры, когда вследствие таких ошибок, допущенных на этапе системного анализа, приходилось в процессе производства изменять до 70...90 % программ.

Целесообразность использования в качестве показателя квалификации значений ***длительности работы в определенной прикладной области*** или специализации подтверждается достаточно высокой ее корреляцией с коэффициентом изменения трудоемкости. При этом квалификация системных аналитиков и непосредственных разработчиков программ в конкретной прикладной области особенно важна не столько как индивидуальная характеристика каждого специалиста, а прежде всего как интегральный показатель команды, реализующей достаточно крупную задачу. Приводимые в разных работах оценки показывают, что при изменении опыта работы в данной области от 1 года до 10 лет производительность труда может повышаться в 1,5—2 раза, что отражается значением рейтингов ***M12***.

Технологическая квалификация (***M13***) программистов в использовании инструментальной системы автоматизации производства программных компонентов отражает опыт применения методов, средств и всего технологического процесса при создании данного типа программных модулей и компонентов. Этот опыт можно характеризовать длительностью работы с конкретной инструментальной системой автоматизации или ее версиями. Такая система базируется на единых технологических концепциях, на опыте и длительности работы с регламентированными технологиями, инструментальными комплексами автоматизации процесса разработки, с ***языками проектирования, программирования и тестирования программ***. Особое значение имеет коллективный опыт организации и выполнения сложных проектов на базе современных автоматизированных технологий и инструментальных средств программной инженерии. Опыт применения конкретного комплекса автоматизации, языков проектирования и программирования может являться существенным фактором при выборе технологии для создания новых компонентов и обеспечении качества программных продуктов (см. рейтинги в табл. 2).

Квалификация специалистов существенно зависит от ***стабильности состава и психологического климата*** в коллективе, от способности членов коллектива к сотрудничеству и дружной совместной работе над единым проектом (***M11*** в табл. 1 и 2). В данном факторе годы работы с конкретной технологической системой отражают не только опыт работы с инструментальными средствами, но и слаженность коллектива при проведении больших комплексных работ. Нарушение технологии, задержка при разработке отдельных модулей или групп программ могут приводить к большим дополнительным затратам времени и значительной задержке производства продукта в целом. Это вызывает простои групп специалистов, как следствие — увеличение совокупных затрат и снижение производительности труда. Однако чаще всего коллективы специалистов уже имеют некоторый технологический опыт, и дальнейшее повышение их квалификации может дать относительно немного — 10...20 % производительности.

Программистская квалификация специалистов (***M10, M14***) и опыт работы с языками программирования из представленных выше характеристик квалификации относительно слабо отражается на производительности труда при выполнении сложного программного проекта. В данной характеристике учитывается не только уровень состояния языка для непосредственного программирования, но и всех компонентов средств, используемых при создании программ (спецификаций, диалога, тестирования компонентов и их комплексирования). После двухтрех лет работы проявляются индивидуальные особенности конкретных специалистов, их творческие способности, ответственность в работе, рациональный подход к использованию средств автоматизации. При производстве сложных комплексов после первых лет работы возрастание программистской квалификации может повысить общую производительность труда на 5...10 %.

Специалисты программной индустрии, связанные с коллективным созданием заказных программных продуктов, должны владеть [1]:

- эффективными способами организации регламентированной работы больших профессиональных коллективов специалистов над целостным продуктом;
- умением распределять сотрудников разной профессиональной специализации по производственным этапам, компонентам и видам работ в жизненном цикле комплексов программ;

- методами планирования работы в условиях ограниченных ресурсов, по графикам в реальном времени с заданными сроками выполнения работ, контролем качества продукта и документированием результатов деятельности;
- методами и навыками верификации и тестирования, гарантирующими выполнение требований к продукту по выполняемым им функциям, по параметрам качества, надежности отдельных компонентов и программного комплекса в целом.

Совокупность перечисленных видов квалификации можно представить как ***обобщенный человеческий фактор команды при экономическом прогнозировании*** в процессе производства сложных программных продуктов. На этот параметр влияет и большое число других трудно учитываемых условий. Сложность психологических экспериментов с регистрацией всех основных условий производства приводит к большому разбросу результатов. В некоторых случаях эти условия изменяют совокупные затраты даже в 2—5 раз. Как следствие, некоторые специалисты в основном на ***опыте разработки относительно простых небольших программ*** утверждают, что при производстве комплексов программ только человеческий фактор программистов определяет длительность и трудоемкость разработки. Отсюда делается вывод о принципиальном отсутствии возможности планирования и прогнозирования процессов и экономики производства программных продуктов. В действительности при крупных разработках ***человеческий фактор в коллективах усредняется*** и появляется возможность его оценки по тем или иным характеристикам. Тем не менее достаточно часто по этой причине возможен разброс производительности труда в 2—3 раза при производстве крупных продуктов аналогичного назначения [3, 5].

Специалисты второй категории — технологи создают, обслуживают и сопровождают технологический инструментарий, который применяется специалистами первой категории. Они обеспечивают применение системы качества предприятия, контролируют и инспектируют процессы производства для минимизации затрат ресурсов. Основные задачи специалистов второй категории должны быть сосредоточены ***на контроле экономических характеристик*** процесса производства продукта и результатов выполнения запланированных работ, а также на принятии организационных и технологических мер для достижения параметров качества, которые обеспечивают выполнение всех требований технического задания на программный продукт.

Специалисты, управляющие обеспечением качества программных продуктов, должны владеть стандартами и методами, принятыми на предприятии разработчика. Такие стандарты и методы поддерживают регистрацию, контроль, документирование всех манипуляций с продуктом, а также воздействия на показатели его качества и экономические характеристики процесса производства на всех этапах производства. Они должны обеспечивать информацией так называемую ***систему качества проекта***, включающую выявление всех отклонений от заданных показателей качества продукта и эффективности реализации процессов производства, а также от предписанной технологии. В результате функционирования этой системы должны приниматься меры либо по устранению отмеченных отклонений, либо по корректировке ранее сформулированных требований. Последнее целесообразно, если устранение отклонений требует больших затрат.

Влияние особенностей коллективов на экономические характеристики производства программных продуктов

В программных продуктах практически отсутствуют затраты на физические материалы и комплектующие компоненты. Все достоинства и недостатки таких продуктов практически ***полностью определяются интеллектуальным трудом специалистов, их квалификацией***, психологическими характеристиками и умением слаженно и ответственно работать в большом коллективе. В модели COCOMO II значительное внимание уделено ***влиянию организации и взаимодействию внутри коллектива квалифицированных разработчиков (F4)*** на трудоемкость создания сложных программных продуктов. Эта характеристика иллюстрируется составляющими (табл. 3). При этом большое значение может иметь личная мотивация и психологические особенности поведения специалистов при работе над проектированием и производством конкретного сложного программного продукта. Принимая во внимание трудности выявления недостатков при формировании эффективного колллектива, при разработке крупного програмного продукта рекомендуется учитывать такие характеристики человеческого фактора, как:

- согласованность целей коллектива;
- способность членов коллектива разработчиков продукта адаптироваться к другим участникам проекта;
- опыт работы в составе данного коллектива;
- степень доверия друг к другу и взаимодействия в составе коллектива.

Эти характеристики обобщены в показатель ***влияния коллективности — сложности взаимодействия специалистов (F4)*** в коллективе, которому поставлены в соответствие рейтинги изменения трудоемкости производства продукта (последняя строка в табл. 3). Наилучшим считается непрерывное корректное взаимодействие организованной команды специалистов с большим опытом работы в данном коллективе при

Таблица 3

Характеристики и влияние "коллективизма" разработчиков программных продуктов на трудоемкость их производства

"Коллективизм"	Значения характеристик				
	Очень низкий	Низкий	Номинальный	Высокий	Очень высокий
Согласованность целей коллектива	Минимальная	Незначительная	Относительная	Значительная	Полная
Способность членов коллектива адаптироваться к целям других	Малая	Незначительная	Относительная	Значительная	Полная
Опыт работы в составе данного коллектива	Нет	Малый	Незначительный	Значительный	Большой
Степень доверия и взаимодействия в коллективе	Нет	В малой степени	В некоторой степени	Значительная	Большая
Обобщенная коллективность работ	Некоторое взаимодействие в коллективе	Сложное взаимодействие	Зачастую коллективная работа	Высокая степень взаимодействия	Непрерывное взаимодействие
Обобщенный коэффициент влияния коллективности F4	5,48	4,38	3,29	2,19	1,10

полной согласованности их целей, планов и методов работы.

При использовании табл. 3 предлагается оценивать характеристику человеческого фактора в коллективах разработчиков программных продуктов — **коллективностью (*F4*).** При этом рекомендуется применять пять уровней с соответствующими рейтингами от единицы до пятикратного увеличения трудоемкости коллективного создания программного продукта, при отсутствии достаточной организации взаимодействия специалистов в коллективе:

- некоторое взаимодействие в коллективе;
- сложное взаимодействие;
- зачастую коллективная работа;
- высокая степень взаимодействия;
- непрерывное организованное взаимодействие.

При этом наилучшей организации взаимодействия в коллективе соответствует ***F4 = 1.*** Эти оценки учитывают наличие в коллективе всех категорий специалистов, включая руководителей и вспомогательный персонал, которые не прикасаются к программам.

Необходимость и возможность учета человеческого фактора в экономике заказных программных продуктов

Для заказчика и разработчиков при заключении контракта необходимо достаточно достоверное ***прогнозирование требований*** к программному продукту и ***экономическое обоснование необходимых для его производства ресурсов*** по трудоемкости, стоимости, срокам реализации и другим показателям. Заказчик заинтересован в получении программного продукта высокого качества при минимальных затратах. Разработчик при этом желает получить максимальную оплату за созданный продукт и достаточные ресурсы на его производство. Конфликт ***интересов поставщика и потребителя*** при оценке экономических характеристик, стоимости и других ресурсов на выполнение проекта требует ***поиска компромисса***. При таком поиске производитель программного продукта не должен получить ресурсы меньше необходимых, а заказчик не должен выделить ресурсы, больше чем те, что необходимы для выполнения проекта. По этой причине оба партнера по проекту ***заинтересованы в достоверном экономическом прогнозировании*** и обосновании целесообразности и эффективности процессов проектирования и производства программного продукта. Для решения этой задачи должен быть подготовлен согласованный между заказчиком и разработчиком договор, в котором определены ***цели и задачи проекта, требуемые характеристики продукта, доступные экономические и другие ресурсы для его реализации*** [6—8]. Эти данные должны быть предварительно проанализированы, сбалансированы и должны обеспечивать реализацию целей проекта при выделенных ресурсах с минимальным допустимым риском.

Однако масштабы целей и функций сложных программных проектов имеют устойчивую тенденцию изменяться и увеличиваться, первоначально выделенные ресурсы могут не обеспечивать их реализацию. Большую часть рисков и негативных по-

следствий производства можно избежать, используя существующие методы оценивания и прогнозирования производственных затрат, а также технологии управления проектами программных продуктов для их успешного завершения. При планировании размеров, сложности и трудоемкости производства конкретных крупных программных продуктов ***интуитивные оценки*** заказчиков, руководителей и специалистов, как правило, отличаются существенными ошибками. Обычно они ***психологически оптимистичны***, и комплекс программ им кажется меньше по размеру, что приводит к нереальным оценкам количества функции, их содержания, размера и сложности продукта в целом, а также отдельных его компонентов. Следствием этого обстоятельства являются ***большие ошибки при планировании затрат*** времени, качества и стоимости создания крупных программных продуктов. Типичны ситуации, когда отставание сроков внедрения промышленных систем управления и обработки информации полностью ***зависит от неготовности и/или недостаточного качества программных продуктов***, которые должны в их работе использоваться.

При формировании крупных производственных коллективов очень важно уметь ***учитывать психологические особенности и профессиональную компетентность руководителей***. Для этого руководителям и специалистам — участникам проектирования и производства крупных программных продуктов полезно знать и использовать ***основы психологии людей,*** участвующих в конкретном производстве сложных продуктов [1, 2]. Они должны уметь различать психологические характеристики и типы личностей в коллективе специалистов, использовать их общие и профессиональные характеристики. Для ***гарантии надлежащего уровня качества программного продукта и безопасности его применения*** руководителям целесообразно знать и учитывать статистические ***свойства дефектов и ошибок*** в комплексах программ, их типы и источники, факторы, влияющие на их появление и обнаружение, а также возможные негативные последствия [2, 5].

Для обеспечения высокого качества многих заказных комплексов программ необходимы соответствующие ***проблемно-ориентированные системы имитации внешней среды для автоматизации тестирования***. Такие системы способны достаточно полно заменить испытания комплексов программ с реальными физическими объектами и системами. Высокая стоимость и риски неблагоприятных событий при испытаниях с реальными объектами почти всегда при этом оправдывают значительные затраты на такие системы. Это обстоятельство тем более значимо, если предстоят испытания критически важных заказных программ с высокими требованиями к качеству функционирования программных комплексов, с длительным жизненным циклом и множеством развивающихся версий.

Для сокращения неопределенностей и прямых ошибок при оценивании качества заказных комплексов программ необходимо до начала испытаний определять ***основные параметры внешней среды и потоки информации,*** при которых комплекс должен функционировать. При этом необходимо принять во внимание положение требований, которые нужно учитывать при оценивании функциональных возможностей качества продукта. Для этого заказчик и разработчик совместно должны ***структурировать, описать и согласовать модель внешней среды*** и ее параметры как в среднем, типовом режиме применения, так и в наиболее вероятных и критических режимах.

Заключение

Человеческий фактор в программной инженерии представляет уникальное явление, сферу изучения и применения. Он концентрируется в интеллектуальных продуктах и не зависит от поставок сырья, материалов и различных физических средств. Его очень трудно выявлять, оценивать влияние на качество продуктов и характеристики комплексов программ, прогнозировать затраты и удостоверять свойства, которых удалось добиться. Однако производство крупных заказных программных продуктов должно планироваться и им необходимо управлять. Такое управление должно осуществляться с учетом свойств и квалификации больших коллективов специалистов-разработчиков. Представленные выше оценки могут способствовать формированию, совершенствованию и применению на конкретном предприятии ***современной методологии и регламентированной инженерной дисциплины,*** которые призваны поддержать эффективные процессы коллективного производства сложных программных продуктов для различных областей их применения. Представленные выше соображения, доводы и числовые показатели могут ориентировать руководителей больших проектов комплексов программ на необходимость оценивать влияние человеческого фактора и на подходы, которые могут быть для этого использованы.

Список литературы

1. **Липаев В. В.** Человеческие факторы в программной инженерии: рекомендации и требования к профессиональной квалификации специалистов. Учебник. М.: СИНТЕГ, 2009. 328 с.

2. **Боэм Б. У.** Инженерное проектирование программного обеспечения. Пер. с англ. / Под ред. А. А. Красилова. М.: Радио и связь, 1985. 512 с.

3. **Boehm B. W., Abts C., Brown A. W. et al.** Software cost estimation with COCOMO II. New Jersey: Prentice Hall, 2000. 502 p.

4. **Липаев В. В.** Прогнозирование экономических характеристик производства заказных программных продуктов // Программная инженерия. 2012. № 3. С. 2—11.
5. **Липаев В. В., Потапов А. И.** Оценка затрат на разработку программных средств. М.: Финансы и статистика, 1988. 224 с.
6. **Соммервилл И.** Инженерия программного обеспечения. 6-е изд. Пер. с англ. М.: Вильямс, 2002. 624 с.
7. **Фатрелл Р. Т., Шафер Д. Ф., Шафер Л. И.** Управление программными проектами: достижение оптимального качества при минимальных затратах. Пер. с англ. М.: Вильямс, 2003. 1136 с.
8. **Липаев В. В.** Программная инженерия сложных заказных программных продуктов: Учеб. пособие. М.: МАКС Пресс, 2014. 312 с.

V. V. Lipaev, Professor, Senior Scientific Researcher, e-mail: vlip28@mail.ru,
Institute for System Programming of the Russian Academy of Sciences, Moscow

The Human Factor in the Economics of Custom Software Products

The human factor in software engineering is a unique phenomenon, the scope of the study and practical application. He concentrates in intellectual products and does not depend on the supply of materials and a variety of physical means. It is very difficult to identify and evaluate the impact on quality of products, to predict costs and certify properties that have been made, as well as the characteristics of complex programs. However, the production of large custom software must be planned and must be managed. Such management should take into account the characteristics and qualifications of large groups of professionals-developers. To do this in Russia [1, 4, 5] and abroad [2, 3, 7] under investigation and the methodical framework to address the human factor in the production of large complex programs. This publication presents approaches that may contribute to the formation, development and application on a particular enterprise of modern software engineering methodologies. Such approaches and produced by them are intended to support effective evaluation processes of collective production of complex software products for different fields of application. Submitted considerations, arguments and numerical indicators can inform managers of large projects for the development and implementation of complex programs to the need to assess the impact of the human factor. Presents tables for their numerical evaluation based on the use of the model COCOMO II. The influence of individual professionals involved in the creation of software products, as well as the characteristics of the organization development teams on the economic characteristics of their production.

Keywords: *economic characteristics, factors of production, software, model COCOMO II*

References

1. **Lipaev V. V.** *Chelovecheskie faktory v programmnoj inzhenerii: rekomendacii i trebovanija k professional'noj kvalifikacii specialistov*: Uchebnik. M.: SINTEG, 2009. 328 p.
2. **Boehm B. W.** *Software Engineering Economics.* New Jersey: Prentice-Hall, 1981.
3. **Boehm B. W., Abts C., Brown A. W. et al.** *Software cost estimation with COCOMO II.* New Jersey: Prentice Hall, 2000. 502 p.
4. **Lipaev V. V.** Prognozirovanie jekonomicheskih harakteristik proizvodstva zakaznyh programmnyh produktov. *Programmnaya ingeneria*. 2012. N. 3. P. 2—11.
5. **Lipaev V. V., Potapov A. I.** *Ocenka zatrat na razrabotku programmnyh sredstv.* M.: Finansy i statistika, 1988. 224 p.
6. **Sommervill I.** *Inzhenerija programmnogo obespechenija.* 6-e izd. Per. s angl. M.: Vil'jams, 2002. 624 p.
7. **Fatrell R. T., Shafer D. F., Shafer L. I.** *Upravlenie programmnymi proektami: dostizhenie optimal'nogo kachestva pri minimal'nyh zatratah*. Per. s angl. M.: Vil'jams, 2003. 1136 p.
8. **Lipaev V. V**. *Programmnaya inzheneria slozhnyh zakaznyh programmnyh produktov.* Ucheb. posobie. M.: MAKS Press, 2014. 312 p.

УДК 004.414.23+004.415

А. О. Соколов, аспирант, инженер-конструктор, e-mail: wedmeed@mail.ru,
Саратовский государственный технический университет им. Гагарина Ю. А.,
ОАО "КБ Электроприбор", г. Саратов

Оценка влияния апериодического диспетчера в системах реального времени

Рассмотрен предложенный автором метод оценки возможности выполнения прикладных задач в вычислительных комплексах с операционными системами реального времени, использующими апериодический диспетчер. Под возможностью выполнения в настоящей работе подразумевается соблюдение ограничений реального времени для каждой задачи при их совместном выполнении. В рассматриваемых системах диспетчер запускается только тогда, когда произошло требующее обработки событие, а не через постоянные промежутки времени, как в системах с классической организацией. Однако в связи с отсутствием возможности точного определения моментов запуска диспетчера, в данных системах сложно прогнозировать потребление им процессорного времени. По этой причине возникает необходимость закладывать большую избыточность для гарантии работоспособности системы. Предлагаемый метод позволяет точно оценить влияние диспетчера на систему путем анализа параметров прикладных задач. Задача решена с применением экзоядерной архитектуры операционной системы и с использованием модели программного обеспечения из работы [1]. В качестве примера в работе проведен сравнительный анализ эффективности систем с различной организацией диспетчера в условиях реального времени.

Ключевые слова: *диспетчер, тест выполнимости, обработка событий, реальное время, системный таймер, экзоядро*

Введение

В данной работе рассматриваются вычислительные системы реального времени (РВ). Программное обеспечение в таких системах обычно представляется как набор (*set*) задач (*tasks*) [1—6]. Под термином *задача* здесь понимается объект, являющийся совокупностью некоторого программного кода, используемой им памяти и набора данных, описывающих состояние процессора при выполнении этого кода — контекста. Аналогом объекта *задача* в системах общего назначения является *процесс* (*process*) или, в некоторых системах, *поток* (*thread*). Однако в отличие от них, процесс выполнения каждой задачи τ_i представляет собой бесконечную последовательность запросов $\tau_{i,j}$ (*instance*) и имеет регламентированные временные ограничения [1, 2, 4]. Термином *запрос* обозначается процесс выполнения задачи в период между двумя последовательными событиями, являющимися событием запуска (инициации) и событием завершения запроса. Временные ограничения (*ограничения реального времени* или *ограничения РВ*) описываются такими параметрами, как, например, период инициации запросов задачи T_i, относительный крайний срок завершения запросов задачи D_i, максимальное время выполнения каждого запроса C_i. Более полное описание стандартных параметров представлено в работе [1]. Для функционирования системы, состоящей из нескольких задач, требуется операционная система (ОС) РВ, которая обеспечивает соблюдение ограничений РВ для обслуживаемых задач.

Основным компонентом любой ОС является диспетчер — программа, реализующая своевременные переключения между задачами [2, 7, 8]. Диспетчеризация включает в себя выбор задач для выполнения, сохранение и восстановление контекста, перенастройку различных устройств (таймеров, модулей защиты памяти). Диспетчер следит за соблюдением ограничений РВ и может запускать, приостанавливать и завершать выполнение запросов большинства прикладных задач (кроме тех, которые организованы в виде обработчиков прерываний). Он анализирует состояние системы и обрабатывает происходящие в ней события. При этом диспетчер сам проводит ряд вычислений. Следовательно, его влияние на систему заключается не только в непосредственном управлении вычислительным процессом, но и в потреблении части вычислительных ресурсов. В системах РВ этому аспекту уделяется большое внимание, так как для подобных систем характерны:

- низкая производительность используемых аппаратных платформ — при работе в сложных условиях в некоторых сферах в силу большого числа внешних воздействующих факторов или вследствие

ограничений на массо-габаритные показатели применение высокопроизводительной аппаратуры может быть невозможно;

- необходимость точного прогнозирования условий функционирования ПО;
- наличие крайних сроков реакции на события — для каждой задачи определен интервал времени, в который относительно момента своей инициации должен гарантированно завершаться каждый запрос задачи, иначе система не выполнит свои функции.

Большинство работ по планированию в условиях РВ (например, работы [2—6, 9]), опирается на классический подход к организации работы диспетчера. Он заключается в периодическом запуске этой служебной задачи через равные промежутки времени (*tick* — квант системного таймера). При этом границы времени выполнения ее запросов являются предсказуемыми. Этот факт позволяет использовать различные упрощения при оценке соблюдения ограничений РВ (далее для краткости — *выполнимости*) для каждой из прикладных задач. Основным недостатком данного подхода является значительное увеличение среднего времени отклика прикладных задач в силу искусственно вводимого квантования. Как следствие, некоторые наборы задач могут стать невыполнимыми при адаптации под классический подход.

Для многих вычислительных систем более эффективным является подход, основанный на применении системного таймера с динамически изменяемым периодом (*dyntick*). Примеры использования этого подхода представлены в работах [10—12]. Его преимуществом является перераспределение затрат вычислительных ресурсов между прикладными и служебными задачами, вследствие чего уменьшается энергопотребление и увеличивается производительность систем. Недостаток такого подхода — сложность прогнозирования возможного времени отклика на апериодические события. Для их своевременной обработки диспетчер должен инициироваться не только прерываниями от таймера, но и посредством прерываний от всех остальных источников. Такой подход, использованный в работах [13, 14], называется апериодическим (*tickless*) или событийным (*event-driven*). Его недостатком является усложнение процесса определения границ выполнения задач во времени (далее, для краткости, границы выполнения запросов задачи во времени, а также влияние на них различных факторов будем называть *поведением* задачи). Для данного подхода отсутствуют аналитические методы оценки выполнимости задач. Однако без результатов такого анализа использование ПО в системах РВ с высокой нагрузкой или имеющих высокие требования к отказоустойчивости становится проблематичным.

В настоящей работе представлено решение задачи, которое заключается в построении теста выполнимости (*feasibility test*) для систем с апериодическим диспетчером. Тест выполнимости — это процедура, в процессе реализации которой проводится анализ соблюдения ограничений РВ для каждой задачи в наборе при максимальной нагрузке на вычислительную систему. Для решения используется универсальная модель ПО РВ, описанная в работе [1]. Эта модель предусматривает возможность введения новых типов задач $type_i$, которые могут иметь нестандартные (отличающиеся от перечисленных в работе [1]) параметры. Это условие необходимо для описания границ времени выполнения диспетчера и для возможности использования теста выполнимости, описанного в работе [15], в качестве основы для решения поставленной задачи. Механизмы оптимизации теста, описанные в работе [15], остаются применимы и для систем с апериодическим диспетчером.

В разд. "Способы организации работы диспетчера" дан сравнительный анализ используемых подходов, их недостатков и преимуществ. В этом разделе также описаны методы, которые позволяли проводить тест выполнимости в системах с периодическим диспетчером.

Одно из основных затруднений при анализе поведения диспетчера заключается в сильном влиянии на него факторов, возникновение которых сложно прогнозировать (число обращений к служебным программам, разброс максимального и минимального времени выполнения прикладной задачи и т. д.). Для частичного устранения этого затруднения предлагается использовать экзоядерную ОС. При этом следует отметить, что ее применение не всегда приемлемо в некоторых реальных приложениях. В разд. "Сокращение обращений к диспетчеру в экзоядерной ОС" описаны возможные ограничения и способы их преодоления.

Теоретическая часть предлагаемого в работе решения описана в разд. "Интеграция диспетчера в тест выполнимости". Для ее проверки, а также как пример применения, проведен эксперимент, описание и результаты которого представлены в разд. "Практическая оценка влияния диспетчера на систему". Однако проведенный эксперимент интересен не только в качестве примера. Он также показывает эффективность апериодического подхода в системах РВ, использующих экзоядерную ОС, при определенных начальных условиях.

Способы организации работы диспетчера

Классический подход, основанный на периодическом запуске диспетчера, используется в большинстве работ по планированию систем РВ [2—6, 9] и де-факто считается основным. Его преимущество заключается в том, что время работы программ, реализующих функции ядра (диспетчеризацию и прочие служебные функции), можно учесть как неявное дополнение ко времени выполнения прикладных задач. Это заметно облегчает процедуру оценки выполнимости и позволяет легко синтезировать различные алгоритмы планирования.

В рассматриваемом подходе процессорное время разделяется на одинаковые кванты времени, равные

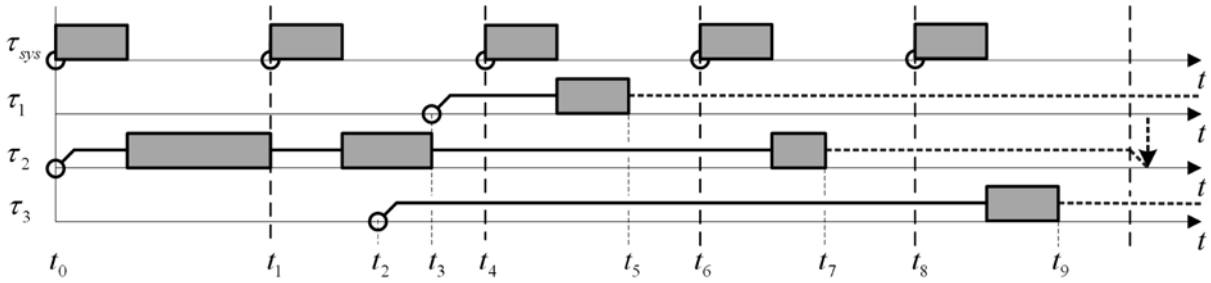

Рис. 1. Учет служебных задач в системе с постоянным периодом системного таймера

периоду системного таймера. В начале каждого кванта времени запускается периодическая служебная задача τ_{sys}, которая и является диспетчером. Эта задача проверяет возникшие в системе события, обрабатывает прерывания, выбирает задачу для выполнения в текущем кванте и передает ее на выполнение процессору.

На рис. 1 представлен пример реализации механизмов планирования при классическом подходе. Моменты времени t_0, t_1, t_4, t_6, t_8 обозначают моменты инициации работы диспетчера. В момент t_2 произошло событие инициации запроса низкоприоритетной задачи τ_3. В момент t_3 задача τ_2 через системный вызов инициирует выполнение задачи τ_1. Моменты времени t_5, t_7 и t_9 соответствуют событиям завершения запросов рассматриваемых задач. Как наглядно видно на рисунке, в периоды времени $t_3...t_4$, $t_5...t_6$ и $t_7...t_8$ процессор не выполняет полезной работы. Это связано с необходимостью ожидания момента смены цикла для выполнения процедуры диспетчеризации.

При использовании периодического подхода необходимо учитывать перечисленные далее обстоятельства.

- Минимальный период системного таймера ограничен максимальным временем выполнения служебной задачи C_{sys} (в любой квант времени должно оставаться время для работы прикладных задач, иначе теоретически возможна ситуация, когда все процессорное время будет занято диспетчером).
- Среднее время реакции задачи τ_i ($1 \leqslant i \leqslant n$, n — общее число прикладных задач) на внешнее событие оказывается соизмеримым с периодом системного таймера T_{sys} (если τ_i инициирована в начале кванта, уже отданного другой, даже низкоуровневой задаче, то τ_i не будет выполняться в этот квант). Как следствие, к максимальному времени отклика (временной интервал между событием инициации и завершения) запросов задачи нужно добавлять T_{sys}:

$$R_i^r = T_{sys} + R_i, \tag{1}$$

где R_i^r — реальное максимальное время отклика запросов задачи τ_i, которое необходимо использовать при определении выполнимости; R_i — расчетное максимальное время отклика запросов задачи τ_i, вычисленное в процессе проведения теста выполнимости, описанного в работе [15].

- Время выполнения служебной программы добавляется к максимальному времени выполнения запросов C_i каждой прикладной задачи τ_i, которое в этом случае определяется по формуле

$$C_i^r = \left\lceil \frac{C_i}{T_{sys} - C_{sys}} \right\rceil C_{sys} + C_i, \tag{2}$$

где C_i^r — расчетное максимальное время выполнения запросов задачи τ_i, которое необходимо использовать в тесте выполнимости при расчетах времени отклика.

- Формула (2) актуальна только в том случае, если в системе нет задач с аппаратной реализацией приоритета (прерываний), кроме периодически запускаемого диспетчера. В других случаях формула (2) сильно усложняется, а ее вычисление становится итерационной процедурой.
- При расчете влияния на низкоприоритетные задачи необходимо округлять расчетное максимальное время выполнения C_i^r вверх до значения, кратного периоду системного таймера момента T_{sys} (если задача завершила работу не в конце системного цикла, другая задача начнет выполнение только в следующем цикле):

$$C_i^l = \left\lceil \frac{C_i^r}{T_{sys}} \right\rceil T_{sys}, \tag{3}$$

где C_i^l — расчетное максимальное время выполнения задачи τ_i, которое необходимо использовать в тесте выполнимости при расчете R_k^r, $1 \leqslant k \leqslant n$, $k \neq i$.

- Каждый системный вызов может быть совершен в начале цикла системного таймера, блокируя выполнение задачи в этот цикл:

$$C_i^r = \left\lceil \frac{C_i}{T_{sys} - C_{sys}} \right\rceil C_{sys} + C_i + aT_{sys}, \tag{4}$$

где a — число системных вызовов (учитываются только те вызовы, для обработки которых выполнение задачи должно быть приостановлено).

- Период задачи T_i должен быть кратен периоду T_{sys}.

Из формул (4) и (2) видно, что при приближении периода системного таймера T_{sys} к максимальному времени выполнения диспетчера C_{sys} параметр C_i^r для любой задачи стремится к бесконечности, а фактор использования (*utilization factor*) процессора служебными задачами $U_{sys} = C_{sys}/T_{sys}$ — к единице.

Таким образом, пользователю необходимо искать компромисс между оптимизацией по времени реакции и по производительности.

Несмотря на все недостатки, основным преимуществом классического подхода являются полная предсказуемость поведения служебной задачи, отсутствие случаев наложения служебных задач друг на друга. Последнее позволяет рассматривать прикладную задачу и обрабатывающий ее диспетчер как единое целое. Это упрощение позволяет использовать тест выполнимости, описанный в работе [15]. При этом единственной модификацией набора задач будет применение условий (1)—(4), что не представляет трудностей. Простота использования описанного подхода обеспечивает широкие возможности для его применения.

Отмеченная ранее модификация классического подхода заключается в изменении порядка инициации диспетчера. При этом он должен запускаться только тогда, когда происходит изменение состояния выполнения задач (инициация, завершение). Зная расположение во времени этих моментов (а это возможно для всех периодических задач), можно каждый раз оптимально настраивать таймер на заданные моменты. При этом если некоторые события могут произойти раньше запланированного времени, то диспетчер должен быть вызван программно (например, при завершении выполнения запроса задачи). Таким образом, время между каждыми двумя последовательными запросами диспетчера динамически изменяется.

Для пояснения можно рассмотреть следующий пример. На рис. 2 представлена та же ситуация, что и на рис. 1, обработка которой теперь ведется в системе с динамическим диспетчером. С момента времени t_0 по момент времени t_1 задача τ_{sys} обрабатывает инициацию запроса задачи τ_2 и настраивает таймер на следующее ближайшее событие, которым является инициация запроса задачи τ_3 в момент t_3. В момент t_2 задача τ_2 через системный вызов инициирует выполнение задачи τ_1. После обработки системного вызова диспетчер запускается повторно, так как наступило событие, на которое был запрограммирован таймер. С момента t_4 таймер настроен на момент t_{11}, который соответствует ближайшему крайнему сроку завершения запроса одной из задач. В моменты времени t_5, t_7 и t_9 выполнение диспетчера инициируется системными вызовами, связанными с завершениями запросов. В момент времени t_6 перестройка таймера не происходит, так как ближайшее прогнозируемое событие по-прежнему произойдет не раньше момента t_{11}. В моменты t_8 и t_{10} таймер настраивается на новый момент времени, располагающийся после момента t_{11} и связанный со следующим ближайшим событием. Как следует из рис. 2, несмотря на увеличение числа вызовов служебной задачи, общее время выполнения набора сократилось по сравнению с тем же показателем на рис. 1.

Реализация данного подхода возможна благодаря наличию в системном таймере функции сигнализатора (*alarm* — тревога, предупреждение). Данная функция предусматривает возможность генерации прерывания при достижении таймером некоторого значения. Регистр, в котором хранится это значение, должен поддерживать перезагрузку без остановки таймера. Например, в микроконтроллере 1986ВЕ1Т в качестве такого устройства может выступать модуль часов реального времени [16]. Детальный обзор решений на основе различных модификаций динамического подхода представлен в работах [10, 11].

Преимущество динамического подхода выражается в том, что оценку влияния диспетчера на выполнение прикладной задачи вместо формулы (2) необходимо проводить по формуле

$$C_i^r = 2C_{sys} + C_i, \tag{5}$$

а при наличии системных вызовов, вместо формулы (4) — по формуле

$$C_i^r = 2C_{sys} + C_i + 2aC_{sys}. \tag{6}$$

Кроме перечисленных формул, в описании динамического подхода отсутствуют ограничения (1) и (3). Это связано с тем обстоятельством, что запрос диспетчера для обработки очередного события может быть вызван сразу после завершения предыдущего. Благодаря этому исчезают отрезки простоя при наличии требующих обработки событий.

При сравнении формул (2) и (5) становится очевидно, что последняя формула дает явное преимущество при увеличении параметра C_i. Результат вычисления C_i^r по формуле (2) оказывается меньше результата, вычисленного по формуле (5), только в том случае, когда выполняется условие $C_i \leqslant T_{sys} - C_{sys}$.

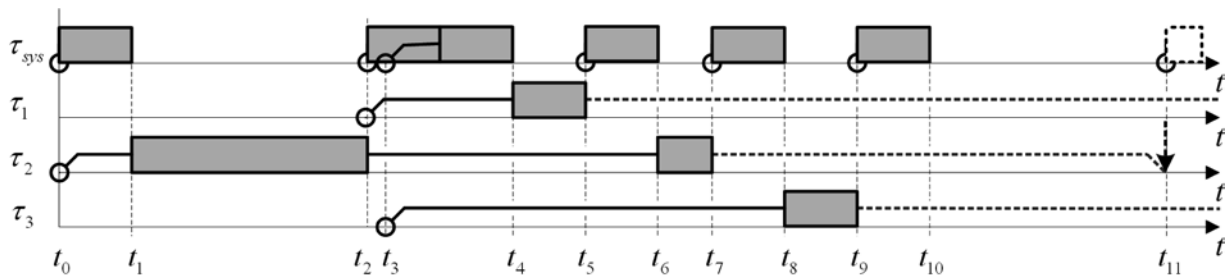

Рис. 2. Распределение служебных задач при использовании динамического и апериодического диспетчера

Существенный недостаток динамического подхода заключается в том, что при наличии в системе апериодических событий, точное время возникновения которых нельзя предсказать, их обработка будет откладываться до ближайшего запуска диспетчера. Это выразится в том, что для регулярных и спорадических задач условие (1) все-таки будет иметь место, однако уже в более "тяжелой" форме:

$$R_i^r = T_{sys\max} + R_i, \quad (7)$$

где $T_{sys\max}$ — максимальный период, на который может быть настроен системный таймер.

Для устранения этого недостатка используется апериодический (событийный) подход. Единственным, но очень значимым отличием его от динамического подхода является возможность инициации диспетчера любым прерыванием в системе. При этом любое апериодическое событие может быть обработано сразу, а условие (7) упраздняется. В системах РВ использование такого механизма дает большой выигрыш. Примеры использования подхода представлены в работах [13, 14].

Задача оценки выполнимости существует как для динамического, так и для апериодического подхода. Ее сложность заключается в отсутствии возможности применить какие-либо упрощения, подобные тем, которые используются в классическом подходе. Диспетчер и управляемые им прикладные задачи имеют разные уровни приоритета. Рассматривать их как единое целое, при условии наложения друг на друга во времени, не представляется возможным.

Большинство работ, использующих данные подходы, не рассматривают условие выполнимости. Например, в работах [10, 11] предполагается наличие больших запасов по времени (эти запасы рассматриваются как потенциал для снижения энергопотребления). В работе [12] рассматривается система общего назначения без ограничений РВ. Во всех этих работах применяется динамический подход.

Примером использования апериодического подхода является, например, ОС Contiki, что подробно рассмотрено в работе [13]. Однако эта ОС без дополнительных надстроек поддерживает только аппаратное вытеснение задач посредством механизма прерываний. Диспетчер в этой системе неотделим от задачи, так как они выполняются с одинаковым уровнем аппаратного приоритета. Оценка выполнимости в этом случае может быть проведена с использованием значительных упрощений. Однако на практике в такой архитектуре может не хватать аппаратных уровней приоритета и каналов прерываний.

В работе [14] рассмотрена архитектура системы, максимально соответствующая апериодическому подходу. Диспетчер для всех задач, инициация которых зависит от времени, запускается таймером с сигнализатором. Для задач, отвечающих за обработку событий, диспетчер инициируется соответствующими прерываниями. По аналогии с работами [10, 11] в работе [14] также предполагается наличие достаточных запасов по времени.

Для решения задачи оценки выполнимости при использовании апериодического подхода в настоящей работе предлагается описать поведение диспетчера при обработке различных прикладных задач. После этого он может быть включен в тест выполнимости. Дополнительным преимуществом такого решения является отсутствие упрощения, используемого в классическом подходе, что представляет возможность включить в тест выполнимости и задачи с аппаратной реализацией приоритета.

Сокращение обращений к диспетчеру в экзоядерной ОС

Основные сложности при описании границ выполнения служебных задач во времени возникают в связи с большим числом непрогнозируемых факторов. Наиболее подвержены их влиянию запросы диспетчера, инициированные системными вызовами. Так как использование системных вызовов полностью зависит от разработчика прикладного ПО, то учитывать их при разработке ОС не представляется возможным. Как следствие, необходимо решение, которое позволит заменить системные вызовы на другие механизмы.

Кроме информирования диспетчера об изменении состояния выполнения прикладной задачи (эти события считаются известными, так как они заложены в характеристики задачи, их описание приведено в работе [1]), системные вызовы служат для запуска специализированных программ. Эти программы осуществляют работу с внешними устройствами или защищенными ресурсами. Они имеют расширенные привилегии и называются драйверами [7, 8]. Именно обращение к драйверам и является "тонким местом" при определении моментов времени, в которые инициируется работа диспетчера.

При использовании различных архитектур ОС драйверы реализуются по-разному. В монолитных (рис. 3) или гибридных архитектурах драйверы чаще всего являются интегрированными [7, 8, 17, 18]. Такие системы обычно являются недостаточно защищенными (код сторонних драйверов имеет привилегии как код ядра ОС). Наибольшей надежностью обладают микроядерные ОС (рис. 3), в которых функции драйвера выделены в отдельные задачи и не являются частью ядра [7, 8, 18]. Платой за максимальный уровень защищенности являются низкая производительность и большое количество накладных расходов вычислительных ресурсов при работе с драйверами.

Минимизация затрат вычислительных ресурсов на обработку системных вызовов достигается применением экзоядерной (рис. 3) архитектуры ОС РВ [19—22]. Идея экзоядерной архитектуры ОС известна уже долгое время, однако она пока не нашла широкого распространения. Эта идея оформилась исключительно в создание дополнительной абстракции между аппаратурой и прикладными ОС [23]. Использование данной архитектуры в низкоуровневых устройствах может быть потенциально выгодно.

Все функции по управлению вычислительным процессом в экзоядре выполняет единственная служебная

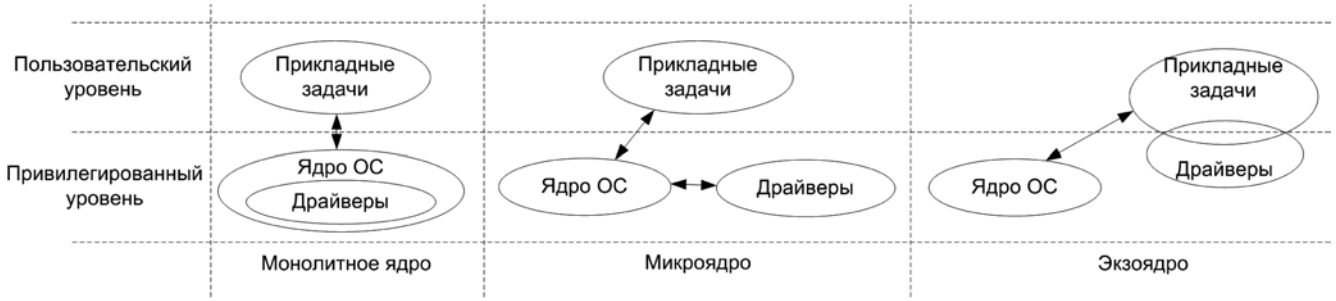

Рис. 3. Работа с драйверами при использовании разных архитектур ОС

задача — диспетчер (планировщик, супервизор). Она обрабатывает только события инициации и завершения запросов всех остальных задач. Эти события являются единственными реализуемыми системными вызовами в экзоядерной ОС. Все остальные функции ОС выделяются в набор библиотек, которые реализуют функции драйверов [19]. Негативным эффектом экзоядерной архитектуры является тот факт, что прикладные задачи должны иметь доступ к привилегированным ресурсам. Такое условие может быть приемлемо во встроенных системах РВ, так как их ПО обычно:

- автономно, а именно — весь набор выполняемых программ строго предопределен, пользователь может влиять только на поток входных данных, а не на процесс вычислений;
- не нуждается в защите от вредоносного ПО, так как его попадание на вычислитель, как правило, невозможно;
- разделяемо, так как каждая задача нуждается в доступе только к определенному и ограниченному набору ресурсов.

Низкая защищенность систем с экзоядерной архитектурой может быть компенсирована некоторыми техническими средствами, включая модули защиты памяти, сторожевые таймеры и т. п.

Преимуществом этой архитектуры является увеличение производительности за счет минимизации числа системных вызовов. Необходимыми остаются только вызовы для осуществления диспетчеризации. Формально перечисленные особенности выражаются в том, что при классическом подходе формула (4) всегда сводится к формуле (2), а при динамическом и апериодическом подходах отсутствует инициация запросов служебных задач при обращении к драйверам.

Интеграция диспетчера в тест выполнимости

При использовании экзоядерной ОС на поведение задачи диспетчера влияют только те факторы, которые обусловлены особенностями прикладных задач. Так как в системах РВ параметры последних описываются достаточно подробно, то процедура описания и прогнозирования поведения служебных задач существенно упрощается.

Предлагается условно провести деление запросов диспетчера τ_{sys} по принципу принадлежности к конкретным прикладным задачам. Для каждой прикладной задачи τ_i, для которой необходима программная диспетчеризация, можно описать две дополнительные служебные подзадачи τ_{asys_i} и τ_{fsys_i}, которые обрабатывают события инициации и завершения запросов этой задачи соответственно. Иллюстрация механизма разделения приведена на рис. 4. Светлым цветом обозначены прикладные задачи, темным цветом — служебные. Физически, после деления диспетчер так и остается единым, но в тесте выполнимости он будет рассматриваться отдельными частями, каждая из которых имеет собственный набор параметров.

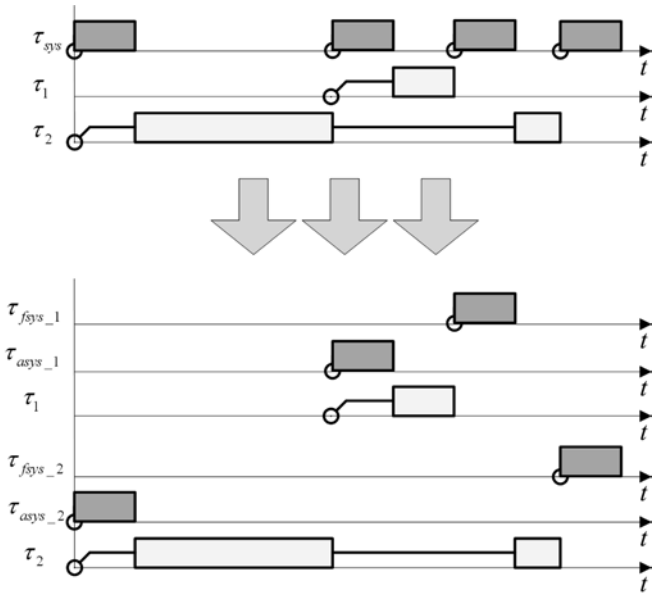

Рис. 4. Условное деление задачи диспетчера

Поведение полученных служебных подзадач будет подчиняться следующим простым условиям:

$$\forall j,\ \begin{cases} a_{asys\_i,j} = a_{i,j} \\ f_{asys\_i,j} \leqslant s_{i,j} \\ a_{fsys\_i,j} = f_{i,j} \end{cases}, \qquad (8)$$

где $a_{x,\,y}$ — момент инициации; $s_{x,\,y}$ — момент начала выполнения; $f_{x,\,y}$ — момент завершения y-го запроса задачи или подзадачи τ_x.

Служебная задача будет включаться в набор задач системы в виде всех пар подзадач τ_{asys_i} и τ_{fsys_i}, параметры которых не должны противоречить условиям (8). Для подзадач τ_{asys_i} эти параметры будут выглядеть следующим образом:

$$\begin{cases} \forall \left(i : type_i \in \{ph, pf\}\right), type_{asys\_i} = ph \\ \forall \left(i : type_i \in \{rh, rf\}\right), type_{asys\_i} = rh \\ \forall \left(i : type_i \in \{sh, sf\}\right), type_{asys\_i} = sh \end{cases}, \quad (9)$$

$$\begin{cases} C_{asys\_i} = C_{sys} \\ \Phi_{asys\_i} = \Phi_i \\ T_{asys\_i} = T_i \\ P_{asys\_i} = P_{sys} \end{cases}, \quad (10)$$

где $type_i \in \{ph, pf, rh, rf, sh, sf\}$ — тип задачи (подзадачи) τ_i; $\Phi_i = a_{i,1}$ — ее фаза; P_i — приоритет.

В настоящей работе обозначения типов соответствуют обозначениям из работ [1, 15]:

ph — периодическая задача жесткого РВ;

pf — периодическая задача строгого РВ;

{*rh*, *rf*} — регулярные задачи жесткого и строгого РВ соответственно;

{*sh*, *sf*} — спорадические задачи жесткого и строгого РВ соответственно.

Для периодических задач период T_i определяет точное время между инициациями их запросов, а для спорадических — минимальное. Для регулярных задач T_i является размером квантов, на которые поделено все время выполнения, а запрос может быть запущен в произвольный момент в пределах каждого кванта. Для задач всех типов выполняется условие $C_i < T_i$. Для задач жесткого РВ дополнительно вводится ограничение на относительный крайний срок завершения запросов задачи D_i, который должен быть меньше или равен периоду T_i. Для задач строгого РВ такое ограничение отсутствует. Запрос задачи строгого РВ может выполняться дольше одного периода. При этом инициации новых запросов игнорируются до завершения текущего запроса, что в результате выражается как $D_i < \infty$.

Следуя условию (9), подзадача τ_{asys_i} наследует тип от соответствующей прикладной задачи τ_i. Однако τ_{asys_i} всегда будет задачей жесткого РВ. Эти обстоятельства связаны с тем, что τ_i и τ_{asys_i} инициируются одними и теми же событиями, с той только разницей, что подзадача диспетчера не может их пропускать. В условии (10) характеристики подзадачи τ_{asys_i} во многом соответствуют характеристикам задачи τ_i по той же причине. Время выполнения и приоритет являются одинаковыми для всех подзадач, так как это фактически одна и та же программа.

Время крайнего срока завершения запросов служебных подзадач D_{sys} не учитывается. Если максимальное время отклика запросов каждой прикладной задачи меньше относительных крайних сроков, что выражается как $R_i < D_i$ и является условием выполнимости, то все запросы диспетчера (которые оказываются автоматически учтены, так как приоритет диспетчера выше) также были завершены вовремя. Пользователя при этом не интересует, как между собой делили время служебные подзадачи. Таким образом, проведение теста выполнимости, из которого исключается подсчет времени отклика диспетчера, становится существенно более простым. Контроль служебной задачи можно осуществлять с помощью доступных аппаратных решений. Для микроконтроллера [16] таким устройством может быть сторожевой таймер.

Для учета подзадачи τ_{asys_i} в тесте выполнимости, описанном в работе [15], достаточно данных, приведенных в формулах (9) и (10). Кроме того, благодаря этой информации подзадача может участвовать в процессах оптимизации взаимного влияния периодических задач при проведении теста выполнимости (но для этого необходимо посчитать максимальное время ее отклика R_{asys_i}), если $type_{asys\_i} = ph$.

Подзадача τ_{fsys_i} может запускаться не чаще, чем задача τ_i, но время ее инициации не может быть точно спрогнозировано. Границы интервала, в котором возможна инициация, сильно различаются в зависимости от типа рассматриваемой прикладной задачи. Поведение подзадачи τ_{fsys_i} в большинстве случаев невозможно описать ни одним из типов задач, определенных в используемой модели. В качестве пояснения может служить приведенный ниже пример.

Пусть имеется периодическая задача строгого РВ τ_i ($type_i = pf$, рис. 5, *а*). Рассматривая ее поведение во времени, можно определить границы возможной инициации запроса подзадачи τ_{fsys_i}. Самая ранняя инициация некоторого запроса τ_{fsys_i} произойдет в том случае, если соответствующий запрос задачи τ_i завершился сразу после запуска (рис. 5, *а*, случай *1*), т. е. все события произошли в один момент:

$$a_{fsys\_i,\,j} = a_{i,\,j}.$$

Самая поздняя инициация будет в том случае, если выполнение запроса задачи τ_i откладывалось и завершилось в крайний срок $d_{i,\,j}$ (рис. 5, *а*, случай *2*):

$$a_{fsys\_i,\,j} = d_{i,\,j}.$$

Так как задача τ_i строгого РВ, то может иметь место условие $d_{i,\,j} > a_{i,\,j+1}$, что приведет к следующему условию:

$$a_{fsys\_i,\,j} > a_{fsys\_i,\,j+1}. \quad (11)$$

Условие (11) показывает, что области возможной инициации запроса служебной подзадачи τ_{fsys_i} (на рис. 5 эти области выделены двусторонней стрелкой над временны́ми осями в каждом из представленных случаев) накладываются друг на друга. Этот факт не может быть описан ни одним из стандартных типов задач. Как следствие, такая подзадача не может быть учтена в тесте выполнимости [15], так как он рассчитан только на стандартные типы.

В другом случае, если прикладная задача τ_i будет регулярной задачей жесткого РВ ($type_i = ph$), подза-

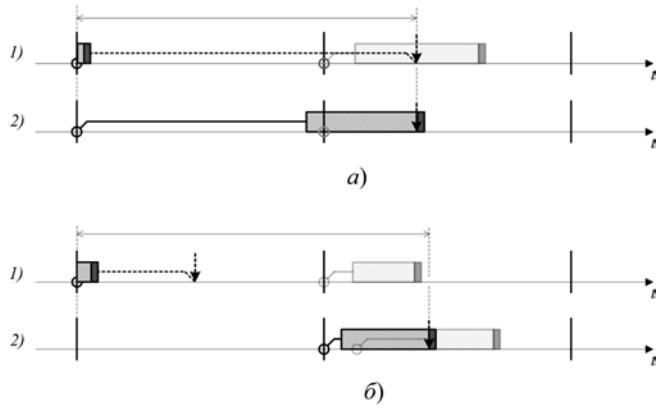

Рис. 5. Границы инициации задачи τ_{fsys_i} (обозначена серым цветом) в зависимости от поведения задачи τ_i (обозначена штриховкой):

а — $type_i = pf$; б — $type_i = rh$

дачу диспетчера τ_{fsys_i} также нельзя описать стандартными типами. Наиболее ранний момент инициации запроса подзадачи τ_{fsys_i} аналогичен случаю в предыдущем примере (рис. 5, *б*, случай *1*). Наиболее поздний — соответствует случаю, когда запрос задачи τ_i был инициирован в конце периода, а его выполнение завершилось в крайний срок (рис. 5, *б*, случай *2*).

Подобный результат получается и в большинстве других случаев. Для учета подзадачи τ_{fsys_i} необходимо определить новые типы задач, описывающие ее параметры в каждом случае. Однако такой подход может привести к сильному усложнению теста выполнимости, а также повысить пессимистичность результатов его выполнения.

Другое решение возможно, если учесть два обстоятельства. Первое, обоснованное при рассмотрении поведения подзадачи τ_{asys_i} — служебная задача может не рассматриваться в тесте выполнимости. Второе — подзадача τ_{fsys_i} может быть инициирована не чаще задачи τ_i. В этом случае служебные подзадачи могут быть описаны небольшим числом обобщенных типов. Обобщенный тип определяет только часть параметров задачи (подзадачи). При этом они не описывают точных интервалов во времени, в которых может произойти инициация запроса подзадач, но полностью характеризуют влияние на другие задачи из набора.

Чтобы рассчитать влияние на другие задачи, необходимо определить плотность распределения запросов подзадачи τ_{fsys_i}. Если в произвольный промежуток времени могут попасть запросы прикладной задачи с $\tau_{i,\,j+1}$ по $\tau_{i,\,j+n}$, то в этот же промежуток может также попасть от $n-1$ до $n+1$ запросов подзадачи τ_{fsys_i}. Из них $\tau_{fsys\_i,\,j+1} - \tau_{fsys\_i,\,j+n}$ являются обработчиками запросов задачи τ_i, полностью входящих в рассматриваемый временной интервал. Запрос $\tau_{fsys\_i,\,j}$ обрабатывает событие завершения запроса $\tau_{i,\,j}$, которое произошло незадолго до рассматриваемого участка времени или в его начале.

На рис. 6 двусторонней стрелкой указан исследуемый промежуток времени (включает максимум

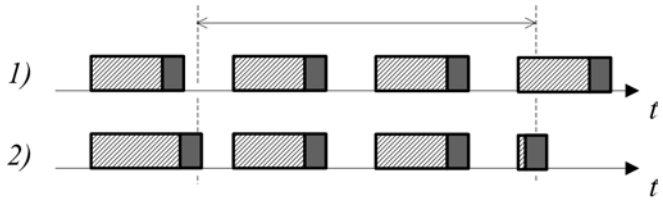

Рис. 6. Определение плотности распределения запросов задачи τ_{fsys_i} (обозначена серым цветом) в зависимости от поведения задачи τ_i (обозначена штриховкой)

три запроса прикладной задачи). При благоприятном стечении обстоятельств (случай *1*) в него попадают только два запроса служебной подзадачи. При наихудшем — четыре (случай *2*). Отсюда следует, что максимальное число запросов подзадачи τ_{fsys_i} за некоторый промежуток времени на единицу больше, чем максимальное число запросов соответствующей прикладной задачи τ_i за этот же промежуток.

К такому же выводу можно прийти и путем обратного рассуждения. Пусть в промежуток времени, включающий запросы $\tau_{i,\,j+1} - \tau_{i,\,j+n}$, попадает $n+2$ запросов подзадачи τ_{fsys_i}. В этом случае последние будут иметь номера либо с $j-1$ по $j+n$, либо с j по $j+n+1$. Запрос $\tau_{fsys\_i,\,j-1}$ не может произойти позже запроса $\tau_{i,\,j}$, при этом $\tau_{i,\,j}$ завершается до рассматриваемого участка. Аналогично, запрос $\tau_{fsys\_i,\,j+n+1}$ не может быть инициирован раньше завершения запроса $\tau_{i,\,j+n+1}$, которое не попадает в исследуемый интервал.

Таким образом, влияние подзадачи τ_{fsys_i} на весь набор задач можно оценить следующими формулами:

$$\forall\left(i : type_i \in \{rh, rf\}\right),\ w_{fsys\_i}(t) = \left\lceil \frac{t}{T_i} + 2 \right\rceil C_{fsys\_i}; \quad (12)$$

$$\forall\left(i : type_i \in \{ph, pf, sh, sf\}\right),$$
$$w_{fsys\_i}(t) = \left\lceil \frac{t}{T_i} + 1 \right\rceil C_{fsys\_i}, \quad (13)$$

где $w_{fsys\_i}(t)$ — время, которое в худшем случае может быть потрачено на выполнение подзадачи τ_{fsys_i} за время t.

В соответствии с формулами (12) и (13) образуются следующие обобщенные типы служебных задач:

- *rt* — обработчик завершения запроса регулярной задачи (*regular task terminator*);
- *spt* — обработчик завершения запроса спорадической или периодической задачи (*sporadic or periodic task terminator*).

Параметры подзадач τ_{fsys_i}, по аналогии со случаем для τ_{asys_i}, могут выглядеть следующим образом:

$$\begin{cases} C_{fsys\_i} = C_{sys} \\ T_{fsys\_i} = T_i \\ P_{fsys\_i} = P_{sys} \end{cases}. \quad (14)$$

С учетом формул (12) и (13), а также при включении подзадач τ_{asys_i} и τ_{fsys_i} в набор задач формула для подсчета времени отклика из работы [15] примет следующий вид:

$$\begin{cases} \forall \left(i : type_i \in \{ph, pf, sh, sf\}\right), \\ R_i = C_i + \sum\limits_{\forall k} \left\lceil \frac{R_i}{T_k} \right\rceil C_k + \sum\limits_{\forall l} \left\lceil \frac{R_i}{T_l} + 1 \right\rceil C_l + \sum\limits_{\forall m} \left\lceil \frac{R_i}{T_m} + 2 \right\rceil C_m \\ \forall \left(i : type_i \in \{rh, rf\}\right), \\ R_i = 2C_i + \sum\limits_{\forall k} \left\lceil \frac{R_i}{T_k} \right\rceil C_k + \sum\limits_{\forall l} \left\lceil \frac{R_i}{T_l} + 1 \right\rceil C_l + \sum\limits_{\forall m} \left\lceil \frac{R_i}{T_m} + 2 \right\rceil C_m \end{cases}, \quad (15)$$

где k: $1 \leqslant k \leqslant n$, $k \neq i$, $P_k \geqslant P_i$, $type_k = ph$, pf, sh, sf; l: $1 \leqslant l \leqslant n$, $l \neq i$, $P_l \geqslant P_i$, $type_l = rh$, rf, spt; m: $1 \leqslant m \leqslant n$, $m \neq i$, $P_m \geqslant P_i$, $type_m = rt$.

В формуле (15) в отличие от формулы из работы [15] также учтено дополнительное влияние регулярных задач на самих себя (предыдущий запрос был инициирован в конце периода, а текущий — в начале). По этой причине необходимо учитывать удвоенное время выполнения при расчете времени отклика.

Формула (15) является основой для модифицированного теста выполнимости, рассчитанного на апериодический подход. Она может быть использована как на практике, для оценки выполнимости реального набора задач, так и для проведения моделирования и исследований подхода. В процессе проведения теста выполнимости могут использоваться все механизмы оптимизации, описанные в работе [15] для соответствующих типов. Теоретически, при оценке влияния на систему подзадач типов *rt* и *spt* тоже могут быть применены эти механизмы. Однако для этого необходимо более точно определить их параметры. Очевидно, в дальнейших работах эти задачи могут быть разделены еще на несколько подтипов.

Практическая оценка влияния диспетчера на систему

Полученные теоретические результаты были проверены путем моделирования. Кроме демонстрации возможности их практического применения, в ходе эксперимента была проанализирована эффективность апериодического подхода в условиях РВ.

Суть эксперимента заключалась в сравнении результатов теста выполнимости при использовании периодического и апериодического подходов. Из результатов похожих экспериментов, описанных в работах [10, 11], следует, что применение динамического подхода не во всех ситуациях позволяет снизить время отклика задач. Очевидно, того же результата следует ждать в системах РВ при использовании событийно-ориентированного диспетчера. Увеличение отклика задач во многих случаях приведет к невыполнимости тестового набора.

По аналогии с экспериментом, описанным в работе [15], в настоящей работе было использовано большое число тестовых наборов задач (20 млн). Параметры задач, при заданных граничных условиях, соответствовали следующим условиям:

$$\begin{cases} type_i \in \{ph, pf, sh, sf, rh, rf\} \\ C_i < T_i \\ D_i \geqslant C_i \\ \forall \left(i : type_i \in \{ph, rh, sh\}\right), D_i \leqslant T_i \\ P_i \leqslant n \end{cases}.$$

Все числовые значения параметров генерировались случайно в пределах заданных граничных условий. При этом для всех параметров, кроме максимального времени выполнения запросов C_i, использовалось равномерное случайное распределение. Характеристики C_i в каждом наборе имели экспоненциальное распределение с параметром $\lambda = n/rand$ в пределах от 1 до T_i, где *rand* — случайная величина с равномерным распределением от 0 до 1, генерируемая для каждого тестового набора задач. Такая генерация позволила добиться равномерного распределения факторов использования $U = \Sigma C_i/T_i$ тестовых наборов задач. Все параметры задач после генерации округлялись до ближайшего натурального числа. Актуальность полученных результатов, в соответствии с методом Монте-Карло, подтверждается большим числом исходных тестовых комбинаций.

Для использования классического подхода необходимо задать некоторое значение периода системного таймера T_{sys}. Очевидно, что результаты теста будут сильно от него зависеть. На практике, чаще всего он мал по сравнению с периодами задач. Для эксперимента были приняты значения $T_{sys} \in \{T_{\min}/50, T_{\min}/20, T_{\min}/10, T_{\min}/5, T_{\min}/2\}$, где $T_{\min}$ — минимальный возможный период задач в эксперименте. Выполнимость каждого набора задач исследовалась для всех перечисленных значений T_{sys}.

Прогнозирование выполнимости при апериодическом подходе требует применения экзоядерной ОС. Однако при ее использовании показатели применимости и быстродействия улучшаются и в классических системах. По этой причине, для объективности эксперимента было принято, что все взаимодействие с аппаратурой и между задачами осуществляется через библиотеки (т. е. в обоих подходах используется экзоядерная ОС).

Так как использование формул (2) и (3) требует наличия только задач с программным приоритетом, то все задачи в тестовых наборах считались управляемыми диспетчером.

С учетом изложенного выше числовые ограничения параметров задач в эксперименте были следующими:

$$\begin{cases} T_{\min} = 100 \leqslant T_i \leqslant 1000 \\ T_i : 100 \\ \Phi_i \leqslant 5T_i \\ \forall \left(i : type_i \in \{pf, rf, sf\}\right), D_i \leqslant 5T_i \\ n = 5 \\ P_{sys} = n + 1 \\ C_{sys} = 1 \end{cases}.$$

Граничные значения были выбраны из тех же соображений, что и в работе [15].

Перед проведением теста выполнимости в обоих случаях выполнялась предварительная обработка набора задач. При использовании периодического диспетчера проводилось увеличение времени выполнения задач с программным приоритетом в соответствии с формулами (2) и (3). Период не округлялся, так как T_{sys} выбран таким, чтобы все возможные T_i были кратны ему. Затем, в соответствии с формулой (1) ко времени отклика этих задач добавлялся T_{sys}.

Для моделирования систем с апериодическим диспетчером в набор задач включались подзадачи τ_{asys_i} и τ_{fsys_i}, параметры которых соответствуют формулам (9), (10) и (14). Расчет времени отклика проводился для всех задач по уравнениям (15). Выполнимость оценивалась только для прикладных задач из исходного набора.

Основное влияние на результаты каждого проведенного теста оказывал исходный набор задач. Основной мерой набора задач, показывающей сложность его планирования, как и в работе [15] был принят фактор использования процессора тестовым набором. Все полученные результаты представлены как зависимость от этого параметра. Это позволяет наглядно сравнивать поведение систем при разных исходных условиях.

Очевидно, что проведение теста имеет смысл только тогда, когда исходный набор выполним. Как следствие, для представления результатов используется не абсолютная оценка, а относительная:

$$f_p(U) = Nm(U)/Ns(U),$$

где $Ns(U)$ — число выполнимых исходных наборов задач с заданным фактором использования; $Nm(U)$ — число наборов задач с заданным фактором использования, выполнимость которых сохраняется после модификации для исследуемого подхода; $f_p(U)$ — их отношение, которое можно характеризовать как *продуктивность* исследуемого подхода.

На рис. 7 сплошными линиями представлены характеристики продуктивности для классического подхода при различных средних значениях периода служебной задачи ($T_{sys} \in \{T_{\min}/20,\ T_{\min}/10,\ T_{\min}/5\}$); пунктирной линией — при $T_{sys} = T_{\min}/2$; штрихпунктирной линией — при $T_{sys} = T_{\min}/50$. Как видно, продуктивность классического подхода сильно различается при разных периодах системного таймера. Сложно однозначно оценить данную зависимость, так как на нее влияют характер запасов по крайним срокам завершения запросов задач, фактор использования каждой конкретной задачи, их отношение между различными задачами в наборе и т.д. В целом, результаты тестирования апериодического подхода (штриховая линия на рис. 7) показывают высокую продуктивность.

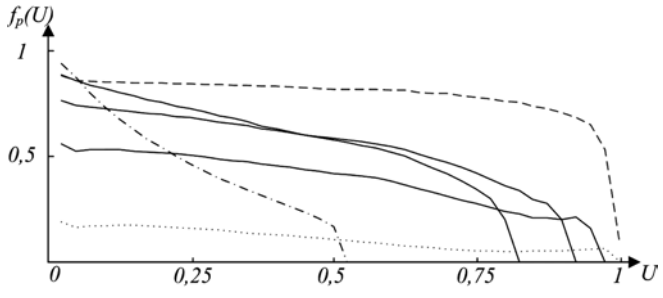

Рис. 7. Продуктивность подходов при $n = 5$

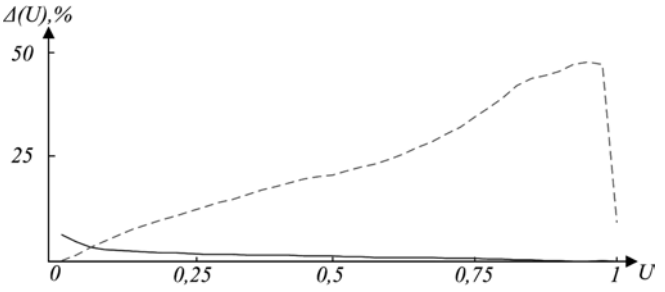

Рис. 8. Сингулярность подходов при $n = 5$ (сплошная линия — периодический подход, штриховая линия — апериодический подход)

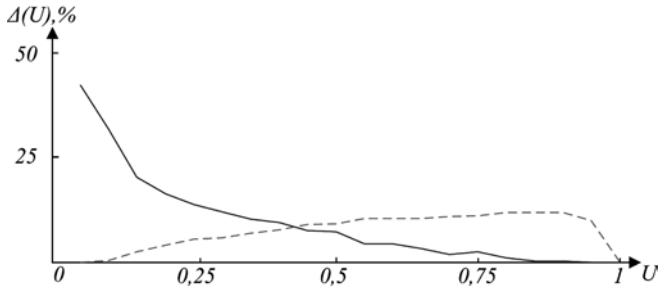

Рис. 9. Сингулярость подходов при $n = 15$ (сплошная линия — периодический подход, штриховая линия — апериодический подход)

Представленные результаты не означают, что апериодический подход всегда лучше. Среди тестовых наборов встречаются такие, которые могут быть выполнены только в системе с периодическим диспетчером (рис. 8, сплошная линия). Обратная ситуация встречается чаще (рис. 8, штриховая линия), но при увеличении фактора использования. Процент таких случаев среди выполнимых исходных наборов для простоты можно назвать сингулярностью подхода и обозначить $\Delta(U)$. Наличие ненулевой сингулярности в обоих подходах не позволяет сделать вывод об однозначном преимуществе одного из них. Решающее значение имеет распределение параметров в исходном наборе задач. Данные результаты сходны с результатами работ [10, 11].

Так как имеет место сингулярность классического подхода, то интересны факторы, определяющие ее. В результате дополнительных экспериментов было установлено, что наибольшее влияние оказывает число задач в исходном наборе. На рис. 9 представлен график сингулярности подходов при $n = 15$. График имеет ломаный вид, вызванный уменьшением числа исходных тестовых наборов до 1,5 млн ради сокра-

щения времени эксперимента (следствие увеличения вычислительной сложности теста выполнимости при увеличении n). Несмотря на это обстоятельство, график отражает общую динамику.

Подобный результат объясняется тем, что увеличение числа задач, даже при сохранении суммарного фактора использования, всегда негативно влияет на сложность наихудшего стечения обстоятельств. Так как при оценке выполнимости при апериодическом подходе число задач увеличивается в троекратном объеме (на каждую добавленную прикладную задачу приходится по две добавочные системные подзадачи), то и увеличение его пессимизма оказывается больше.

Тем не менее, даже при большом числе задач, апериодический подход кажется более применимым в случаях, когда исходный набор задач обладает высоким фактором использования и требует для выполнения бóльшую часть доступного процессорного времени.

Заключение

В данной работе продемонстрировано применение универсальной модели ПО РВ для построения метода оценки возможности выполнения набора задач в системах с апериодическим диспетчером. Для этого введены дополнительные типы задач, с помощью которых возможно описать поведение задачи диспетчера. Выведена формула расчета времени отклика прикладных задач в условиях учета работы диспетчера.

Результаты получены при использовании такого упрощения, как применение экзоядерной ОС. За счет исключения большинства системных вызовов оно позволило избавиться от ряда факторов, влияние которых на поведение диспетчера нельзя спрогнозировать.

Применение результатов работы наиболее вероятно во встроенных системах РВ. Это связано с их высоким уровнем защищенности и с автономным характером построения программного обеспечения. Первое условие делает возможным использование экзоядерной ОС. Второе условие гарантирует неизменность параметров задач на протяжении всего периода эксплуатации. Ярким представителем ВС РВ являются цифровые регуляторы, описанные в работе [1].

В проведенном эксперименте, кроме демонстрации возможности учета динамического диспетчера в тесте выполнимости, проанализирована возможность применения апериодического подхода в системах РВ. Результатом анализа является наличие ненулевой сингулярности периодического и апериодического подходов, что свидетельствует о невозможности полного исключения одного из подходов для практических применений. Апериодический подход дает лучшие результаты при большой загрузке процессора и малом числе исходных задач.

Представленные результаты создают значительный задел для будущих исследований. При этом они могут иметь два направления — изучение нестандартных способов организации диспетчера и расширение области применения алгоритма оценки выполнимости. К первому направлению в ближайшей перспективе можно отнести сравнение эффективности подходов в системах, которые обладают определенной спецификой. Такие исследования могут быть проведены при модификации эксперимента, описанного в настоящей работе, путем изменения используемых ограничений параметров тестовых задач. Интерес представляет также исследование факторов, влияющих на эффективность подходов.

Второе направление касается расширения модели и теста выполнимости путем изучения и введения новых типов задач. Помимо этого, в ходе выполнения теоретической части работы было установлено, что для проведения теста выполнимости не нужно точных данных о поведении задачи (достаточно сведений о ее влиянии на низкоуровневые задачи в наихудшем случае). Данное утверждение создает предпосылки для переработки и более полной систематизации модели.

Список литературы

1. **Соколов А. О.** Модель программного обеспечения систем реального времени // Программная инженерия. 2014. № 7. С. 9—16.
2. **Buttazzo G. C.** Hard real-time computing systems: predictable scheduling algorithms and applications. New York: Springer, 2011. 521 p.
3. **Isovic D., Fohler G.** Handling mixed sets of tasks in combined offline and online scheduled real-time systems // Real-Time systems. 2009. Vol. 43, N. 3. P. 296—325.
4. **Кавалеров М. В., Матушкин Н. Н.** Применение алгоритма получения условия допустимости стандартного ограничения реального времени для примеров линейных интервальных ограничений // Вестник ПНИПУ. Электротехника, информационные технологии, системы управления. 2012. № 6. С. 104—114.
5. **Tindell K. W., Burns A., Wellings A. J.** An extendible approach for analyzing fixed priority hard real-time tasks // Real-Time Systems. 1994. Vol. 6, N. 2. P. 133—151.
6. **Fohler G.** Predictably Flexible Real-Time Scheduling // Advances in Real-Time Systems. Berlin: Springer, 2012. P. 207—221.
7. **Таненбаум Э.** Современные операционные системы. 3-е изд. СПб.: Питер, 2010. 1120 с.
8. **Таненбаум Э., Вудхалл А.** Операционные системы: разработка и реализация. СПб.: Питер, 2007. 704 с.
9. **Wang W., Mok A. K., Fohler G.** Generalized pre-scheduler // Real-Time Systems, 2004. ECRTS 2004. Proceedings. 16th Euromicro Conference on. Catania: IEEE, 2004. P. 127—134.
10. **Gracioli G., Santos D. M., de Matos R. et al.** One-shot time management analysis in EPOS // Chilean Computer Science Society, 2008. SCCC'08. International Conference. IEEE, 2008. P. 92—99.
11. **Fröhlich A. A., Gracioli G., Santos J. F.** Periodic timers revisited: The real-time embedded system perspective // Computers & Electrical Engineering. 2011. Vol. 37, N. 3. P. 365—375.
12. **Siddha S., Pallipadi V., Ven A.** Getting maximum mileage out of tickless // Linux Symposium. Ottawa: Citeseer, 2007. P. 201—207.
13. **Dunkels A., Gronvall B., Voigt T.** Contiki-a lightweight and flexible operating system for tiny networked sensors // Local Computer Networks, 2004. 29th Annual IEEE International Conference on. Tampa: IEEE, 2004. P. 455—462.
14. **Will H., Schleiser K., Schiller J.** A real-time kernel for wireless sensor networks employed in rescue scenarios // Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on. Zurich: IEEE, 2009. P. 834—841.
15. **Соколов А. О.** Оценка выполнимости наборов задач реального времени // Программная инженерия. 2014. № 9. С. 23—29.
16. **32-разрядный** контроллер для авиационного применения 1986BE1T, K1986BE1T, K1986BE1TK, K1986BE1QI,

К1986ВЕ1Н4. Спецификация ТСКЯ.431296.008СП. ЗАО "ПКК Миландр". URL: http://milandr.ru/uploads/Products/product_241/spec_1986BE1.pdf (дата обращения 30.08.2014).
17. **Нестеров П. В., Шаньгин В. Ф., Горбунов В. Л.** и др. Микропроцессоры: в 3-х кн. Кн. 1. Архитектура и проектирование микро-ЭВМ. Организация вычислительных процессов: Учебник для вузов / Под ред. Л. Н. Преснухина. М.: Высшая школа, 1986. 495 с.
18. **Олифер В. Г., Олифер Н. А.** Сетевые операционные системы. СПб.: Питер, 2002. 544 с.
19. **Engler D. R.** The Exokernel operating system architecture: PhD thesis; Место защиты: Massachusetts Institute of Technology. Massachusetts, 1998. 92 p.
20. **Engler D. R., Kaashoek M. F.** Exokernel: An operating system architecture for application-level resource management // ACM. 1995. Vol. 29, N. 5. P. 251—266.
21. **Kaashoek M. F., Engler D. R., Ganger G. R. et al.** Application performance and flexibility on exokernel systems // ACM. 1997. Vol. 31, N. 5. P. 52—65.
22. **Engler D. R., Kaashoek M. F.** Exterminate all operating system abstractions // Proc. of Hot Topics in Operating Systems, 1995. (HotOS-V). Fifth Workshop on. Washington: IEEE, 1995. P. 78—83.
23. **Chakraborty P.** Research purpose operating systems — A wide survey // Computer Science & Telecommunications. 2010. Vol. 26, N. 3. P. 147—160.

A. O. Sokolov, Postgraduate Student, Design Engineer, e-mail: wedmeed@mail.ru, Yuri Gagarin State Technical University of Saratov, "KB Electropribor", Saratov

A Feasibility Test for Event-Driven Systems

In this paper a feasibility test for event-driven systems is presented. These systems use tickless-timer along with interrupts for dispatcher initiations. This approach complicates scheduling but it allows an optimal distribution of processor time. A dispatcher runs only then events that require handling is existing. Feasibility test at such systems is difficult because the dispatcher has a complex behavior.

The proposed solution is intended for real-time systems. Some difficulties are solved using the exokernel architecture. The dispatcher is divided into several virtual subtasks which have parameters is possible to determine. These subtasks are integrated into the extensible software model and the feasibility test.

As an example the comparative analysis of approaches is presented. It shows that the event-driven approach gives better results under a heavy load and at a small number of tasks.

Keywords: *dispatcher, scheduling, feasibility test, real-time, tickless timer, event-driven, exokernel, embedded system, operating system*

References

1. **Sokolov A. O**. Model' programmnogo obespechenija sistem real'nogo vremeni. *Programmnaya ingeneria*. 2014. N. 7. P. 9—16.
2. **Buttazzo G. C.** *Hard real-time computing systems: predictable scheduling algorithms and applications.* New York: Springer, 2011. 521 p.
3. **Isovic D., Fohler G.** Handling mixed sets of tasks in combined offline and online scheduled real-time systems. *Real-Time systems.* 2009. Vol. 43, N. 3. P. 296—325.
4. **Kavalerov M. V., Matushkin N. N.** Primenenie algoritma poluchenija uslovija dopustimosti standartnogo ogranichenija real'nogo vremeni dlja primerov linejnyh interval'nyh ogranichenij. *Vestnik PNIPU. Jelektrotehnika, informacionnye tehnologii, sistemy upravlenija*. 2012. N.6. P. 104—114.
5. **Tindell K. W., Burns A., Wellings A. J**. An extendible approach for analyzing fixed priority hard real-time tasks. *Real-Time Systems.* 1994. Vol. 6, N. 2. P. 133—151.
6. **Fohler G.** Predictably Flexible Real-Time Scheduling. *Advances in Real-Time Systems.* Berlin: Springer, 2012. P. 207—221.
7. **Tanenbaum Je.** *Sovremennye operacionnye sistemy.* 3-e isd. SPb.: Piter, 2010. 1120 p.
8. **Tanenbaum Je., Vudhall A.** *Operacionnye sistemy: razrabotka i realizacija.* SPb.: Piter, 2007. 704 p.
9. **Wang W., Mok A. K., Fohler G.** Generalized pre-scheduler. *Real-Time Systems, ECRTS 2004. Proc. of. 16th Euromicro Conference on. Catania: IEEE*, 2004. P. 127—134.
10. **Gracioli G., Santos D. M., de Matos R. et al.** One-shot time management analysis in EPOS. *Chilean Computer Science Society,* 2008. *SCCC'08. International Conference*. IEEE, 2008. P. 92—99.
11. **Frohlich A. A., Gracioli G., Santos J. F.** Periodic timers revisited: The real-time embedded system perspective. *Computers & Electrical Engineering.* 2011. Vol. 37, N. 3. P. 365—375.
12. **Siddha S., Pallipadi V., Ven A.** Getting maximum mileage out of tickles. *Linux Symposium*. Ottawa: Citeseer, 2007. P. 201—207.
13. **Dunkels A., Gronvall B., Voigt T.** Contiki-a lightweight and flexible operating system for tiny networked sensors. *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*. Tampa: IEEE, 2004. P. 455—462.
14. **Will H., Schleiser K., Schiller J.** A real-time kernel for wireless sensor networks employed in rescue scenarios. *Local Computer Networks, 2009. LCN 2009. IEEE 34th Conference on*. Zurich: IEEE, 2009. P. 834—841.
15. **Sokolov A. O.** Ocenka vypolnimosti naborov zadach real'nogo vremeni. *Programmnaya ingeneria*. 2014. N. 9. P. 23—29.
16. **32-razrjadnyj** kontroller dlja aviacionnogo primenenija 1986VE1T, K1986VE1T, K1986VE1TK, K1986VE1QI, K1986VE1N4. Specification TSKJa.431296.008CP. ZAO "PKK Milandr". URL: http://milandr.ru/uploads/Products/product_241/spec_1986BE1.pdf (date of access 30.08.2014).
17. **Nesterov P. V., Shan'gin V. F., Gorbunov V. L. et al.** *Mikroprocessory: In 3 books.* Book. 1. *Arhitektura i proektirovanie mikro-EVM. Organizacija vychislitel'nyh processov: Uchebnik dlja vuzov.* Eds. by L. N. Presnuhin. M.: Vysshaja shkola, 1986. 495 p.
18. **Olifer V. G., Olifer N. A.** *Setevye operacionnye sistemy.* SPb.: Piter, 2002. 544 p.
19. **Engler D. R.** The Exokernel operating system architecture: PhD thesis; Thesis introducing: Massachusetts Institute of Technology. Massachusetts, 1998. 92 p.
20. **Engler D. R., Kaashoek M. F.** Exokernel: An operating system architecture for application-level resource management. *ACM.* 1995. Vol. 29, N. 5. P. 251—266.
21. **Kaashoek M. F., Engler D. R., Ganger G. R. et al.** Application performance and flexibility on exokernel systems. *ACM.* 1997. Vol. 31, N. 5. P. 52—65.
22. **Engler D. R., Kaashoek M. F.** Exterminate all operating system abstractions. *Proc. of Hot Topics in Operating Systems, 1995. (HotOS-V), Fifth Workshop on.* Washington: IEEE, 1995. P. 78—83.
23. **Chakraborty P.** Research purpose operating systems — A wide survey. *Computer Science & Telecommunications.* 2010. Vol. 26, N. 3. P. 147—160.

УДК 004.415.2

Г. А. Юрьев, канд. физ.-мат. наук, доц., e-mail: g.a.yuryev@gmail.com,
А. С. Панфилова, программист, e-mail: panfilova87@gmail.com,
П. А. Мармалюк, канд. техн. наук, зав. лаб., e-mail: pavel.marmalyuk@gmail.com,
Московский городской психолого-педагогический университет

Архитектура программного обеспечения для анализа результатов окулографических исследований*

Представлена архитектура программного обеспечения, предназначенного для автоматизации расчетов, связанных с анализом данных психологических исследований в области глазодвигательной активности человека. Приведены структура классов этого программного обеспечения и структура базы данных, которая может использоваться для хранения экспериментальных и расчетных результатов. Описаны подходы к созданию графического интерфейса для программного обеспечения, написанного в среде программирования R.

Ключевые слова: *программное обеспечение, видеоокулография, язык программирования R, диаграмма классов, база данных, графический пользовательский интерфейс*

Введение

Одним из направлений деятельности современной экспериментальной психологии является исследование траекторий движения взора испытуемых в рамках различных экспериментальных ситуаций. Движения глаз человека и их направленность позволяют раскрыть структуру взаимоотношений индивида со средой, человека с миром. Тесные связи окуломоторики с центральной нервной системой, с содержанием психических процессов, а также с многообразными формами активности (поведением, деятельностью, общением) позволяют через анализ движений глаз изучать механизмы работы мозга и их нарушения, выявлять динамику состояний, закономерности восприятия, мышления, представлений, прослеживать интенции, намерения и установки личности [1]. В данной статье представлены принятые авторами архитектурные решения ядра системы, предназначенной для анализа окулографических данных, полученных с помощью видеорегистраторов движений глаз. Рассмотрена разработанная диаграмма классов, обеспечивающих экспорт/импорт экспериментальных данных, их хранение, группировку, первичный анализ и визуализацию. Описана структура базы данных для этой системы. Предполагается реализация системы с использованием свободно распространяемой среды программирования для статистических вычислений R [2], что позволит хорошо интегрировать конечный программный продукт в исследовательские процессы профильных лабораторий, использующих среду R для анализа данных. Также обсужден ряд особенностей, отражающих специфику этой среды, с учетом которых проводилось проектирование. Авторами рассмотрен ряд подходов к реализации графических пользовательских интерфейсов для приложений, созданных при помощи среды программирования R.

* Публикация подготовлена в рамках поддержанного РГНФ научного проекта № 14-06-12012.

Постановка задачи

Целью описанного в данной статье этапа разработки программной системы является создание ее архитектуры. Такая система призвана обеспечивать пользователей аппаратно-программного комплекса видеорегистрации движений глаз средствами анализа экспериментальных данных. Традиционно, производители такого оборудования предоставляют и программное обеспечение, дающее, как правило, весьма ограниченные возможности по предварительной обработке и выдаче пользователю данных проведенной записи. Это программное обеспечение обычно является закрытым, а экспортированные данные требуют дополнительного анализа, который сложно провести с помощью стандартных статистических пакетов. Авторы статьи ставят своей задачей разработку открытого программного обеспечения, представляющего перечисленные далее широкие возможности по проведению анализа данных. Важно отметить необходимость реализации графического пользовательского интерфейса для экспертов-психологов, которые могут испытывать затруднения при работе с командной строкой.

Единичный айтрекинговый эксперимент можно обобщенно определить как особого рода взаимодей-

ствие экспериментатора и испытуемого в соответствии с планом (дизайном) эксперимента. Подобные эксперименты проводятся с использованием специализированного аппаратно-программного (айтрекер, средства синхронизации, компьютеры и мониторы, штатное программное обеспечение) и материального (бумажные инструкции, стимульный материал) обеспечения. В результате эксперимента формируются следующие экспериментальные данные: "сырые" и "событийные" данные; данные об испытуемых; стимулы и взаимнооднозначное соответствие айтрекинговых данных и стимулов, использованных в эксперименте.

Итоговая версия системы анализа призвана:

- обеспечивать импорт первичных траекторий взора ("сырых" данных, представленных как временные ряды) и производных показателей движения глаз, рассчитываемых с помощью внешних программных решений, а также использованного в экспериментах графического стимульного материала;
- вносить дополнительную информацию об испытуемых (количественные оценки релевантных для анализа факторов) и выделять на стимульном материале пользовательские зоны интереса;
- визуализировать первичные траектории движений глаз для заданных экспериментальных ситуаций, а также типичные траектории, построенные для групп испытуемых, выявленных в результате кластерного анализа или указанных пользователем;
- рассчитывать на основе первичных траекторий и данных о фиксациях и саккадических движениях глаз традиционные интегральные показатели глазодвигательной активности (число и продолжительность фиксаций, интегральную длину пути, число саккад в различных направлениях, амплитуду, скорость и ускорение саккад), рассчитывать основные статистики показателей по выделенным зонам интереса и по экспериментальным ситуациям в целом;
- рассчитывать матрицы вероятностей перехода первого порядка между заданными зонами интереса или между ячейками регулярной сетки, наложенной на плоскость стимула; рассчитывать матрицы представления преемника (*successor representation matrix*, аналог фундаментальной матрицы цепи Маркова) для последовательности переходов между заданными зонами интереса; строить карты распределения внимания (*attention map*) для заданных экспериментальных ситуаций; выполнять статистический анализ сходства карт распределения внимания;
- задавать гипотетические стратегии посещения зон интереса и рассчитывать степень следования им;
- проводить выборочный иерархический кластерный анализ с использованием в качестве исходного набора признаков для построения мер подобия траекторий такие признаки, как традиционные интегральные показатели, элементы матрицы переходных вероятностей или матрицы представления преемника, показатели степени следования пространственным стратегиям переходов между зонами интереса, дополнительные импортированные показатели, статические карты распределения внимания;
- визуализировать результаты кластерного анализа, способствуя выбору наилучшего кластерного решения (варианта разбиения выборки на подгруппы);
- сохранять в системе результаты расчетов и кластерного анализа в любом из перечисленных представлений для последующего использования в качестве эталонов при проведении классификации вновь зарегистрированных траекторий; экспортировать результаты расчетов;
- проводить классификацию вновь зарегистрированных траекторий по выделенным группам на основе сравнения перечисленных количественных показателей с эталонными распределениями, а также путем расчета мер правдоподобия анализируемых траекторий заданным динамическим вероятностным распределениям.

Для решения перечисленных выше задач имеет смысл разделить процесс создания системы на три отдельных, перечисленных далее, хорошо формализуемых этапа.

1. Реализация ядра системы, обеспечивающего экспорт/импорт экспериментальных данных, их хранение, группировку, структуризацию, поэлементный доступ к хранимым единицам. Система предназначена для решения следующих подзадач:
   - обеспечение экспорта данных распространенных форматов (SMI BeGaze/IDF Converter, Tobii software, Interactive Minds NYAN) и их единообразного представления для последующей обработки;
   - проектирование и реализация программных средств, представляющих возможность группировки данных по ряду критериев (данные одного эксперимента, данные одного испытуемого, данные испытуемых, которые характеризуются единым фактором и др.);
   - создание структуры хранения воспроизводимой схемы эксперимента для упрощения обмена исходными данными и результатами их обработки, а также механизмов, обеспечивающих такое хранение.

2. Реализация функций, обеспечивающих предварительную обработку "сырых" экспериментальных данных (фильтрация, сглаживание, обнаружение окуломоторных событий, расчет интегральных показателей глазодвигательной активности), представляющих собой в случае видеоокулографического исследования набор временных рядов. Этот этап включает реализацию методов анализа загруженных экспериментальных данных, среди которых можно выделить две группы:
   - методы сглаживания и фильтрации выбросов сырых траекторий;
   - методы обнаружения окуломоторных событий (фиксации, саккады, глиссады, моргания, следящие движения алгоритмами I-DT [3], I-VT [4], Adaptive I-DT [5], I-MST, I-KFG, I-VVT, I-VMP и I-VDT [6]), а также расчет ряда интегральных показателей событий.

3. Реализация комплексных методов анализа, применяемых к группам траекторий или окуломо-

торным событиям (кластерный анализ, построение карт распределения внимания, построение матриц вероятностей перехода между заданными зонами интереса или между ячейками регулярной сетки, наложенной на плоскость стимула, и др.), включая создание средств статистического анализа выборочных данных для выявления сходств/различий в окуломоторной активности испытуемых различных экспериментальных групп.

Особенности среды разработки

В данном разделе приведено краткое описание особенностей среды разработки R, которые оказали существенное влияние на принятие архитектурных решений.

В качестве средства разработки авторами была выбрана прикладная среда для статистического анализа данных R. Такой выбор позволил обеспечить высокую степень интеграции разработанного инструментария с решениями, уже существующими в рамках профильных лабораторий, а также обеспечить соответствие выбранного инструментария стандартам в области анализа данных.

Для описания классов в среде R используется нотация языка S4 [7]. По умолчанию, структура такого класса предполагает наличие слотов (свойств) и, по необходимости, значений этих свойств, которые будут установлены при создании экземпляра. Методы при этом описываются за пределами тела класса (функционально-ориентированные методы [8], вместо обычных для объектно-ориентированного программирования класс-ориентированных). Эти методы представляют собой специализированный вид функции, для которой программист может определить список классов, имеющих право ее вызывать. Такой подход в данном случае служит одним из механизмов обеспечения ситуационного (*ad hoc*) полиморфизма, позволяя разным классам иметь методы с одинаковым интерфейсом (или только названиями), но различной реализацией.

Методы классов используемой среды не имеют доступа к свойствам (слотам) объекта из своей реализации, если эти свойства не были переданы как параметр. Объект передается методу как параметр, по значению. При необходимости внесения изменений в свойства объекта после модификации, он возвращается в качестве результата исполнения функции и перезаписывается на место вызвавшего объекта. Другой вариант заключается в возвращении не всего объекта, а значения, согласующегося по типу с одним из его слотов, с последующей его перезаписью. Это ограничение позволяет интерпретатору языка защищать глобальный контекст исполняемой программы от любых изменений, не указанных программистом напрямую. Все методы и свойства объектов являются публичными, ввиду отсутствия модификаторов доступа. Контроль за выделением и освобождением памяти для передачи больших объемов данных по значению осуществляется интерпретатором.

Изложенное выше позволяет констатировать слабую поддержку (на уровне синтаксиса) средой разработки R полноценных механизмов инкапсуляции. Такой вывод следует из отсутствия модификаторов доступа к свойствам и методам класса, от низкой степени разделения интерфейсной и функциональной частей реализации, а также в силу того, что существуют значительные ограничения на передачу параметров по ссылке. Перечисленные особенности среды R сыграли значительную роль в определении структуры классов и интерфейсов их методов, которые описаны в следующем разделе.

Структура классов

Все перечисленные далее диаграммы даны в нотации C#. Базовым классом, отвечающим за представление описания одного окулографического исследования, является класс MetaExperiment, структура которого представлена на рис. 1.

Этот класс представляет описание окулографического эксперимента, который включает его название (Name); описание (Description); автора (Author); условия проведения эксперимента (ExpCond); список стимулов (Stimuli); список областей интереса (AOIL); специализированную таблицу (TAS), устанавливающую связи между стимулами, зонами интереса и, возможно, иными переменными, зависящими от контекста эксперимента. Класс MetaExperiment обеспечивает решение подзадачи ядра системы, связанной с хранением метаданных структуры эксперимента.

Подробнее следует остановиться на понятии *условий эксперимента*. Как легко заметить, условия (как и многие другие параметры) представлены списком, структура, содержание и способ адресации в котором

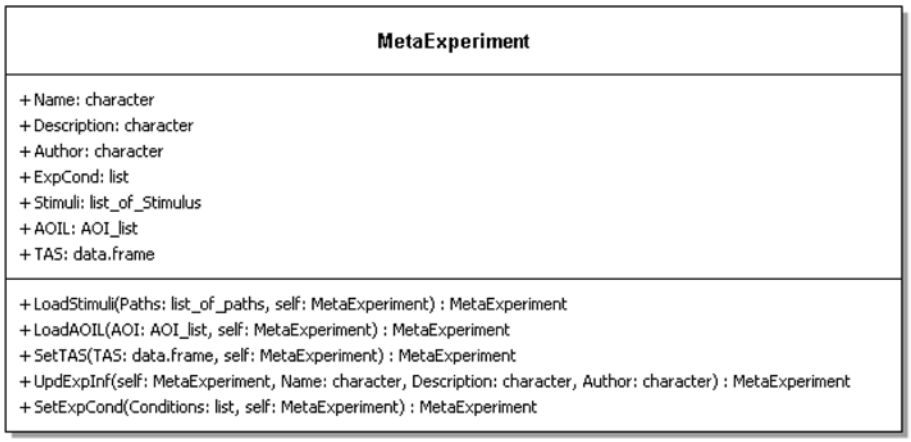

Рис. 1. Структура базового класса MetaExperiment

могут существенно меняться. Такое представление обеспечивает гибкость программного проекта, позволяя существенно модифицировать интерфейс функций без значительного изменения реализации. На настоящее время под условиями эксперимента понимают следующие параметры: частоту дискретизации записи; удаленность испытуемого от стимульного монитора; размеры и разрешение экрана; высоту положения глаз испытуемого относительно нормали к центру экрана; особенности освещения и использованного оборудования. Указанные параметры нужны для корректной предварительной обработки. При необходимости модифицировать или дополнить метод (функцию), принимающий этот список в качестве входного аргумента, список пополняется новым значением, а в реализации метода делается необходимое дополнение.

Под *списком стимулов* понимается набор объектов, составляющих стимульный материал. В наиболее типичном случае это набор используемых в процессе эксперимента изображений, включая кадры стимульного видеоряда, задачи теста или другие контролируемые экспериментальные факторы.

Зачастую при проведении окулографических исследований, кроме временного ряда координат взора испытуемого, экспериментатора интересует частота попадания взора в некоторую область на изображении. Например, частота, с которой испытуемый в процессе эксперимента возвращался к исследованию области носа на продемонстрированном в качестве стимула портрете. Может представлять интерес и динамика движения взора в данной конкретной области или ее частях. Сама область может задаваться достаточно сложным образом: от прямоугольника (в простейшем случае), до некоторой математической функции, определяющей границы *области интереса*. Возможны комбинации различных типов зон в рамках одного стимула. Эксперимент может подразумевать изменение этих зон в рамках одного стимула при возникновении какого-либо дополнительного условия. Таким условием может быть, например, восприятие фрустрирующего звукового стимула. В части постановки задачи, определение областей интереса на этапе проектирования ядра системы служит обязательным условием для последующей реализации, например, функций построения матриц вероятностей переходов между зонами стимула, представляющих исследовательский интерес.

Понятие зон интереса отражается в отношениях между классами, представленными на рис. 2.

В данном случае AOI это класс, представляющий область интереса произвольного типа. Содержание его свойств меняется в зависимости от типа и формы необходимой зоны, которые определяются свойством Characteristics, являющимся списком. Например, в случае прямоугольной формы области, он может хранить координаты верхнего левого и нижнего правого углов, а в случае формы круга — содержать его радиус и координаты центра.

Свойства *типа* и *формы* (Type и Shape) используются для определения логики обработки класса AOI. Класс AOI_set это комбинация различных зон в рамках одного кадра, фактически список объектов класса AOI и методы работы с ним. Последний класс AOI_list представляют собой список всех комбинаций зон, использованных в эксперименте, т. е. список объектов класса AOI_set.

Для установления *связей между зонами интереса, стимулами и экспериментальными ситуациями* (дополнительными внешними условиями) предусмотрено свойство TAS. Оно представляет собой таблицу, устанавливающую для каждой экспериментальной ситуации соответствие стимула, набора зон интереса и типа внешнего условия, определяющего экспериментальную ситуацию.

На рис. 3, см. третью сторону обложки, представлены различные экспериментальные ситуации,

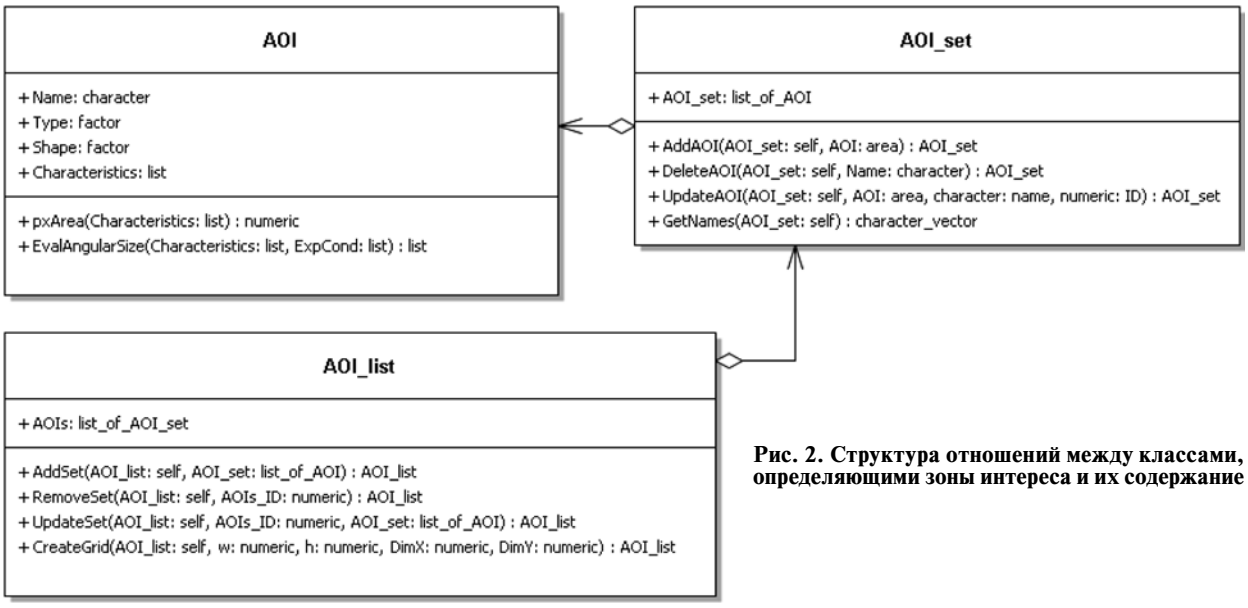

Рис. 2. Структура отношений между классами, определяющими зоны интереса и их содержание

Таблица 1

Пример таблицы, используемой для связи экспериментальных ситуаций, стимулов и зон интереса

TrialID	AOI_setID	StimulID	TrialDescription
1	2	4	Предъявляется изображение, транслируется звук моря, установлен набор зон интереса AOI_set (ID=2)
2	2	4	Предъявляется изображение, слышен звук колокола
3	3	4	Предъявляется изображение, слышен звук моря, меняется набор зон интереса
...	...	...	...

в которых используются одни и те же изображения в качестве стимульного материала, но меняются условия эксперимента. Первая экспериментальная ситуация: предъявление изображения трех человек (Simul (ID = 4)), при этом испытуемый слышит звуки моря на фоне. Экспериментатора интересует динамика перемещения взора по областям (AOI_set (ID = 2)). Во второй экспериментальной ситуации изображение и области интереса остаются такими же, но меняется воспроизводимый звук. В третьей ситуации меняется набор областей интереса (AOI_set (ID = 3)).

Первые две ситуации подразумевают идентичные наборы зон интереса и отличаются только определенным внешним воздействием (звуком). В третьей ситуации звук идентичен использованному в первой, а изменяется набор зон интереса. Для всех трех ситуаций используется один и тот же стимул. Фрагмент таблицы TAS (являющейся слотом класса MetaExperiment) для данного случая будет соответствовать табл. 1.

От рассмотренного базового класса MetaExperiment наследуется класс ExperimentData (рис. 4), который расширяется дополнительными свойствами, хранящими выборочные данные. Предполагается, что это данные эксперимента, проведенного в соответствии с логикой, описанной в родительском классе. Фактически, он обеспечивает хранение, загрузку, модификацию и выборку сырых данных, т. е. служит для решения первой и второй подзадач, возложенных на проектируемое ядро.

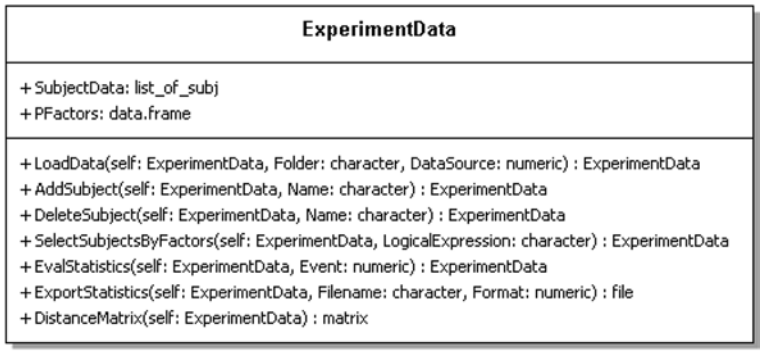

Рис. 4. Структура класса MetaExperiment

Дополнительные слоты — это список испытуемых (и собранных для них данных), а также набор соответствующих этим субъектам факторов, которые могут представлять произвольную, выбранную исследователем характеристику.

Данные участников эксперимента представляются списком объектов класса Subject, изображенным на рис. 5.

Свойства класса Subject определяют идентификационные данные участника исследования и список траекторий его взора, полученных в ходе исследования. Под траекторией в данном случае понимается не только сам временной ряд, но и дополнительные данные, связанные с ним.

Сам временной ряд хранится в форме объекта класса Trajectory, имеющего набор методов для выделения его характеристик. Данные исследования могут быть записаны с использованием различного оборудования, в частности, монокулярного (для фиксации траектории движения взора одного глаза) или бинокулярного (для фиксации данных по обоим глазам испытуемого). Эта ситуация отражена через объявление двух классов для этих случаев (Monocular и Binocular соответственно). Использование именно двух различных, независимых классов, вместо логичного в данном случае наследования одного от другого, обусловлено значительными различиями в логике обработки таких данных. При этом оба класса содержат близкий набор свойств. К их числу относятся: номер экспериментальной ситуации; одна или две траектории (в зависимости от использованного способа регистрации); условия проведения эксперимента (ExpCond), которые могут быть либо взяты из общего описания условий, находящегося в рассмотренном ранее слоте класса MetaExperiment, либо переопределены вручную для конкретной траектории.

Кроме временного ряда, траектория содержит свойство *список событий* (EventList), предназначенное для хранения информации о найденных по траектории окуломоторных событиях (артефакты записи, саккады, глиссады, фиксации, моргания).

Данное описание отражает взгляд авторов на декомпозицию задачи хранения и обработки данных окулографических исследований и последующее ее решение в рамках среды разработки R.

Структура базы данных

Среда R позволяет использовать различные СУБД для работы с данными: PostgreSQL, MS Access, MS SQL Server, SQLite, MySQL. Рассмотрим далее основные библиотеки, обеспечивающие связь среды разработки программ R с базой данных.

Существующие решения для работы с базой данных. Модуль PL/R позволяет внедрять функции на языке R в СУБД PostgreSQL, позволяющие задавать определенные правила для добавления, извлечения и удаления записей [9]. Использование MySQL возможно через пакет RMySQL [10], позволяющий соз-

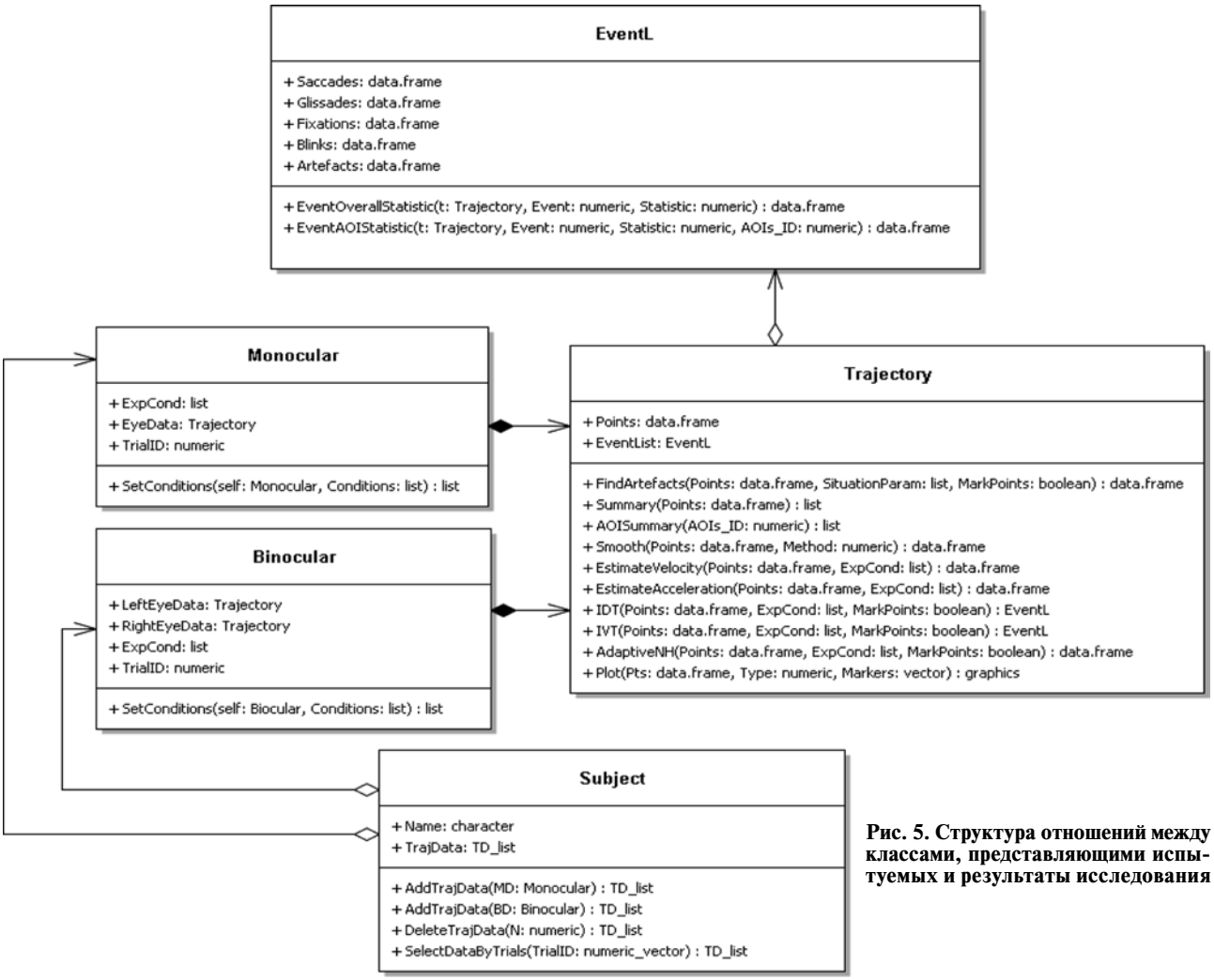

Рис. 5. Структура отношений между классами, представляющими испытуемых и результаты исследования

давать соединение с базой данных, составлять запросы на языке SQL, получать результаты выборки. Компоненты RSQLite и sqldf [11] работают с базой данных SQLite. При этом RSQLite отличает простота в использовании. При установке соединения с базой данных необходимо использовать команду

db <- *dbConnect*(SQLite(), *dbname* = "Test.sqlite"),

в которой требуется указать имя файла *.sqlite. Пакет RSQLite имеет удобный интерфейс для импорта данных из файлов формата .csv или .xls в базу данных. С помощью команды

dbWriteTable(*conn* = db, *name* = "Class",
value = "class.csv", *row.names* = FALSE,
header = TRUE)

представляется возможность записать данные из файла (параметр *value*) в определенную таблицу базы данных (параметр *name*). Более универсальным является модуль RODBC [12], который позволяет использовать драйвер ODBC для подключения к различным базам данных с использованием команды

channel <- *odbcConnect*("test",
uid = "ripley", *pwd* = «secret»),

где первый параметр функции *odbcConnect*() обозначает кодовое название драйвера, используемого для подключения, параметры *uid* и *pwd* содержат информацию о логине и пароле, заданным для данной БД. Пакет поддерживает драйверы, представленные в табл. 2.

RODBC позволяет проводить все необходимые операции с данными, имеет множество настроек и функций для удобной работы с результатом запроса.

Структура базы данных. Разработка структуры базы данных предполагает реализацию возможности хранения записей глазодвигательной активности для воспроизведения эксперимента и дальнейшего анализа. Основными сущностями разработанной структуры являются MetaExperiment и Subject, связанные между собой через таблицу отношений, позволяющую в дальнейшем поставить в соответствие эксперименты и испытуемых, принимавших в них участие (рис. 6).

Сущность MetaExperiment хранит описание окулографического эксперимента и содержит его название (name), описание (description), автора (author). Экспе-

Таблица 2

Драйверы пакета RODBC

Название параметра	Драйвер
test	MySQL ODBC 3.51 Driver
sqlite3	SQLite3 ODBC Driver
testpg	PostgreSQL ANSI
testacc	Microsoft Access Driver (*.mdb)
SQLServer	SQL Native Client
testpgw	PostgreSQL Unicode

римент связан с условиями его проведения через таблицу TRS, устанавливающую связь между стимулами (Stimulus) и списками зон интереса (AOI_list) (рис. 7).

Подробное рассмотрение понятия областей интереса представлено в разд. "Структура классов". Разработанная схема предполагает реализацию программных механизмов хранения данных эксперимента с описанием используемых стимулов, соответствующих им зон интереса, которые представляются с помощью различных объектов. Целью является возможность дальнейшего воспроизведения эксперимента, а также соотнесения параметров глазодвигательной активности с конкретной экспериментальной ситуацией.

Для обеспечения возможности извлечения данных об условиях проведения эксперимента для конкретного испытуемого используются сущности MetaExp_Person_ExpCondition и PFactors (рис. 8).

Зависимая сущность PFactors содержит номер фактора (id_factors) из таблицы Factors и его значение (factorValue). В качестве факторов может выступать результат отнесения испытуемого к какому-либо классу в рамках проведенного эксперимента, а также другие характеристики испытуемого, которые учитываются в данном эксперименте. Зависимая сущность MetaExp_Person_ExpCondition обеспечивает соотнесение с определенным испытуемым параметров проведения заданного эксперимента (ExperimentalConditions): удаленность испытуемого от экрана (distance); частота дискретизации айтрекера (freq); другие простые атрибуты (another).

Как отмечалось ранее, проведение окулографического эксперимента возможно с использованием двух вариантов записи глазодвигательной активности — монокулярной и бинокулярной. Для каждого из них обозначены таблицы Monocular и Binocular, которые ставятся в соответствие определенному испытуемому через сущность TrajData.

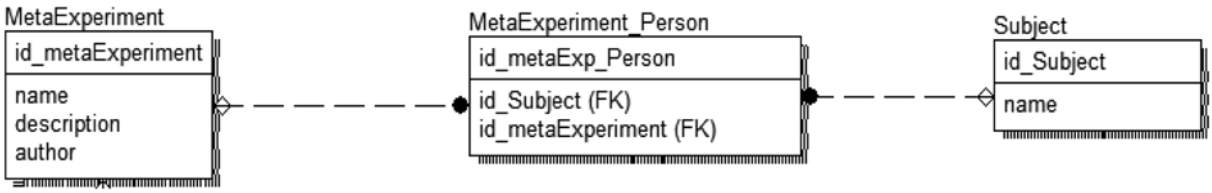

Рис. 6. Структура отношений между таблицей испытуемых и таблицей настроек эксперимента

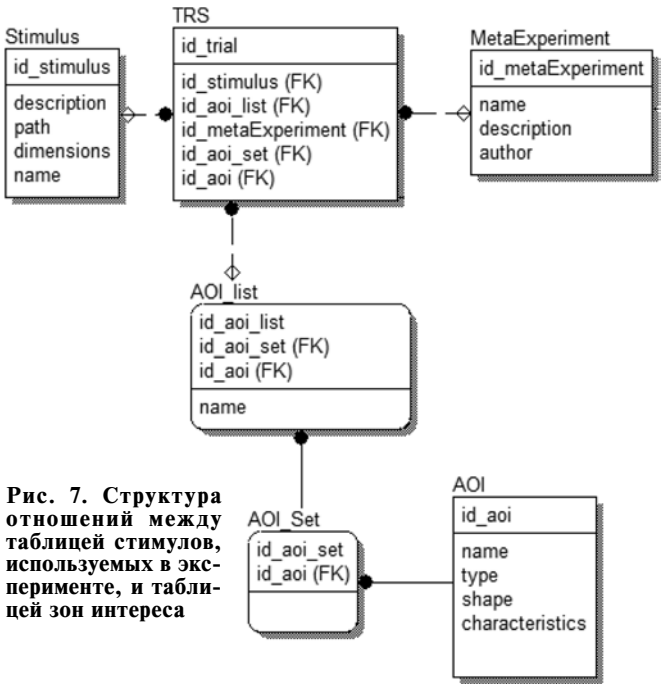

Рис. 7. Структура отношений между таблицей стимулов, используемых в эксперименте, и таблицей зон интереса

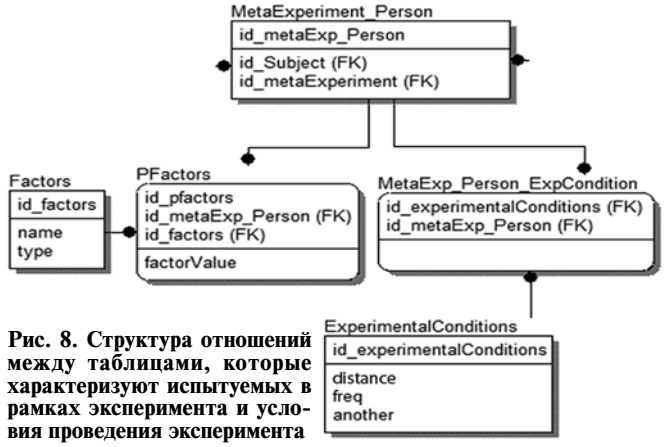

Рис. 8. Структура отношений между таблицами, которые характеризуют испытуемых в рамках эксперимента и условия проведения эксперимента

Хранение в базе данных записей о глазодвигательной активности нецелесообразно, так как эта информация достаточно велика по объему и зависит от числа фиксируемых параметров, длительности записи и частоты дискретизации. Например, при частоте дискретизации, равной 1000 Гц, 100 записей в монокулярном режиме, каждая из которых длится 3 мин,

Рис. 9. Структура отношений между таблицами, описывающими информацию о траектории и типах записей

будут содержать более 18 млн значений. Авторами предлагается хранить в сущности Trajectory ссылку на файл с записями эксперимента, а также информацию о формате получаемых данных для возможности их дальнейшего конвертирования в собственный формат разрабатываемой системы. Таблица Trajectory связана с сущностью TRS по типу один-ко-многим для возможности соотнесения траектории с определенной экспериментальной ситуацией. Связь с сущностями Left и Right необходима для дальнейшего анализа данных, требующего информацию о записях правого и левого глаза в отдельности. Предлагаемая структура хранения информации о траекториях, позволяющая соотносить ее с испытуемыми и различными видами записей глазодвигательной активности, представлена на рис. 9.

Информация о таких событиях, как моргание (Blink), глиссада (Glissade), саккада (Saccade), фиксация (Fixation) и артефакты (Artefact), может быть выделена в рамках анализа монокулярных и бинокулярных траекторий. Данные события могут быть соотнесены с траекториями, записанными в монокулярном режиме, или в отдельности для правого и левого глаза, через сущность Event (рис. 10).

Таким образом, разработанная авторами структура базы данных позволяет извлекать информацию о траекториях, относящихся к определенной экспериментальной ситуации, сопровождающейся демонстрацией стимулов, и относящихся к стимулам областей интереса. Данную информацию можно связать с испытуемыми, принимавшими участие в исследовании, их характеристиками и параметрами исследования, которые обозначаются для каждого тестируемого.

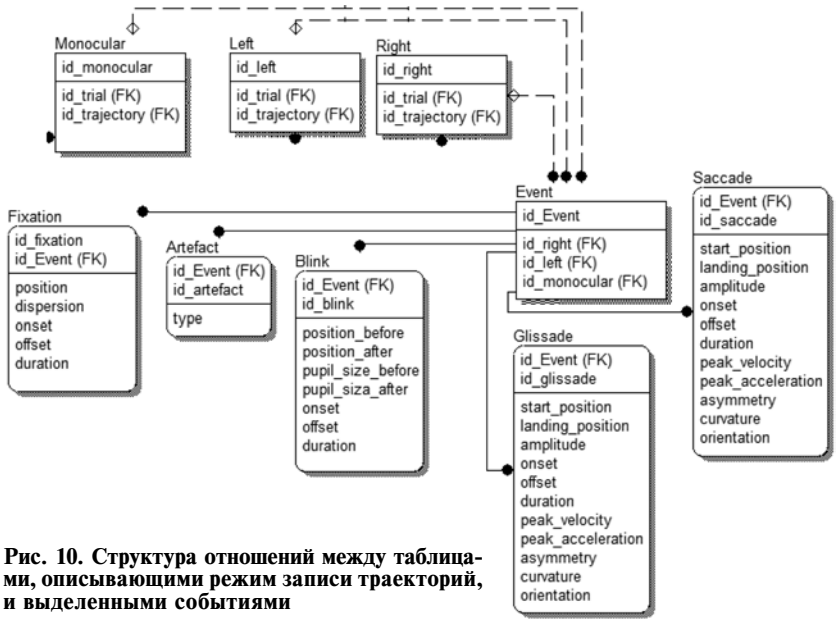

Рис. 10. Структура отношений между таблицами, описывающими режим записи траекторий, и выделенными событиями

Графический интерфейс

Разработка системы анализа результатов окулографических исследований предполагает реализацию большого числа математических методов, результаты применения которых в большинстве случаев желательно представлять графически. Необходимо также предоставить пользователю возможность работы с данными траекторий, экспериментов и испытуемых в интерактивном режиме. Представляет интерес реализация возможности задания зон интереса для стимулов в режиме *drag-and-drop* (перетаскивание). Данные требования предполагают реализацию графического интерфейса пользователя, включающего формы ввода данных, различные меню, таблицы и графики. Использование языка R изначально предполагало работу в режиме командной строки. Однако на настоящее время доступен ряд графических интерфейсов пользователя, например, пакеты Rcmdr, RKWard и RStudio.

Модуль R Commander [13] реализован в пакете Rcmdr [14] и в качестве отдельного приложения. При загрузке данного модуля, пользователь работает в модальном окне среды R и получает доступ к меню, которое позволяет импортировать данные разных форматов, проводить статистический анализ, строить различные графики, модели и др. (рис. 11).

Пакет Rcmdr предлагает возможность добавления дополнительных пунктов меню, создание диалоговых и модальных окон. Однако данный модуль продемонстрировал отсутствие должной стабильности в работе.

Пакет RKWard [15] представляет собой отдельный программный продукт, который использует для работы библиотеки среды KDE. В нем присутствует браузер текущего окружения, редактор данных, который умеет перехватывать создание графических окон и добавлять к ним функции сохранения содержимого гра-

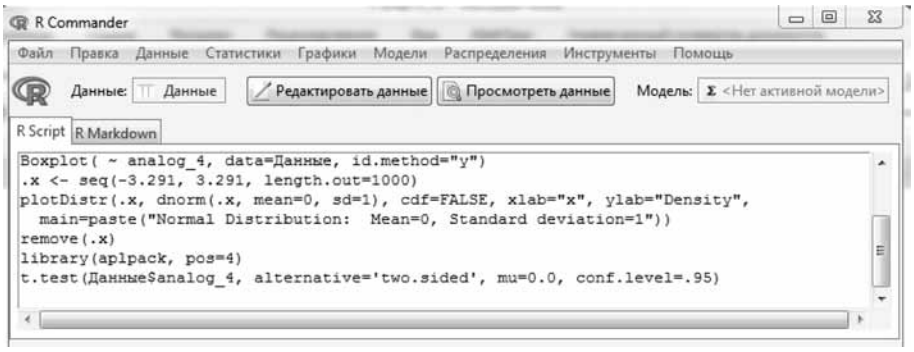

Рис. 11. Интерфейс модуля R Commander

фического окна в файлы популярных форматов (.pdf, .eps, .jpeg, .png). Интерфейс RKWard достаточно гибкий. Пользователь может его расширять за счет написания плагинов. Собственно, все встроенные средства анализа, доступные сразу же после запуска из меню, это плагины, написанные авторами RKWard [16]. Представленная система имеет ряд недостатков, среди которых можно выделить следующие:

- необходимость установки KDE;
- необходимость установки среды RKWard;
- сложность адаптации для решения новых задач.

Пакет RStudio [17] доступен в двух версиях: RStudio Desktop, в которой программа выполняется на локальной машине как обычное приложение; RStudio Server-версия, установливаемая на удаленном Linux-сервере и позволяющая осуществлять доступ к RStudio через браузер. Пакет RStudio написан на языке программирования C++ и использует фреймворк Qt для графического интерфейса пользователя. Он предоставляет большой набор дополнительных функций для работы в командной строке: управление файлами; завершение кода; автоматическое создание функций; инструментальные средства навигации по коду; контроль версий. Инструментарий Shiny [18] позволяет создавать веб-приложения с использованием языка R и HTML CSS, JavaScript, что предоставляет широкие возможности для реализации интерфейса пользователя и вывода графики в интерактивном режиме. Недостатком пакета Shiny является необходимость использования веб-сервера с установленным программным продуктом RStudio Server, что потребует загрузки файла с результатами эксперимента, объем которого может достигать 1 Гбайта.

В связи с тем, что существующие решения зачастую требуют установки дополнительного программного обеспечения, сложны в доработке и расширении, необходимо разработать графический интерфейс пользователя, способный работать в приложении R, предоставляемом разработчиками.

В результате анализа определены несколько пакетов, которые наиболее часто используются для разработки графического пользовательского интерфейса (GUI). Пакеты RGTk2 [19, 20] и tcltk2 [21] позволяют создавать модальные окна с пунктами меню, диалоговые окна, окна загрузки/сохранения файлов, использовать различные элементы управления, создавать табличное представление данных. Пакет gWidgets2 [22, 23] расширяет возможности упомянутых выше модулей и предлагает ряд функций для разработки пользовательского интерфейса.

Заключение

Авторами разработана архитектура программной системы, предназначенной для анализа результатов видеоокулографических исследований, учитывающая специфику среды разработки программ R, выбранной для реализации проекта.

Разработана структура базы данных, предназначенной для хранения результатов исследований и результатов их обработки.

Проанализированы подходы к созданию графических пользовательских интерфейсов для приложений, реализованных в среде программирования R.

Полученные результаты являются исходными для разработки программного обеспечения. Представленный в статье подход может быть использован для реализации в среде программирования R других проектов, связанных с анализом взаимосвязей между объективно измеряемыми параметрами испытуемого и характеристиками его личности. Задачи установления или проверки наличия таких взаимосвязей часто ставятся в рамках психологических исследований.

Предложенная архитектура и выбранные средства реализации обеспечат гибкость, расширяемость и кроссплатформенность готового программного проекта.

Список литературы

1. **Барабанщиков В. А., Жегалло А. В.** Регистрация и анализ направленности взора человека. М.: Институт психологии РАН, 2013. 316 с.

2. **Буховец А. Г., Москалев П. В., Богатова В. П., Бирючинская Т. Я.** Статистический анализ данных в системе R: Уч. пособие / Под ред. А. Г. Буховец. Воронеж: Изд-во ВГАУ, 2010. 124 с.

3. **Salvucci D. D., Goldberg J. H.** Identifying fixations and saccades in eye-tracking protocols // In Proceedings of the 2000 Symposium on Eye Tracking Research and Applications. 2000. P. 71—78.

4. **Olsen A.** Tobii I-VT Fixation Filter: Algorithm Description, 2012. URL: http://www.tobii.com/en/eye-tracking-research/global/library/white-papers/the-tobii-i-vt-fixation-filter (дата обращения 27.06.2014).

5. **Nyström M., Holmqvist K.** An adaptive algorithm for fixation, saccade, and glissade detection in eyetracking data // Behavior Research Methods. 2010. Vol. 42, № 1. P. 188—204.

6. **Komogortsev O. V.** Komogortsev Oleg's Web Page. Eye Movement Classification Software (offline classification). 2013. URL: http://cs.txstate.edu/~ok11/emd_offline.html (дата обращения 10.07.2014).

7. **Eddelbuettel D., Francois R.** Seamless R and C++ Integration with Rcpp. Series: Use R!, Vol. 64. Springer, 2013. 220 p.

8. **ECOOP'92.** European Conference on Object-Oriented Programming: Utrecht, The Netherlands, June 29—July 3, 1992 // Proceedings (Lecture Notes in Computer Science). Springer, 1992. 429 p.

9. **Conway J. E.** PL/R User's Guide — R Procedural Language. URL: http://www.joeconway.com/plr/doc/plr-US.pdf (дата обращения 10.07.2014).

10. **James D. A., DebRoy S.** RMySQL. URL: http://cran.r-project.org/web/packages/RMySQL/RMySQL.pdf (дата обращения: 10.07.2014).

11. **David A. J., Falcon S.** RSQLite. URL: http://cran.r-project.org/web/packages/RSQLite/RSQLite.pdf (дата обращения 10.07.2014).

12. **Ripley B., Lapsley M.** RODBC. URL: http://cran.r-project.org/web/packages/RODBC/RODBC.pdf (дата обращения 10.07.2014).
13. **Fox J.** The R Commander: A Basic-Statistics Graphical User Interface to R // Journal of Statistical Software, 2005. N. 14(9). URL: http://www.jstatsoft.org/v14/i09/paper (дата обращения 10.07.2014).
14. **Fox J., Bouchet-Valat M.** Rcmdr. — URL: http://cran.r-project.org/web/packages/Rcmdr/Rcmdr.pdf (дата обращения 10.07.2014).
15. **Rodiger S., Friedrichsmeier T., Kapat P., Michalke M.** RKWard: A Comprehensive Graphical User Interface and Integrated Development Environment for Statistical Analysis with R // Journal of Statistical Software, 2012. N. 49(9). URL: http://www.jstatsoft.org/v49/i09/paper (дата обращения 10.07.2014).
16. **Шипунов А. Б., Балдин Е. М., Волкова П. А. и др.** Наглядная статистика. Используем в R! — М.: ДМК Пресс, 2014. 298 с.
17. **Rstudio**. URL: http://www.rstudio.com (дата обращения 10.07.2014).
18. **RStudio**, Inc. shiny. URL: http://cran.r-project.org/web/packages/shiny/shiny.pdf (дата обращения 10.07.2014).
19. **Lawrence M., Lang D. T.** RGtk2: A Graphical User Interface Toolkit for R // Journal of Statistical Software. 2010. N. 37(8). URL: http://www.jstatsoft.org/v37/i08/paper (дата обращения 10.07.2014).
20. **Lawrence M., Lang D. T.** RGtk2. URL: http://cran.r-project.org/web/packages/RGtk2/RGtk2.pdf (дата обращения 10.07.2014).
21. **Grosjean P.** tcltk2. URL: http://cran.r-project.org/web/packages/tcltk2/tcltk2.pdf (дата обращения 10.07.2014).
22. **Huang B., Cook D., Wickham H.** tourrGui: A gWidgets GUI for the Tour to Explore High-Dimensional Data Using Low-DimensionalProjections // Journal of Statistical Software, 2012. N. 49(6). URL: http://www.jstatsoft.org/v49/i06/paper (дата обращения 10.07.2014).
23. **Seminar** for statistics (SfS). URL: http://stat.ethz.ch/R-manual/R-devel/library/methods/html/refClass.html (дата обращения 27.06.2014).

G. A. Yuryev, Associate Professor, e-mail: g.a.yuryev@gmail.com,
A. S. Panfilova, Programmer, e-mail: panfilova87@gmail.com,
P. A. Marmalyuk, Head of the Laboratory, e-mail: pavel.marmalyuk@gmail.com,
Moscow State University of Psychology and Education

Architecture of Software for Oculography Data Analysis

Considered is architecture of software designed for automation of calculations related to psychological research in the field of eye movement analysis. The structure of developed software classes, its content, and database structure that can be used to store experimental and computational results are under consideration.

Authors aim to develop open source software for scientific organizations involved in eye-tracking studies. This software will include: storage and tools for describing of the experiment formal scheme, the primary means of data analysis for subjects eye motion trajectories (such as filtering, smoothing, etc.), as well as set of the standard for this area algorithms to identify integral oculomotor activity indicators and more complex methods of sample analysis.

Article focuses on the description of the author's decisions about the architecture of the system core and its database.

Keywords: *software, eye tracking, R programming language, class diagram, database, graphical user interface*

References

1. **Barabanshhikov V. A., Zhegallo A. V.** Registracija i analiz napravlennosti vzora cheloveka. M.: Institut psihologii RAN, 2013. 316 p.
2. **Buhovec A. G., Moskalev P. V., Bogatova V. P., Birjuchinskaja T. Ja.** *Statisticheskij analiz dannyh v sisteme R:* Uch. posobie / Eds. A. G. Buhovec. Voronezh: Izd-vo VGAU, 2010. 124 p.
3. **Salvucci D. D., Goldberg J. H.** Identifying fixations and saccades in eye-tracking protocols. *In Proceedings of the 2000 Symposium on Eye Tracking Research and Applications*. 2000. P. 71—78.
4. **Olsen A.** Tobii I-VT Fixation Filter. Algorithm Description, 2012. URL: http://www.tobii.com/en/eye-tracking-research/global/library/white-papers/the-tobii-i-vt-fixation-filter (date of access 27.06.2014).
5. **Nystrom M., Holmqvist K.** An adaptive algorithm for fixation, saccade, and glissade detection in eyetracking data. *Behavior Research Methods*. 2010. Vol. 42, N. 1. P. 188—204.
6. **Komogortsev O. V.** Komogortsev Oleg's Web Page. Eye Movement Classification Software (offline classification). 2013. URL: http://cs.txstate.edu/~ok11/emd_offline.html (date of access 10.07.2014).
7. **Eddelbuettel D., Francois R.** *Seamless R and C++ Integration with Rcpp*. Series: Use R!, Vol. 64. Springer, 2013. 220 p.
8. **ECOOP'92.** European Conference on Object-Oriented Programming: Utrecht, The Netherlands. June 29 — July 3, 1992. *Proceedings (Lecture Notes in Computer Science)*. Springer, 1992. 429 p.
9. **Conway J. E.** PL/R User's Guide — R Procedural Language. URL: http://www.joeconway.com/plr/doc/plr-US.pdf (date of access 10.07.2014).
10. **James D. A., DebRoy S.** RMySQL. URL: http://cran.r-project.org/web/packages/RMySQL/RMySQL.pdf (date of access 10.07.2014).
11. **David A. J., Falcon S.** RSQLite. URL: http://cran.r-project.org/web/packages/RSQLite/RSQLite.pdf (date of access 10.07.2014).
12. **Ripley B., Lapsley M.** RODBC. URL: http://cran.r-project.org/web/packages/RODBC/RODBC.pdf (date of access 10.07.2014).
13. **Fox J.** The R Commander: A Basic-Statistics Graphical User Interface to R. *Journal of Statistical Software*. 2005. N. 14(9). URL: http://www.jstatsoft.org/v14/i09/paper (date of access 10.07.2014).
14. **Fox J., Bouchet-Valat M.** Rcmdr. URL: http://cran.r-project.org/web/packages/Rcmdr/Rcmdr.pdf (date of access 10.07.2014).
15. **Rodiger S., Friedrichsmeier T., Kapat P., Michalke M.** RKWard: A Comprehensive Graphical User Interface and Integrated Development Environment for Statistical Analysis with R. *Journal of Statistical Software*, 2012. N. 49(9). URL: http://www.jstatsoft.org/v49/i09/paper (date of access 10.07.2014).
16. **Shipunov A. B., Baldin E. M., Volkova P. A. et al.** *Naglja-dnaja statistika. Ispol'zuem v R!* M.: DMK Press, 2014. 298 p.
17. **Rstudio**. URL: http://www.rstudio.com (date of access 10.07.2014).
18. **RStudio**, Inc. Shiny. URL: http://cran.r-project.org/web/packages/shiny/shiny.pdf (date of access 10.07.2014).
19. **Lawrence M., Lang D. T.** RGtk2: A Graphical User Interface Toolkit for R. *Journal of Statistical Software*, 2010. N. 37(8). URL: http://www.jstatsoft.org/v37/i08/paper (date of access 10.07.2014).
20. **Lawrence M., Lang D. T.** RGtk2. URL: http://cran.r-project.org/web/packages/RGtk2/RGtk2.pdf (date of access 10.07.2014).
21. **Grosjean P.** tcltk2. URL: http://cran.r-project.org/web/packages/tcltk2/tcltk2.pdf (date of access 10.07.2014).
22. **Huang B., Cook D., Wickham H.** tourrGui: A gWidgets GUI for the Tour to Explore High-Dimensional Data Using Low-DimensionalProjections. *Journal of Statistical Software*. 2012. N. 49 (6). URL: http://www.jstatsoft.org/v49/i06/paper (date of access 10.07.2014).
23. **Seminar** for statistics (SfS). URL: http://stat.ethz.ch/R-manual/R-devel/library/methods/html/refClass.html (date of access 27.06.2014).

УДК 616-71

А. В. Кирюхин, аспирант, мл. науч. сотр., e-mail: anton-vk@yandex.ru,
А. В. Алпатов, канд. техн. наук, доц., e-mail: alpatov-alexey@yandex.ru,
Рязанский государственный радиотехнический университет

Проектирование автоматизированной системы для оценки аффективных состояний человека на основе измерения пульса при просмотре видеоизображений

Продемонстрированы возможности среды разработки программного обеспечения LabVIEW для прототипирования автоматизированной системы, позволяющей получать оценки аффективных состояний человека. Отмечены преимущества среды LabVIEW в условиях синхронной регистрации и анализа сигналов различного типа, таких как пульс и видеоизображение. Приведены примеры проектирования блок-диаграмм и особенности использования в них конкретных библиотек и функций.

Ключевые слова: *LabVIEW, аффективные состояния, пульс, видеоизображение лица, вариабельность ритма сердца, IMAQ-функции, VLC-проигрыватель, блок-диаграмма*

Введение

В последние десятилетия многими исследователями в области нормальной физиологии, безопасности на транспортных и промышленных объектах решается ряд научных задач, связанных с оценкой эмоциональных реакций человека на различные воздействия окружающей среды и социума. К этому направлению относятся вопросы управления стрессовыми ситуациями, возникающими в разных сферах деятельности человека; оценки эмоционального восприятия разного рода информации; изучение психофизиологических механизмов, лежащих в основе формирования эмоционального поведения человека и многие другие задачи, связанные с психоэмоциональным состоянием человека. Одной из таких актуальных в настоящее время задач — оценке эмоционального восприятия человеком различной информации — посвящено большое число работ [1, 2]. Выделилось целое научное направление, связанное с исследованием эмоционального восприятия человека при воздействии на него стимулов разного типа (визуальные, звуковые, тактильные, смешанные). При этом до, после или во время подачи стимула проводится объективная или субъективная оценка различных параметров и реакций человека. Объективная оценка, как правило, осуществляется по результатам регистрации психофизиологических параметров (пульс, электроэнцефалограмма, кожно-гальваническая реакция). Субъективная оценка получается после обработки большого числа психологических тестов, самоопроса и анкетирования. Одним из наиболее сложных эмоциональных стимулов является видеоизображение со звуковым сопровождением. Как правило, для проведения таких исследований необходима большая группа специалистов в области проектирования систем автоматизированного сбора данных и биомедицинской инженерии, имеющих навыки работы с физиологическими сигналами и видеоизображениями, что усложняет реализацию поставленных задач.

В исследовании, результаты которого представлены в настоящей статье, удалось сконцентрировать всю работу в рамках небольшой группы специалистов за счет использования единой среды разработки программ, позволившей создать приложение для регистрации и анализа необходимых данных.

Особенности разрабатываемой системы

Целью работы, результаты выполнения которой представлены далее, является разработка автоматизированной системы, реализующей синхронную регистрацию психофизиологических параметров человека совместно с просмотром аудиовизуального стимула и последующую оценку эмоционального восприятия предоставляемой испытуемым информации.

Автоматизированная система (далее для краткости изложения — система) осуществляет регистрацию психофизиологических параметров испытуемого во время исследования, таких как пульс и мимика лица. Наиболее важным в решении поставленной

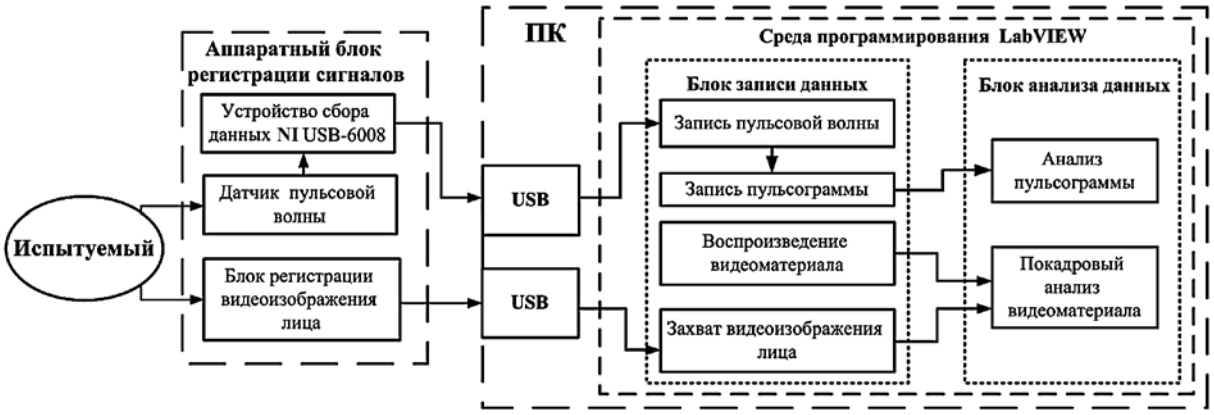

Рис. 1. Структура системы

задачи является поиск характерных особенностей в сигнале пульса, его производных параметрах и в видеоизображении лица, свидетельствующих об изменении эмоциональных реакций на стимул и их временное сопоставление с визуальными сценами предоставляемого материала.

При исследованиях такого рода происходит изменение аффективных состояний (аффектов) испытуемого. Аффект, как и любой другой эмоциональный процесс, представляет собой психофизиологический процесс внутренней регуляции деятельности и отражает бессознательную субъективную оценку текущей ситуации. Его уникальными характеристиками являются кратковременный характер и высокая интенсивность, в сочетании с выраженными проявлениями в поведении и работе внутренних органов [3].

Важно, что аффекты оказывают сильное влияние на физиологические параметры, в частности, на пульс. В ответ на эмоциональное воздействие происходит резкое изменение пульса [4].

Экспериментальные исследования эмоционального состояния и, в частности, аффективных состояний при просмотре различных видеообразов проводятся в таких областях, как физиология, психология, маркетинговые исследования. Для проведения подобного эксперимента была создана исследовательская система, включающая в себя программную и аппаратную части и содержащая компоненты, которые реализуют:

- просмотр видеоизображения;
- регистрацию физиологических и антропометрических показателей;
- совместный анализ полученных данных.

Особенностью данного аппаратно-программного комплекса является реализация синхронной регистрации пульса и видеоизображения лица испытуемого во время просмотра видеостимула (слайд-фото, ролика, короткометражного или полнометражного фильма).

Разработка системы включала в себя анализ сигналов пульса и видеоизображения лица, принципиально отличающихся способами регистрации и обработки, что потребовало соответствующего специализированного программного обеспечения. Структура предлагаемой системы показана на рис. 1.

Система, кроме испытуемого, включает в себя три основных блока: аппаратный блок регистрации сигналов; блок записи данных и блок анализа данных. Аппаратный блок регистрации сигналов содержит устройства, позволяющие проводить считывание всех необходимых психофизиологических параметров человека. Блок записи данных реализован на базе среды разработки программного обеспечения, он осуществляет сбор и сохранение в памяти персонального компьютера (ПК) регистрируемой с испытуемого информации. Блок анализа данных проводит обработку и графическое представление записанной ранее в блоке записи данных информации.

Для реализации поставленной цели была выбрана LabVIEW (*Laboratory Virtual Instrumentation Engineering Workbench*) — среда разработки программного обеспечения, представляющая хорошие возможности работы с видеоизображениями и цифровыми сигналами. Ее использование позволило параллельно проводить цифровую обработку сигналов и цифровую обработку видеоизображений. Программа, написанная в среде LabVIEW, называется и является виртуальным прибором (*Virtual Instrument*) и состоит из следующих двух частей:

- блок-диаграммы, описывающей логику работы виртуального прибора;
- лицевой панели, описывающей внешний интерфейс виртуального прибора [5].

Универсальный характер и простота разработки в среде LabVIEW позволили успешно реализовать программные механизмы, которые в рамках единого модуля обеспечивают выполнение перечисленных выше базовых функций.

Анализируемые параметры

В качестве анализируемых были выбраны два сигнала — пульсовая волна и видеоизображение лица испытуемого. Для задачи регистрации пульсовой волны был разработан фотоплетизмографический датчик. Браслет со светодиодом и фотоприемником

Таблица 1

Параметры регистрируемых сигналов

Тип сигнала	Технические параметры сигнала				
Видеоизображение лица	Частота кадров 30 FPS	Максимальное разрешение 128×720	Режим фокусировки ручной	Тип файла записи видео — avi	Фильтрация изображения отсутствует
Пульсовая волна	Частота дискретизации 60 Гц	Разрядность АЦП 10 бит	Амплитуда усиленного сигнала 0...2,5 В	Тип файла записи сигнала — txt	Используемые фильтры: цифровой фильтр нижних частот, фильтр скользящего среднего

закрепляется на запястье, что в некоторой степени позволяет устранить артефакты движения, которые вносят свое влияние в форму пульсовой волны. Сигнал с датчика поступает на устройство сбора данных NI USB-6008, которое сопрягается с персональным компьютером.

Блок регистрации видеоизображения лица представляет собой USB-HD-камеру, закрепляемую на верхней части монитора, на котором воспроизводится предлагаемый видеофрагмент. Технические параметры регистрируемых сигналов приведены в табл. 1.

В процессе проектирования были сформулированы задачи по обработке данных сигналов и предложена аппаратно-программная платформа для решения этих задач в рамках автоматизированной системы на основе среды разработки LabVIEW. К числу таких задач относятся:

- регистрация пульсовой волны;
- запись пульсограммы;
- захват видеоизображения лица;
- воспроизведение видеоматериала;
- покадровый анализ видеоматериала;
- анализ пульсограммы.

Рассмотрим реализацию каждой из задач, входящую в блок записи данных на рис. 1.

Регистрация пульсовой волны

Для определения значений пульса и накопления их в массив значений, называемый пульсограммой, необходимо осуществить регистрацию пульсовой волны испытуемого. Для этого, кроме разработки аппаратной части решения требовалось создать программный код в среде LabVIEW, реализующий регистрацию и отображение сигнала пульсовой волны на графическом индикаторе. Внешний вид лицевой панели блока записи данных представлен на рис. 2, см. третью сторону обложки.

С помощью библиотеки DAQmx среды LabVIEW была реализована регистрация данных в LabVIEW с устройства NI USB-6008, к которому подключается датчик пульсовой волны. Фрагмент блок-диаграммы инициализации и записи сигнала пульсовой волны показан на рис. 3, см. третью сторону обложки.

Функция DAQmx Create Virtual Channel.vi, предназначенная для создания канала регистрации с устройства сбора данных, позволяет задать граничные значения амплитуды сигнала, выбрать номер канала регистрации и тип регистрируемого сигнала (напряжение, ток, температура и т. д.). С помощью функции DAQmx Timing.vi задается частота дискретизации пульсовой волны, равная 60 Гц. Для открытия сессии регистрации сигнала при запуске виртуального прибора размещена функция DAQmx Start Task.vi.

При нажатии кнопки "Стоп" основного цикла сессия должна быть закрыта, поэтому за пределами цикла размещена функция DAQmx Clear Task.vi, осуществляющая завершение процесса регистрации. Перечисленные выше функции располагаются вне цикла, в отличие от функции DAQmx Read.vi, которая проводит считывание указанного числа выборок за один цикл и размещается внутри основного цикла.

Задачу регистрации сигнала с устройств сбора данных в среде LabVIEW можно решать и другими способами, в частности, с использованием DAQ Assistant — графического интерфейса настройки измерительных задач и каналов. Преимуществом его использования является возможность быстрой настройки канала регистрации сигнала и просмотр требуемого результата. Однако, в отличие от DAQmx, использование интерфейса DAQ Assistant является менее гибким и ограниченным по функциональным возможностям средством решения поставленной задачи.

Запись пульсограммы

В целях улучшения формы сигнала и избавления от нежелательных помех непосредственно перед детектированием пиковых значений пульсовой волны происходит фильтрация с помощью цифрового фильтра нижних частот и фильтра скользящего среднего.

Поскольку пульсовая волна и электрокардиосигнал являются сложными биологическими сигналами, амплитудные и временные параметры которых зависят от многих факторов (в том числе от состояния человека), то задача их обработки с позиций программирования не является тривиальной. По этой причине разработчики среды LabVIEW включили в нее функции, необходимые для обработки биомедицинских сигналов и данных. Оригинальный способ реализации блока записи пульсограммы за-

ключается в применении встроенных функций Online Multiscale Peak Detection.vi из палитры вейвлет-преобразования, предназначенной для поиска пиков на цифровом сигнале, и Extract Heart Rate.vi, предназначенной для выделения ритма сердца из сигналов пульсовой волны или электрокардиосигнала [6]. Палитрой в среде LabVIEW называется набор функций и элементов, сгруппированных по общему назначению (палитры логических элементов, обработки сигналов, работы с массивами и т. д.).

Необходимо отметить одну особенность, влияющую на стабильность работы рассматриваемого блока. Она заключается в реализации плавающей автоподстройки границы срабатывания пик-детектора. Пик-детектором, в общем случае, называется устройство, реализованное программным или аппаратным способом и позволяющее определять максимальные или минимальные амплитудные значения сигнала в заданных условиях.

Необходимость такой подстройки была вызвана тем, что пульсовая волна изменяет свою форму и амплитуду в зависимости от конкретного испытуемого и использование жестко заданного значения границы (*Threshold*), необходимого для срабатывания пик-детектора, неприемлемо. Плавающая граница вычисляется путем усреднения значения амплитуды сигнала, взятого за два периода (120 отсчетов).

В зависимости от формы сигнала и уровня его зашумленности на пульсовой волне детектируются максимумы или минимумы, по которым определяются мгновенные значения частоты сердечных сокращений (пульса).

Для наглядности и удобства просмотра на графический индикатор, кроме исходного сигнала, выведены значения плавающей границы пик-детектора и маркерами подмечены найденные пиковые значения пульсовой волны.

Захват видеоизображения лица

Для реализации процесса захвата видеоизображения с USB-камеры синхронно с регистрацией пульсовой волны было исследовано несколько типов драйверов, входящих в состав среды LabVIEW. Такая USB-камера представляет собой малоразмерную цифровую видеокамеру, подключенную к ПК по средствам USB-интерфейса и способную в реальном времени фиксировать изображения лица испытуемого.

Особенности аппаратного сопряжения каждого драйвера представлены в табл. 2.

В табл. 2 видно, что драйверы NI-IMAQ работают только с оборудованием, производимым компанией National Instruments. Драйвер NI-IMAQ I/O позволяет подключать устройства видеозахвата к платам сбора данных PCI и PCIe. Для подключения USB-камер подходит только драйвер NI-IMAQdx, который и используется для решения поставленной задачи. Фрагмент блок-диаграммы захвата видеоизображения лица показан на рис. 4, см. четвертую сторону обложки.

Таблица 2

Устройства, поддерживаемые драйверами класса IMAQ (IMageAcQuision) — технология захвата изображения, аппаратная часть которой включает модули захвата изображения) [7]

Тип драйвера	Поддерживаемые устройства
NI-IMAQ	NI сетевые фрейм-грабберы NI параллельные цифровые фрейм-грабберы NI аналоговые фрейм-грабберы NI 17xx смарт-камеры
NI-IMAQdx	Ethernet-камеры с поддержкой GigE Vision FireWire IEEE 1394 камеры USB 2.0-камеры с поддержкой Microsoft DirectShow USB 3.0-камеры с поддержкой USB3 Vision
NI-IMAQ I/O	NI PCIe-8255R NI PCI-8254R NI PCIe-8237R

Функция IMAQdx Open Camera.vi предназначена для запуска сессии регистрации видеоизображения. Если в системе присутствует несколько видеокамер, функция IMAQdx Open Camera.vi позволяет выбирать необходимую пользователю камеру из списка. С помощью узла свойств (*Property Node*) указывается разрешение записываемого изображения. Узлами свойств в LabVIEW называются элементы блок-диаграмм, позволяющие программно управлять свойствами объектов лицевой панели, такими как цвет, видимость, местоположение, формат представления чисел и т. д. [8]. Функция IMAQ AVI Create позволяет установить частоту кадров видеоизображения и указать путь к файлу. После захвата изображения с USB-камеры с использованием функции IMAQdx Grab, оно поступает на функцию записи avi-файла IMAQ AVI Write Frame.

Таким образом, совместное использование функций из палитр IMAQdx и IMAQ AVI позволило построить компактную блок-диаграмму, способную воспроизводить и записывать видеоизображение с USB-камеры. Такой подход предоставил возможность избежать использования стороннего приложения, осуществляющего захват и запись изображения с камеры, что, безусловно, являлось бы непроизводительным решением.

Воспроизведение видеоматериала

Для проведения эксперимента кроме записи пульсовой волны и изображения лица требуется просмотр испытуемым определенного видеоматериала, воспроизведение которого запускалось бы исследователем после настройки им всех необходимых параметров сигналов. Поскольку мимика лица испытуемого свидетельствует об изменении эмоционального восприятия видеоматериала, то важной особенностью реали-

зации данной задачи является синхронное воспроизведение и запись данных. Это позволяет в дальнейшем проводить покадровый анализ записанных данных и предоставляемого стимула, сопоставляя определенные значения пульсограммы с конкретными моментами видеоролика и мимикой лица человека.

При решении поставленной задачи были изучены несколько возможных вариантов ее реализации в среде LabVIEW. Реализации, использующие среду ActiveX (для определения пригодных к использованию программных компонентов, написанных на разных языках программирования) совместно с Windows Media Player на лицевой панели виртуального прибора, показали нестабильную работу. Такая нестабильность проявлялась в запаздывании момента запуска воспроизведения и в трудностях, связанных с отображением и окончанием просмотра видеофайла. Программы, реализованные на компонентах NI-IMAQ AVI, отличались недостаточным быстродействием, что приводило к запаздыванию воспроизведения видеоролика [9].

Успешного решения поставленной задачи удалось добиться с использованием библиотеки платформы .NET и подключаемому к ней VLC-проигрывателю (VideoLAN Client — свободный кроссплатформенный медиаплеер). Такая комбинация компонентов, кроме воспроизведения видео-аудио-изображений, также позволяет в среде LabVIEW транслировать изображение с IP-камер, работать с программным средством DirectShow и проводить кодирование/декодирование видео-аудио информации [10]. Представленная реализация кода отличается высоким быстродействием в вызове VLC-проигрывателя и отсутствием запаздывания при просмотре видеоматериала.

В качестве видеоматериалов испытуемым предоставляются различные ролики длительностью 5...10 мин.

Рассмотрим далее реализацию блока анализа данных. Данный блок включает в себя синхронный анализ пульсограммы и видеоматериалов.

Покадровый анализ видеоматериала

Покадровый анализ видеоматериала совмещает в себе синхронный просмотр видеоролика и видеоизображения мимики лица. При реализации просмотра двух и более видеофрагментов за основу была выбрана палитра функций IMAQ AVI. Как было отмечено ранее, применение функций IMAQ AVI не обладает высоким быстродействием, однако для выбора определенного кадра видеофрагментов эта особенность не оказалась критичной. Лицевая панель блока анализа данных представлена на рис. 5, см. четвертую сторону обложки.

Смещение кадров происходит в соответствии с положением курсора, перемещаемого на пульсограмме. Фрагмент блок-диаграммы данного блока представлен на рис. 6, см. четвертую сторону обложки.

Положение курсора, которое отображается на графическом индикаторе, определяется с помощью узла свойств Cursor Position X. По значению положения курсора и значению частоты воспроизведения кадров видеофрагмента определяется номер кадра, отображаемого на экране.

Техническая особенность решения задачи покадрового анализа, позволяющая минимизировать программный код и синхронизировать между собой анализируемые данные, заключается в использовании структуры событий Event Case. Преимуществом использования данной структуры совместно с непрерывным циклом по условию (*While Loop*) является возможность исполнять реализованный разработчиком код в ответ на множество событий (нажатие кнопки, изменение положения курсора на графическом индикаторе, изменение значения числовых терминалов и т. д.). Данная структура использует минимальное количество системных ресурсов компьютера. Код исполняется только в моменты возникновения событий, в отличие от цикла по условию, который исполняется в течение всей работы программы.

Анализ пульсограммы

Задача анализа пульсограммы включает в себя реализацию нескольких блоков, которые выполняют расчет графических зависимостей и дискретных параметров. Схема анализа пульсограммы показана на рис. 7.

Основные составляющие данной схемы: построение основной гистограммы, выполненное на базе статистических методов анализа ритма сердца; построение скатерграммы — одного из графических видов регрессионного анализа; вычисление дискретных параметров. Кроме этого, важными составляющими схемы являются спектральный и флуктуационный анализы, получившие в настоящее время широкое распространение.

Для синхронного просмотра данных проводится интерполяция значений пульса с шагом 1 с. Это позволяет корректно синхронизировать все просматриваемые данные (рис. 8).

Проектирование алгоритмов, применяемых для анализа пульсограммы, проводилось на основе методических рекомендаций по анализу вариабельности сердечного ритма [11]. Математическая база для расчета флуктуационной функции и скейлингового показателя основывается на модифицированном методе флуктуационного анализа в режиме реального времени [12, 13]. Отличительной особенностью реализации метода в среде LabVIEW является использование структуры MathScript, позволяющей интегрировать программный код, написанный в среде MATLAB. Очевидным преимуществом использования данной структуры, при наличии реализованных в среде MATLAB алгоритмов, является экономия трудовых ресурсов. Недостатком данной структуры является трудность отладки и модификации кода разработчиком, не знакомым со средой MATLAB, отсутствие наглядности текстового кода, присущей графическим языкам программирования, а также более длительное время исполнения кода.

Рис. 7. Структурная схема анализа пульсограммы

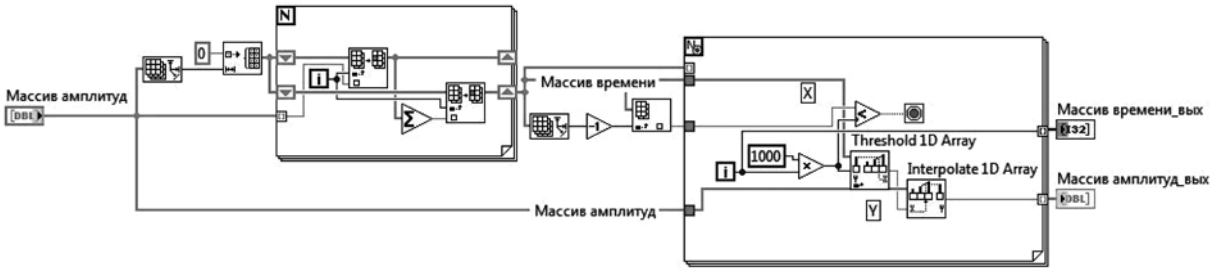

Рис. 8. Фрагмент блок-диаграммы, реализующий интерполяцию исходной пульсограммы

Оригинальностью в решении задачи анализа пульсограммы является использование оконного метода для вычисления стресс-индекса и скейлингового показателя. Данный метод предполагает выбор исследователем размера окна (например, 128, 256 цифровых отсчетов), в котором проводится расчет выбранных параметров в цикле со смещением окна на один отсчет вперед. Таким образом, формируются зависимости, характеризующие изменение эмоционального восприятия за определенный временной диапазон, равный размеру окна. Оконный метод применяется для разных дискретных параметров, стандартизированных для анализа вариабельности ритма сердца (PNN50, RMSSD, SDNN, CV, MxDMn, D, RRcp).

Заключение

Представлены результаты разработки автоматизированной системы для оценки аффективных состояний человека на основе измерения пульса при просмотре видеоизображений. Приведена структурная схема системы, включающая аппаратную и программную части. В табличной форме наглядно показаны технические характеристики регистрируемых сигналов, а также варианты сопряжения видеоустройств с драйвером класса IMAQ среды LabVIEW. Были подробно описаны решения задач, необходимых для реализации блока записи данных, с описанием отличительных особенностей и оригинальных деталей выбранных решений. Графически продемонстрирован внешний вид лицевых панелей блоков записи и анализа данных. Приведена структурная схема анализа пульсограммы, включающая оконный метод расчета стресс-индекса и скейлингового показателя, графические зависимости и дискретные параметры вариабельности сердечного ритма.

Отличительной особенностью проектирования данной системы являлась работа с сигналами различного типа. Как правило, реализация таких систем требует непосредственного участия специалистов по цифровой обработке сигналов и цифровой обработке изображений. Однако использование среды разработки программ LabVIEW за счет широкого набора драйверов, библиотек, шаблонов программирования, а также справочной информации и средств технической поддержки [14], позволила минимизировать число участников, занятых созданием целевой системы.

Результаты, полученные в ходе разработки системы, используются для проведения исследований на кафедре Биомедицинской и полупроводниковой электроники Рязанского государственного радиотехнического университета совместно с кафедрой Нормальной физиологии с курсом психофизиологии Рязанского государственного медицинского университета им. академика И. П. Павлова. Данные

исследования проводятся в области моделирования целенаправленной деятельности человека, а также в маркетинговых исследованиях для изучения воздействия на людей рекламной продукции.

Работа выполнена при поддержке ООО "Дарта Системс" и МИП ООО "Наномед".

Список литературы

1. **Nasoz F., Alvarez K., Lisetti C., Finkelstein N.** Emotion recognition from physiological signals for presence technologies // Journal of Cognition, Technology, and Work — Special Issue on Presence. 2004. Vol. 6 (1). P. 4—14.

2. **Nakahara H., Furuya S., Obata S. et al.** Emotion-related changes in heart rate and its variability during performance and perception of music // The neurosciences and music III: disorders and plasticity. 2009. Vol. 1169. P. 359—362.

3. **Балабанова Л. М.** Судебная патопсихология (вопросы определения нормы и отклонений). Донецк: Сталкер, 1998. 432 с.

4. **Эмоции** и чувства. URL: http://www.psychologos.ru/articles/view/emocii_i_chuvstva_e.p._ilin/-1 (дата обращения 01.09.14).

5. **LabVIEW.** URL: http://ru.wikipedia.org/wiki/LabVIEW (дата обращения 01.09.14).

6. **Пик-детектор** в реальном времени. URL: http://labviewportal.eu/viewtopic.php?f = 81&t = 5576 (дата обращения 01.09.14).

7. **What is** the difference between NI-IMAQ, NI-IMAQdx, and NI-IMAQ I/O? URL: http://digital.ni.com/public.nsf/allkb/0564022DAFF513D2862579490057D42E (дата обращения 01.09.14).

8. **Тревис Дж.** LabVIEW для всех. М.: ДМК Пресс, 2005. 544 с.

9. **Форум** российского сообщества LabVIEW-разработчиков. URL: http://labviewportal.ru/ (дата обращения 01.09.14).

10. **Всемирный** форум LabVIEW-разработчиков. URL: https://forums.ni.com/ (дата обращения 01.09.14).

11. **Баевский Р. М.** Анализ вариабельности сердечного ритма при использовании различных электрокардиографических систем // Вестник аритмологии. 2001. № 24. С. 65—87.

12. **Алпатов А. В., Митрофанова М. Ю.** Метод флуктуационного анализа сердечного ритма в режиме реального времени // Биомедицинская радиоэлектроника. 2011. № 7. С. 66—71.

13. **Лапкин М. М., Вихров С. П., Алпатов А. В., Митрофанова М. Ю.** Фрактально-флуктуационный анализ нелинейных компонентов сердечного ритма для параметризации функционального состояния человека // Российский медико-биологический вестник им. академика И. П. Павлова. 2012. № 2. С. 96—106.

14. **Русскоязычный** сайт среды графического программирования LabVIEW. URL: http://www.labview.ru (дата обращения 01.09.14).

A. V. Kiryukhin, Postgraduate Student, Junior Researcher, e-mail: anton-vk@yandex.ru,
A. V. Alpatov, Associate Professor, e-mail: alpatov-alexey@yandex.ru, Ryazan State Radio Engineering University

System for the Affective State Assessment of Humans on the Basis Heart Rate Measurement upon Video Presentation

The purpose of this article consists in development of the system for affective states assessment, according to pulsometry and facial expressions of the men. The system design comprises the problem of the hardware realization of signal detection block, data recording and analysis. The advantages of LabVIEW are well observed in terms of the synchronous recording and analysis of the different types of signals, such as pulse and video. The sequence of virtual instrument construction for recording the pulse wave using a DAQ device NI USB-6008 is presented in detail. Much attention is given to the use of libraries and functions for operating with video images. The features of the window method for the analysis of heart rate variability using fluctuation analysis are briefly described. The patterns of the block diagrams design and the peculiarities of the specific libraries and functions usage are given. This article is the concern of the physicians, engineers of medical equipment and specialists occupied in the areas of human activity where the information about the men's emotional state increases the work safety.

Keywords: *LabVIEW, affective state, pulse, video facial, heart rate variability, IMAQ-functions, VLC-player, block diagram, data acquisition, virtual instrument*

References

1. **Nasoz F., Alvarez K., Lisetti C., Finkelstein N.** Emotion recognition from physiological signals for presence technologies. *Journal of Cognition, Technology, and Work — Special Issue on Presence*. 2004. Vol. 6 (1). P. 4—14.

2. **Nakahara H., Furuya S., Obata S. et al.** Emotion-related changes in heart rate and its variability during performance and perception of music. *The neurosciences and music III: disorders and plasticity*. 2009. Vol. 1169. P. 359—362.

3. **Balabanova L. M.** *Sudebnaja patopsihologija* (*voprosy opredelenija normy i otklonenij*). Doneck: Stalker, 1998. 432 p.

4. **Jemocii** i chuvstva. URL: http://www.psychologos.ru/articles/view/emocii_i_chuvstva_e.p._ilin/-1 (date of access 01.09.14).

5. **LabVIEW**. URL: http://ru.wikipedia.org/wiki/LabVIEW (date of access 01.09.14).

6. **The peak** detector in real time. URL: http://labviewportal.eu/viewtopic.php?f = 81&t = 5576 (date of access 01.09.14).

7. **What** is the difference between NI-IMAQ, NI-IMAQdx, and NI-IMAQ I/O? URL: http://digital.ni.com/public.nsf/allkb/0564022DAFF513D2862579490057D42E (date of access 01.09.14).

8. **Travis J.** *LabVIEW dlya vseh*. M.: DMK Press, 2005. 544 p.

9. **Forum** of Russian community of LabVIEW-developers. URL: http://labviewportal.ru/(date of access 01.09.14).

10. **World** forum of LabVIEW-developers. URL: https://forums.ni.com/ (date of access 01.09.14).

11. **Baevskij R. M.** Analiz variabel'nosti serdechnogo ritma pri ispol'zovanii razlichnyh jelektrokardiograficheskih system. *Vestnik aritmologii*. 2001. N. 24. P. 65—87.

12. **Alpatov A. V., Mitrofanova M. Ju.** Metod fluktuacionnogo analiza serdechnogo ritma v rezhime real'nogo vremeni. *Biomedicinskaja radiojelektronika*. 2011. N. 7. P. 66—71.

13. **Lapkin M. M., Vikhrov S. P., Alpatov A. V., Mitrofanova M. Yu.** Fraktal'no-fluktuacionnyj analiz nelinejnyh komponentov serdechnogo ritma dlja parametrizacii funkcional'nogo sostojanija cheloveka. *Rossiyskiy mediko-biologicheskiy vestnik imeni akademika I. P. Pavlova*. 2012. N. 2. P. 96—106.

14. **Russian** site of graphical programming environment LabVIEW. URL: http://www.labview.ru (date of access 01.09.14).

УДК 004.912

И. С. Кононенко[1, 2], науч. сотр., e-mail: irina_k@cn.ru,
Н. В. Саломатина[1, 3], канд. физ.-мат. наук, науч. сотр., e-mail: nataly@math.nsc.ru,
Е. А. Сидорова[1, 2], канд. физ.-мат. наук, ст. науч. сотр., e-mail: lena@iis.nsk.su
[1] Новосибирский государственный университет,
[2] Институт систем информатики им. А. П. Ершова СО РАН, г. Новосибирск,
[3] Институт математики им. С. Л. Соболева СО РАН, г. Новосибирск

Опыт создания тематических словарей для рубрикации коротких описаний веб-сайтов

Предложены методы построения мультиязычных тематических словарей, базирующиеся как на использовании автоматических средств для создания тематических коллекций и их статистической обработки, так и на экспертных знаниях, представленных в ресурсах типа тезаурусов и каталогов. Создаваемые словари предназначены для классификации коротких текстов (пользовательских запросов). Приведены характеристики построенных русско- и англоязычных словарей, а также результаты их тестирования на подборке запросов.

Ключевые слова: *тематический словарь, тематическая коллекция текстов, метапоиск, тезаурус, каталог, классификация запросов, интернет-деятельность*

Введение

Под тематическими понимают словари, содержащие терминологию некоторой предметной области (ПО) и нацеленные на решение определенной задачи обработки текстового материала. Одним из самых распространенных примеров применения тематических словарей (ТС) является автоматическая рубрикация или классификация запросов пользователей в поисковых сервисах (когда запрос сопоставляется с рубриками из заданного множества).

Ориентируясь на современные методы автоматической рубрикации и создания ТС, рассматривают две основные технологии: на основе методов машинного обучения, см., например, работы [1, 2]; основанные на знаниях, включая "инженерные" [3, 4] или экспертные [5]. Например, инженерный подход к построению ТС с применением тезауруса общеполитической лексики, предложенный в работе [6], отличается тем, что сначала тема коротко описывается вручную, а затем расширяется с помощью иерархических связей, зафиксированных в тезаурусе. Основой для машинного обучения служит обширная и последовательно отрубрицированная коллекция текстов соответствующего жанра. Задача, которая рассматривается далее, направлена на разработку методов формирования ТС, ориентированных на проблему рубрикации коротких описаний (далее *запросов*), описывающих сайты, реализующие различные виды повседневной интернет-деятельности. Особенностью поставленной задачи является отсутствие представительной коллекции текстов запросов. Создание обучающего корпуса текстов для системы рубрикации затруднено двумя следующими обстоятельствами: реальной коллекции коротких описаний такого рода не существует; согласованное ручное рубрицирование такой коллекции трудоемко. В отсутствие обучающих данных приходится использовать методы, связанные с созданием словарей ключевых слов и их настройкой лингвистами и инженерами знаний. Для рассматриваемой в настоящей работе задачи, состоящей в определении тематики текстов особого, запросного жанра, как и для случая заголовков и других микротекстов, плохо применимы методики распознавания, основанные на подсчете весов [7]. Соответствие описания рубрике должно определяться лишь фактом вхождения в текст терминов, приписанных этой рубрике составителями словаря системы классификации. В такой ситуации, как отмечено в работе [8], на первый план выходит уменьшение трудоемкости, создание различных помощников для автоматизации деятельности составителей словаря и экспертов. *Цель исследования*, результаты которого представлены в данной работе, — разработать в помощь эксперту методы и программные средства для построения тематических мультиязычных словарей.

В настоящей работе методы формирования ТС будут продемонстрированы на примере предметной области "Интернет-деятельность" и жанра запросов, которые формируют пользователи при описании своего сайта как средства реализации деятельности. Запросы могут задаваться пользователем на русском или английском языках.

1. Методика создания тематических словарей

Предлагаемая методика создания ТС включает:

1) автоматизацию сбора коллекции текстов, содержащих тематически значимые слова;

2) статистическую обработку текстов в целях наполнения и обучения словаря;

3) расширение словаря по тезаурусным связям;

4) автоматическую проверку качества построенного словаря.

Известно, что для некоторых научных предметных областей имеются ресурсы, которые достаточно полно описывают ПО и помогают строить словари более узких и более широких тематик. При описании различных видов повседневной интернет-деятельности таким ресурсом может служить универсальный тезаурус английского языка WordNet [9]. Кроме того, в отличие от научных ПО, где основная лексика содержится в специальной литературе, не всегда доступной поисковым системам, описания повседневной деятельности легко могут быть найдены с помощью метапоиска, т. е. с использованием известных поисковых систем в сети Интернет. Дополнительно могут быть привлечены каталоги ресурсов типа Яндекс-каталог (yaca.yandex.ru) или DMOZ (www.dmoz.org), в которых потребуется осуществлять поиск не ресурсов, а рубрик, соответствующих "базовым" рубрикам (т. е. рубрикам рубрикатора ПО интернет-деятельности). Преимущество использования таких каталогов заключается в том, что в них ресурсы, размещенные в определенной рубрике, предварительно проверяются экспертами, а значит степень релевантности их относительно соответствующей рубрики очень высока.

При решении поставленной задачи были использованы методы и программы анализа текстов, возможности ресурсов типа Яндекс-каталог, WordNet и систем метапоиска. Начальное наполнение ТС составили предложенные экспертами ключевые слова — несколько терминов, отражающих тему рубрики и/или являющихся синонимами ее названия.

2. Формирование тематической коллекции

Тематическая коллекция — множество текстов, представляющих соответствующую сайту деятельность и размеченных индексами подходящих им рубрик. Для формирования тематической коллекции авторы использовали два подхода: метапоиск в сети Интернет для сбора англоязычной коллекции и поиск по рубрикаторам ресурсов для русскоязычной коллекции.

2.1. Метапоиск

Сборка коллекции текстов для создания англоязычных ТС проводилась с помощью метапоиска. Тематическая коллекция формировалась из наиболее релевантных результатов поиска по заданному названию темы путем загрузки страниц, их очистки и извлечению основной текстовой информации [10]. У пользователя (эксперта) была возможность задавать размер текстовой коллекции, выбирать кодировку (utf-8, windows-1251), указывать, нужно ли оставлять теги и/или группировать содержание страниц в зависимости от принадлежности к определенным разделам меню. Запрос отправлялся к трем поисковым системам — Google, Bing, Yandex. Из объединенных результатов поиска удалялись дубликаты (по полному совпадению url) и тексты с релевантностью (вычисляемой как скалярное произведение между вектором термов запроса и вектором термов текста страницы) ниже заданного порога. На следующем этапе модуль разбора структуры страницы позволял выделить основной контент, меню, текущий раздел меню, список ссылок и пр. Результат предоставлялся пользователю в виде заархивированной коллекции текстов.

Собранная англоязычная коллекция содержала порядка 100 текстов на каждую тему.

2.2. Поиск по рубрикатору

Получение исходного массива русскоязычных текстов осуществлялось на базе каталога Яндекса (yaca.yandex.ru). Было сделано предположение о том, что базовые тематические слова, соответствующие содержанию сайтов Рунета, могут быть найдены в каталоге Яндекса, представляющим собой репрезентативную коллекцию аннотированных ссылок на сайты, вручную собранных редакторами каталога. Схема построения тематической коллекции по рубрикатору выглядит следующим образом.

1. Категории рубрикатора ПО сопоставлялись с рубриками Яндекс-каталога. Поиск проводился по наименованию или по ключевым словам рубрики.

2. Если какая-либо рубрика непосредственно не соответствовала рубрике Яндекс-каталога, поиск проводился по всему каталогу или по некоторой более широкой рубрике с помощью наименований или ключевых слов этой рубрики. Например, поиск по ключевому слову *секонд* в рамках Яндекс-рубрики "Покупки" позволяет найти описания сайтов для рубрики "Секонд-хенд".

3. Для каждой рубрики из выделенной категории или найденного подмножества описаний вынимались первые 20...30 аннотаций сайтов с заголовками и добавлялись в коллекцию.

В результате был получен массив размеченных текстов, который содержал исходные наименования и описания рубрик, а также порядка 3000 аннотаций из каталога Яндекс.

3. Формирование словаря

Используемый инструментарий для формирования предметных словарей [11] работает в режиме обучения ТС для извлечения терминов и тематического индексирования помещаемых в него терминов, а также в режиме классификации для тестирования определения рубрики-темы входного запроса.

При обработке коллекции, собранной с помощью метапоиска для создания англоязычной версии словаря, реализованы два варианта анализа полученного контента сайтов — полнотекстовый и частичный. Сформированная из содержания заданного раздела меню частичная текстовая коллекция, как правило, уже является списком терминов. Например, если речь идет о конкретных услугах типа "Car Rental Service", "Cargo Delivery Services", "IT and Computer Repair Services" и т. п., то практически все необходимые для описания темы термины сосредоточены в разделе меню "Services". Для каждого термина сохраняется информация о частоте встречаемости. Повторяемость терминов на разных сайтах свидетельствует об их преимущественном использовании в обиходе. Полученный таким образом словарь предъявляется на просмотр эксперту.

Полная текстовая коллекция также может обрабатываться только частично. Например, из нее извлекаются лишь слова и словосочетания, помеченные указанными в схеме разбора страницы специальными тегами разметки: заголовок, ссылка, элемент списка. Если частичный анализ подборки не дает удовлетворительных результатов, осуществляется вариант полнотекстовой обработки данных метапоиска.

3.1. Обучение словаря

Под обучением понимается процесс формирования словаря со статистическими показателями. Это словарь, элементам (терминам) которого ставится в соответствие статистическое распределение по классам (темам). Терминами словаря могут быть слова и словокомплексы (СК), встречающиеся в текстах определенной тематики с частотой выше заданного порога. В качестве СК могут выступать как словосочетания, найденные по морфолого-синтаксическим шаблонам (использовались для составления русскоязычных ТС), так и устойчивые терминоподобные *N*-граммы, извлеченные из текстовой коллекции (для наполнения англоязычных ТС).

Обучение происходит на основе обучающей тематической коллекции — массива текстов с исходной экспертной тематической разметкой.

Можно выделить следующие этапы обучения.

1. Морфологический анализ текста и сборка СК.

2. Для каждого термина *x*, обнаруженного в тексте, отнесенного к теме *t*, и самой темы *t* корректируются их статистические показатели в словаре. Если термина *x* в словаре не было, то он в него добавляется.

3. В результате обработки всех текстов обучающей выборки для каждого встреченного термина накапливается статистическая информация по каждой теме, в текстах которой данный термин встретился. При этом вычисляемая на основе этой информации весовая характеристика термина *x* для класса *t* будет отражать коэффициент вероятности отнесения текста, включающего термин *x*, к классу *t*.

4. Полученное достаточно грубое распределение весов анализируется лингвистом для уточнения и исключения случаев, ухудшающих распознавание. При этом используется конкорданс (множество всех контекстов), классификатор и выбираются случаи, получившие неправильную или недостаточно ясную классификацию.

Одной из значимых особенностей создаваемых словарей является наличие многословных терминов — словокомплексов. Именно такие термины составляют большую часть предметной терминологии, когда осуществляется работа с ограниченными предметными областями.

Первоначальный поиск СК в обучающей выборке проводился на основе правил согласования морфологических признаков в структуре словосочетания. Наиболее распространенными структурами здесь являются: сочетания существительного с прилагательным (*азиатская кухня*); существительного с существительным в косвенном падеже (*бригада отделочников, уборка территории, помощь инвалидам*); существительного с другим существительным в качестве приложения (*авиа кейтеринг, аудио формат, фитнес-центр*). Имеются также термины, состоящие из трех и более слов (*товар повседневного спроса, школа боевых искусств, аксессуары для чайной церемонии*). Результатом анализа является набор словосочетаний — кандидатов в термины-словокомплексы.

Словарная статья СК содержит следующие признаки:

- нормальная форма СК;
- список составляющих однословных терминов;
- правило, согласно которому образуется данный СК.

В процессе классификации поиск СК осуществляется, в первую очередь, на основе его лексического состава, для которого далее проводится проверка согласования, заданного правилом.

В случае отсутствия правил для первоначальной сборки СК применяется *N*-граммный анализ контента [12]. Текст представляется спектром *N*-словных цепочек X_N ($N = 1, ..., N_{max}$, где N_{max} — длина максимального повтора). Для каждой *N*-граммы фиксируется частота и позиции вхождения в текст. На просмотр эксперту выдаются устойчивые (функционирующие в большем числе разнообразных контекстов) *N*-граммы, упорядоченные по степени неравномерности распределения в тексте. Например, используется упорядочение по убыванию среднеквадратичного отклонения [13], что способствует преимущественному попаданию специальных терминов в начало списка, общих терминов — в середину списка, а общеязыковых сочетаний — в конец. Для ускорения работы эксперт ограничивается просмотром лишь фиксированной доли X_N из начала словаря (в рассматриваемом случае — 300...500 терминов).

3.2. Экспертная настройка словаря

Предварительная настройка предполагает: перенос в стоп-словарь так называемых стоп-слов (предлоги, союзы, частицы, междометия, некоторые местоимения); удаление ложных гипотез, таких как неверные омонимы, неверно предсказанные слова и некорректные СК и т. п.

Основная работа эксперта по настройке словаря ведется в следующих направлениях:

- удаление незначимых лексических единиц;
- корректировка тематических признаков (удаление, добавление);
- расширение словаря по тезаурусным связям (преимущественно синонимами, гипонимами и функционально связанными словами).

В русскоязычном варианте для имевшейся иерархии тем была предусмотрена общая рубрика "Универсальный". К ней относятся термины *сайт*, *интернет*, *веб-сайт*, *интернет-страница*, которые могут быть частью любого запроса, независимо от темы. Остальные рубрики для удобства сгруппированы в категории ("Услуги", "Розничные магазины", "Организации" и т. п.).

Термины, в особенности однословные, отличаются многозначностью и/или политематичностью. Так, для слова *тур* в словаре зафиксировано два значения:

"путешествие" (рубрики "Туристическое агентство" и "Блог о путешествиях") и "концертное турне" (рубрики "Веб-сайт группы", "Веб-сайт музыканта или исполнителя" и "Кинотеатр, театр или концертный зал"). Слово *аренда* характеризует все виды аренды недвижимости (три рубрики), проката автомобилей (две рубрики), а также услуги хостинг-провайдеров по сдаче в аренду выделенных серверов. Слово *кухня* характеризует как магазины мебели, так и предприятия питания, но *национальная кухня, высокая кухня* — только предприятия питания (в отличие от СК *немецкая кухня*, который не снимает неоднозначности).

Многие слова соотносятся с очень широким спектром тем (*работа, обслуживание, организация, компания, магазин, услуги, товар, салон, кабинет, фирма, центр* и т. п.). Тематический потенциал этих слов может соответствовать целой категории (*магазин* — "Розничные магазины") или выходить за рамки одной категории (*компания, фирма, центр*). Некоторые термины размечены характерными для них одной или несколькими темами, но не исключены в запросах произвольной тематики. В этом случае им дополнительно приписывается тема "Универсальный". Так трактуются лексические обозначения пола (*мужчина, мужской, женщина, женский*), возраста (*детский, молодежный, ребенок, взрослый*), национально-территориальной принадлежности (*национальный, Европа, европейский, немецкий* и т. п.) и др. Например, слово *женский* встретилось в описаниях, помеченных темами "Магазин одежды" и "Медицинский центр", но его тематический потенциал существенно шире. Прилагательные и существительные национально-территориальной принадлежности соотносятся с рубриками "Языковая школа", "Обучение за рубежом", "Ресторан", "Кафе или бар", "Кейтеринг-компания" и в то же время не исключены в запросах иных тематик: *итальянский ресторан, доставка блюд европейской кухни, обучение немецкому языку* vs *одежда из Европы, продажа немецкой мебели, клуб аргентинского танго.*

С точки зрения эксперта, словарь не отражает полную реальную картину политематичности терминов, она должна быть существенно скорректирована вручную. В процессе корректировки тематической разметки множество приписанных термину тематических признаков может модифицироваться. В частности, эксперт удаляет тематические ориентации, автоматически приписанные лексеме, входящей в состав СК: тема "Туристическое агентство" слова *дом,* приписанная в контексте *дом отдыха.* Слова, признанные экспертом тематически незначимыми, из словаря удаляются. В частности, удаляются незначимые лексемы, входящие в состав словокомплексов (*мягкий* в СК *мягкая мебель, убор* в СК *головной убор, сортовой* в СК *сортовой металл* и т. п.).

Считается, что все предлоги, местоимения, междометия, частицы, союзы сами по себе не несут какой-либо смысловой нагрузки и должны быть отнесены к стоп-словам. Ввиду прагматической специфики ряда рубрик, представляющих информацию, относящуюся к личному пространству пользователя (категории "Увлечения и хобби" и "Личное"), некоторые личные и притяжательные местоимения, такие как *мой, наш, свой, я*, рассматриваются как тематически значимые: *мой сайт-резюме*; *дневник нашей свадьбы*; *сайт обо мне, моих родных и друзьях.*

3.3. Расширение словаря по тезаурусным связям

Расширение словаря как часть настройки проводится с помощью доступных словарей тезаурусного типа. В отсутствие необходимых ресурсов для русского языка частичная синонимизация словаря может быть осуществлена вручную путем пополнения следующих групп тематической лексики:

- синонимичные обозначения соответствующих рубрикам видов деятельности;
- ролевые существительные и СК как обозначения типовых участников конкретных или более общих (категориальных) видов деятельности, а именно субъектов, бенефициантов и продуктов;
- гипонимы для обозначения направлений и продуктов деятельности;
- меронимы (для некоторых тем).

Так, тематическая лексика была дополнена однокоренными синонимами различных видов деятельности: *охрана — охранять, охранный, охранная деятельность; защита, защищать — защитить; лечить — вылечить, излечить, излечивать.* Кроме того, проведено расширение словаря за счет добавления имен участников различных видов деятельности: субъектов (*строитель, продавец, уборщик, застройщик, страховщик, страховая компания, охранник, охранное предприятие, торговый центр, универсальный магазин, корпорация, партнерство*), бенефициантов (*покупатель, арендатор, страхователь*), объектов (*пациент, жилое помещение*) и продуктов (*изделие металлопроката, ювелирная продукция*).

Гипонимы внесены в словарь для обозначения как направлений деятельности (*медицина — акушерство, андрология, стоматология, венерология, гинекология, косметология, маммология*), так и разновидностей продуктов деятельности (*изделия металлопроката — балка, швеллер, лист, труба, арматура, уголок, проволока*; *ювелирное изделие — украшение — серьга, браслет, колье, кольцо, подвеска*; *одежда — юбка, брюки, костюм, сорочка*).

Словари некоторых тем дополнены меронимами, например, *автомобиль — бак, бампер, двигатель, автостекло, кабина, карбюратор, колесо, кузов, шина.*

Для английского языка термины в ТС были дополнены из ресурса WordNet, снабженного программным доступом, понятиями, связанными с исходными отношениями часть-целое, частное-общее, а также "подобными" (*similar*) словами. Эксперт, просматривая всю совокупность понятий, извлеченных по связям, исключал тех претендентов на тематическую лексику, которые попали в список в силу характерных для языка явлений омонимии и полисемии. Задача упрощалась тем обстоятельством, что каждый термин в WordNet сопровождается определением или толкованием. Автоматизация работы эксперта состояла в извлечении из списка связанных понятий WordNet, прежде всего, слово-

сочетаний, содержащих слова, выявленные с помощью метапоиска. Полезными часто оказывались и однокоренные слова.

Число вариантов, выдаваемых WordNet по тезаурусным связям, варьировалось от нескольких десятков до нескольких сотен, доля терминов из разряда подобных (*similar*) была незначительна по объему. Проявление многозначности/омонимии, главных источников "шума", хорошо демонстрирует пример ключевых слов темы "Band/Musician Fan Website". Основное значение *fan* для данной темы — *фанат*. Другое значение слова *fan* — фен, поэтому все, что касается парикмахерского искусства, тоже попадало в список экспертного анализа (аналогично, *band* — "группа/банда", другое значение — "повязка"). Доля "шума" составила от 40...50 до 80...90 % в зависимости от рассматриваемой темы. Удалить этот "шум" автоматически не представляется возможным.

Для некоторых тем словарь, извлеченный из WordNet, хорошо коррелировал со словарем, построенным по релевантным теме сайтам. Таковы темы со значительной долей "устоявшейся" лексики, например, "Accounting Service", "Artist's Website", "Writer's Website". Слабо представлены в WordNet темы, в которых важно отразить максимально большое число меронимов, например, "Auto Parts Store", "Sports Team or Club Team", "Tanning Salon".

Объем слов и словосочетаний в пересечении ТС — показатель их различающей способности при использовании в классификационных целях. Построенный вариант англоязычных ТС отличался тем, что доля случаев с большим пересечением словарей (более 50 %) была не слишком велика и составила порядка 11 %. Эти словари близки по тематике: "Band Website", "Band/Musician Fan Website" и "Musician's/Performer's Website"; "Sports Team Fan Website" и "Sports Team or Club Team"; "Movie Blog" и "Movie Fan Website". Всего порядка 14 % слов и словосочетаний вошли более чем в один словарь. В среднем, доля слов в пересечении разных ТС не превышала 3 %. Например, слово *journal* входит в словари шести тем: "Generic Blog", "Library Website", "Movie Blog", "Music Blog", "Travel Blog", "Writer's Website"; *health* и *news* — в словари семи тем; *education* — восьми. Рекордсменами являются следующие слова: *blog* (присутствует в 15 ТС), *location* и *music* (14), *design* (13).

"Степень политематичности" термина (присутствие его в словарях разных тем) может быть учтена путем "взвешивания" слова. В англоязычной версии словаря в целях более успешной классификации запросов веса терминов в каждом ТС были вычислены согласно мере TFIDF [14]: очевидно, вес слова для данной темы тем выше, чем чаще оно встречается в текстах, и, соответственно, словаре этой темы и реже в составе словарей других тематик.

4. Апробация подходов и оценка словарей

Материал для апробации предлагаемых подходов содержал 108 тем, снабженных ключевыми словами на русском и английском языках.

4.1. Характеристики словарей

Суммарный объем полученных англоязычных ТС составил 11 263 слова и словосочетания и 6740 для русского языка. В среднем приходилось немногим более 100 слов на каждую тему для английского языка и 60 — для русского. Таким образом, объем словарей ключевых слов увеличился более чем на порядок. К "плохо" расширяемым темам для английского языка можно отнести такие темы, как "Carpenter/Framing Constructions", "Car Wash", "Bricklayer and Tiler", их словари содержали не более 50 понятий. Максимальное наполнение ТС наблюдалось для тем, так или иначе касающихся здоровья: "Pharmacy", "Health Care Provider", "Sports and Outdoor Equipment Store", объемы словарей составляли 200...400 терминов.

Существенно изменилась структура ТС по сравнению со словарями ключевых слов в плане наличия словосочетаний: они составили почти 46 % от всего объема англоязычных словарей и около 50 % русскоязычных. Большая часть словосочетаний — двухсловные (38,5 % в англоязычных словарях и 37,1 % в русскоязычных). Доля трехсловных словосочетаний — 5,6 % для английского языка и 11,3 % для русского. Доля сочетаний, состоящих из четырех слов в англоязычных и русскоязычных словарях, невелика (немногим более 1 %). Для русского языка более длинные сочетания не рассматривались. Для английского языка пяти-, шести- и семисловные сочетания составили около 1 %, это, например, *rings with side-ways bent cross section* (тема "Metal Forming Company"), *complete air handler cabinet biocide disinfection* ("Heating, Ventilation and Air Conditioning (HVAC) Services"), *Association for the Advancement of Retired Persons* ("Association/Foundation/League"), *preparing financial reports for a business* ("Accounting Service").

4.2. Классификация результатов тестирования

Корпус тестовых запросов, использованных при оценке ТС, включал по 3—5 запросов по каждой теме (всего 368 запросов на русском языке и 330 на английском языке). Запросы представляют собой микротексты — короткие или сверхкороткие фразы (3—4 слова), написанные на обладающем определенной спецификой фрагменте естественного языка. Такой фрагмент аналогичен языку запросов к традиционной поисковой системе. Его синтаксис сводится к номинативным (чаще) или глагольным (реже) конструкциям, которые нередко выстраиваются в уточняющие последовательности, предназначенные для описания интересующего пользователя вида деятельности. К таким описаниям относятся: дескрипция-заголовок (*Пункт проката автомобилей*; *Доставка пиццы на дом*; *Фан-клуб группы Золотой век*; *Сайт живописца Ольги Ивановой*; *Бригада плотников в Н-ске*; *Сайт ИП, продажа и изготовление под заказ мягкой мебели*); прямая или косвенная директива на базе глагольной конструкции (*Создать сайт мотосалона*; *Хочу создать сайт для продажи женской одежды через Интернет*).

Режим классификации позволяет по заданному запросу получить список тем, к которым он может быть отнесен. Соответствие запроса рубрике опре-

Таблица 1

Количественные характеристики результатов классификации

Характеристики	Тип результатов							Сумма
	1	2	3	4	5	6	7	
Число ответов	80	200	23	42	14	4	5	368
Доля ответов, %	21,7	54,4	6,3	11,4	3,8	1	1,4	100

делялось лишь фактом вхождения в текст терминов, приписанных этой рубрике составителями словаря. Результаты классификации для русского языка подразделяются на следующие семь типов:

1) единственный правильный ответ;

2) правильный ответ выигрывает с отрывом;

3) объективная неоднозначность (множественный альтернативный правильный ответ);

4) субъективная неоднозначность (несколько ответов с одним правильным);

5) правильный ответ есть в выдаче, но проигрывает;

6) правильного ответа нет в выдаче;

7) пустая выдача.

Количественные характеристики эксперимента приведены в табл. 1.

Типы 1—3 представляют варианты правильных ответов системы, типы 4—7 — неправильных. Неадекватные ответы системы могут быть связаны с рядом причин: пустые ответы (тип 7) классификатора связаны с наличием в запросе незнакомых словарю тематически значимых слов, в том числе иноязычных (*Сайт хостела*, *Spa салон)*, или аббревиатур (*ГБОУ СОШ № 15*). В некоторых случаях разрешение возникающих трудностей предполагает донастройку словаря.

При неоднозначности классификации выделяются случаи так называемой объективной неоднозначности (тип 3), когда все равнозначимые ответы (с одинаковой оценкой релевантности) признаются экспертами правильными (разумеется, относительно данного рубрикатора). Так, запрос *обучение иностранным языкам* справедливо относится классификатором к двум рубрикам: "Языковая школа" и "Обучение за рубежом". Наличие в запросе уточнения *за рубежом, в Германии* vs *в Москве* (*обучение иностранным языкам за рубежом/в Москве*) соответствующим образом уточняет и ответ классификатора. На примере рубрик, связанных с образовательной деятельностью, становится очевидной необходимость подключения к словарю простого тезауруса географических названий, который обеспечит выбор правильной рубрики с учетом территориальной принадлежности. Так, для запроса *"Учеба в Германии. Сайт Баварского образовательного центра"*, единственно правильным ответом является "Обучение за рубежом".

Еще один пример запроса типа 3: из трех рубрик, имеющих отношение к тематике дизайна, запрос *услуги дизайнера* объективно относится ко всем трем. Напротив, для запроса *услуги дизайнера одежды* релевантной может быть только рубрика "Веб-сайт дизайнера". Правильный ответ классификатора обеспечивается наличием соответствующей тематической ориентации у слова *одежда*.

Следующие примеры демонстрируют субъективную неоднозначность классификации (тип 4), при которой не все равнозначимые ответы признаются правильными.

Словокомплекс *продажа одежды* имеет ориентации на две темы: "Магазин одежды" и "Секонд-хенд". Соответствующий запрос *продажа одежды* классификатор относит к обеим темам, хотя, по мнению эксперта, этот запрос подразумевает обычный "Магазин одежды". В то же время запрос с уточнением *продажа одежды секонд-хенд* приводит к однозначной теме "Секонд-хенд".

Подобная неоднозначность характеризует и ряд рубрик, соответствующих сайтам персон или организаций, при наличии параллельных рубрик для сайтов поклонников персон или организаций, например "Спортивная команда или клуб" и "Веб-сайт поклонников спортивной команды", не различаемых запросами типа *сайт волейбольной команды "Уралочка"*. Выходом в этой ситуации является "отрицательная" ориентация лексической единицы на темы, которым она противоречит (отрицание темы). Так, наличие в запросе лексемы *фанат, поклонник* и т. п. отрицает тему "Спортивная команда или клуб".

В ряде случаев проблема классификации возникает в силу присутствия в запросе имен собственных, имеющих тематически важные омонимы: *Фан-клуб группы "Золотой век"* (*золотой*), *Неофициальный сайт фанатов Романа Иванов* (*роман*), *Сайт автошколы "Лидер"* (*лидер*). Распознавание собственных имен в запросе (при наличии кавычек, строчных букв) может улучшить ситуацию.

При тестировании англоязычных словарей рассматривались три градации качества классификации:

1) первое место в векторе претендентов на заданную тематику ("хорошее" распознавание);

2) вхождение верного результата классификации в пятерку претендентов, начиная со второго места в результирующем векторе ("удовлетворительная" классификация);

3) "плохая" классификация — отсутствие верного ответа в первой пятерке.

Результаты тестирования ТС представлены в табл. 2.

Для некоторых тем классификация всех запросов была проведена верно, например, "Car Wash", "Courier Delivery Service", "Hosting Provider". В этих случаях

Таблица 2

Число и доля ответов для трех градаций качества классификации

Число и доля ответов	Тип результатов			Сумма
	1	2	3	
Без учета весов	125 (37,9 %)	105 (31,8 %)	100 (30,3 %)	330 (100 %)
С учетом весов	151 (45,8 %)	96 (29,1 %)	83 (25,1 %)	330 (100 %)

запросы содержали термины с максимальным весом в данной теме.

Причины неудовлетворительной классификации для английского языка, в отличие от русскоязычной версии, отчасти связаны с отсутствием морфологического и синтаксического анализа запроса. Нормализация слов запроса не проводилась, как и выделение словосочетаний. Вес запроса определялся суммой весов входящих в него слов (вес служебных слов считался равным нулю). Очевидно, что при совпадении терминов-словосочетаний в запросе и словаре, вес термина для идентификации темы должен оцениваться с учетом факта многословности и быть выше суммы весов входящих в него слов.

Далее представлены другие факторы неудовлетворительной классификации запросов (отчасти сходные с русскоязычной версией).

- Неоднозначность запроса часто возникает в случае короткой формулировки, в которой содержатся только политематические (универсальные) слова. Например, запрос *service* предположительно должен был идентифицировать тему "Business or Company", но это слово содержится в названии 16 тем (например, "Accounting Service", "Cargo Delivery Services", "Courier Delivery Service"). Искомая тема в векторе претендентов заняла лишь седьмое место, вес термина *service* в ТС "Business or Company" оказался ниже, чем в словарях других тематик.
- Запрос относится к периферии темы. В этом случае веса входящих в запрос терминов объективно низки в ТС. Например, сложно отнести запрос *page of young mothers* к теме "Birth Announcement".
- Неполнота словаря: отсутствие термина или варианта его написания. "Пустой" результат классификации (нет темы в векторе претендентов) получен для 11 англоязычных запросов (3,3 %). Специальные термины, например, *accountant* (тема "Accounting Service"), *bookcrossing* ("Library Website"), *bard* ("Musician's/Performer's Website"), *audiophile* ("Music Blog") не были зафиксированы в словарях, равно как и термин *gymnasium*, употребляемый в некоторых странах Европы и практически неупотребляемый в Англии и Америке ("Primary or High School"). Примером неучтенного варианта написания может служить термин *spa-studio* ("Spa or Sauna").
- Наличие близких по содержанию тем, например, "Car Repair Service" и "Appliance Repair and Service Center"; "Dry Cleaning and Laundry Services" и "Housekeeping and Cleaning Services". В этом случае часто в запросе преобладают слова, имеющие бóльший вес в других ТС. В этом случае, как правило, в векторе, отражающем результаты классификации, первые два места занимают близкие темы (часто, первое — ошибочный результат, второе — правильный). Например, суммарный вес запроса *catalogue of apartments and houses for rent, agency website* для темы "House or Apartment Rental" (как предполагалось, правильно идентифицируемой), равный 0,28734, оказался бóльшим для темы "Real Estate Agency" — 0,53502.

Очевидно, что в условиях классификации микротекстов (запросов) необходимо проводить их синтаксический анализ, а именно выделение словосочетаний, при наличии нескольких — выделение главного, несущего основную смысловую нагрузку. Корректировка весов слов согласно статистике правильной и неправильной классификации запросов также могла бы изменить ситуацию в лучшую сторону.

Заключение

Предложенные в статье методы автоматизации позволяют обучать и настраивать тематические словари, предназначенные для идентификации темы запроса. Методы основаны на программных технологиях, разработанных в ИСИ СО РАН: системе Klan, которая используется для создания предметных словарей [11], и метапоисковой системе, позволяющей формировать коллекции текстов на основе поискового запроса [10]. Апробация словарей, созданных экспертами для русского и английского языков, свидетельствует об эффективности предлагаемой методики. Также очевидна применимость предложенных методов для различных предметных областей.

Дальнейшее направление исследований связано с пополнением тематических словарей, углублением анализа микротекстов-запросов, приведением весов терминов в соответствие со статистикой запросов, присоединением словарей имен собственных (географических названий, имен персон, названий организаций).

Работа выполнена в Новосибирском государственном университете при финансовой поддержке Министерства образования и науки Российской Федерации (договор № 02.G25.31.0054).

Список литературы

1. **Поляков П. Ю., Плешко В. В., Ермаков А. Е.** RCO на РОМИП 2009 // Труды РОМИП 2009: семинар в рамках всерос. науч. конф. RCDL'2009. СПб.: НУ ЦСИ. 2009. С. 122—134.

2. **Агеев М. С., Добров Б. В., Лукашевич Н. В.** Выбор факторов для классификации веб-страниц и веб-сайтов // Труды РОМИП 2010: семинар в рамках всерос. науч. конф. RCDL'2010. Казань: Изд-во Казанского ун-та, 2010. С. 28—39.

3. **Лукашевич Н. В.** Тезаурусы в задачах информационного поиска. М.: Изд-во МГУ, 2011. 512 с.

4. **Агеев М. С., Добров Б. В., Лукашевич Н. В.** Поддержка системы автоматического рубрицирования для сложных задач классификации текстов // RCDL'2004 Электронные библиотеки: перспективные методы и технологии, электронные коллекции: шестая всерос. науч. конф. Пущино, 2004. URL: http://www.cir.ru/docs/ips/publications/2004_rcdl_rubric.pdf (дата обращения 05.08.2014).

5. **Дунаев Е. В., Шелестов А. А.** Автоматическая рубрикация web-страниц в интернет-каталоге с иерархической структурой // Сборник "Интернет-математика 2005". М.: ООО "Яндекс", 2005. С. 382—398.

6. **Агеев М. С., Добров Б. В., Красильников П. В. и др.** УИС РОССИЯ в РОМИП 2007—2008: поиск и классификация // Труды РОМИП: семинар в рамках всерос. науч. конф. RCDL'2008. СПб.: НУ ЦСИ, 2008. С. 44—58.

7. **Белов А. А., Волович М. М.** Автоматическое распознавание тематики сверхкоротких текстов // Труды междунар. конф. "Компьютерная лингвистика и интеллектуальные технологии". Наро-Фоминск, 2007. URL: http://www.dialog-21.ru/digests/dialog2007/materials/html/05.htm (дата обращения 23.09.2014).

8. **Агеев М. С.** Методы автоматической рубрикации текстов, основанные на машинном обучении и знаниях экспертов: дис. ... канд. физ.-мат. наук. М., 2004. 136 с.

9. **WordNet.** URL: http://wordnet.princeton.edu/wordnet/ (дата обращения 12.09.2014).

10. **Ахмадеева И. Р., Загорулько Ю. А., Саломатина Н. В. и др.** Подход к формированию тематических коллекций текстов

на основе web-публикаций // Вестник НГУ. Серия: Информационные технологии. 2013. Т. 11, Вып. 4. С. 59—70.

11. **Сидорова Е. А.** Подход к построению предметных словарей по корпусу текстов // Труды междунар. конф. "Корпусная лингвистика — 2008". СПб.: СПбУ, Факультет филол. и искусств, 2008. С. 365—372.

12. **Гусев В. Д., Саломатина Н. В.** L-граммное представление текстов на естественном языке и его возможности // Материалы всерос. научн. конф. "Квантитативная лингвистика: исследования и модели" (КЛИМ—2005). Новосибирск: Изд-во НГПУ, 2005. С. 256—270.

13. **Остапенко В. Е.** Выделение и классификация терминов с помощью элементарных квантитативных моделей // НТИ. Сер. 2. 1989. № 11. С. 24—28.

14. **Солтон Дж.** Динамические библиотечно-поисковые системы. М.: Мир, 1979.

I. S. Kononenko[1, 2], Researcher, e-mail: irina_k@cn.ru,
N. V. Salomatina[1, 3], Researcher, e-mail: nataly@math.nsc.ru,
E. A. Sidorova[1, 2], Senior Researcher, e-mail: lena@iis.nsk.su
[1] Novosibirsk State University,
[2] A. P. Ershov Institute of Informatics Systems, SB RAS, Novosibirsk,
[3] Sobolev Institute of Mathematics, SB RAS, Novosibirsk

Experiment of Constructing Subject Dictionaries for Categorization of Short Descriptions of Web-Sites

In this study original procedures and techniques for automatized construction of multilingual subject dictionaries are proposed that rely on tools for automatic compiling and statistical processing of subject specific text collections as well as expert knowledge represented in thesauri and catalogue-like resources. Subject dictionaries are intended for use in the process of automatic categorization of short descriptions of user Internet activities (user queries). An experience of constructing Russian and English subject dictionaries is described and exemplified. Special questions outlined in the paper concern selection of domain texts that contain important subject specific terms, automatic lexicon extraction (including single- and multi-word terms), and lexicon extension on base of thesaural relations. Experimental results of dictionary-based query categorization tested across 108 domains for the two languages are discussed.

Keywords: *subject dictionary, subject specific text collection, metasearch, thesaurus, catalogue, query categorization, Internet activity*

References

1. **Poljakov P. Ju., Pleshko V. V., Ermakov A. E.** RCO na ROMIP 2009. *Trudy ROMIP 2009: seminar v ramkah vseross. nauch. konf. RCDL'2009.* SPb: NU CSI. 2009. P. 122—134.

2. **Ageev M. S., Dobrov B. V., Lukashevich N. V.** Vybor faktorov dlja klassifikacii veb-stranic i veb-sajtov. *Trudy ROMIP 2010: seminar v ramkah vseross. nauch. konf. RCDL'2010.* Kazan': Izd-vo Kazanskogo un-ta, 2010. P. 28—39.

3. **Lukashevich N. V.** *Tezaurusy v zadachah informacionnogo poiska.* M.: Izd-vo MGU, 2011. 512 p.

4. **Ageev M. S., Dobrov B. V., Lukashevich N. V.** Podderzhka sistemy avtomaticheskogo rubricirovanija dlja slozhnyh zadach klassifikacii tekstov. *RCDL'2004. Jelektronnye biblioteki: perspektivnye metody i tehnologii, jelektronnye kollekcii: shestaja vseross. nauch. konf.* Pushhino, 2004. URL: http://www.cir.ru/docs/ips/publications/2004_rcdl_rubric.pdf (date of access 05.08.2014).

5. **Dunaev E. V., Shelestov A. A.** Avtomaticheskaja rubrikacija web-stranic v internet-kataloge s ierarhicheskoj strukturoj. *Sbornik "Internet-matematika 2005".* M.: OOO Yandex, 2005. P. 382—398.

6. **Ageev M. S., Dobrov B. V., Krasil'nikov P. V. et al.** UIS ROSSIJa v ROMIP 2007—2008: poisk i klassifikacija. *Trudy ROMIP: seminar v ramkah vseross. nauch. konf. RCDL'2008.* SPb.: NU CSI, 2008. P.44—58.

7. **Belov A. A., Volovich M. M.** Avtomaticheskoe raspoznavanie tematiki sverhkorotkih tekstov. *Trudy mezhdunar. konf. "Komp'juternaja lingvistika i intellektual'nye tehnologii".* Naro-Fominsk, 2007. URL: http://www.dialog-21.ru/digests/dialog2007/materials/html/05.htm (date of access 23.09.2014).

8. **Ageev M. S.** Metody avtomaticheskoj rubrikacii tekstov, osnovannye na mashinnom obuchenii i znanijah jekspertov: dis. ... kand. fiz.-mat. nauk. M.: MGU, 2004. 136 p.

9. **WordNet.** URL: http://wordnet.princeton.edu/wordnet/ (date of access: 12.09.2014).

10. **Ahmadeeva I. R., Zagorul'ko Ju. A., Salomatina N. V. et al.** Podhod k formirovaniju tematicheskih kollekcij tekstov na osnove web-publikacij. *Vestnik NGU.* Serija: *Informacionnye tehnologii.* 2013. Vol. 11, Is. 4. P. 59—70.

11. **Sidorova E. A.** Podhod k postroeniju predmetnyh slovarej po korpusu tekstov. *Trudy mezhdunar. konf. "Korpusnaja lingvistika — 2008".* SPb.: SPbU, Fakul'tet filol. i iskusstv, 2008. P. 365—372.

12. **Gusev V. D., Salomatina N. V.** L-grammnoe predstavlenie tekstov na estestvennom jazyke i ego vozmozhnosti. *Materialy vseros. nauchn. konf. "Kvantitativnaja lingvistika: issledovanija i modeli"* (KLIM—2005). Novosibirsk: Izd-vo NGPU, 2005. P. 256—270.

13. **Ostapenko V. E.** Vydelenie i klassifikacija terminov s pomoshh'ju jelementarnyh kvantitativnyh modelej. *NTI.* Ser. 2. 1989. N. 11. P. 24—28.

14. **Solton J.** *Dinamicheskie bibliotechno-poiskovye sistemy.* M.: Mir, 1979.

ООО "Издательство "Новые технологии". 107076, Москва, Стромынский пер., 4
Технический редактор *Е. М. Патрушева.* Корректор *Е. В. Комиссарова*

Сдано в набор 06.11.2014 г. Подписано в печать 17.12.2014 г. Формат 60×88 1/8. Заказ PI115.
Цена свободная.

Оригинал-макет ООО "Адвансед солюшнз". Отпечатано в ООО "Адвансед солюшнз".
119071, г. Москва, Ленинский пр-т, д. 19, стр. 1. Сайт: **www.aov.ru**