ISSN 2220-3397

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Программная инженерия

ПрИн
3
2022
Том 13

Рисунки к статье **А. Е. Близнюка, В. А. Жмудя, М. В. Трубина, В. Г. Трубина**
«ПРОГРАММИРОВАНИЕ МИКРОКОНТРОЛЛЕРОВ STM32F10x
С ПОМОЩЬЮ ВСТРОЕННОГО ЗАГРУЗЧИКА ПО USART»

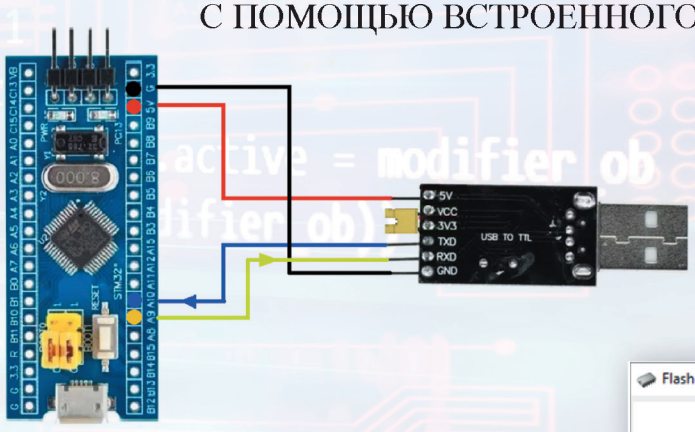


Рис. 12. Схема подключения преобразователя к отладочной плате

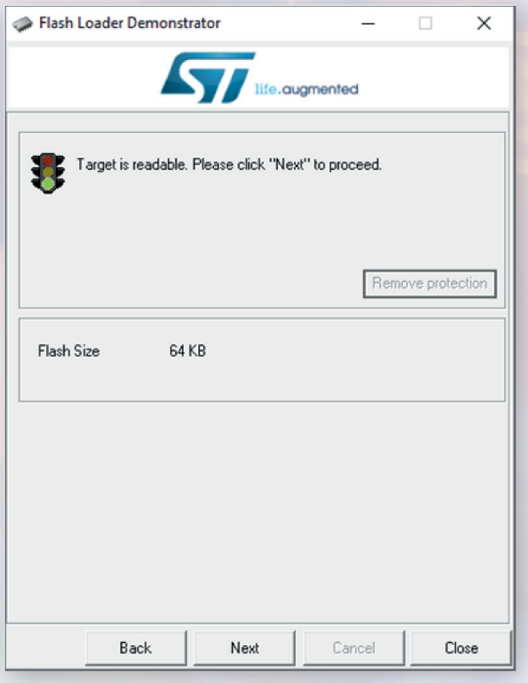


Рис. 14. Уведомление об успешном подключении

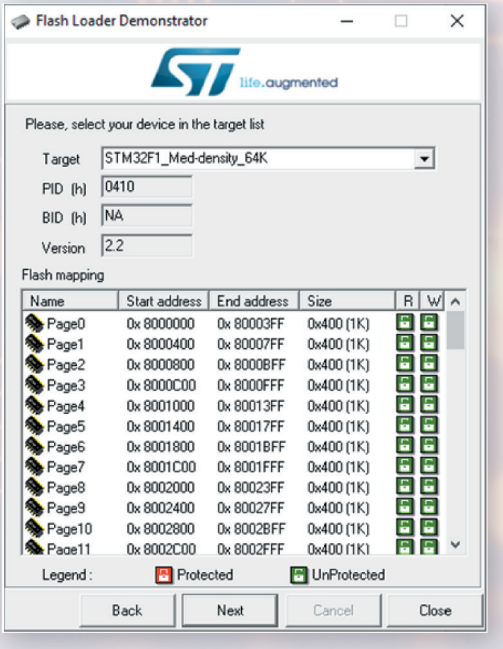


Рис. 15. Информация о микроконтроллере в программе «Flash Loader Demonstrator»

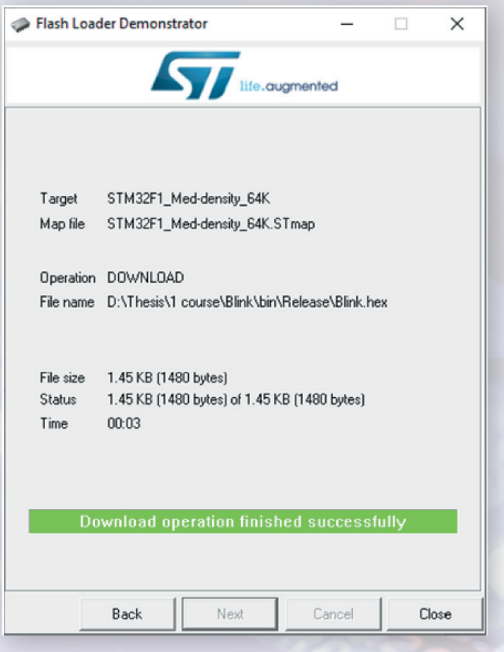


Рис. 17. Удачное завершение процесса программирования микроконтроллера

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Издается с сентября 2010 г. DOI 10.17587/issn.2220-3397 ISSN 2220-3397

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

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 подписных агентствах (индекс по Объединенному каталогу "Пресса России" — **22765**) или непосредственно в редакции (для юридических лиц).
Тел.: (499) 270-16-52.
Http://novtex.ru/prin/rus E-mail: prin@novtex.ru
Журнал включен в систему Российского индекса научного цитирования и базу данных RSCI на платформе Web of Science.
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

Vol. 13
N 3
2022

Published since September 2010 **DOI 10.17587/issn.2220-3397** **ISSN 2220-3397**

CONTENTS

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

УДК 004.75 DOI: 10.17587/prin.13.107-118

Ю. С. Затуливетер, канд. техн. наук, ст. науч. сотр., вед. науч. сотр., zvt@ipu.rssi.ru,
Е. А. Фищенко, канд. техн. наук, вед. науч. сотр., elena.fish@mail.ru,
Институт проблем управления им. В. А. Трапезникова Российской академии наук, Москва

Синергия цифровой универсальности глобальной компьютерной среды

В качестве объекта исследования рассмотрена глобальная компьютерная среда (ГКС) в целом. Дан анализ общесистемных аспектов осуществления посредством ГКС цифровой трансформации социотехносферы. Представлены факторы деструктивного влияния фундаментальных системотехнических закономерностей стихийного роста ГКС на функционирование и развитие социотехносферы. Особое внимание уделено изучению особенностей проявления положительного синергетического сетевого эффекта, выраженного законом Меткалфа, в условиях изначально открытой внутрисистемной разнородности сетевых ресурсов существующей ГКС. Установлено, что с увеличением размеров больших распределенных систем, реализуемых в ГКС, фактор разнородности сетевых ресурсов становится источником отрицательной (системно-деструктивной) сетевой синергии, которая обесценивает положительную синергию закона Меткалфа. На примере экосистем, реализуемых в ГКС посредством облачных систем и технологий, показано, что в условиях изначально открытой разнородности сетевых ресурсов дальнейшее наращивание размеров больших распределенных систем ведет к непреодолимому росту сложности системно-функциональной интеграции сетевых ресурсов и неконтролируемому снижению устойчивости социотехносферы. Представлены принципы концептуального реинжиниринга ГКС, направленного на устранение причин возникновения системной разнородности и бесшовное/кибербезопасное распространение алгоритмической универсальности, замкнутой во внутрикомпьютерных ресурсах, на любое сколь угодно большое подмножество компьютеров ГКС. Такой реинжиниринг позволит обеспечить нейтрализацию отрицательной синергии и максимизацию положительной синергии эффекта Меткалфа в больших распределенных системах без ограничений на их размеры.

Ключевые слова: *компьютерная среда, закономерности развития, социотехносфера, цифровая трансформация, положительная сетевая синергия, закон Меткалфа, разнородность сетевых ресурсов, отрицательная синергия, облачные системы, концептуальный реинжиниринг, модель глобально универсальных распределенных вычислений, единое алгоритмическое пространство, бесшовное программирование, кибербезопасность*

Введение

Глобальная компьютерная среда (ГКС) в ходе стихийно протекающей цифровой трансформации социотехносферы служит всеохватывающей универсальной основой для раскрытия системообразующего потенциала положительной сетевой синергии, выраженной законом Меткалфа: "Польза от применения сетей пропорциональна квадрату числа сетевых узлов" [1].

В условиях изначально открытой разнородности сетевых ресурсов наращивание масштабов влияния цифровой трансформации на все разнообразие малых и больших социосистем осуществляется посредством наращивания числа и видов глобально распределенных облачных цифровых экосистем[1]. К таким экосистемам относятся коммуникационные и маркетинговые, технические, финансовые и экономические системы, системы госуправления, управления производственными и бизнес-процессами и др.

В ходе стремительного, но де-факто системно-несбалансированного, роста ГКС цифровая трансформация осуществляется с опорой на слабо формализованные и трудно контролируемые системотехнические возможности крайне разнородной[2] ГКС. При этом беспрецедентное по масштабам и темпам влияние такой компьютерной среды в качестве абсолютно новой — глобально сильносвязной и всеохватывающей — цифровой информационной инфраструктуры человечества кардинально и вместе с тем хаотично меняет кибернетические свойства не только раз-

[1] Этим термином обозначаются большие распределенные системы сетевой цифровизации бизнес-моделей различной направленности с облачным воплощением "центродоминирующих" сетевых архитектур "клиент-сервер".

[2] Разнородность сетевых ресурсов выражается растущим разнообразием трудно сопрягаемых цифровых форм представления данных и программ, аппаратных, программных и информационных платформ.

личных социосистем[1], но и мировой социосистемы в целом [2].

Во многом стихийные, стратегически не координируемые системотехнические процессы цифровой глобализации мирового информационного пространства посредством существующей ГКС (с изначально открытой разнородностью сетевых ресурсов и потому не обладающей общим системно-целостным свойством функциональной полноты [2]) вышли далеко за рамки исторического опыта и возможностей имеющегося (доцифрового) инструментария управления устойчивым развитием социосистем.

Внутренние системотехнические процессы системно несбалансированного развития ГКС де-факто стали доминирующим фактором неконтролируемого роста проявлений отрицательной сетевой синергии "побочных" сетевых эффектов и их тотального воздействия на функционирование/развитие социосистем и мировой социосистемы в целом.

Прямым следствием такого воздействия является экспоненциальный рост в ГКС потоков и объемов глобально распределенной информации [3]. Глобальные, быстро растущие потоки информации в отчетах Всемирного экономического форума (*World Economic Forum*) и Института будущего (*Institute for the Future*) безальтернативно рассматриваются только как новые возможности для развития сложившихся [3] и появления перспективных [4] массовых цифровых секторов мировой экономики.

Однако в этих и других исследованиях маркетологической направленности не учитываются фундаментальные факторы проявления отрицательной синергии "побочных" сетевых эффектов, которые возникают вследствие глубинных внутрисистемных противоречий и системотехнических дисбалансов стихийного развития разнородной ГКС [2, 5].

Один из таких "побочных" эффектов глобально проявляется уже три десятилетия (с момента появления WWW) в виде экспоненциально растущих потоков/объемов слабо формализованной цифровой информации, которая мало пригодна для глубокой и полномасштабной алгоритмической переработки в совокупных ресурсах ГКС. Все большая часть такой информации, возникающей в ходе функционирования/развития больших и малых социосистем, остается непереработанной в целях управления их устойчивым развитием. В глобально сильносвязном информационном пространстве ГКС с новой, исторически беспрецедентной, метрикой "все влияет на все и сразу" [5] это оборачивается[2] прогрессирующим снижением качества процессов управления мировой экономики [6—8].

В результате нескольких десятилетий стихийной цифровизации мировая социосистема демонстрирует неспособность согласованно и своевременно отвечать на глобальные вызовы "побочных" сетевых эффектов системно не сбалансированной компьютерной среды, а также обеспечивать общедоступную эффективность согласованного кризисного управления в условиях вирусной пандемии.

На фоне критического перепроизводства недостаточно полно перерабатываемой цифровой информации новейшие вызовы негативно воздействуют на различные социосистемы и мировую социосистему в целом [2, 5]. Такие вызовы, остающиеся без адекватных ответов, ведут к деградации принципов и оснований социального прогресса, сложившихся в доцифровые времена.

Цифровая трансформация как стратегия формирования нового устойчивого цифрового мироустройства требует принципиально новых методов и моделей, а также компьютерно-сетевых средств управления устойчивым развитием. Их реализация возможна только на основе неограниченно растущего системообразующего потенциала обновленной в своих концептуальных основах ГКС.

Стратегия концептуального реинжиниринга ГКС — это устранение фундаментальных причин непрерывного воспроизводства изначально открытой системной разнородности и полномасштабная системно-целостная интеграция глобально распределенных функциональных, вычислительных и информационных ресурсов в целях осуществления всего разнообразия процессов управления устойчивым развитием социотехносферы в условиях глобальной информационной сильносвязности.

В настоящей статье изложен концептуальный подход, направленный на системно-целостное рассмотрение фундаментальных закономерностей развития ГКС и выявление причин внутрисистемных дисбалансов в развитии ГКС. Показаны пути устранения этих причин путем реинжиниринга основополагающих принципов формирования и развития глобальных компьютерных сетей. Такие пути открываются, как показано далее, на основе математического обобщения принципов алгоритмической универсальности цифровых компьютеров, канонизированных в классической модели универсальных цифровых компьютеров Дж. фон Неймана.

1. Внутрисистемные дисбалансы развития глобальной компьютерной среды как ключевой фактор социальной дестабилизации

Цифровые экосистемы, реализуемые в разнородной ГКС, проникают практически во все сферы повседневной жизни. При этом, как отмечается в работе [9], их современное развитие в первую очередь обеспечивает опережающий рост "разрозненных данных", не являющихся "систематически организованной информацией", так как данные во многих

[1] Социосистемы — структурированные сообщества людей, объединяемые общими устремлениями/целями к совместному функционированию и развитию, а также выбором путей их воплощения. Основу социосистем составляют врожденные универсальные способности Homo Sapience к индивидуальному и коллективному восприятию, абстрагированию и преобразованию информации, что составляет основу уникального свойства информационной универсальности человека, не имеющего природных аналогов.

[2] В интенсивных потоках множественных локальных событий спонтанно возникают и быстро распространяются непредсказуемые цепочки неконтролируемых причинно-следственных связей, которые все чаще вызывают лавины деструктивных процессов на глобальных уровнях [5].

случаях оторваны от контекста. Поэтому "огромное количество данных не может быть своевременно обобщено и преобразовано в полезную для общества информацию, из которой становится возможным получение систематизированных знаний". Во взаимодействии между людьми и компьютерами, а также компьютеров между собой не хватает "универсальной совместимости того, что передается, и того, что исполняется". Устранение указанных проблем критически важно для развития ГКС, которая должна наращивать "онтологически и семантически осмысленные информационные коммуникации вместо обезличенных".

Цифровая трансформация осуществляет переход к новым — более гибким и эффективным — бизнес-моделям создания, модернизации и расширения сфер влияния социальных и техногенных систем [10]. Такие модели цифровизации направлены на адаптацию больших систем к быстрым и масштабным изменениям глобального информационного контекста, в котором они функционируют и развиваются.

Продвижение цифровой трансформации в решающей степени определяется и вместе с тем ограничивается доступными системообразующими возможностями ГКС, а также ориентированными на эти возможности моделями управления устойчивым развитием сетевых распределенных систем.

В работе показано, что главным препятствием на путях развития таких систем становятся внутрикомпьютерные и внутрисистемные дисбалансы стихийного роста ГКС.

В стихийном, системно несбалансированном развитии опережающий количественный рост размеров существующей ГКС не сопровождается качественным совершенствованием ее общесистемных свойств и возможностей.

Фундаментальные внутрисистемные дисбалансы развития ГКС проявляют себя следующими факторами [2, 5, 10]:

- непрерывное воспроизводство изначально открытой разнородности на всех системных уровнях, начиная с аппаратного;
- ГКС в целом, составленная из универсальных компьютеров, связанных сетями, не обладает системно-целостным свойством функциональной полноты (бесшовной универсальной программируемости), которым обладает каждый компьютер в ее сетевых узлах;
- неконтролируемый рост экспоненциальных потоков и объемов разнородной, слабо формализованной информации, мало пригодной для алгоритмической обработки, во много раз превышает совокупные социально-субъектные и технические возможности ее переработки в целях управления устойчивым развитием социотехносферы;
- с увеличением масштабов применения ГКС проблемы дальнейшего наращивания размеров больших систем и обеспечения их кибербезопасности становятся практически неразрешимыми вследствие неприемлемого роста затрат на практическое решение комбинаторно-сложных задач системно-функциональной интеграции сетевых ресурсов с изначально открытой разнородностью.

На внешних относительно ГКС — социальных — уровнях перечисленные факторы проявления внутренних дисбалансов ГКС становятся причиной прогрессирующего снижения устойчивости социосистем и мировой экономики, которое происходит на фоне ускоряющейся экспансии крайне разнородной компьютерной среды.

Гипертекстовое информационное пространство WWW породило беспрецедентный феномен глобальной информационной сильносвязности, когда "все влияет на все и сразу" [5]. Глобальная информационная сильносвязность радикально меняет кибернетические свойства социосистем. В результате мировая социотехносфера становится все более нестабильной. В условиях глобальной информационной сильносвязности небольшие случайные причины могут вызвать глобальные неконтролируемые лавинообразные реакции деструктивного характера. Эти и другие последствия априори необъявленных "побочных" эффектов цифровой глобализации выдвигают принципиально новые требования к кибернетическим моделям и методам, а также к компьютерно-сетевым средствам управления устойчивым развитием социотехносферы [5, 6—8, 10].

Отсутствие у существующей ГКС в целом общесистемного свойства функциональной полноты (бесшовной универсальной программируемости) привело к экспоненциальному росту потоков и объемов слабо формализованной, разнородной по формам представления глобально распределенной информации. Такое развитие компьютерной среды с непредсказуемыми последствиями для мирового рынка оборачивается опережающим ростом кризиса перепроизводства слабо формализованной глобально распределенной информации. По причине малой пригодности для глубокой и полномасштабной алгоритмической переработки в ГКС в целях управления устойчивым развитием социосистем такая информация становится "непосильной ношей" для мирового сообщества.

Все это заставляет думать, что стихийное, внутрисистемно несбалансированное системотехническое развитие ГКС становится новейшим деструктивным фактором мирового влияния и одной из главных новейших причин снижения устойчивости социосистем. В результате системно несогласованных подходов к цифровой глобализации своевременно не перерабатываемая информация ведет к росту глобального "информационного шума". Такой шум неизбежно снижает качество принятия решений на всех уровнях управления процессами функционирования и развития социосистем.

Кризис перепроизводства информации становится проявлением информационного коллапса и универсально-корневой причиной нарастания финансово-экономических, геополитических, социальных и других кризисов, ведущих к снижению устойчивости социосистем и мировой социосистемы в целом [2, 5, 6—8, 10]. Факты критического снижения качества управления устойчивым развитием подтверждаются перманентной чередой новейших (с цифровым генезисом) мировых финансово-экономических кризисов [6—8]. Эта серия глобальных финансово-экономических потрясений в явочном

порядке и весьма "громко"[1] стартовала в 2000 г. с интернетовского "пузыря доткомов" [11] и уже два десятилетия продолжает неотвратимо расширять масштабы своего деструктивного влияния.

В 2010-е гг. новейшие "цифрогенные" кризисы, резистентные к известным доцифровым финансово-экономическим и политическим методам купирования, стали трансформироваться в предпосылки общесистемного кризиса современного мироустройства. Все больше переставая подчиняться известным методам регулирования, кризисные проявления сопровождаются нарастанием санкций, торговых и "гибридных" войн всех со всеми.

Результатом неконтролируемого развития процессов глобальной цифровизации является лавинообразное нарастание противоречий между традиционными (доцифровыми) и новейшими (цифровыми) методами, моделями и средствами управления функционированием/развитием социосистем. Такие противоречия проявляется в ускоряющемся росте социально-экономической нестабильности мировой социосистемы.

Деградация традиционных — доцифровых — структур и институтов управления под воздействием системно несогласованной экспансии цифровых технологий во многом происходит в силу внутрисистемных диспропорций развития ГКС [2, 5, 6—8, 10].

В своей существующей системно несбалансированной системотехнической архитектуре ГКС разрушает доцифровые — общесистемные балансы и институты управления устойчивым развитием. При этом ГКС, оставаясь крайне разнородной, не обладает функциональной полнотой и системообразующим потенциалом для создания новых, системно увязанных и сбалансированных цифровых систем и институтов своевременной и полномасштабной переработки неконтролируемых потоков/объемов информации в целях устойчивого развития.

Процессы деградации доцифровых структур и институтов управления устойчивым развитием приближаются к точке невозврата. Сложившаяся ситуация требует принципиального, кибернетически обоснованного реинжиниринга базовых внутрисистемных принципов формирования и развития ГКС [2, 5, 10]. Такой реинжиниринг направлен прежде всего на устранение внутрисистемных диспропорций развития ГКС. Он должен обеспечить эволюционную (с сохранением накопленных достижений) трансформацию ГКС в целом в универсально программируемую и кибербезопасную цифровую среду глобально распределенного управления социально значимыми преобразованиями на основе кибернетически обоснованных моделей устойчивого и безопасного развития социотехносферы.

2. Движущие силы массовой цифровизации

Быстрый рост масштабов влияния ГКС основан на синергии мультипликативного комбинирования многих одновременно действующих типов фундаментальных и других глобально/долгосрочно действующих движущих сил производства—потребления информации. Среди них выделим следующие:

- природная (доцифровая) информационная универсальность Homo Sapiens, которая выражается в способности универсально передавать и принимать, воспринимать, абстрагировать, обрабатывать, накапливать, а также классифицировать информацию в ее различных формах проявления, что обеспечивает стабильность и ускорение социального прогресса;
- искусственная (цифровая) алгоритмическая универсальность, которая в компьютерном исполнении (на основе классической модели универсального компьютера Дж. фон Неймана [12]) открыла массовые возможности автономного выполнения алгоритмов обработки цифровой информации разнообразного назначения;
- эмпирический закон Мура, который в течение пяти десятилетий (с 1970-х гг.) определял стратегическую стабильность экспоненциальных темпов прогресса полупроводниковых технологий (долгосрочная рыночная основа массового производства—потребления компьютеров с микропроцессорными архитектурами, реализующими классическую цифровую универсальность в модели фон Неймана);
- глобальная компьютерная среда, логические и технологические основы для формирования/расширения которой на протяжении длительного времени предопределяются базовым стеком (набором) сетевых протоколов TCP/IP, обеспечивающим возможность неограниченного роста размеров сетей (с изначально открытой — легализованной — разнородностью компьютерных ресурсов), к которым в качестве узловых агентов может присоединяться любое число субъектов или объектов различного назначения — от массовых компьютерных устройств, датчиков и исполнительных механизмов до серверных/облачных центров и суперкомпьютеров;
- эмпирический закон Меткалфа [1], который раскрывает квадратичный потенциал роста синергии компьютерных сетей в отношении социально значимых процессов, происходящих в ГКС, включая бизнес-процессы.

3. Закон Меткалфа

С увеличением размеров ГКС ее влияние на все сферы деятельности стремительно растет.

Польза от применения сетей в различных сферах деятельности человека выражается положительным синергетическим сетевым эффектом, выраженным эмпирическим законом Меткалфа [1]: *"Польза от применения сетей пропорциональна квадрату числа сетевых узлов: $V \sim N^2$"*. Значение N^2 соответствует максимально возможному числу парных соединений между узлами сети. Квадратичная польза синергии сетевого эффекта быстро возрастает с увеличением числа межузловых соединений в сетях. Меткалф установил эту закономерность (рис. 1) во время

[1] Ущерб от вложений в первую волну массовых бизнес-проектов e-commerce в пространстве WWW составил около 5 трлн долл. США (https://www.forex.blog/dot-com-bubble-puzyr-dotkomov-1995-2000/).

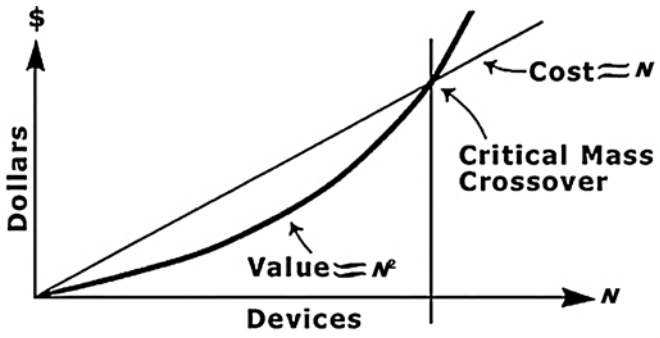


Рис. 1. Положительная синергия сетевого эффекта [1, 13]

продвижения технологии локальной сети Ethernet, разработанной под его началом [1], которая стала массовым продуктом и одним из ведущих сетевых стандартов, что в значительной мере способствовало формированию и расширению ГКС.

Линейный график изначально показывал пропорциональное увеличение стоимости покупки сетевых карт, с помощью которых компьютеры узлов подключаются к локальной сети. Как подтвердила практика, ценные качества сетевого взаимодействия между компьютерами квадратично возрастают с увеличением числа узлов, как показано на рис. 1.

Положительный эффект сетевой синергии достигается при увеличении числа сетевых узлов N, когда оно превышает значение "критической массы" N_{cm}: $N > N_{cm}$. Применительно к локальным сетям $N_{cm} \sim 30$ [1]. При этом с увеличением N положительный эффект пропорционален разнице между N^2 и затратами N: $\sim (N^2 - N)$. С развитием локальных сетей эта зависимость хорошо подтвердилась на практике [1].

В 1990-е гг. (в первое десятилетие становления и развития WWW) закон Меткалфа стал одним из основных мотиваторов глобальной цифровизации сетевого бизнеса. В этот период движущие силы массовой цифровизации проявлялись с максимальными (экспоненциально растущими) темпами закона Мура: удвоение эффективности характеристик полупроводниковых интегральных схем каждые 1,5...2 года.

Во время стремительного увеличения размеров ГКС, происходящего под давлением движущих сил массовой цифровизации, в системотехнических глубинах крайне разнородных сетевых ресурсов ГКС начали появляться и не менее стремительно нарастать силы торможения.

Одним из ведущих факторов торможения является изначально открытая (легализованная) разнородность сетевых ресурсов ГКС (аппаратных, программных, информационных). Разнородность вызывает все большее глобальное торможение всех перечисленных активных движущих сил, за исключением закона Мура. Однако этот закон уже прекращает свое действие по фундаментальным физическим причинам, поскольку с уменьшением размеров до атомарных уровней транзистор утрачивает свой функционал.

4. Силы торможения развития глобальной компьютерной среды

В ходе цифровизации бизнес-процессов и формирования крупных цифровых бизнес-экосистем на основе облачных технологий специалисты активно изучают особенности и проблемы сетевой синергии.

Публикации, посвященные изучению особенностей воплощений закона Меткалфа в ГКС и перспектив развития глобальных цифровых экосистем на основе облачных архитектур и технологий, как показывают работы [13, 14], ограничиваются рассмотрением только двух источников движущих сил массовой цифровизации, а именно, эмпирическими законами Мура и Меткалфа. Один из таких подходов представлен в работе [13]. Игнорирование других источников фундаментального влияния, перечисленных выше, сужает рамки таких рассмотрений, сводя их к исследованию частных, фрагментарных проявлений. Кроме перечисленных факторов практически отсутствуют исследования фундаментальных сил торможения, противостоящих движущим силам, которые возникают по причинам глубинных внутрисистемных диспропорций развития ГКС.

Такие фрагментарные подходы не позволяют видеть всей полноты как системообразующего, так и деструктивного воздействия стихийного роста ГКС на социотехносферу.

Проанализируем методологию и результаты двух таких типовых, имеющих корпоративно-односторонний (без учета внутрисистемных закономерностей развития ГКС) характер исследований.

4.1. Анализ сетевой синергии глобальной компьютерной среды

Работа [13] по кругу поднятых вопросов является показательным примером таких "корпоративных" исследований, направленных на продвижение облачных систем. В этой работе с позиций продвижения цифровой экономики автором анализируются различные аспекты влияния движущих сил законов Мура и Меткалфа на формирование и развитие цифровой синергии компьютерных сетей в рамках облачной концепции.

В результате такого анализа автор приходит к выводу, что в современной практике массового применения ГКС по мере увеличения размеров цифровых экосистем наблюдается все большее снижение достигаемой эффективности по сравнению с той, которая предполагается этими законами. Тем самым косвенно допускается тот факт, что существуют пределы роста размеров и масштабов влияния современных крупномасштабных сетевых экосистем.

В работе [13] также полагается, что облачные технологии продолжают поддерживать положительный сетевой эффект, а существующие методы и технологии частично преодолевают разные сдерживающие силы, которые возникают с увеличением масштабов цифровых экосистем в ходе решения системотехнических проблем функциональной интеграции сетевых ресурсов. Оценка перспектив дальнейшего развития облачных цифровых экосистем в этой работе

основана на заявленной "теории оптимизации функциональной совместимости". Такой подход предполагает возможность весьма затратной и ограниченной по возможностям "шлюзовой" интеграции экосистем, если на нее "существует массовый спрос".

Эта работа следует общепринятым представлениям об отсутствии де-факто конкурентоспособной альтернативы концепции облачных систем. Апробация этой концепции как ведущего направления глобальной цифровизации и продвижения сетевых бизнес-моделей проводилась в течение предыдущих 15...20 лет [9, 15—17]. Становление облачной концепции осложнялось несостоявшимися ожиданиями сверхбыстрого роста традиционных (доцифровых) секторов экономики [6—8] вслед за беспрецедентными успехами закона Мура — главного ускорителя цифровой индустрии и компьютерного рынка.

Системотехнические ограничения облачных сетевых архитектур изначально были известны компьютерным специалистам. Бизнес-сообщество также не забыло компьютерный парадокс Р. Солоу [18, 19], который предупреждает о нетривиальной сложности проблем совместимости доцифровой экономики и цифровых технологий.

Нетривиальность проблем достижения такой совместимости состоит в том, что в рамках только существующих экономических воззрений/институтов развития или только стихийного развития цифровых технологий полномасштабное осуществление системно-целостной интеграции доцифровых и цифровых моделей управления устойчивым развитием невозможно. Об этом свидетельствует тридцатилетний опыт стихийного развития ГКС [6—8].

Решающим фактором формирования и доминирования глобального тренда развития облачного концепта стало то, что в условиях открытой разнородности ГКС конкурентоспособных моделей глобализации цифровой экономики, альтернативных высокозатратным облачным технологиям, не существует.

Наращивание масштабов применения облачных архитектур и технологий во многом опирается на агрессивные маркетологические методы и кампании продвижения в сознание участников рынка синергии экспоненты закона Мура, а также параболы закона Меткалфа. Однако на современном этапе развития ГКС ситуация меняется в связи с завершением периода активного роста синергии этих законов.

Следует отметить, что "однобокие" маркетологические способы продвижения не учитывают влияние других ключевых движущих сил массовой цифровизации, определяющих объективные закономерности развития ГКС, представленные в работах [2, 5] и других работах, на которые даны ссылки в следующих разделах. По этой причине такие подходы утрачивают действенность по мере естественного исчерпания системообразующего потенциала закона Мура и нарастания отрицательной синергии противодействия положительному сетевому эффекту Меткалфа.

Многочисленные факты функционирования цифровых облачных экосистем в 2000-е и 2010-е гг. показывают, что стремительный рост масштабов влияния ГКС на социосистемы, основанный на доминировании облачных технологий, лежащих в основе систем обработки глобально распределенной информации, сопровождается неуклонным снижением стабильности социотехносферы в целом [6—8].

Нельзя также не отметить, что монополия облачных систем в методах и средствах осуществления цифровой трансформации социотехносферы неизбежно становится глобальным инструментом манипуляций общественным сознанием, что несет растущие риски узурпаций цифрового информационного пространства[1].

В настоящее время облачные архитектуры приближаются к исчерпанию своего системообразующего потенциала. Для формирования новых стратегий и перспективных моделей дальнейшего наращивания масштабов глобализации цифровой трансформации необходим концептуальный реинжиниринг крайне разнородной ГКС.

Далее в работе показано, что жизненный цикл рыночных успехов облачных технологий принципиально ограничен. Поэтому маловероятно, что приемлемый уровень их прибыльности сохранится для продвижения в текущем десятилетии и, тем более, в следующем.

4.2. Изменение исходных предпосылок закона Меткалфа в условиях глобальной компьютерной среды

Анализ статистических данных, представленных в работе [14], позволяет сделать определенные выводы при оценке перспектив развития облачных технологий. Эта работа направлена на проверку действия закона Меткалфа в ГКС. В ней представлены данные, сравнивающие прибыльность и стоимость цифрового бизнеса двух крупных социальных сетей Facebook и Tencent в течение 2003—2014 гг. Как отмечают авторы исследования, результаты сравнения этих различных сетей показывают успешный бизнес с достаточно высокими экономическими показателями в сопоставимых пропорциях.

Дополнительный анализ статистических данных, представленных в этой работе, показывает, что рентабельность бизнеса в сетях Facebook и Tencent за десять лет не соответствует квадратичному росту. Более того, статистика годовых отчетов показывает тенденцию к снижению положительного эффекта в диапазоне 40...20 %. Это подходящие уровни для бизнеса, но это не квадратичный рост, а устойчивое снижение. Возникает вопрос: "Как этот факт связан с квадратичной синергией Меткалфа?"

---

[1] Глобальное отчуждение персональной информации, которая концентрируется в облачных недрах гипертрофированных реализаций клиент-серверной сетевой архитектуры поисковиков и социальных сетей, открывает неограниченные возможности манипуляций общественным сознанием. Отсутствие должной защиты личного цифрового информационного пространства каждого человека и неконтролируемая социальными институтами концентрация персональных данных становится одной из главных угроз тотальной цифровизации, осуществляемой посредством облачных систем.

В соответствии с методологией и выводами работы [13] напрашивается ответ о влиянии многих конкретных (частного характера) факторов торможения. Эти различные факторы, будучи выявленными, могут быть устранены с помощью подходящих для каждого случая технологий "оптимальной функциональной совместимости". Однако в таком ответе не просматриваются основные причины фундаментального отклонения фактических данных [14] от квадратичного роста синергии Меткалфа.

В понимании авторов настоящей статьи, полученные расхождения с законом Меткалфа убедительно указывают на происходящее де-факто глобальное увеличение сил торможения отрицательной синергии. Причины в том, что массовый фундаментальный эффект торможения связан с легализованной априори открытой разнородностью ГКС. С увеличением размеров распределенных экосистем в ГКС возникает необходимость решения многовариантных задач системно-функциональной интеграции крайне разнородных сетевых ресурсов ГКС на аппаратном, программном и информационном уровнях. Это требует экспоненциально растущих (в общем случае) затрат на преодоление комбинаторно сложного проклятия размерности. На практике это означает появление предельных (непреодолимых) уровней интеграции разнородных сетевых ресурсов, которые ограничивают масштабы роста облачных систем и технологий, примеры анализа которых рассмотрены в работах [9, 15—17].

На рис. 2 показаны пределы роста облачных систем, реализуемых в условиях изначально открытой (легализованной) разнородности сетевых ресурсов ГКС, обеспечивающих положительный эффект сетевой синергии.

На рис. 2:

N_{exp} — начало действия отрицательной синергии (начало экспоненциального увеличения стоимости систем из-за комбинаторной сложности системно-функциональной интеграции разнородных ресурсов ГКС);

N_{max} — это верхний предел возможной прибыльности (стоимость крупнейших облачных платформ и экосистем достигает 4 трлн. долл. США и более [15], что свидетельствует о приближении к этому пределу);

$N_{cm} < N^+ < N_{max}$ — диапазон экономически эффективного увеличения размера облачных систем.

Рис. 2. Пределы роста облачных систем обработки распределенной информации в ГКС

Важно отметить, что на рис. 2 при $N \sim N_{exp}$ линейный рост затрат по закону Меткалфа прекращается. В дальнейшем, когда $N > N_{exp}$, можно наблюдать, что начинается экспоненциальный рост затрат на преодоление силы торможения отрицательной синергии, связанной с комбинаторной сложностью задач системно-функциональной интеграции многоуровневой разнородности ресурсов ГКС.

Особая значимость соотношений, представленных на рис. 2, заключается в том, что возникает четкое понимание того, что в условиях разнородности сетевых ресурсов ГКС экономически обоснованное увеличение размера N^+ облачных экосистем ограничено диапазоном $N_{cm} < N^+ < N_{max}$.

Судя по колоссальным затратам на создание, эксплуатацию и модернизацию [15], многие облачные экосистемы уже приблизились к верхним пределам. По этой причине необходимо искать новые возможности для наращивания системно сбалансированной цифровой трансформации посредством увеличения масштабов функциональной интеграции ресурсов ГКС. Такая трансформация может осуществляться за счет устранения причин непрерывного воспроизводства разнородности и последующего "обнуления" экспоненты отрицательной синергии.

5. Пути устранения системных дисбалансов в развитии глобальной компьютерной среды

Разнородность форм представления глобально распределенных данных, программ, процессов и систем в глобальных сетях, воплощаемая в различных, изначально несовместимых аппаратных, программных и информационных платформах, является одним из главных проявлений несбалансированного развития ГКС. Из трех возможных фундаментальных видов действий с информацией, а именно хранение, передача и преобразование, глобализованы только первые два. В результате универсальная программируемость, присущая каждому узлу ГКС, не распространяется на любое сколь угодно большое подмножество ГКС. Поэтому необходима системно-целостная глобализация этих действий [2, 20].

Разнородность является доминирующим источником отрицательной синергии ГКС. Чрезмерный рост стоимости экосистем связан с комбинаторной сложностью функциональной интеграции разнородных сетевых ресурсов. Разнородность и другие внутрисистемные дисбалансы ГКС активируют вторичные источники отрицательной синергии. К их числу относятся такие как кризис перепроизводства информации, невозможность обеспечения требуемых уровней кибербезопасности и др.

Дальнейшее рассмотрение обосновывает принципиальную возможность пересмотра и обновления основополагающих принципов, лежащих в основе ГКС, с изначально открытой (легализованной) разнородностью и осуществления реинжиниринга ГКС на основе такого обновления.

Стратегической целью такого реинжиниринга является выявление и устранение коренных причин разнородности ГКС, в том числе формирование

обновленной аксиоматики системно-сбалансированного развития ГКС с последующим эволюционным переходом существующей ГКС на качественно новые системообразующие уровни. При этом необходимо обеспечить регулярные системные возможности наследования аппаратных, программных и информационных наработок предшествующих поколений.

Ключевой задачей предлагаемого реинжиниринга является кумулятивно-бесшовное и кибербезопасное распространение свойства алгоритмической универсальности с внутрикомпьютерных ресурсов сетевых узлов на сколь угодно большие компьютерные сети ГКС. Свойство кумулятивной универсальности позволяет при необходимости согласованным образом задействовать совокупный вычислительный, функциональный и информационный потенциал всех узлов ГКС для решения всего разнообразия задач управления устойчивым развитием социотехносферы, функционирующей в условиях глобальной информационной сильносвязности.

5.1. О технологиях функциональной интеграции

В условиях изначально легализованной разнородности ГКС нельзя построить единую универсальную модель бесшовно программируемых глобально распределенных вычислений. Поэтому большие распределенные системы различного назначения создаются путем профилированного конфигурирования и системно-функциональной интеграции разнородных сетевых ресурсов с использованием растущего арсенала разнообразных облачных технологий.

Это дорогостоящие, трудно совместимые проблемно-ориентированные технологии, которые очень дорого разрабатывать и применять. Они требуют создания большого числа трудно сопрягаемых стандартов, решения сложных технологических проблем масштабируемости, композиции по требованию, безопасности и т. д.

Такие проблемы решаются, как правило, с использованием эвристических и согласительных подходов к стандартизации. К сожалению, они приводят к увеличению числа системно разрозненных и конкурирующих программных и технологических цифровых платформ для интеграции сетевых ресурсов, которые не отвечают растущим требованиям реентерабельности и кибербезопасности. Также они предполагают обязательные априорные системно-технические ограничения функциональности систем, что ориентирует их на создание излишне большого числа дорогостоящих и трудно развиваемых узкопрофильных распределенных систем и подсистем.

5.2. Устранение разнородности на основе новой модели распределенных вычислений

Непрерывно растущая на многих системотехнических уровнях разнородность ГКС является результатом изначального отсутствия в классической модели универсальных компьютеров Дж. фон Неймана [12] универсальных форм математически регламентированного представления данных и программ, а также способов аппаратного воплощения универсально программируемых вычислений.

В работах [2, 5, 20] авторов настоящей статьи представлено концептуальное описание универсальной модели бесшовно программируемых распределенных вычислений в сколь угодно больших одноранговых — Peer-to-Peer (P2P) — компьютерных сетях. Эта модель рассматривается в том числе и как перспективная альтернатива концепции облачных систем. Предлагаемая модель построена путем "поглощающего" математического обобщения классической модели универсального компьютера Дж. фон Неймана с использованием математически замкнутого компьютерного базиса исчисления древовидных структур (ИДС) [2, 20].

Формализм ИДС посредством деревьев с любым числом задаваемых вершин (рис. 3) позволяет осуществить математическую унификацию структурных форм представления компьютерной информации (данных и программ), а также методов бесшовного программирования/выполнения программ (в процедурном стиле фоннеймановской модели) как во внутренних ресурсах компьютеров, так и в сколь угодно больших компьютерных сетях.

На рис. 3 показан переход от математической формы представления деревьев (рис. 3, *а*) к бинарной форме компьютерного представления (рис. 3, *б*). "Многоарные" связи между вершинами (рис. 3, *а*) обозначены штриховыми отрезками. Переход осуществляется посредством удаления всех штриховых связей между вершинами и добавления связей, обозначенных сплошными отрезками. Это взаимнооднозначный переход.

Значениями вершин таких деревьев (на рис. 3 обозначены буквами) могут быть слова допустимых типов данных (в виде битовых строк разной длины), а также однородные массивы однотипных (равной длины) слов. Значения строк могут интерпретироваться согласно списку допустимых типов данных как символьные или числовые слова или массивы, а также как двоичные коды программ, предназначаемых для запуска в различных указываемых программистом узлах ГКС и ОС. Число вершин и размеры двоичных строк и массивов не имеют принципиальных ограничений.

Исходный постулат модели ИДС формулируется следующим образом: универсальной цифровой формой представления компьютерной информации

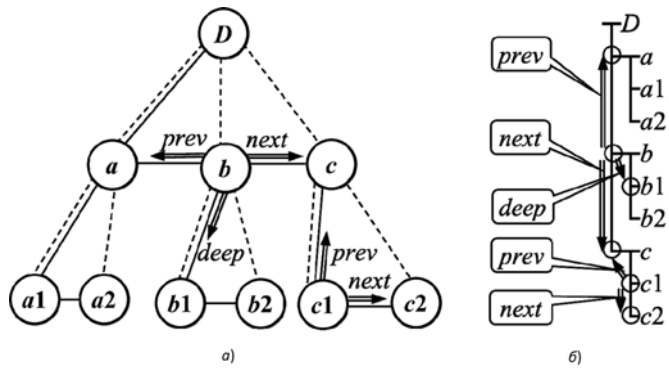


Рис. 3. Древовидная структура (*а*) и компьютерная форма ее представления (*б*) [10]

(данных и программ) являются древовидные структуры в виде компьютерной формы представления двоичных деревьев (рис. 3, *б*).

Формализм ИДС представляет собой математически замкнутый компьютерный базис операций формирования и преобразования произвольных древовидных структур [2, 20] в виде двоичных деревьев (рис. 3, *б*). По сути, компьютерный базис ИДС можно рассматривать как фундаментальный математический стандарт представления и обработки компьютерной информации.

Фундаментальность выбранной формы представления компьютерной информации (данных и программ) состоит в характеристическом свойстве минимальной структурной сложности деревьев: связность между всеми вершинами обеспечивается минимально возможным числом дуг (число дуг на 1 меньше числа связываемых вершин). Потеря любой из дуг нарушает целостность дерева.

Достоинства модели ИДС:

- являясь минимально достаточным (для сохранения универсальности) математическим обобщением классической модели универсального компьютера (модели Дж. фон Неймана), она в полной мере наследует процедурный стиль программирования и, соответственно, простейшую в аппаратной реализации логику управления исполнением программ;
- открывает пути к конструктивной математически замкнутой унификации (стандартизации) форм представления и способов обработки компьютерной информации, что ведет к устранению причин непрерывного воспроизводства разнородных форм представления данных и программ, а также аппаратных, программных и информационных платформ;
- открывает возможности формирования в совокупных ресурсах существующей ГКС математически однородного, бесшовно программируемого алгоритмического пространства распределенных вычислений, которое обнуляет многоуровневые комбинаторные компоненты сложности функциональной интеграции ресурсов ГКС.

Новую модель можно рассматривать как основу для системно-целостного и сбалансированного реинжиниринга [20, 21, 23] существующих логических основ разнородной ГКС. Данная модель, наследуя локальное свойство цифровой алгоритмической универсальности классической модели Дж. фон Неймана, привносит математическую регламентацию форм представления данных/программ и способов их взаимодействия. В работах [2, 5, 20, 21] показано, как в рамках новой модели свойство локальной алгоритмической универсальности компьютеров из каждого узла ГКС бесшовно и кибербезопасно распространяется на любое подмножество компьютеров сколь угодно больших сетей.

Предложенное обобщение классической модели открывает возможность построения нового класса универсальных сетевых компьютеров, характеризующихся немикропроцессорной архитектурой [20—22]. Такие компьютерные архитектуры на аппаратном уровне воплощают возможности математически замкнутой модели ИДС в части кумулятивного расширения указанных свойств программируемости и кибербезопасности на большие сети. Требуемое расширение достигается путем формирования в ГКС единого, математически однородного, бесшовно программируемого и кибербезопасного алгоритмического пространства распределенных и параллельных вычислений посредством использования в сетевых узлах новых компьютеров с немикропроцессорной архитектурой [20-23].

5.3. Глобальная компьютерная универсальность как основа новой сетевой синергии

Вместе с устранением причин разнородности форм представления и способов работы с компьютерной информацией в новой модели глобально распределенной универсальности исчезают и вторичные источники отрицательной синергии:

- экспоненциальный рост слабо формализованной глобально распределенной информации;
- неспособность обеспечить требуемые уровни реентерабельности и кибербезопасности в условиях растущей разнородности ГКС.

Формирование единого и универсального, математически однородного алгоритмического пространства распределенных и параллельных вычислений позволит кумулятивно и бесшовно — для компьютерной среды в целом — распространить свойства универсальной программируемости и кибербезопасности на сетевые ресурсы без ограничений на специфику задач и размеры систем переработки глобально распределенной информации.

Свойство бесшовной и кибербезопасной программируемости ГКС в целом можно рассматривать как новую — глобальную, компьютерную универсальность, которая расширяет локальную универсальность своих компьютерных узлов на сколь угодно большие сети. Бесшовное программирование позволит обходиться без обременительных стандартизаций дорогостоящих технологий системно-функциональной интеграции изначально разнородных сетевых ресурсов.

Таким образом, ГКС с качественно новыми системообразующими возможностями глобально распределенной универсальности становится основой принципиально новой сетевой синергии [8], которая открывает перспективы неограниченного расширения границ достижимости потенциала положительной синергии закона Меткалфа.

На рис. 4 показан синергетический эффект глобальной компьютерной универсальности обновленной ГКС. Обновление осуществляется посредством реинжиниринга ГКС [10, 22, 23] путем включения новых сетевых компьютеров с немикропроцессорной архитектурой в состав "обычных" сетевых узлов.

Новые компьютеры с немикропроцессорной архитектурой, напрямую связанные между собой через сети, позволяют бесшовно программировать сколь угодно большие распределенные системы переработки глобально распределенной информации в целях устойчивого развития. Изначально обеспечивая бесшовное программирование и кибербезопасность на

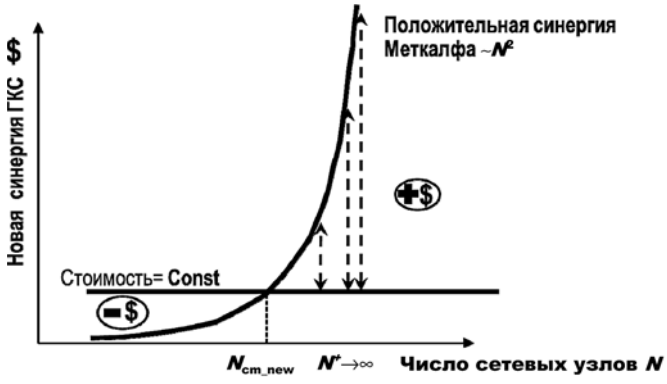


Рис. 4. Глобально распределенная компьютерная универсальность как основа новой синергии ГКС

аппаратных уровнях, такие компьютеры позволят с минимальными усилиями (без разработки многочисленных дополнительных, громоздких и уязвимых "промежуточных" слоев системного программного обеспечения) осуществлять функциональную интеграцию сетевых ресурсов посредством развитых средств автоматизации программирования больших распределенных систем без ограничений на их размеры.

Такая интеграция сетевых ресурсов станет возможной в автоматических режимах компоновки глобально распределенных программ с привлечением соответствующих библиотек динамического конфигурирования сетевых ресурсов под решаемые задачи. При этом, главным образом, остаются затраты только на разработку моделей и алгоритмов решаемых прикладных задач, а также создание прикладного программного обеспечения. Благодаря новым системообразующим возможностям глобальной универсальности, воплощаемой в средствах автоматизации бесшовного программирования, стоимость разработок прикладных программ перестает зависеть от размера больших распределенных систем (линия Const на рис. 4).

При этом существующие и ранее наработанные вычислительные, программные и информационные ресурсы сохраняются в узлах сетей, но уже в качестве "ведомых", управление которыми делегируется новым сетевым компьютерам, имеющим в пределах своего сетевого узла системный статус "ведущий".

Сетевые компьютеры с новой, глобально распределенной, универсальностью позволяют:

- устранить экспоненциальный эффект тормозящей силы отрицательной синергии, возникающей вследствие разнородности сетевых ресурсов, и снять существующие ограничения на размер и функционал больших распределенных систем, что позволит практически неограниченно расширять сферы квадратичного действия положительной синергии Меткалфа;
- избавиться от необходимости увеличения инвестиций, пропорциональных размеру больших систем, горизонтальная линия "Стоимость = Const" (см. рис. 4) определяет фиксированный (минимально достаточный) уровень первоначальных инвестиций, что позволяет масштабировать полученные функциональные решения на сколь угодно большие сети без заметных дополнительных затрат.

Заключение

Глобально распределенная компьютерная универсальность открывает принципиально новые системообразующие возможности для реинжиниринга ГКС и формирования в ней универсального, бесшовно программируемого и кибербезопасного алгоритмического пространства распределенных и параллельных вычислений. В этом пространстве становится возможным устранение отрицательной синергии разнородности ГКС, что является необходимым условием для системно-сбалансированной цифровой трансформации социотехносферы, направленной на безопасное и устойчивое развитие социосистем и мировой социосистемы в целом в условиях глобальной информационной сильносвязности.

Основные результаты:

- представлен оригинальный подход к изучению фундаментальных закономерностей развития компьютерной среды как глобального системно-целостного объекта;
- впервые идентифицированы и показаны способы устранения внутрисистемной отрицательной синергии ГКС, что открывает возможности для минимизации затрат на создание и модификацию сколь угодно больших систем распределенных систем, которые составят альтернативу облачным системам и технологиям и откроют перспективы для массовой реализации системно-сбалансированных моделей цифровой трансформации.

Список литературы

1. **Metcalfe B.** Metcalfe's law after 40 years of Ethernet// Computer. 2013. Vol. 46, No. 12. P. 26—31. DOI: 10.1109/MC.2013.374.

2. **Затуливетер Ю. С.** Проблемы глобализации управления в математически однородном поле компьютерной информации // Проблемы управления. 2005. № 1. Ч. I. Кибернетизация социосистемы. С. 1—12. URL: http://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=pu&paperid=402&option_lang=rus. Ч. II. К единому функциональному пространству. № 2. С. 13—23. URL: http://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=pu&paperid=418&option_lang=rus.

3. **The Global** Technology Report 2012. Living in a Hyperconnected World. Geneva. 2012. URL: https://www3.weforum.org/docs/Global_IT_Report_2012.pdf

4. **Institute** for the future. The hyperconnected world of 2030—2040. URL: https://www.iftf.org/fileadmin/user_upload/downloads/ourwork/IFTF_Hyperconnected_World_2020.pdf#:~:text=The%20hyperconnected%20world%20is%20a%20future%20in%20which,more%20devices%2C%20and%20more%20interactions%20between%20the%20two

5. **Затуливетер Ю. С., Фищенко Е. А.** Проблемы программируемости, безопасности и надежности распределенных вычислений и сетецентрического управления. Ч. 1. Анализ проблематики // Проблемы управления. 2016. № 3. С. 49—57. URL: https://www.ipu.ru/sites/default/files/publications/38154/19991-38154.pdf

6. **Затуливетер Ю. С.** Ноосферные проблемы цифровой трансформации // Материалы 6-го Международного научного конгресса "Глобалистика-2020: Глобальные проблемы и будущее человечества" (Москва, МГУ). М.: МООСИПНН Н. Д. Кондратьева. 2020. С. 38—49. DOI: 10.46865/978-5-901640-33-3-2020-38-49. URL: https://www.ipu.ru/sites/default/files/publications/60013/49159-60013.pdf

7. **Затуливетер Ю. С.** Кибернетический анализ закономерностей теории длинных волн Кондратьева в контексте глобальной информационной сильносвязности // Научное наследие Н. Д. Кондратьева и современность: Сб. науч. тр. участников X Международной Кондратьевской конференции, посвященной 125-летию со дня рождения Н. Д. Кондратьева, Москва, 25—30 сентября 2017 г. С. 158—169. URL: https://www.ipu.ru/sites/default/files/publications/44065/27730-44065.pdf

8. **Затуливетер Ю. С.** Новая синергия длинных волн Н. Д. Кондратьева // Труды 11-й Международной Кондратьевской конференции "Возможные сценарии будущего России и мира: междисциплинарный дискурс" (Москва, 2020). М.: МООСИПНН Н. Д. Кондратьева, 2020. С. 169—183. DOI: 10.46865/978-5-901640-34-0-2020-169-183. URL: https://www.ipu.ru/sites/default/files/publications/60020/49168-60020.pdf

9. **Skala K., Sojat Z.** The Rainbow: integrating computing into the global ecosystem // MIPRO 2019 (20—24 May 2019. Opatija. Croatia). P. 233—240. DOI: 10.23919/MIPRO.2019.8756947.

10. **Затуливетер Ю. С., Фищенко Е. А.** К компьютерно-сетевым архитектурам для цифровой трансформации больших систем // Программные системы: теория и приложения. 2020. Т. 11, № 3 (46). С. 85—131. DOI: 10.25209/2079-3316-2020-11-3-85-131.

11. **Кризис** доткомов 2000-х — пузырь, который лопнул. Причины и последствия // URL: https://internetboss.ru/krisis-dotkomov/

12. **Беркс А., Голдстейн Г., Нейман Дж.** Предварительное рассмотрение логической конструкции электронного вычислительного устройства // Кибернетический сборник. М.: Мир. 1964. Вып. 9. С. 7—67.

13. **Yoo C. S.** Moore's law, Metcalfe's law, and the theory of optimal interoperability. 2016. URL: https://ctlj.colorado.edu/wp-content/uploads/2015/12/v2.Final-Yoo-11.25.15-JRD.pdf

14. **Zhang X. Z., Liu J. J., Xu Z. W.** Tencent and Facebook data validate Metcalfe's law // Journal of computer science and technology. 2015. Vol. 30, Is. 2. P. 246—251. DOI: 10.1007/s11390-015-1518-1.

15. **Everything** you need to know about Digital Platforms. 2016. URL: http://stephane-castellani.com/everything-you-need-to-know-about-digital-platforms/

16. **Srikanth.** Top 10 Cloud Computing Challenges in 2020. 30.09.2019. URL: https://www.techiexpert.com/top-10-cloud-computing-challenges-in-2020/

17. **Ray P. P.** An introduction to dew computing: Definition, concept and implications // IEEE Access. 2017. Vol. 6. P. 723—737. URL: DOI: 10.1109/ACCESS.2017.2775042.

18. **Solow R.** We'd Better Watch Out. New York Review of Books. July 12, 1987. URL: http://digamo.free.fr/solow87.pdf

19. **Парадокс** производительности. URL: https://ru.abcdef.wiki/wiki/Productivity_paradox

20. **Затуливетер Ю. С.** Компьютерный базис сетецентрического управления //Труды 2-й Всероссийской конференции с международным участием "Технические и программные средства систем управления, контроля и измерения" (УКИ-2010, Москва). М.: ИПУ РАН, 2010. С. 17-37. URL: https://www.ipu.ru/sites/default/files/publications/38190/20052-38190.pdf

21. **Затуливетер Ю. С., Фищенко Е. А.** Проблемы программируемости, безопасности и надежности распределенных вычислений и сетецентрического управления Ч. 2. Подход к общему решению // Проблемы управления. 2016. № 4. С. 58—69. URL: http://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=pu&paperid=983&option_lang=rus

22. **Затуливетер Ю. С., Фищенко Е. А.** Универсальное алгоритмическое пространство распределенных и параллельных вычислений // Информационные технологии и вычислительные системы. 2018. № 2. С. 78—93. DOI: 10.14357/20718632180207.

23. **Zatuliveter Yu. S., Fishchenko E. A.** Towards Strategic Reengineering the Global Computer Environment for Control of Sustainable Development of Social Systems // IFAC-PapersOnLine. 2021. Vol. 54, No. 13. P. 129—133. DOI: https://doi.org/10.1016/j.ifacol.2021.10.432

Synergy of Digital Universality of the Global Computer Environment

Yu. S. Zatuliveter, zvt@ipu.rssi.ru, **E. A. Fishchenko**, elena.fish@mail.ru, V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, 117997, Russian Federation

Corresponding author:

Zatuliveter Yury S., Ph.D. of Engineering Sciences, Leading Researcher, Senior Research Officer, V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, 117997, Russian Federation
E-mail: zvt@ipu.rssi.ru

Received on December 14, 2021
Accepted on January, 28, 2022

The global computer environment (GCE) as a whole is considered an object of research. The system-wide aspects of implementing the digital transformation of the sociotechnosphere through the GCE are analyzed. Factors of the destructive influence of the fundamental system-technical laws of the spontaneous growth of the GCE on the functioning/development of the sociotechnosphere are presented. Special attention is paid to the study of the features of the apperance of a positive synergetic network effect, expressed by Metcalf's law, in the conditions of the initially open intra-system heterogeneity of the network resources of the existing GCE. It is established that with the increase in the size of large distributed systems implemented in the GCE, the factor of heterogeneity of network resources becomes a source of negative (system-destructive) network synergy, which devalues the positive synergy of Metcalf's law. Examples of ecosystems implemented in the GCE through cloud technologies are considered. It is shown that in the conditions of initially open heterogeneity of network resources, the further increase in the size of large distributed systems leads to an insurmountable increase in the complexity of the system-functional integration of network resources and an uncontrolled decrease in the stability of the sociotechnosphere. The principles of conceptual reengineering the GCE are presented to eliminate the causes of systemic heterogeneity and seamless/cybersecurity distribution of algorithmic universality, closed in intracomputer resources, to any arbitrarily large sub-

set of GCE computers. Such reengineering will allow neutralizing the negative synergy and maximizing the positive synergy of the Metcalf effect in large distributed systems without restrictions on their size.

Keywords: *computer environment, development trends, sociotechnosphere, digital transformation, positive network synergy, Metcalf's law, heterogeneity of network resources, negative synergy, cloud systems, conceptual reengineering, model of globally universal distributed computing, unified algorithmic space, seamless programming, cybersecurity*

For citation:

Zatuliveter Yu. S., Fishchenko E. A. Synergy of Digital Universality of the Global Computer Environment, *Programmnaya Ingeneria,* 2022, vol. 13, no. 3, pp. 107—118.

DOI: 10.17587/prin.13.107-118

References

1. **Metcalfe B.** Metcalfe's Law after 40 Years of Ethernet, *Computer*, 2013, vol. 46, no. 12, pp. 26—31, DOI: 10.1109/MC.2013.374.

2. **Zatuliveter Yu. S.** The problems of control paradigm globalization in the mathematically uniform field of computer information, *Problemy upravleniya (Control Sciences)*, 2005, no. 1. Part I. Cybernation of sociosystem, pp. 1—12. Part II. To the common function space, no. 2, pp. 13—23 (in Russian).

3. **The Global** Technology Report 2012. Living in a Hyperconnected World, Geneva. 2012, available at: https://www3.weforum.org/docs/Global_IT_Report_2012.pdf

4. **Institute** for the Future. The Hyperconnected World of 2030—2040, available at: https://www.iftf.org/fileadmin/user_upload/downloads/ourwork/IFTF_Hyperconnected_World_2020.pdf#:~:text=The%20hyperconnected%20world%20is%20a%20future%20in%20which,more%20devices%2C%20and%20more%20interactions%20between%20the%20two

5. **Zatuliveter Yu. S., Fishchenko E. A.** Problems of Programmability, Security and Reliability of Distributed Computing and Network-centric Control. Part 1: Problem Analysis, *Problemy upravleniya*, 2016, no. 3, pp. 49—57 (in Russian).

6. **Zatuliveter Yu. S.** Noospheric Problems of Digital Transformation, *Materialy 6-go Mezhdunarodnogo nauchnogo kongressa "Globalistika-2020: Global'nye problemy i budushchee chelovechestva",* Moscow, MOOSIPNN N. D. Kondrat'eva, 2020, pp. 38-49, DOI: 10.46865/978-5-901640-33-3-2020-38-49. (in Russian).

7. **Zatuliveter Yu. S.** Cybernetic analysis of the law-regularities of Kondratiev's theory of long waves in the context of global information strongly connectedness, *Nauchnoe nasledie N. D. Kondrat'eva i sovremennost': Sbornik nauchnyh trudov uchastnikov X Mezhdunarodnoj Kondrat'evskoj konferencii, posvyashchennoj 125-letiyu so dnya rozhdeniya N. D. Kondrat'eva*, Moscow, 25—30 September, 2017, pp. 158—169 (in Russian).

8. **Zatuliveter Yu. S.** New Synergy of Long Waves by N. D. Kondratiev, *Trudy 11-j Mezhdunarodnoj Kondrat'evskoj konferencii "Vozmozhnye scenarii budushchego Rossii i mira: mezhdisciplinarnyj diskurs" 2020,* Moscow, MOOSIPNN N. D. Kondrat'eva, 2020, pp. 169—183, DOI: 10.46865/978-5-901640-34-0-2020-169-183 (in Russian).

9. **Skala K., Sojat Z.** The Rainbow: Integrating Computing into the Global Ecosystem, *MIPRO 2019 (20—24 May 2019. Opatija. Croatia),* pp. 233—240. DOI: 10.23919/MIPRO.2019.8756947.

10. **Zatuliveter Yu. S., Fishchenko E. A.** Towards Computer-network Architectures for the digital transformation of large-scale systems, *Program Systems: Theory and Applications*, 2020, vol. 11, no. 3(46), pp. 85—131. DOI: 10.25209/2079-3316-2020-11-3-85-131 (in Russian).

11. **The dotcom** crisis of the 2000s is a bubble that has burst. Causes and consequences, 2021, available at: https://internetboss.ru/krisis-dotkomov/

12. **Burks A. W., Goldstine H. H., von Neumann J.** Preliminary discussion of the logical design of an electronic computing instrument, *A. H. Taub, John von Neumann Collected Works*, The Macmillan Co., New York, 1963. Volume V, pp. 34—79.

13. **Yoo C. S.** Moore's Law, Metcalfe's Law, and the Theory of Optimal Interoperability, 2016, available at: https://ctlj.colorado.edu/wp-content/uploads/2015/12/v2.Final-Yoo-11.25.15-JRD.pdf.

14. **Zhang X. Z., Liu J. J., Xu Z. W.** Tencent and Facebook Data Validate Metcalfe's Law, *Journal of computer science and technology*, 2015, vol. 30, is. 2, pp. 246—251, DOI: 10.1007/s11390-015-1518-1.

15. **Everything** you need to know about Digital Platforms, 2016, available at: http://stephane-castellani.com/everything-you-need-to-know-about-digital-platforms/

16. **Srikanth.** Top 10 Cloud Computing Challenges in 2020, 30.09.2019, available at: https://www.techiexpert.com/top-10-cloud-computing-challenges-in-2020/

17. **Ray P. P.** An introduction to dew computing: Definition, concept and implications, *IEEE Access*, 2017, vol. 6, pp. 723—737, DOI: 10.1109/ACCESS.2017.2775042.

18. **Solow R.** We'd Better Watch Out, *New York Review of Books*, July 12, 1987, available at: http://digamo.free.fr/solow87.pdf

19. **Productivity** paradox, available at: https://ru.abcdef.wiki/wiki/Productivity_paradox (in Russian).

20. **Zatuliveter Yu.S.** Computer Basis of Network-centric Control, *Trudy 2-j Vserossijskoj konferencii s mezhdunarodnym uchastiem "Tekhnicheskie i programmnye sredstva sistem upravleniya, kontrolya i izmereniya", UKI-2010*, Moscow, ICS, 2010, pp. 17—37 (in Russian).

21. **Zatuliveter Yu. S., Fishchenko E. A.** Problems of Programmability, Security and Reliability of Distributed Computing and Network-centric Control. Part 2: the Approach to General Solution, *Problemy upravleniya (Control Sciences)*, 2016, no. 4, pp. 58—69 (in Russian).

22. **Zatuliveter Yu. S., Fishchenko E. A.** The Universal Algorithmic Space of Distributed and Parallel Computing, *Informacionnye Tekhnologii i Vichslitel'nye Sistemy*, 2018, no. 2, pp. 78—93, DOI: 10.14357/20718632180207 (in Russian).

23. **Zatuliveter Yu. S., Fishchenko E. A.** Towards Strategic Reengineering the Global Computer Environment for Control of Sustainable Development of Social Systems, *IFAC-PapersOnLine*, 2021, vol. 54, no. 13, pp. 129—133, DOI: 10.1016/j.ifacol.2021.10.432.

УДК 519.682.6 DOI: 10.17587/prin.13.119-123

М. В. Сергиевский, канд. техн. наук, доц., sermax@yandex.ru, Национальный исследовательский ядерный университет "МИФИ", Москва

Метаклассы в UML и в языках программирования

Важными этапами процесса разработки объектно-ориентированных информационных систем являются проектирование и программирование. На этапе проектирования строится модель предметной области, в которой могут быть использованы метаклассы. Но, как известно, в языке UML нет прямой поддержки метаклассов. В статье описан способ, как в ряде случаев можно перейти от моделей с метаклассами к моделям с обычными классами.

Приведены примеры, показывающие, какими возможностями обладают языки программирования Python, Scala и Objective-C для реализации таких моделей. Кроме того, анализируется, как по-разному в этих языках трактуется понятие метакласса.

Ключевые слова: *класс, метакласс, объект, UML, наследование, отношение классификации, предметная область*

Введение

Введем несколько определений, связанных с понятиями объекта, класса и метакласса [1].

Класс — сущность, определяющая структуру и функциональность входящих в его состав однотипных элементов, которые называются объектами.

Объект — экземпляр класса.

Метакласс — это класс, экземплярами которого являются классы.

Между классами может существовать отношение наследования, показывающее, что один класс может наследовать структуру и поведение другого. Отношение наследования может существовать и между метаклассами.

Отношение классификации INSTANCE OF устанавливается между экземпляром класса и самим классом, т. е. оно существует, с одной стороны, между объектами и классами, с другой стороны, между классами и метаклассами.

В настоящее время существует несколько концепций метакласса [2]. По крайней мере две из них получили практическое воплощение.

Первая концепция предусматривает наличие только одного метакласса. Каждый класс всегда имеет строгий шаблон, задаваемый выбранной объектной моделью или языком программирования. Он определяет, например, допустимо ли множественное наследование, какие существуют ограничения на именование классов, как описываются поля и методы, набор существующих типов данных и многое другое. Таким образом, класс можно рассматривать как объект, у которого есть свойства: имя, список полей и их типы, список методов, список аргументов для каждого метода и т. д. Также класс может обладать поведением, т. е. поддерживать реализацию методов. А поскольку для любого объекта существует шаблон, описывающий свойства и поведение этого объекта, значит, его можно определить и для класса. Такой шаблон, задающий различные классы, называется метаклассом. В реальных предметных областях использовать такую концепцию метакласса для построения модели предметной области смысла не имеет, поскольку метакласс в данном случае выступает в роли абстрактной категории, являющейся как бы надстройкой над всеми классами. Ее существование при реализации можно предусмотреть по умолчанию.

При использовании второй концепции допускается существование множества метаклассов. Подобно тому, как объекты со сходными свойствами группируются и являются экземплярами определенного класса, так и классы могут группироваться и приписываться к вполне конкретному метаклассу. Эта концепция близка к идее общего предка для группы классов. И чаще всего она реализуется без введения понятия метакласса с помощью отношения наследования между классами. Именно такая концепция метакласса будет преимущественно рассматриваться в данной работе. Таким образом, классы как объекты будем относить к определенному метаклассу. И таких метаклассов в общем случае может быть несколько.

UML и понятие метакласса

Метакласс используется при моделировании с помощью UML только на уровне метамодели [3]. То есть для точного определения, какие сущности (классы, отношения и т. п.) могут использоваться в модели. В UML все элементы модели являются экземплярами какого-либо метакласса. Например, чтобы в модели могли быть использованы классы, на уровне метамодели должен существовать метакласс Class.

Таким образом, непосредственно использовать метаклассы при построении модели предметной области с помощью UML нельзя. Тем не менее метаклассы как элементы абстракции могут быть необходимы. Существует несколько возможностей представлять метаклассы в диаграммах классов

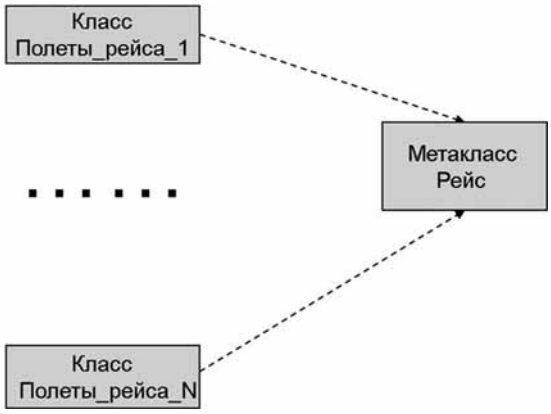


Рис. 1. Модель предметной области с метаклассом

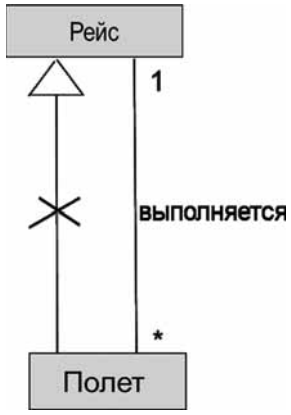


Рис. 2. Модель предметной области "Авиарейсы"

UML. Лучше всего рассмотреть эти возможности на реальных примерах.

Предположим, что предметная область включает данные о авиарейсах и полетах, выполняемых в рамках этих рейсов. Все множество авиаполетов можно разбить на подмножества, относящиеся к одному рейсу. Тогда конкретный рейс можно рассматривать как класс, а все множество рейсов — как метакласс, состоящий из объектов-классов — конкретных рейсов. Модель предметной области в данной трактовке представлена на рис. 1.

Как средствами UML описать такую ситуацию? Поскольку в UML нет возможности использовать метаклассы для описания предметной области, нужно довольствоваться обычными классами и объектами [4—6].

Проведем упрощение модели. Будем рассматривать множество рейсов не как метакласс, а как класс, объектами которого являются отдельные рейсы, идентификаторами которых служат номера рейсов. Для простоты будем считать, что совмещенных рейсов нет. Отдельные рейсы в данном случае являются обычными объектами, а не классами. В рамках стандартного UML можно предложить три модели, описывающие такую ситуацию.

Модель 1. В этой модели для представления рейсов используется абстрактный класс Рейс. Тогда между ним и другими классами предметной области (в первую очередь, полетами) могут существовать только отношения наследования (в терминологии UML — обобщения) и зависимости. Отношений ассоциации в данном случае быть не может, поскольку объектов абстрактного класса не создается; достаточно использовать только отношение обобщения.

Модель 2. В этой модели между классами предметной области в принципе могут быть предусмотрены два отношения: обобщение и ассоциация. Существование отношения ассоциации сразу же говорит то, что будут создаваться объекты родительского класса, т. е. класса Рейс. И объекты класса-потомка Полет будут обязательно связаны с объектами родителя (все зависит от кратности). Тогда нет смысла вообще использовать отношение обобщения, чтобы не дублировать атрибуты у объектов потомка и его потенциального родителя. Недостающие атрибуты объект, который потенциально является потомком, заимствует у прежнего родителя, с которым он будет связан только отношением ассоциации (рис. 2).

Модель 3. В этой модели полеты одного рейса объединяются в классы, и таких классов столько же, сколько различных рейсов. Целесообразно поступать именно так, если рейсы имеют определенную специфику, выражающуюся в наличии уникальных атрибутов и операций. Но поскольку в UML метаклассы не поддерживаются, вместо метакласса в этой модели используется абстрактный класс-предок. Преимущество такого подхода в том, что он дает возможность унифицировать часть операций для всех классов-полетов (рис. 3).

Расширим представление о предметной области, чтобы она точнее описывала реальную ситуацию. Включим в нее данные о рейсах, полетах, выполняемых в рамках определенных рейсов, и пассажирах. Класс Рейс включает следующие атрибуты: номер; пункт отправления, пункт назначения, день недели; время вылета. Класс Полет содержит такие атрибуты: дата; время в пути; доп. параметры, характеризующие полет. Класс Пассажир содержит данные о пассажирах и билетах. Понятно, что атрибуты рейса неявно являются и атрибутами полета. Но, если существует отношение ассоциации между рейсом и полетом, то повторять эти атрибуты в классе Полет нецелесообразно (рис. 4).

Приведем еще один пример [7]. Предположим, что предметная область включает данные о литературных произведениях и книгах (не сборниках), как материальных объектах. Литературное произведение

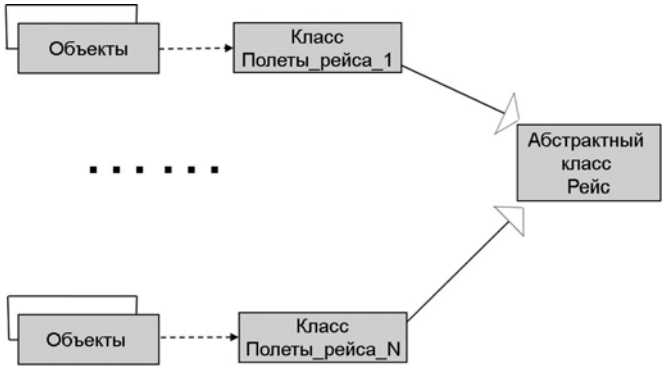


Рис. 3. Модель предметной области "Авиарейсы " с множеством классов

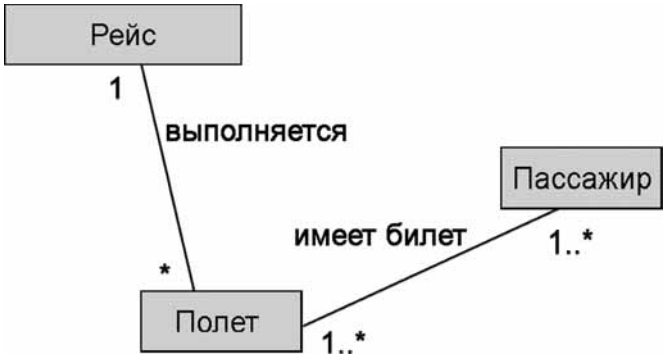


Рис. 4. Расширенная модель предметной области "Авиарейсы"

как класс характеризуется следующими атрибутами: название, годы написания, оглавление, объем в знаках. Книга, как класс материальных объектов, может иметь такие атрибуты: число страниц, переплет, состояние, идентификационный номер. Очевидно, что атрибуты произведения являются и атрибутами книги. Можно рассматривать произведение, как метакласс, состоящий из классов — конкретных произведений, а книги — как экземпляры конкретных произведений. Но разумнее поступить так же в предыдущем случае. То есть, либо отказаться от концепции метакласса, а рассматривать только два обычных класса и связать их не отношением обобщения, а отношением ассоциации, чтобы не дублировать значения атрибутов, относящихся к литературному произведению, в объектах-книгах. Либо объединить объекты-книги в классы, и таких классов будет столько же, сколько различных произведений, а вместо метакласса использовать общий абстрактный класс-предок.

Метаклассы в Python

Как известно, структура объекта полностью определена в его классе. Следовательно, структура любого класса должна быть определена в его метаклассе. Класс должен иметь имя, набор полей и набор методов, т. е. метакласс должен предписывать любому классу иметь именно эти атрибуты. В таком случае метакласс выступает в роли шаблона. Именно такая концепция реализована в языке Python [8].

Классы в Python — это объекты, а значит, как и любой другой объект, класс можно создавать динамически. Именно это и делается в Python во время выполнения оператора `class`. Впоследствии с классом, как с объектом, можно проводить различные операции, например, добавлять поля и методы. А поскольку класс — это объект, то должен существовать специальный класс (метакласс), экземпляром которого он является. Отметим, что с метаклассом никаких операций не совершается, и, что очень важно: метакласс в Python всего один. Таким образом, метакласс в Python — это некая абстракция, но вполне реальная. Иерархия отношений классификации между объектами, классами и метаклассами приведена на рис. 5.

Но есть и другой способ задания классов. Класс в Python можно не объявлять стандартным образом, а напрямую создавать как специальный объект, используя метод `type`. Модель "Авиарейсы", приведенная на рис. 3, на языке Python может быть реализована путем генерации классов следующим образом:

```
type (Var, (Flight), {Date, Time}),
```

где `Var` — переменная, последовательно принимающая значения `"FlightA"... "FlightU"`, которые являются именами классов, определяющих относящиеся к одному рейсу объекты-полеты; `Flight` — абстрактный класс рейсы; `Date` (дата полета), `Time` (время в пути) — атрибуты классов полеты. Поскольку заранее число классов неизвестно, то их надо создавать динамически. Число созданных классов, таким, образом в разных случаях будет различным.

Параметрами метода `type` являются: имя создаваемого класса, имя класса-предка, списки полей и методов создаваемого класса (в данном случае методов нет). Тогда в принципе можно трактовать `type` как метакласс, который используется для генерации классов. А уже потом при обращении к сгенерированному классу можно, в свою очередь, создавать обычный объект.

Моделирование метаклассов на языке Scala

В большинстве случаев метаклассы нужны, чтобы сначала создавать различные классы, а потом их объекты. И желательно делать это в как можно более общем виде. Но существует хорошо известный шаблон проектирования — Factory [9], который как раз и позволяет делать подобные вещи, т. е. в зависимости от условий создавать объекты разных классов. Чаще всего имя класса, объект которого должен быть создан, или информация о нем передаются из внешнего источника. Можно сказать, что в какой-то степени этот шаблон моделирует работу с метаклассами.

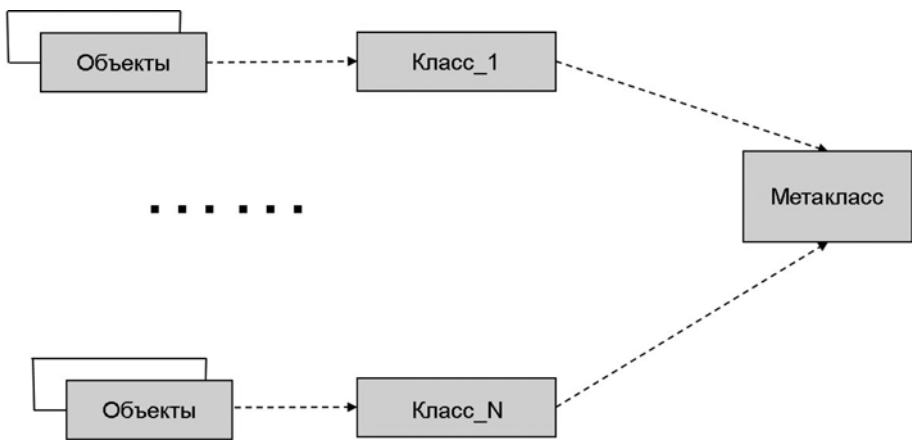


Рис. 5. Иерархия отношений классификации в Python:
---------► — отношение классификации INSTANCE OF

Продемонстрируем, как это может быть сделано на языке Scala [10] для задачи о рейсах и полетах.

```
abstract class Flight{
    val par: String
}
object Flight {
   case class FlightA(par: String) extends
   Flight
. . . . . .
      case class FlightU (par: String)
      extends Flight def apply(par:
      String): Flight={
      if (par=="FlightA")
            FlightA(par)
. . . . . .
           else
            FlightU(par)
  }
}
```

В данном случае, в зависимости от значения параметра `par` создаются объекты разных классов: от `FlightA` до `FlightU`. Делается это следующим образом:

```
var ObjFl = Flight(par)
```

Метод `apply` возвращает значение типа `Flight`, потомками которого являются классы `FlightA,...,FlightU`. Правда, недостатком такого подходя является необходимость предварительно определить все классы. Case-классы, а не обычные классы, использованы здесь, чтобы упростить запись кода, избежав явного создания объектов.

Данные о рейсах, относящиеся к различным множествам объектов-полетов — экземплярам классов `FlightA...FlightU` — удобно хранить в соответствующих классам объектах-спутниках. Например, для класса `FlightA` объект-спутник будет определяться так:

```
Object FlightA {
val NFlight=303
val DepTime=15.30
}
```

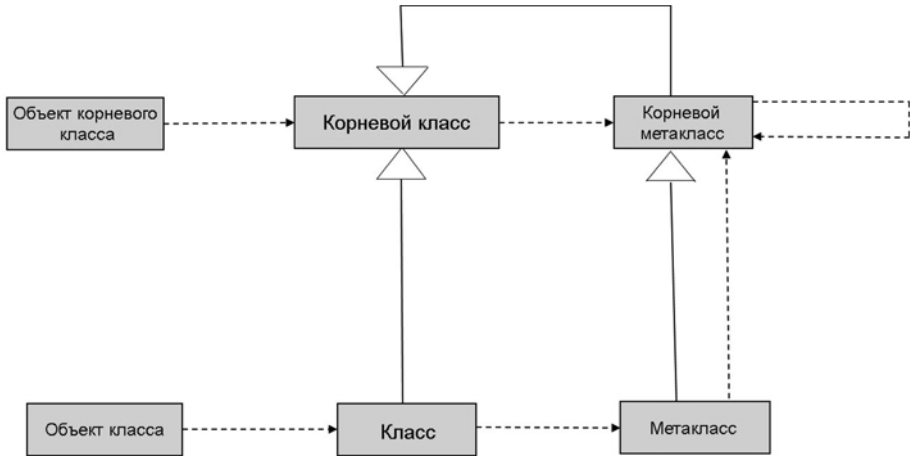


Рис. 6. Иерархия отношений объект — класс — метакласс в Objective-C

Метаклассы в языке Objective-C

Иная концепция метакласса используется в языке Objective-C [11]. Основной постулат: каждый элемент в Objective-C является объектом. Также считается, что у каждого класса есть свой уникальный метакласс. В теле каждого объекта Objective-C (т. е. в области памяти, которая для него распределяется) обязательно есть ссылка на класс. Поскольку класс — это тоже объект, у него должна быть ссылка на его класс, т. е. метакласс. Определяя класс, по умолчанию создаем парный для него метакласс. Говоря упрощенно, метакласс характеризуется теми статическими методами и полями, которые есть в классе. Таким образом, в Objective-C принята концепция раздельного хранения обычных методов, которые могут вызываться только тогда, когда объект создан, и методов класса, которые могут быть вызваны до создания объектов.

Наличие метаклассов обязательно, поскольку именно метаклассы хранят статические поля и методы классов. Следовательно, для каждого класса должен быть создан уникальный метакласс, так как каждый класс потенциально может иметь уникальный набор статических полей и методов.

Для создания пары класс — метакласс используется метод `objc _ allocateClassPair`. Для рассматриваемого примера, связанного с предметной областью "Авиарейсы", данные, относящиеся ко всем объектам класса (такие как номер рейса `NFlight`, время отправления `DepTime`), размещаются в метаклассе. Напомним, что в Objective-C метаклассы обычно создаются средой выполнения по умолчанию и не имеют имен. Иерархии отношений классификации и наследования между объектами, классами и метаклассами в языке Objective-C приведены на рис. 6.

Обратим внимание на то, что только корневой класс не имеет класса-предка, а корневой метакласс является объектом самого себя.

Заключение

Концепция метакласса появилась достаточно давно, но реального применения до последнего времени практически не находила. Причин здесь несколько. Основная состоит в том, что в реальных предметных областях аналогов метаклассов, по существу, нет. Поэтому нет метаклассов и в универсальном средстве моделирования программных систем — UML. Но как элементы абстракции метаклассы могут возникать. В принципе можно допустить существование множества метаклассов. То есть обычные классы могут группироваться и приписываться к вполне конкретному метаклассу. Показано, как в ряде случаев можно перейти от моделей с метаклассами к стандартым UML-моделям.

В отличие от UML в некоторых языках программирования явно

или неявно понятие метакласса все-таки используется. Тут, в первую очередь, надо говорить о языках Python и Objective-C. В Objective-C понятие метакласса далеко от канонического: допускается существование множества метаклассов, поскольку автоматически в пару к каждому обычному классу добавляется метакласс, хранящий статические поля и методы. В Python принята концепция единого метакласса для всех классов. В этом случае метакласса в модели предметной области создавать не нужно, но он по умолчанию будет присутствовать в реализации.

Список литературы

1. **Olive A.** Conceptual Modeling of Information Systems. Springer Science & Business Media, 2007. 455 p.
2. **Forman I. R., Danforth S. H.** Putting Metaclasses to Work, Addison-Wesley, 1998. 447 p.
3. **Rumbaugh J., Jacobson I., Booch G.** Unified Modeling Language. 2nd edition, Boston, Addison-Wesley, 2004, 742 p.
4. **Sergievskiy M.** N-ary Relations of Association in Class Diagrams: Design Patterns. // International Journal of Advanced Computer Science and Applications. 2016. Vol. 7, No. 2. P. 265—268.
5. **Sergievskiy M.** Modeling Unified Language Templates for Designing Information Systems. // Automatic Documentation and Mathematical Linguistics. 2020. Vol. 54, No. 1. P. 26—35.
6. **Sergievskiy M., Kirpichnikova K.** Optimizing UML Class Diagrams. // ITM Web of Conferences. 2018. Vol. 18. Article Number. 03003.
7. **Сергиевский М. В.** Шаблоны унифицированного языка моделирования для проектирования программных систем. // Научно-техническая информация. Серия 2: Информационные процессы и системы. 2020. № 1. С. 19—27.
8. **Рамальо Л.** Python. К вершинам мастерства. М.: ДМК-Пресс, 2015. 768 с.
9. **Gamma E., Johnson R., Helm R., Vlissides J.** Design Patterns. Elements of Reusable Object-Oriented Software. Addison-Wesley, 2001. 416 p.
10. **Одерски М., Спун Л., Веннерс Б.** Scala. Профессиональное программирование, 3-е изд. СПб.: Питер, 2017. 1100 с.
11. **Gallagher M.** What is a meta-class in Objective-C? 2010. URL: https://www.cocoawithlove.com/2010/01/what-is-meta-class-in-objective-c.html

Metaclasses in UML and in Programming Languages

M. V. Sergievskiy, sermax@yandex.ru, National Research Nuclear University MEPhI (Moscow Engineering Physics Institute), Moscow, 115409, Russian Federation

Corresponding author:

Sergievskiy Maxim V., Associated Professor, National Research Nuclear University MEPhI (Moscow Engineering Physics Institute), Moscow, 115409, Russian Federation

Received on January 16, 2022
Accepted on January 26, 2022

Design and programming are important stages of the development process of object-oriented information systems. At the design stage, a domain model is built, in which metaclasses can be used. But, as widely known, there is no direct support for metaclasses in UML. The article describes how in some cases it is possible to switch from models with metaclasses to models with regular classes. Several variants of such a transition are given. Two concepts of the metaclass are considered, which have now found application.

Examples are given showing what capabilities the Python, Scala and Objective-C programming languages have for implementing such models. In addition, it analyzes how the concept of a metaclass is interpreted differently in programming languages.

Keywords: *class, metaclass, object, UML, inheritance, classification relation, domain area*

For citation:

Sergievskiy M. V. Metaclasses in UML and in Programming Languages, *Programmnaya Ingeneria*, 2022, vol. 13, no. 3, pp. 119—123.

DOI: 10.17587/prin.13.119-123

References

1. **Olive A.** *Conceptual Modeling of Information Systems,* Springer Science & Business Media, 2007, 455 p.
2. **Forman I. R., Danforth S. H.** *Putting Metaclasses to Work*, Boston, Addison-Wesley, 1998, 447 p.
3. **Rumbaugh J., Jacobson I., Booch G.** *Unified Modeling Language*, 2nd edition, Boston, Addison-Wesley, 2004, 742 p.
4. **Sergievskiy M.** N-ary Relations of Association in Class Diagrams: Design Patterns, *International Journal of Advanced Computer Science and Applications*, 2016, vol. 7, no. 2, pp. 265—268.
5. **Sergievskiy M.** Modeling Unified Language Templates for Designing Information Systems, *Automatic Documentation and Mathematical Linguistics*, 2020, vol. 54, no. 1, pp. 26—35.
6. **Sergievskiy M., Kirpichnikova K.** Optimizing UML Class Diagrams, *ITM Web of Conferences*, 2018, vol. 18, article number 03003.
7. **Sergievsky M. V.** UML Templates for Information Systems Design, *Nauchno-tekhnicheskaya informaciya. Seriya 2: Informacionnye processy i sistemy*, 2020, no. 1, pp. 19—27 (in Russian).
8. **Ramalho L.** *Python. To the heights of mastery,* Moscow, DMK-Press, 2015, 768 p. (in Russian).
9. **Gamma E., Johnson R., Helm R., Vlissides J.** *Design Patterns. Elements of Reusable Object-Oriented Software,* Addison-Wesley, 2001, 416 p.
10. **Odersky M., Spoon L., Winners B.** *Scala. Professional programming,* 3rd edition, Saint Petersburg, Piter, 2017, 1100 p. (in Russian).
11. **Gallagher M.** What is a meta-class in Objective-C? 2010, available at: https://www.cocoawithlove.com/2010/01/what-is-meta-class-in-objective-c.html

УДК 004.658 DOI: 10.17587/prin.13.124-131

Р. Э. Асратян, канд. техн. наук, вед. науч. сотр., rubezas@yandex.ru, Институт проблем управления им. В. А. Трапезникова Российской академии наук, Москва

Защищенный сетевой канал для web-сервисов на основе SSL/TLS в среде Linux

Рассмотрен подход к организации безопасного взаимодействия в распределенных системах через общедоступную сеть, использующий организацию защищенных каналов связи на основе протокола SSL/TLS. В отличие от технологии VPN, описываемый подход строго ориентирован на поддержку HTTP-взаимодействий, что позволяет подключить средства аутентификации и разграничения прав доступа к информационным ресурсам на основе сертификатов открытого ключа клиента в качестве готовых технических решений. Описана реализация подхода в среде Linux и результаты экспериментального исследования.

Ключевые слова: *распределенные системы, информационная безопасность, web-сервисы, технология SSL/TLS, сертификаты открытого ключа, Linux*

Введение

Внимание к средствам информационной безопасности в Интернете постоянно растет и приобретает почти всеобщий характер [1-4]. Эту тенденцию легко проследить: все последние годы число сайтов, использующих защищенный протокол HTTPS, непрерывно увеличивалось, и сегодня они заняли главенствующее положение в Интернете. Пожалуй, единственной областью, где классический HTTP еще сохраняет свои позиции, остаются распределенные системы (РС), построенные на основе web-сервисов [5, 6], но и здесь ситуация постепенно меняется. По крайней мере в среде Windows технология WCF (*Windows Communication Foundation*) [7—9] завоевывает все большее число сторонников среди разработчиков РС, которые все чаще делают выбор в пользу HTTPS в качестве сетевого "транспорта" для взаимодействий клиента и сервиса.

В среде Linux [10] ситуация складывается несколько иначе. Это во многом связано с тем, что поддержка WCF в среде программирования MonoDevelop — в основном средстве разработки РС на основе архитектуры .NET — пока еще находится в стадии становления с не вполне ясными перспективами. Тем не менее поддержка SOAP-сервисов в Linux хорошо зарекомендовала себя уже в течение ряда лет (что стало одним из факторов повышения интереса к среде Linux, которое можно наблюдать в последние годы).

Одной из важных задач, которые приходится решать разработчикам РС в Linux, является задача организации безопасного взаимодействия как с вновь создаваемыми, так и с уже созданными web-сервисами, ориентированными на протокол HTTP/SOAP. Общепринятый подход к решению этой задачи заключается в использовании VPN [11, 12], т. е. технологии создания защищенных каналов взаимодействия с ограниченным доступом через общедоступную сеть. Главным преимуществом этого подхода является его высокая универсальность: так как средства защиты в VPN подключаются на нижних уровнях иерархии протоколов OSI (как правило, не выше сетевого), то его "бенефициарами" оказываются все протоколы вышестоящих уровней. Альтернативный подход основан на применении протокола SSL/TLS для построения высокоуровневых защищенных сетевых туннелей [13, 14]. В отличие от VPN, средства защиты подключаются в этом случае на транспортном уровне иерархии протоколов, что также обеспечивает весьма высокую универсальность подхода: защищенный канал может быть использован для взаимодействия по любому протоколу уровня приложения.

Однако высокая универсальность рассмотренных подходов имеет и отрицательную сторону. Она не позволяет продвинуться в направлении создания готовых полезных решений в области защиты информации в той же степени, которая доступна более специализированным технологиям, сфокусированным на поддержку строго определенных протоколов приложения. Например, в области авторизации и тонкого разграничения прав доступа к данным на уровне не только отдельных web-сервисов, но и отдельных сервисных функций (методов). Известные универсальные средства организации защищенных сетевых каналов не предоставляют такой возможности (во всяком случае, не предоставляют в форме готового к использованию решения).

В данной статье описана технология построения защищенных каналов взаимодействия для РС, использующих технологию web-сервисов, в среде Linux. Основная идея подхода основана на использовании SSL/TLS для надстройки защищенного канала над уже открытым TCP-соединением [13, 15]. Однако

в отличие от известных подходов, строгая ориентация на поддержку web-сервисов позволяет включить в защищенный канал средства анализа HTTP/SOAP-траффика и готовые средства авторизации и тонкого разграничения прав доступа к данным, основанные на разборе HTTP/SOAP-заголовков в информационных запросах и на значениях реквизитов клиентов, содержащихся в предъявленных сертификатах открытого ключа [1]. Другими словами, предлагаемый подход представляет собой попытку получить выигрыш в функциональных возможностях канала за счет сознательного проигрыша в универсальности. Важно отметить, что подобное повышение функциональных возможностей канала, по-видимому, не имеет альтернативы в ситуации, когда требуется добавить средства тонкого разграничения прав доступа в РС без доработки уже существующих клиентских и серверных компонентов.

Описываемый защищенный канал основан на использовании специальных шлюзов, обеспечивающих переключение с HTTP на HTTPS на стороне клиента и переключение с HTTPS на HTTP на стороне web-сервера и составляющих защищенный канал взаимодействия (HTTPS-канал) для компонентов РС. Предполагается, что как программы клиента, так и web-серверы размещены в одних защищенных частных сетях (или даже на одних сетевых узлах) с обслуживающими их шлюзами и только взаимодействие между шлюзами осуществляется через общедоступную сеть (рис. 1). Важно подчеркнуть, что эти дополнительные возможности не диктуются потребностям конкретного проекта, они реализованы в форме готовых к использованию "общих" решений.

Основы организации защищенного канала

Функционирование защищенного канала проиллюстрировано на рис. 1. Его работа организована в соответствии с изложенными далее основными положениями.

- Защищенный канал опирается на соединение двух базисных технологий: SSL/TLS и технологии proxy-серверов. По своему статусу в системе и клиентский, и сервисный шлюзы представляют собой proxy-серверы — постоянно активные программы ("демоны"), выполняющие функции посредников между клиентскими и сервисными компонентами РС. В данном случае смысл "посредничества" заключается в криптозащите данных.
- Объектами обработки в канале являются не отдельные IP-пакеты, а электронные документы: информационные запросы к web-сервисам и результаты их обработки (ответы) в формате HTTP/SOAP (рис. 1).
- Защита данных в канале основывается на сертификатах открытого ключа клиентского и серверного шлюзов. Хотя передача сертификата клиента серверу, вообще говоря, является опциональной в современных версиях протоколов SSL/TLS, в данном случае она является обязательной: серверный шлюз немедленно разрывает соединение с клиентским шлюзом, если в результате процедуры "рукопожатия" (*handshaking*) сертификат последнего не был получен. Именно на характеристиках владельцев сертификатов (а не на учетных записях, паролях и т. п.) строятся средства аутентификации и контроля прав доступа к сервисам (разумеется, корректность данных в сертификатах обязательно проверяется на основе сертификата "доверенной" организации).
- Средства защиты опираются на возможности библиотек libssl и libcrypt для ОС Linux и сосредоточены исключительно в рамках защищенного канала. Ни клиентские, ни серверные компоненты РС не должны иметь дело ни с крипто-функциями, ни с сертификатами или закрытыми ключами. Это требование означает, что средства контроля прав доступа к сервисам, основанные на сертификатах, должны быть реализованы в самом канале, а не в клиентских и/или сервисных компонентах. Важно отметить, что в описанном подходе некорректные (нарушающие права доступа) запросы вообще не допускаются до сервера, они отвергаются еще на уровне канального шлюза. Разумеется, как программные модули шлюзов, так и их конфигурационные файлы (см. далее) должны быть доступны по записи только администраторам РС.
- Каждый клиентский шлюз способен взаимодействовать не с одним, а со многими серверными шлюзами поочередно. Выбор серверного шлюза осуществляется на основе интернет-имени адресуемого web-сервера с использованием таблицы маршрутизации, содержащейся в конфигурационном файле клиентского шлюза. Важная особенность описываемого подхода заключается в том, что функции маршрутизации "переплетаются" здесь с функциями защиты: упомянутая таблица содержит не только интернет-имена (или адреса) серверных шлюзов, но и требования к их сертификатам, сформулированные в терминах ограничений на реквизиты владельца (защита от сфальцифицированного серверного шлюза). С точки зрения пользователя самым важным компонентом шлюза является его конфигурационный файл ssltunnel.cfg, который содержит параметры (настройки), управляющие его работой (рис. 1). Параметры задаются в форме

```
ключевое слово = значение
```

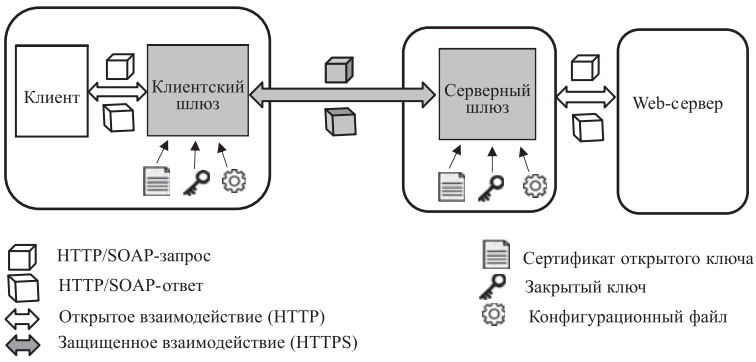


Рис. 1. Основы организации защищенного канала

Основные настройки шлюза защищенного канала

№	Ключевое слово	Настройка	Значение по умолчанию
1	ListenPort	Номер порта для входящих соединений	8128
2	ListenIp	IP-адрес (или имя) сетевой карты	127.0.0.1
3	ReadTimeout	Таймаут чтения из сокета, с	180
4	WriteTimeout	Таймаут записи в сокет, с	60
5	CertificateFile	Полное имя файла сертификата открытого ключа шлюза (или имя директории для поиска)	/media
6	PrivateKeyFile	Полное имя файла закрытого ключа шлюза (или имя директории для поиска)	/media
7	TrustedDir	Полное имя директории доверенных сертификатов	/etc/ssl/certs

Каждая настройка размещается в отдельной строке. В таблице перечислены основные параметры и связанные с ними ключевые слова.

Первые четыре параметра вполне характерны для сетевых серверных программ. Сертификат и закрытый ключ шлюза загружаются из файлов с расширением ".pem" с помощью функций SSL_CTX_use_certificate_file() и SSL_CTX_use_Privatekey_file() библиотеки libssl соответственно (они могут быть подготовлены с помощью утилиты openssl). Отметим, что вместо имен файлов могут быть указаны имена директорий: в этом случае шлюз осуществит поиск файлов в соответствующей директории и ее поддиректориях. Сертификат открытого ключа будет загружен из первого же найденного файла, имя которого заканчивается на "cert.pem" (например, My_cert.pem), а закрытый ключ — из первого же найденного файла, имя которого заканчивается на "key.pem" (например, My_key.pem). Такая организация позволяет считывать сертификаты и ключи со съемных носителей, задавая имя директории, в которую они монтируются (например, "/media"). Несмотря на имеющиеся в Linux развитые средства защиты директорий от несанкционированного доступа, хранение клиентских сертификатов и закрытых ключей на съемных носителях представляется наиболее безопасным.

Кроме параметров, указанных в таблице, конфигурационный файл может содержать разделы, отражающие специфику клиентского и серверного шлюзов. Эти разделы прямо относятся к поиску и проверке подлинности серверного шлюза (раздел [Forward] для клиентского шлюза) или к проверке прав клиента на доступ к web-сервисам и отдельным сервисным функциям (раздел [Access] для серверного шлюза). Лучше всего пояснить структуру этих разделов на конкретном примере.

Рассмотрим следующий пример. Предположим, что большая организация Titan имеет центральный офис в Москве и филиалы в областных центрах страны, а на web-серверах каждого офиса имеется web-сервис Staff и набор сервисных функций, предназначенных для регистрации и учета служащих этого филиала. К их числу относится ряд достаточно простых функций, обеспечивающих добавление, удаление и коррекцию записей о служащих в базе данных филиала (AddPerson, DeletePerson и CorrectPerson) и функцию ViewPersons, позволяющую получить список служащих офиса. Кроме того, на web-серверах каждого офиса имеется web-сервис Stat, который содержит функцию Report, позволяющую сформировать отчет о кадровой статистике филиала за любой период времени (например, о динамике средней зарплаты служащих в разрезе подразделений). Предположим также, что служащие компании имеют возможность обращаться к web-сервисам компании дистанционно со своих служебных или домашних рабочих станций через Интернет с использованием защищенных HTTPS-каналов. В целях безопасности web-серверы каждого офиса "спрятаны" в его частной сети, а доступ к сервисам из Интернета осуществляется через серверный шлюз защищенного канала, размещенный на выделенном сервере, имеющем подключения и к частной сети, и к Интернету. Будем считать, что компания снабжает каждого служащего личным закрытым ключом и сертификатом открытого ключа, удостоверенным электронной подписью центрального офиса. Файл сертификата главного офиса имеется на всех рабочих станциях и серверных шлюзах в директории /etc/ssl/certs/main_office. На рис. 2 приведен пример конфигурационного файла клиентского шлюза (размещенного на клиентской рабочей станции), а на рис. 3 — пример конфигурационного файла серверного шлюза.

Раздел [Forward] включает последовательность строк, каждая из которых содержит описание одного серверного шлюза. Это описание связывает интернет-имя адресуемого web-сервера с основными характеристиками обслуживающего его серверного шлюза — IP-адресом (или интернет-именем) и требования к реквизитам владельца (в формате, соответствующем стандарту X509), которым должен удовлетворять сертификат сервера. Например, если клиентская программа выполняет вызов web-сервиса с URL, равным http://www.titan.spb.ru/staff.asmx, то в соответствии со второй строкой в разделе [Forward]:

```
ListenPort=8128
CertificateFile=/media
PrivateKeyFile=/media
TrustedDir=/etc/ssl/certs/main_office

[Forward]
www.titan.moscow.ru 193.182.12.1 C=Russia,L=Moscow,O=Titan,CN=TitanSrv
www.titan.spb.ru 47.172.33.2 C=Russia,L=StPeterburg,O=Titan,CN=TitanSrv
. . .
www.titan.omsk.ru 146.58.130.3:8443 C=Russia,L=Omsk,O=Titan,CN=TitanSrv
```

Рис. 2. Пример конфигурационного файла клиентского шлюза

```
ListenPort=443
ListenIp=47.172.33.2
CertificateFile=/media
PrivateKeyFile=/media
TrustedDir=/etc/ssl/certs/main_office

[Access]
www.titan.spb.ru/staff.asmx C=Russia,L=StPeterburg,O=Titan
  AddPerson   C=Russia,L=StPeterburg,O=Titan,OU=Inform Department
  DeletePerson C=Russia,L=StPeterburg,O=Titan,OU=Inform Department
  CorrPerson C=Russia,L=StPeterburg,O=Titan,OU=Inform Department
  ViewPersons
www.titan.spb.ru/stat.asmx
  Report  C=Russia,L=StPeterburg,O=Titan,CN=Admin*:* |
         C=Russia,O=Titan,CN=Top*:*
```

Рис. 3. Пример конфигурационного файла серверного шлюза

- клиентский шлюз защищенного канала выполнит сетевое соединение с серверным шлюзом с IP-адресом 47.172.33.2 (порт 443 предполагается по умолчанию) и инициирует создание защищенного канала в соответствии с технологией SSL/TLS;
- если поле Subject Name в сертификате, полученном от серверного шлюза в результате процедуры "рукопожатия" (*handshaking*), будет содержать реквизиты владельца, отличные от C=Russia, L=StPeterburg, O=Titan, CN=TitanSrv (страна Russia, город StPeterburg, организация Titan, имя/название TitanSrv), то клиентская программа получит сообщение о сфальсифицированном серверном шлюзе и соединение с последнем будет немедленно разорвано.

Раздел [Access] на рис. 3 содержит описание ограничений доступа к web-сервисам санкт-петербургского филиала. Здесь легко увидеть строки, содержащие интернет-имена двух web-сервисов (www.titan.spb.ru/staff.asmx и www.titan.spb.ru/stat.asmx). Кроме имени сервиса каждая из этих строк может содержать требования к реквизитам клиентов, которым разрешено обращаться к данному сервису в уже знакомой нам форме: содержание поля Subject Name в сертификате, полученном от клиентского шлюза в результате процедуры "рукопожатия" (*handshaking*), должно соответствовать этим требованиям. После строки, содержащей имя web-сервиса, могут следовать строки, содержащие имена отдельных функций. Эти строки также могут содержать требования к реквизитам клиентов, специфичные для той или иной функции. В приведенном примере только служащие санкт-петербургского офиса из подразделения "Inform Department" имеют право вызывать функции AddPerson, DeletePerson и CorrPerson, а функция ViewPerson доступна любому служащему офиса.

Права доступа к функции Report определяются несколько сложнее. В данном примере предполагается, что в соответствии с корпоративным стандартом реквизит CN (*Common Name*) в сертификате служащего должен быть задан в формате "должность:имя". В приведенном примере право доступа к функции имеют все служащие санкт-петербургского офиса, название должности которых начинается со слова "Admin" (символ "*" означает "любая подстрока"), а также служащие любого офиса компании, название должности которых начинается со слова "Top" (символ "|" в данном случае означает "или").

Сообщение о сфальсифицированном сервере или о нарушении прав доступа к сервисам передаются в программу клиента в форме HTTP-ответа с кодом 500 (Внутренняя ошибка сервера), содержащего текст сообщения в формате SOAP. Этот ответ создает исключительную ситуацию в программе клиента (если вызов сервисной функции заключен в блок "try—catch", то программа клиента легко может получить текст сообщения из параметра предложения catch). Разумеется, все это справедливо и в отношении любых других диагностических сообщений от шлюзов, связанных с неудачной попыткой сетевого соединения, с ошибкой чтения из сокета или записи в сокет, с некорректным сертификатом и т. п. Если диагностическое сообщение порождено клиентским шлюзом, то оно передается непосредственно программе клиента, а если серверным шлюзом, то оно сначала передается в клиентский шлюз, а потом программе клиента.

Логика работы шлюзов

Как и у всякой программы на языке C++ (использование libssl естественно приводит к выбору именно этого языка программирования), сразу после запуска шлюза управление получает "главная" функция main. Она немедленно выполняет ряд действий, характерных для фоновых программ:

- переключение в режим "демона" с разрывом связей с монитором, клавиатурой и мышью;
- установка собственных обработчиков так называемых "сигналов" ОС Linux для потребного реагирования на экстренные ситуации (например, для создания записи в журнале о получении сигнала SIGTERM перед прекращением работы по команде оператора).

Кроме того, функция main выполняет чтение всех настроек из конфигурационного файла, загружает закрытый ключ, сертификат открытого ключа и управляющую информацию (из разделов [Forward] или [Access]) во внутренние структуры данных и выполняет подключение и инициализацию средств поддержки SSL/TLS из библиотеки libssl. После этого она выполняет действия, характерные для любого сетевого сервера, основанного на TCP/IP [14, 15]: создает сетевой сокет для приема запросов на соединение, привязывает его к порту, номер которого задан в параметре ListenPort (системный вызов bind()); включает режим прослушивания порта (системный вызов listen()); входит в "вечный цикл", обеспечивающий обнаружение и обслуживание запросов на сетевое соединение (системный вызов accept()).

Как и всякий многоканальный сервер, шлюз создает отдельный обрабатывающий программный поток ("нить") для обслуживания каждого запроса таким образом, что сразу несколько запросов могут одновременно находиться в состоянии обслуживания. Каждая обрабатывающая программная нить (вернее, ее основная функция) получает от функции main главный параметр — номер сетевого сокета для "удаленного общения" с программой, запросившей соединения и обслуживания.

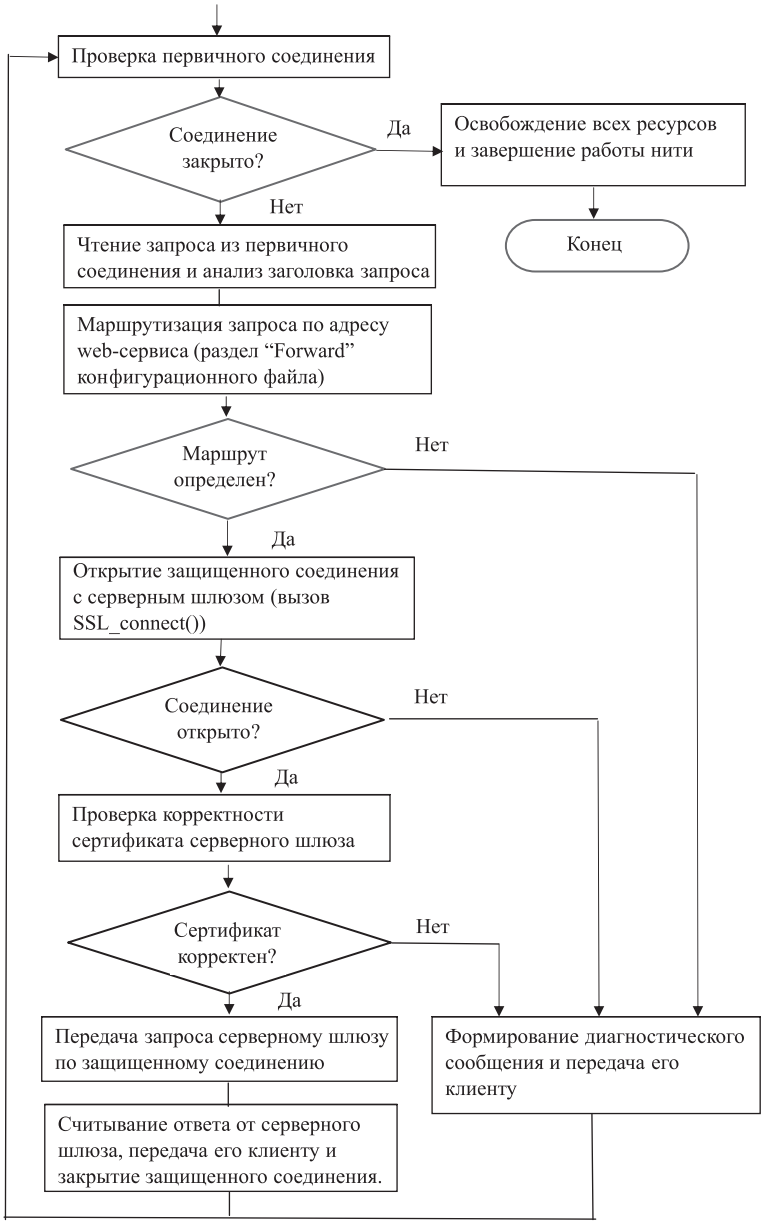


Рис. 4. Алгоритм работы обрабатывающей нити в клиентском шлюзе

Блок-схемы алгоритмов работы обрабатывающих нитей в клиентском и серверных шлюзах приведены на рис. 4 и 5 соответственно. В их "поведении" имеются общие черты: каждая нить соединяет в себе функции сервера и клиента одновременно. Она получает сокет "первичного" соединения от функции main, запрашивает "вторичное" соединение с удаленным сервером и выполняет перенос данных из первичного соединения во вторичное и обратно. Для клиентского шлюза первичное соединение используется для связи с программой клиента, а вторичное — для связи с серверным шлюзом. Для серверного шлюза первичное соединение используется для связи с клиентским шлюзом, а вторичное — с web-сервером. На этом общие черты в поведении нитей заканчиваются.

Главное отличие в логике работы обрабатывающих нитей в клиентском и серверном шлюзах вытекает из того, что в первом случае первичное соединение является незащищенным, а вторичное — защищенным, а во втором случае — наоборот. Поэтому обращение к функциям библиотеки libssl (вызов SSL_connect()) в клиентском шлюзе осуществляется на более поздних этапах обработки запроса, чем в серверном шлюзе. Как видно на блок-схеме рис. 4, создание защищенного канала выполняется после анализа и маршрутизации информационного запроса (поиска адекватного серверного шлюза), а в блок-схеме на рис. 5 — сразу же после создания обрабатывающей нити.

Временные оценки

Как видно из приведенного описания, использование защищенного канала предполагает включение двух промежуточных серверов-шлюзов между клиентской программой и web-сервером. Такая организация не может не вызвать вопроса о размере неизбежной дополнительной задержки в обработке запросов к web-серверу. Для оценки этой задержки была выполнена реализация защищенного канала на основе библиотек libssl и libcrypt (пакет libssl-dev версии 1.1.1d, версия TLS 1.2) в Linux Debian 10 на языке C++ и проведена серия экспериментов в лабораторной сети с использованием web-сервера apache2 версии 2.4.

Так как значение дополнительной задержки при вызове web-сервиса, очевидно, зависит от размеров информационного запроса и результата его выполнения (сетевого трафика), были использованы несколько сервисных функций с различным объемом передаваемых данных в обоих направлениях и различными временами обработки. На рис. 6 приведено время обращения к сервисной функции с малым временем обработки (100 мс) для различных объемов передаваемых данных (10 и 200 Кбайт) в двух режимах: без защиты ("напрямую") и через защищенный канал. На рис. 7 приведено время обращения в тех же условиях к более "медленной" функции (время обработки 500 мс).

Как видно на рис. 6 и 7, при трафике 10 Кбайт после подключения защищенного канала время выполнения запроса к функции с временем обработки 100 мс увеличилось на 13 мс (12 %), а при трафике 200 Кбайт — на 27 мс (21 %). При вызове же функции с временем обработки 500 мс дополнительная задержка составила 14 мс (2,7 %) при трафике 10 Кбайт и 29 мс (5,5 %) при трафике 200 Кбайт. Как и следовало ожидать, при вызове более "медленной" функции потери быстродействия выглядят более умеренными.

На рис. 8 приведены результаты экспериментальной оценки быстродействия защищенного канала, но не в режиме одиночных запросов, а в условиях

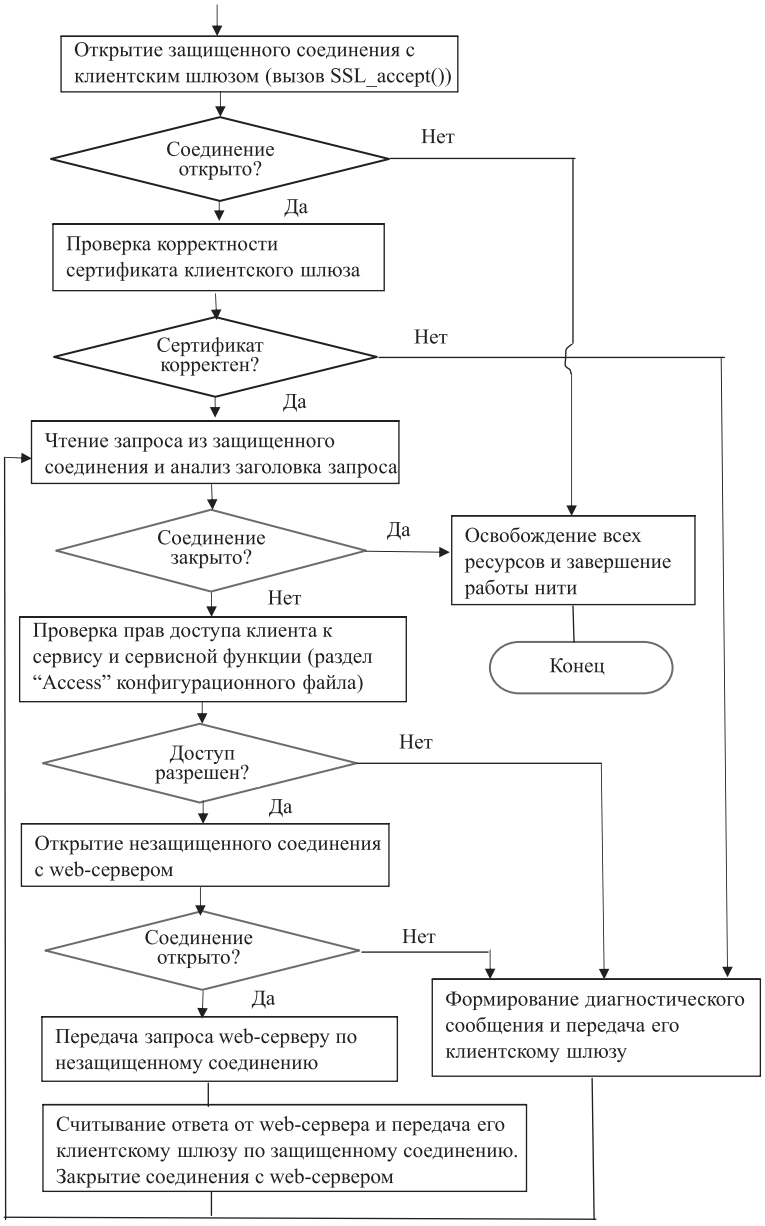


Рис. 5. Алгоритм работы обрабатывающей нити в серверном шлюзе

высокой нагрузки: при параллельной обработке пакетов одновременно поступивших информационных запросов. Кривые, представленные на рис. 8,

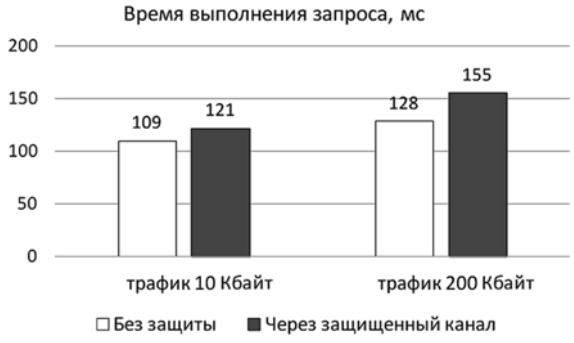


Рис. 6. Время выполнения запроса к "быстрой" функции (100 мс)

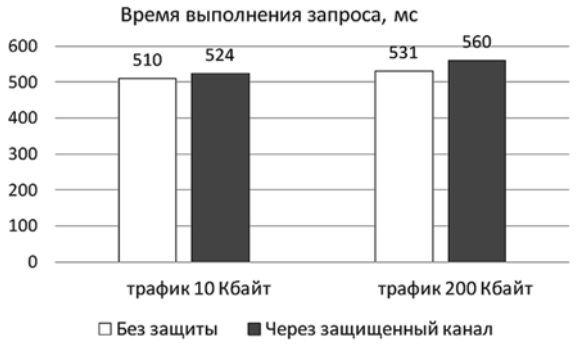


Рис. 7. Время выполнения запроса к более "медленной" функции (500 мс)

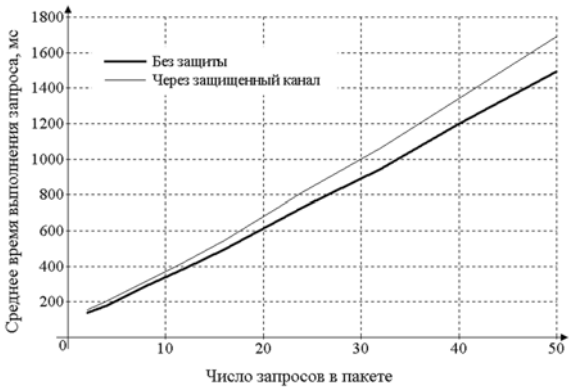


Рис. 8. Среднее время выполнения запроса в пакете одновременных запросов

отражают характерную зависимость среднего времени выполнения запроса от их числа в пакете при вызове "быстрой" сервисной функции со временем обработки 100 мс и объемом передаваемых данных 50 Кб в двух режимах: без защиты ("напрямую") и через защищенный канал. Как видно на рис. 8, обе кривые демонстрируют устойчивый рост, так как с увеличением числа запросов увеличивается и число обрабатывающих нитей, а значит и нагрузка на сервер. Однако относительное увеличение среднего времени выполнения запроса вследствие подключения защищенного канала не превышает 15 %. Примечательно, что это относительное увеличение остается довольно стабильным с ростом числа запросов в пакете.

Как видно из приведенных результатов, при вызове "быстрых" сервисных функций с временем обработки 0,1 с относительные потери быстродействия могут быть существенными, особенно при большом объеме передаваемых данных (см. рис. 6 и 8). Однако при увеличении времени обработки сервисной функции до 0,5 с относительные потери снижаются до нескольких процентов даже при трафике 200 Кбайт. По-видимому, именно сервисные функции со временем обработки 0,5 с и выше составляют область наиболее эффективного применения описанного подхода.

Заключение

Разумеется, при разработке любой РС закладываются и реализуются те или иные средства аутентификации и авторизации еще на уровне клиентских компонентов. Как правило, эти средства бывают "привязаны" к особенностям конкретного проекта. Описанный подход к организации проверки и разграничения прав доступа к сервисам и сервисным функциям в рамках защищенного сетевого канала позволяет наложить общие ограничения на поведение любых клиентов, использующих этот канал. Это особенно важно в тех случаях, когда потребителям информации предоставляется возможность разработки собственных клиентских компонентов для доступа к опубликованным web-сервисам и информационным ресурсам. Как уже отмечалось, в описанном подходе некорректные (нарушающие права доступа) запросы вообще не допускаются до сервера, а отвергаются на уровне канального шлюза. Это дает возможность добавить средства авторизации к уже существующим сервисам, не изменяя их.

Применение описанного выше защищенного канала не сильно усложняет задачу сетевого администрирования. Самое общее требование к сетевой инфраструктуре РС заключается в том, что межсетевые экраны не должны запрещать сетевые соединения по адресам и портам, которые используются канальными шлюзами. Другими словами, клиент должен иметь возможность соединения с клиентским шлюзом, клиентский шлюз — с серверным шлюзом, а северный шлюз — с сервером.

Несколько слов о подключении клиентского шлюза: как следует из логики работы канала, сертификат открытого ключа клиентского шлюза определяет права пользователя на доступ к web-сервисам и сервисным функциям. Фактически это означает, что запущенный клиентский шлюз должен обслуживать только клиентские программы, запущенные определенным пользователем. Наиболее естественный способ удовлетворить это требование заключается в размещении клиентского шлюза на рабочей станции пользователя (см. рис. 1) и задании значения 127.0.0.1 в качестве адреса "прослушивания" (значение по умолчанию) с тем, чтобы только локальные клиентские программы могли им пользоваться. Если же не один, а несколько пользователей имеют право поочередного доступа к рабочей станции (достаточно редкая ситуация в наши дни), то следует воспользоваться возможностью предоставления сертификатов и ключей на съемном носителе. Разумеется, на такой рабочей станции нужно запретить дистанционный вход через удаленный рабочий стол, ssh или telnet [14].

Важно отметить, что описанный канал может быть использован не только в клиентских компонентах РС для вызова функций web-сервисов, но и в служебных программах. Например, утилита wsdl, широко используемая для считывания формализованного описания web-сервиса по сети и автоматической генерации текста связующего модуля, вполне уверенно работает через описанный канал (при условии, что в системных сетевых настройках указаны адрес и порт клиентского шлюза в качестве параметров proxy-сервера). Другими словами, предложенная технология может быть использована не только при эксплуатации РС, но и при их разработке.

В имеющейся на настоящее время реализации защищенного канала функции клиентского и сервисного шлюзов выполняются одной и той же программой (разница только в конфигурационных настройках). Шлюзы приспособлены к работе в режиме "демонов" — невидимых фоновых программ, разрывающих связи с монитором, клавиатурой и мышью сразу после запуска. Во время работы шлюз занимает всего около 1,2 Мбайт оперативной памяти.

Список литературы

1. **Козлов А. Д., Орлов В. Л.** Методы и средства обеспечения информационной безопасности распределенных корпоративных систем. М.: ИПУ РАН, 2017. 156 с.
2. **Салимова Ш. А.** Кибербезопасность в России: актуальные угрозы и пути обеспечения в современных условиях // Достижения вузовской науки 2021: сб. статей XVII Международного научно-исследовательского конкурса, Пенза, 20 января 2021 г. Пенза: Наука и Просвещение, 2021. С. 207—214.
3. **Жаранова А. О., Птицына Л. К.** Анализ влияния распределенности на качество функционирования комплексных систем защиты информации // Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2020): сб. науч. статей IX Международной научно-технической и научно-методической конференции. СПб: СПБГУТ, 2020. С. 324—327.
4. **Згоба А. И., Маркелов Д. В., Смирнов П. И.** Кибербезопасность: угрозы, вызовы, решения // Вопросы кибербезопасности. 2014. № 5. С. 30—38.
5. **Шапошников И. В.** Web-сервисы Microsoft.NET. СПб: БХВ-Петербург, 2002. 336 с.
6. **Мак-Дональд М., Шпушта М.** Microsoft ASP.NET 3.5 с примерами на C# 2008 и Silverlight 2 для профессионалов. М.: Вильямс, 2009. 1408 с.
7. **Liu M.** WCF multi-layer services development with entity framework. Birmingham: Packt Publishing, 2014. 378 p.
8. **Lowy J., Montgomery M.** Programming WCF services: design and build maintainable service-oriented systems. N. Y.: O'Reilly Media, 2015. 1018 p.
9. **Костин Е. И.** Создание службы WCF для использования в клиент-серверном приложении // Инновационные подходы в решении научных проблем: сб. тр. по материалам IV Международного конкурса научно-исследовательских работ, Уфа, 01 марта 2021 г. Уфа: Научно-издательский центр "Вестник науки", 2021. С. 121—126.
10. **Negus C.** Linux Bible. N. J.: Wiley, 2015. 912 p.
11. **Запечников С. В., Милославская Н. Г., Толстой А. И.** Основы построения виртуальных частных сетей. — М: Горячая линия-Телеком, 2011. 248 с.
12. **Акушуев Р. Т.** Принцип работы VPN и его особенности // Modern Science. 2020. № 7. С. 312—314.
13. **Baka P., Schatten J.** SSL/TLS under lock and key: a guide to understanding SSL/TLS cryptography. Keyko books, 2020. 132 p.
14. **Хант К.** TCP/IP. Сетевое администрирование. СПб.: Питер, 2007. 816 с.
15. **Снейдер Й.** Эффективное программирование TCP/IP. Библиотека программиста. СПб.: Символ-Плюс, 2002. 320 с.

Secure Network Channel for Web Services based on SSL/TLS in a Linux Environment

R. E. Asratian, rea@ipu.ru, V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, 117997, Russian Federation

Corresponding author:

Asratian Ruben E., Leading Researcher, V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, 117997, Russian Federation
E-mail: rea@ipu.ru

Received on January 27, 2022
Accepted on February 14, 2022

An approach to the organization of secure interaction in distributed systems via a public network is considered, based on the organization of secure communication channels based on SSL/TLS technology. Unlike VPN technology, the described approach is strictly focused on supporting only HTTP/SOAP interactions in distributed systems, which allows you to implement authentication and authorization based on HTTP-header data and client public key certificates as ready-made technical solutions. The approach implies the use of special gateways that provide switching from HTTP to HTTPS on the client side and switching from HTTPS to HTTP on the web server side and make up a "transparent" communication channel for system components. It is assumed that both client programs and web servers are located in the same secure private network (or even on the same network node) with the gateways serving them, and only the interaction between the gateways is carried out through the public network. The work of gateways is based on the use of SSL/TLS technology to add a secure channel over an already open TCP connection. The main idea of the approach is that in this case, security tools are connected at high levels of the OSI protocol hierarchy, which allows gateways to analyze high-level parameters of information requests and responses of web servers contained in HTTP-headers. And this, in turn, allows you to add additional "intelligence" to the gateways associated with authentication of servers and clients, as well as with the differentiation of access rights to information resources up to individual functions (methods) of web services based on the data contained in "Subject Name" attribute of public key certificates. The implementation of the approach in the Linux environment and the results of an experimental study are described. In particular, the study showed that when calling service functions with a runtime of 0.5 seconds or higher, the secure channel increases the total query execution time by only a few percent, even with a rather large amount of data being transmitted (up to 200 kilobytes).

Keywords: *distributed systems, information security, web services, SSL/TLS technology, public key certificates, Linux*

For citation:

Asratian R. E. Secure Network Channel for Web Services based on SSL/TLS Technology in a Linux Environment, *Programmnaya ingeneria*, 2022, vol. 13, no. 3. 124—131.

DOI: 10.17587/prin.13.124-131

References

1. **Kozlov A. D., Orlov V. L.** *Methods and tools for ensuring information security of distributed corporate systems*, Moscow, IPU RAN, 2017, 156 p. (in Russian).
2. **Salimova S. A.** Cybersecurity in Russia: current threats and ways to ensure in modern conditions, *Dostizhenija vuzovskoj nauki 2021: sbornik statej XVII Mezhdunarodnogo nauchno-issledovatelskogo konkursa,* Penza, 20 January 2021, Penza, Nauka i Prosveshhenie, 2021, pp. 207—214 (in Russian).
3. **Zharanova A. O., Pticyna L. K.** Analysis of the impact of distribution on the quality of functioning of complex information security systems, *Aktual'nye problemy infotelekommunikacij v nauke i obrazovanii (APINO 2020): sbornik nauchnyh statej IX Mezhdunarodnoj nauchno-tehnicheskoj i nauchno-metodicheskoj konferencii*, Saint Petersburg, SPBGUT, 2020, pp. 324—327 (in Russian).
4. **Zgoba A. I., Markelov D. V.** Cyber security: threats, challenges, decisions, *Voprosy kiberbezopasnosti*, 2014, no. 5, pp. 30—38 (in Russian).
5. **Shaposhnikov I. V.** *Web-services Microsoft.NET*, Saint Petersburg, BHV-Peterburg, 2002, 336 p. (in Russian).
6. **Mak-Donald M., Szpuszta M.** *Pro ASP.NET 3.5 in C#2008 Includes Silverlight 2*, Moscow, Willams, 2009, 1408 p. (in Russian).
7. **Liu M.** *WCF multi-layer services development with entity framework*. Birmingham: Packt Publishing, 2014, 378 p.
8. **Lowy J, Montgomery M.** *Programming WCF services: design and build maintainable service-oriented systems*, N. Y., O'Reilly Media, 2015, 1018 p.
9. **Kostin E. I.** Creating a WCF service for use in a client-server application, *Innovacionny'e podxody v reshenii nauchnyh problem: sbornik trudov po materialam IV Mezhdunarodnogo konkursa nauchno-issledovatelskih rabot,* Ufa, 01 March 2021, Ufa, Nauchno-izdatelskij centr "Vestnik nauki", 2021, pp. 121—126 (in Russian).
10. **Negus C.** *Linux Bible*, N.J., Wiley, 2015, 912 p.
11. **Zapechnikov S. V., Miloslavskaya N. G., Tolstoy A. I.** *Virtual private networks building principles*, Moscow, Goryachaya linia, 2011, 248 p. (in Russian).
12. **Akushuev R. T.** The principle of VPN operation and its features, *Modern Science*, 2020, no. 7, pp. 312—314.
13. **Baka P., Schatten J.** *SSL/TLS under lock and key: a guide to understanding SSL/TLS cryptography*, Keyko books, 2020, 132 p.
14. **Hant K.** *TCP/IP. Network administration*, Saint Petersburg, Piter, 2007, 816 p. (in Russian).
15. **Snader J.** *Effective TCP/IP programming*, Saint Petersburg, Simvol-Pljus, 2002, 320 p. (in Russian).

УДК 004.358 DOI: 10.17587/prin.13.132-141

А. Е. Близнюк[1], магистрант, bliznyuksaha@mail.ru, **В. А. Жмудь,** д-р техн. наук, зам. ген. директора по науке[2], гл. науч. сотр.[3], oao_nips@bk.ru,
М. В. Трубин[1], аспирант, morkai@bk.ru, **В. Г. Трубин,** ст. препод. [1], директор[4], trubin@ngs.ru
[1] Новосибирский государственный технический университет,
[2] АО "Новосибирский институт программных систем" (НИПС),
[3] Институт лазерной физики СО РАН, г. Новосибирск,
[4] ООО "КБ Автоматика"

Программирование микроконтроллеров STM32F10x с помощью встроенного загрузчика по USART

Представлен способ программирования микроконтроллеров STM32F10x с помощью встроенного загрузчика по последовательному интерфейсу USART с использованием программы Flash Loader Demonstrator от фирмы ST Microelectronics. Описан процесс создания командного файла для автоматизации процесса программирования. Довольно популярным является способ программирования микроконтроллера через интерфейс SWD с применением аппаратного программатора ST-Link. Однако в силу существующего на настоящее время дефицита микросхем, стоимость электронных компонентов заметно возросла, что сделало этот метод программирования более дорогостоящим. Таким образом, на данный момент способ программирования микроконтроллеров с помощью встроенного загрузчика через USART с применением преобразователя USB to RS-232_TTL является достаточно востребованным. Описанный в статье способ программирования может представлять интерес для студентов и инженеров.

Ключевые слова: *микроконтроллер, STM32, загрузчик, последовательный интерфейс, USART, USB to RS-232_TTL, CH340G, Flash Loader Demonstrator, COM-порт, ST Microelectronics*

Введение

В настоящее время практически любые устройства содержат микроконтроллеры. Существует большое число компаний, которые их производят. Достаточно популярными на рынке являются микроконтроллеры STM32 фирмы ST Microelectronics [1].

В процессе разработки устройств с микроконтроллерами обязательным этапом является их программирование. Микроконтроллеры STM32 можно запрограммировать разными способами. Наиболее известными являются перечисленные далее.

1. Через встроенный загрузчик (*bootloader*). Загрузчик — это специальная программа, которая располагается в постоянной памяти микроконтроллера и может самостоятельно перепрограммировать его внутреннюю Flash-память. Программирование осуществляется с помощью управляющих команд, которые передаются через последовательный интерфейс USART [2]. Для связи с компьютером используется переходник USB to RS-232_TTL [3].

2. С помощью внешнего программатора.

ST-Link — это внутрисхемный отладчик и программатор для микроконтроллеров семейств STM8 и STM32 [4]. В ST-Link для обмена данными с микроконтроллерами STM32 используются интерфейсы JTAG и SWD. В работе [5] представлено практическое руководство по программированию микроконтроллеров STM с помощью внешнего программатора ST-Link.

Стоимость ST-Link на настоящее время примерно в два раза больше, чем цена на преобразователи USB to RS-232_TTL. Еще одним преимуществом преобразователя перед внешним программатором является то обстоятельство, что первый позволяет не только запрограммировать микроконтроллер, но и обмениваться данными между микроконтроллером и персональным компьютером при исполнении программы пользователя простым способом.

Существует множество преобразователей USB to RS-232_TTL. Они используются для передачи данных от компьютера на разные устройства и обратно. Также данный преобразователь может использоваться для обновления программы микроконтроллера.

На рынке существует два вида преобразователей, которые различаются по уровню напряжения сигналов в линиях RS-232. *Это напряжение может быть двухполярным или однополярным.* Примером RS-232 с двухполярными сигналами является аппаратный COM-порт на материнской плате персонального компьютера (рис. 1) [6]. Часто аппаратный COM-порт присутствует на материнских платах, но не выводится на заднюю панель системного блока [7].

Пример двухполярного сигнала в линии RS-232 приведен на рис. 2.

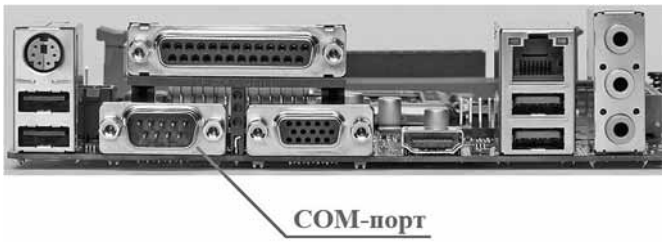


Рис. 1. COM-порт персонального компьютера

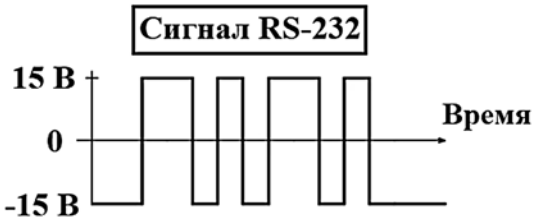


Рис. 2. Двухполярные сигналы уровня RS-232

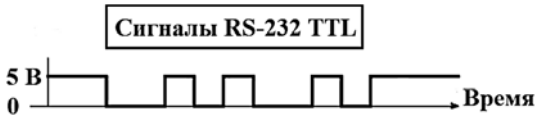


Рис. 3. Однополярные сигналы уровня TTL

Существуют преобразователи, которые имеют однополярные сигналы в линиях RS-232 (рис. 3). С интерфейсом и стандартом RS-232 можно ознакомиться в работе [8].

Перед покупкой преобразователя следует обратить внимание на то, что для программирования микроконтроллера необходим преобразователь с USB в RS-232 *однополярного типа*. В дальнейшем в статье будем называть данный тип преобразователей USB to RS-232_TTL.

На данный момент большой популярностью пользуется преобразователь на микросхеме CH340G фирмы *WCH* (рис. 4).

Данный преобразователь имеет следующие преимущества:

- низкая цена;

Рис. 4. Преобразователь USB to RS-232_TTL на микросхеме CH340G

- широко распространен;
- простая установка драйверов;
- переключение уровней сигналов приема и передачи данных 5 В или 3,3 В с помощью одной перемычки.

Подготовка преобразователя к работе

При подключении преобразователя "USB to RS-232_TTL" к USB-порту компьютера, устройство определяется как "USB to Serial COM Port". После чего автоматически начинается поиск и установка драйверов для преобразователя, но рекомендуется предварительно самостоятельно скачать подходящие драйверы, так как в автоматическом режиме процесс установки не всегда завершается корректно.

Драйверы можно скачать с официального сайта производителя. Для этого на сайте необходимо перейти в раздел "Поиск", который находится в правом верхнем углу веб-страницы [9].

В появившейся поисковой строке необходимо набрать "CH340". После получения результатов поиска нужно выбрать "CH341SER.EXE" (рис. 5).

На следующей открывшейся странице будет представлена информация о выбранном файле. Для загрузки драйверов следует нажать "download" (рис. 6).

file category	file content	version	upload time
DataSheet			
CH340DS1.PDF	CH340 datasheet, USB bus converter chip which realizes USB to serial port/printer port, etc., integrated crystal oscillator, chip information can be customized. Drivers support Windows/Linux/Android/Mac, etc. The datasheet is the the description of USB to serial port.	2.7	2021-07-05
Driver&Tools			
CH341SER.EXE	CH340/CH341 USB to serial port Windows driver, supports 32/64-bit Windows 10/8.1/8/7/VISTA/XP, Server 2016/2012/2008/2003, 2000/ME/98	3.5	2019-03-18

Рис. 5. Результаты поиска

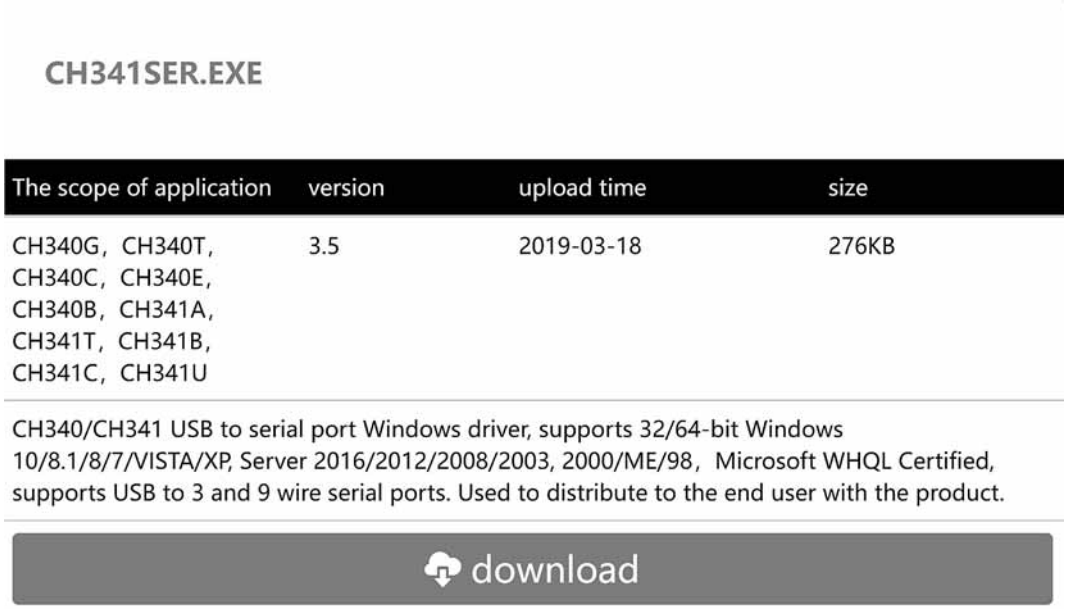


Рис. 6. Информация о выбранном файле

Можно заметить, что драйверы предложены для модели CH341, но они так же подходят для модели CH340G (рис. 7).

После установки программного обеспечения необходимо провести тестирование преобразователя на его работоспособность. Для этого соединим между собой выводы TX и RX (рис. 8).

Далее потребуется любое терминальное приложение для работы с COM-портом компьютера. В данном

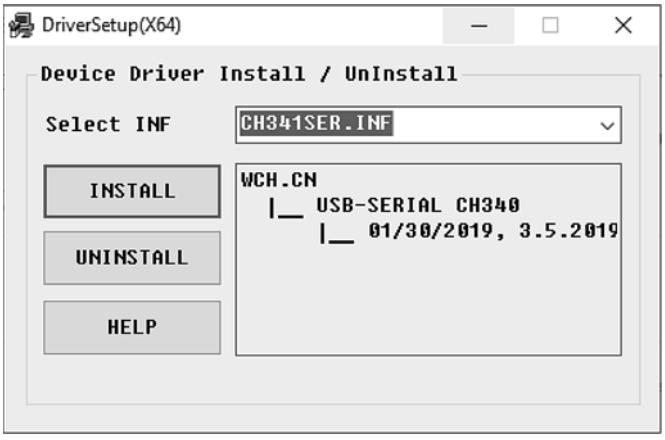


Рис. 7. Драйверы для адаптера CH340G

Рис. 8. Схема подключения для тестирования адаптера

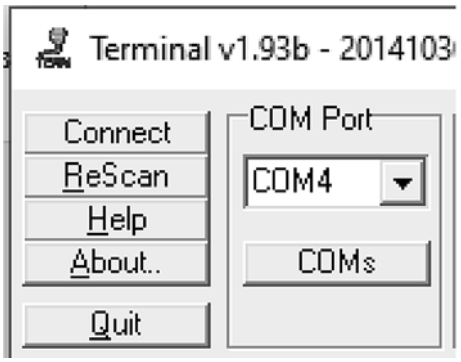


Рис. 9. Соединение с COM-портом в программе Terminal v1.93b

случае используется программа Terminal v1.93b [10]. После этого необходимо подключится к COM-порту. Для этого необходимо нажать "ReScan", чтобы программа обновила список используемых COM-портов, после чего выбрать необходимый порт в выпадающем списке. При нажатии на кнопку "Connect" выполнится соединение с портом (рис. 9).

Далее следует отправить любые символы, которые необходимо ввести в текстовое окно "1" для отправки символов. После этого (при исправном преобразователе) данные символы должны появиться в текстовом окне "2" (рис. 10).

Программирование STM32 с помощью встроенного загрузчика по USART

Для работы с микроконтроллером основными источниками информации являются документы: руководство пользователя "Reference Manual" и техническая документация техническая документация "Data Sheet". В них описаны: характеристики микроконтроллера; назначение выводов; электрические па-

Рис. 10. Результат тестирования работоспособности преобразователя

раметры; описание управляющих регистров; карта памяти; описание периферийных подсистем и т. д.

Перед началом программирования на преобразователе USB to RS-232_TTL необходимо поставить перемычку уровней сигналов в положение 3,3 В как показано на рис. 8.

Существует три основных режима работы, которые приведены в табл. 1.

Отладочная плата на STM32 имеет две перемычки - BOOT_0 и BOOT_1, которые задают режимы работы микроконтроллера после подачи питания на плату, либо после нажатия на кнопку RESET. Необходимо выставить их в режим "2" (рис. 11) [11].

Таблица 1

BOOT-режимы

Режим	Положение		Что запустится после сброса (RESET)
	BOOT_0	BOOT_1	
1	0	0 или 1	Программа из Flash-памяти
2	1	0	Программа загрузчика
3	1	1	Программа в памяти RAM

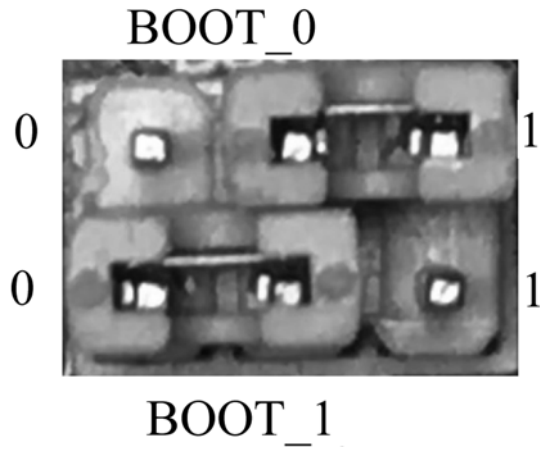


Рис. 11. Перемычки BOOT

Согласно информации из документа AN2606 (табл. 2) [12], загрузчик использует для связи USART под номером 1. На передачу (TX) настроен вывод PA9, а на прием (RX) — вывод PA10.

Согласно данным табл. 2, необходимо соединить вывод RX (PA10) микроконтроллера с выводом TXD преобразователя и вывод TX (PA9) микроконтрол-

Таблица 2

Конфигурация STM32F10x в режиме загрузчика

Подсистема	Состояние	Примечание
USART1	Включен	Первоначальная конфигурация USART1: 8 бит информации, четный паритет и один стоповый бит
Вывод PA10 USART1_RX	Вход	Используется в режиме цифрового входа без подтягивающего резистора
Вывод PA9 USART1_TX	Двухтактный выход push-pull	Используется в режиме альтернативного цифрового выхода

лера с выводом RXD преобразователя. Необходимо также соединить GND, 5V микроконтроллера и преобразователя, как показано на рис. 12, см. вторую сторону обложки.

Программирование микроконтроллера осуществляется с помощью программы Flash Loader Demonstrator, которую можно скачать с официального сайта ST Microelectronics [13].

Данный способ программирования подходит для фирменных микроконтроллеров компании ST Microelectronics. На рынке существует также большое число аналогов STM32, для которых необходимы другие программные средства для программирования, узнать о них можно на сайтах производителей. Например, для программирования микроконтроллера CH32F103C8T6 необходима программа WCHISPTool, которую можно найти на сайте компании WCH.

Подключив преобразователь с отладочной платой к ПК, необходимо запустить программу Flash Loader Demonstrator. После этого необходимо выбрать COM-порт. Остальные параметры оставить без изменения (рис. 13). Для продолжения необходимо нажать "*Next*". Если все операции по подготовке выполнены верно, то будет выведена страница со светофором зеленого цвета (рис. 14, см. вторую сторону обложки). Для продолжения необходимо нажать "Next", далее будет представлена информация о микроконтроллере и его Flash-памяти (рис. 15, см. вторую сторону обложки).

Далее необходимо выбрать, что необходимо выполнить. В нашем случае — загрузка на устройство. Необходимо также выбрать файл типа .hex с программой для микроконтроллера (рис. 16). Для продолжения необходимо нажать "Next". В случае успешной загрузки программы будет выведена страница с зеленой шкалой прогресса (рис. 17, см. вторую сторону обложки).

Чтобы загруженная программа запустилась необходимо вернуть перемычки в режим "1" (см. табл. 1) и нажать кнопку RESET на плате.

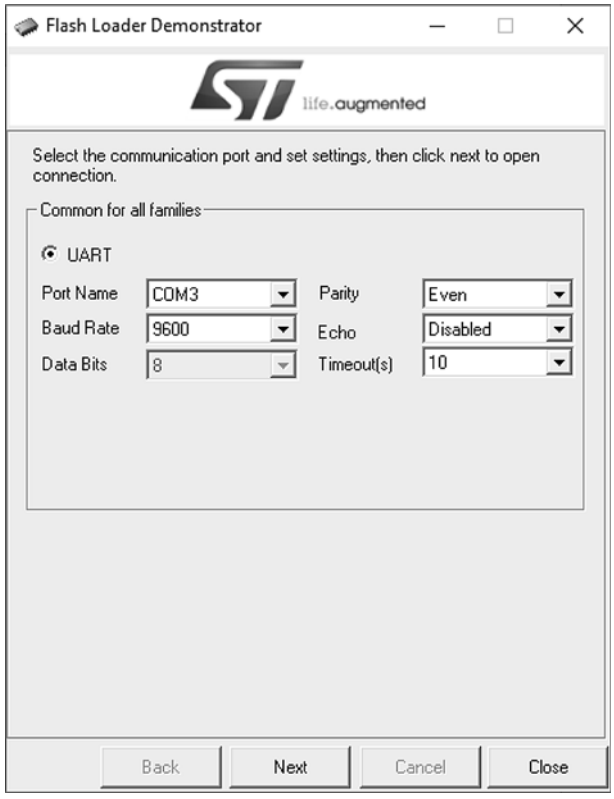


Рис. 13. Выбор параметров COM-порта в программе Flash Loader Demonstrator

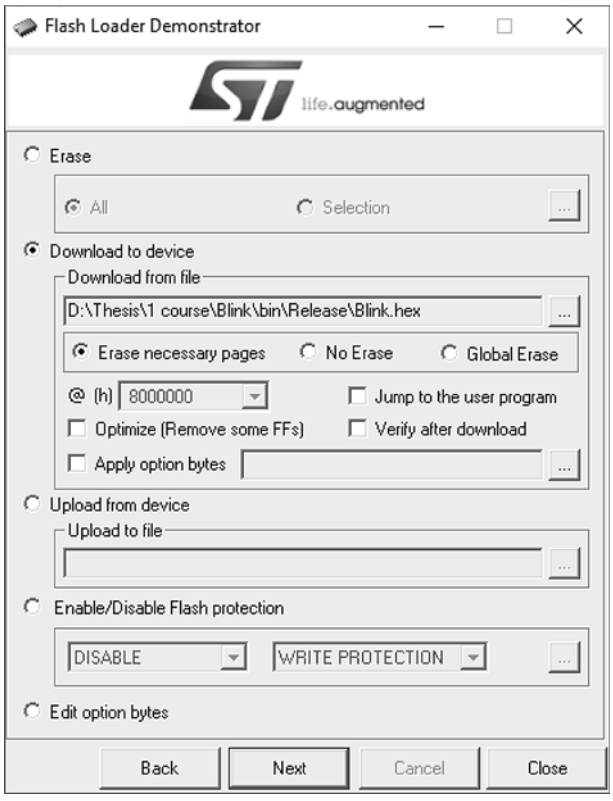


Рис. 16. Выбор файла для загрузки в микроконтроллер

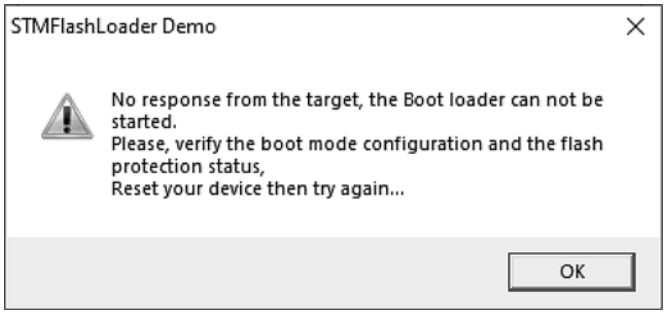


Рис. 18. Сообщение об ошибке

Если приложение вывело уведомление об ошибке (рис. 18), то:

— проверьте правильность соединения микроконтроллера и преобразователя;

— проверьте вольтметром наличие напряжения питания 5 В;

— проверьте положение перемычек BOOT_0 и BOOT_1;

— перезапустите микроконтроллер после смены положения перемычек нажатием кнопки "RESET"на плате.

Также можно автоматизировать данный процесс программирования микроконтроллера путем создания командного файла [14]. В нем необходимо вызвать программу Flash Loader Demonstrator с управляющими ключами, которые позволяют задать режимы и параметры процесса программирования микроконтроллера. Информация о ключах находится в руководстве пользователя программы Flash Loader Demonstrator в разделе Command-line usage [15].

Вызов программы STMFlashLoader.exe с управляющими ключами обеспечивает такие же функциональные возможности, что и приложение с графическим интерфейсом.

Необходимые для поставленной задачи опции описаны в табл. 3.

Для того чтобы запрограммировать микроконтроллер через командный файл, сначала необходимо в этом файле вызвать программу STMFlashLoader.exe, указав к ней полный путь (рис. 19, 20).

В первой строке выполняется проверка на наличие необходимого файла. Вторая строка содержит путь до файла, который необходимо запустить.

Далее определяются параметры COM-порта путем добавления в файл строки, приведенной на рис. 21.

В данной строке определяются такие параметры, как номер COM-порта, скорость передачи, число бит

Таблица 3

Опции Flash Loader Demonstrator

Опция	Описание опции	Параметр	Описание параметра
-c	Определяет COM-порт. Опция -c позволяет выбрать COM-порт на компьютере, который программа использует для связи с микроконтроллером. По умолчанию используется порт COM1. Чтобы выбрать другой COM-порт и параметры соединения, необходимо использовать -c с другими параметрами. Опция -c поддерживает несколько аргументов. Это означает, что можно задать более одного аргумента в одной команде (-c —pn 1 —br 115200 --to 7000)	--pn	Номер порта (значения {1, 2...}, по умолчанию 1)
		--br	Скорость передачи (значения {115200, 57600...}, по умолчанию 57600)
		--db	Число бит информации (значения {5, 6, 7, 8}, по умолчанию 8)
		--pr	Число четности (значения {NONE, ODD, EVEN}, **по умолчанию EVEN**)
		--sb	Число стоповых бит (значения {1, 1.5, 2}, по умолчанию 1)
		--ec	Наличие эха (значение ON или OFF, по умолчанию OFF)
		--to	Максимальное время ожидания ответа (мс, значения {1000, 2000, 3000...}, по умолчанию 5000)
-i	Определяет микроконтроллер, который будет использоваться. Например, STM8_32K, STM32_Med-density_128K, STM32_ High-density_512K, STM32_Low-density_16K и т. д.	-	-
-e	Команда стирания памяти. В соответствии с заданными аргументами, команда может быть использована для стирания определенной страницы памяти или для стирания всей Flash-памяти. Эта операция может занять секунду или более в зависимости от объема памяти	--all	Стереть всю память
-d	Загружает содержимое указанного файла во Flash-память микроконтроллера по указанному адресу	--fn filename	Filename — полный путь до загружаемого файла

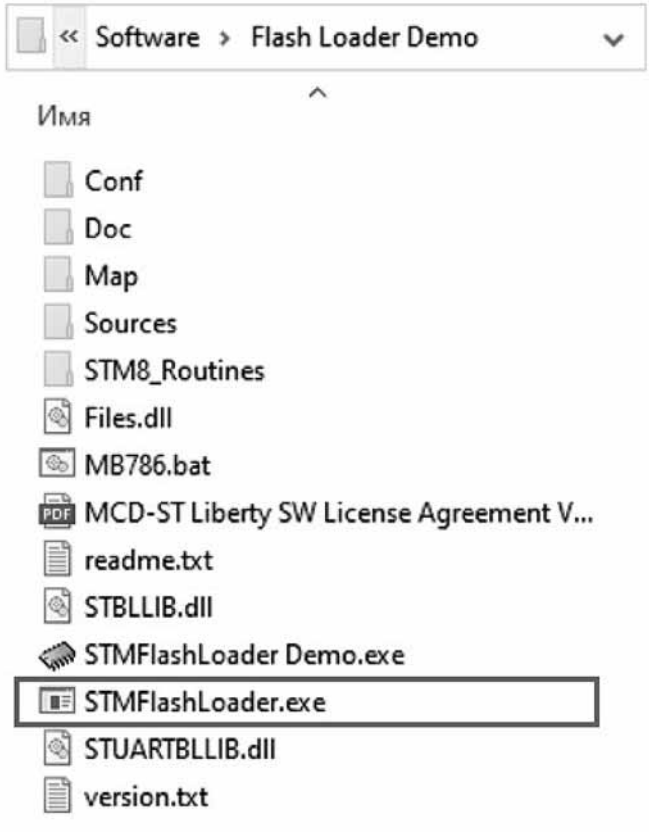


Рис. 19. Местонахождение файла STMFlashLoader.exe

информации, проверка на четность, число стоповых бит и максимальное время ожидания ответа. В параметре "--pn" использована переменная COM_Number, которая содержит номер COM-порта. Это сделано, чтобы была возможность удобно изменить номер используемого порта в начале командного файла. Номер используемого порта можно посмотреть в "Диспетчере устройств". Открывается он путем выполнения следующих действий.

1. Нажать на значок "Компьютер", в проводнике или на рабочем столе, правой кнопкой мыши и выбрать "Свойства" (рис. 22).

2. Выбрать пункт "Диспетчер устройств" в открывшемся окне.

Для просмотра используемых портов в "Диспетчере устройств" необходимо перейти в раздел COM и LPT, где и содержится информация о всех используемых COM-портах (рис. 23).

Далее для определения модели микроконтроллера и для выбора загружаемого файла добавим в командный файл строку, приведенную на рис. 24.

Информация о модели микроконтроллера находится в технической спецификации [16]. Информация о семействе микроконтроллера density находится на первой странице рис. 25.

Информация о размере Flash-памяти находится в разделе Ordering information scheme (рис. 26).

Далее необходимо найти название семейства и размер памяти в папке Map программы Flash Loader Demonstrator (рис. 27).

После этого с помощью опции "-e" очищаем память микроконтроллера, а затем указываем файл для загрузки с полным путем.

В результате получим командный файл, который представлен ниже.

```
::  COM-port number
SET /A COM_Number = 3

if exist "C:/Program Files (x86)/Flash Loader/Software/Flash
Loader Demo/STMFlashLoader.exe"^
  "C:/Program Files (x86)/Flash Loader/Software/Flash Loader
Demo/STMFlashLoader.exe"^
  -c --pn %COM_Number% --br 115200 --db 8 --pr EVEN --sb 1 --ec
OFF --to 1000^
  -i STM32F1_Med-density_64K -e --all -d --fn "tar-
get/target.hex"

if exist "C:/Program Files/Flash Loader/Software/Flash Loader
Demo/STMFlashLoader.exe"^
  "C:/Program Files/Flash Loader/Software/Flash Loader
Demo/STMFlashLoader.exe"^
  -c --pn %COM_Number% --br 115200 --db 8 --pr EVEN --sb 1 --ec
OFF --to 1000^
  -i STM32F1_Med-density_64K -e --all -d --fn "tar-
get/target.hex"

pause
```

```
if exist "C:/Program Files (x86)/Flash Loader/Software/Flash Loader Demo/STMFlashLoader.exe"^
  "C:/Program Files (x86)/Flash Loader/Software/Flash Loader Demo/STMFlashLoader.exe"^
```

Рис. 20. Команда запуска программы STMFlashLoader.exe

```
-c --pn %COM_Number% --br 115200 --db 8 --pr EVEN --sb 1 --ec OFF --to 1000^
```

Рис. 21. Выбор параметров COM-порта

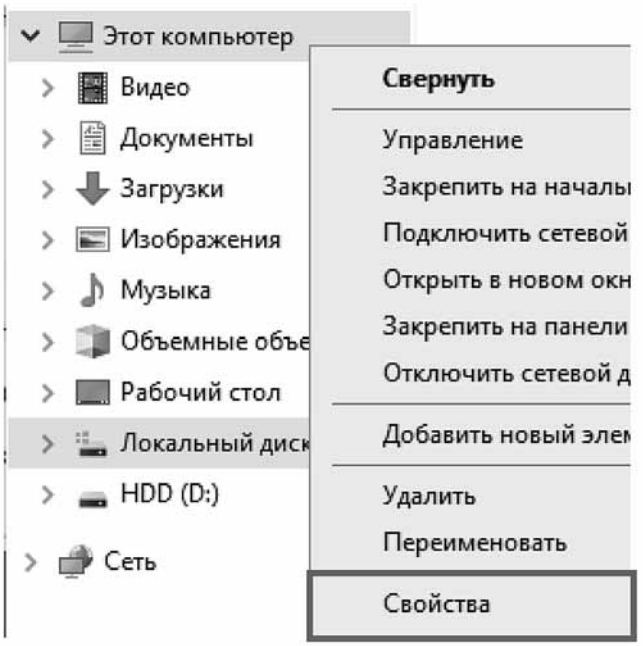


Рис. 22. Путь к диспетчеру устройств

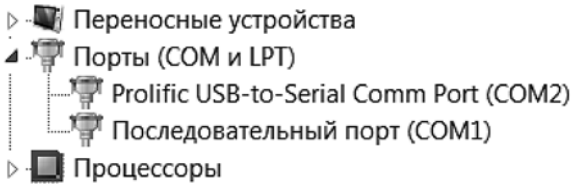


Рис. 23. Информация о COM-портах в "Диспетчере устройств"

В данном файле с помощью первой строки вводится переменная, которая содержит номер COM-порта ("/A" — указывает, что параметр является вычисляемым числовым выражением). Помимо этого, в файле содержится две разные команды, первая выполнится в 32-разрядной операционной системе, а вторая в 64-разрядной операционной системе.

Специальной символ "^" позволяет разбить длинную строку на несколько строк.

На рис. 28 приведен пример успешного завершения программы.

```
-i STM32F1_Med-density_64K -e --all -d --fn "target/target.hex"
```

Рис. 24. Определение микроконтроллера и выбор файла с программой

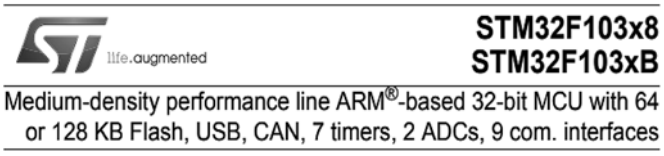


Рис. 25. Информация о семействе микроконтроллера

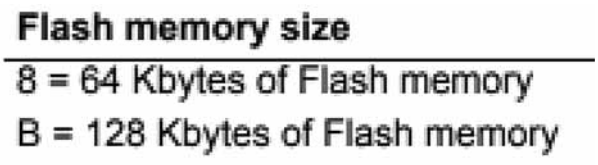


Рис. 26. Информация о размере Flash-памяти микроконтроллера

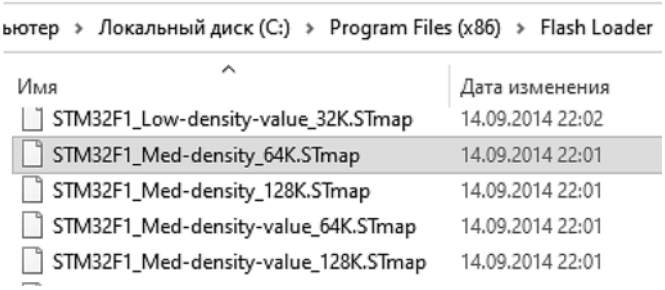


Рис. 27. Папка Map

Следует обратить внимание, что используемая в настоящей статье схема имеет особенности. Так как в данной схеме общий провод блока питания, материнской платы, преобразователя CH340G и микроконтроллера объединены, то попадание высокого напряжения на выводы микроконтроллера может привести к выходу из строя элементов схемы, *включая материнскую плату* (рис. 29, см. третью сторону обложки).

Для предотвращения данной ситуации рекомендуется использовать гальваническую развязку [17]. Существует три основных вида гальванической развязки: трансформаторная, оптическая, конденсаторная.

Примером трансформаторной гальванической развязки является модуль на микросхеме ADUM3160 [18] (рис. 30, см. третью сторону обложки).

При использовании данного модуля схема будет выглядеть, как приведено на рис. 31 (см. третью сторону обложки).

Чтобы удостовериться, что для данной схемы, нет гальванической связи, необходимо

```
C:\Tools\Work\PWM>SET /A COM_Number = 3

C:\Tools\Work\PWM>if exist "C:/Program Files (x86)/Flash Loader/Software/Flash Loader Demo/STMFlashLoader.exe" "C:/Progr
am Files (x86)/Flash Loader/Software/Flash Loader Demo/STMFlashLoader.exe"  -c --pn 3 --br 115200 --db 8 --pr EVEN --sb
1 --ec OFF --to 1000  -i STM32F1_Med-density_64K -e --all -d --fn "target/target.hex"
Opening Port                            [OK]
Activating device                               [OK]

 ERASING ...
erasing all pages                               [OK]

 DOWNLOADING ...

downloading      page/sector 0   @0x  8000000    size 0.55(KB)   [OK]

C:\Tools\Work\PWM>if exist "C:/Program Files/Flash Loader/Software/Flash Loader Demo/STMFlashLoader.exe" "C:/Program Fil
es/Flash Loader/Software/Flash Loader Demo/STMFlashLoader.exe"  -c --pn 3 --br 115200 --db 8 --pr EVEN --sb 1 --ec OFF -
-to 1000  -i STM32F1_Med-density_64K -e --all -d --fn "target/target.hex"

C:\Tools\Work\PWM>pause
Для продолжения нажмите любую клавишу . . .
```

Рис. 28. Процесс выполнения программы

с помощью мультиметра измерить сопротивление между общим проводом блока питания и общим проводом, который приходит на микроконтроллер. *Это необходимо делать обязательно при выключенном питании.* При отсутствии гальванической связи мультиметр покажет сопротивление более 2 МОм, что является хорошим результатом.

Заключение

Программирование микроконтроллеров STM32F10x с помощью преобразователя USB to RS-232_TTL является простым и недорогим решением, что позволяет не покупать отдельное устройство ST-Link для процесса программирования.

Использование преобразователя USB to RS-232_TTL позволяет не только выполнять программирование микроконтроллера, но и обмениваться данными между персональным компьютером и микроконтроллером при выполнении программы пользователя простым способом.

Для реализации представляемого способа программирования микроконтроллеров, необходим преобразователь USB to RS-232 *однополярного* типа.

На рынке существует большое число аналогов микроконтроллеров STM32. Способ программирования, описанный в статье, подходит для микроконтроллеров фирмы ST Microelectronics. Для программирования аналогов рекомендуется обратиться за дополнительной информацией на сайты производителей.

Список литературы

1. **STM32** 32-bit Arm Cortex MCUs. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html

2. **Универсальный** асинхронный приемопередатчик. URL: https://ru.wikipedia.org/wiki/Универсальный_асинхронный_приемопередатчик

3. **Микроконтроллер** и Bootloader. URL: https://microtechnics.ru/mikrokontroller-i-bootloader-opisanie-i-princip-raboty/

4. **ST-LINK/V2** in-circuit debugger/programmer for STM8 and STM32. URL: https://www.st.com/en/development-tools/st-link-v2.html

5. **Торгаев С. Н., Тригуб М. И., Мусоров И. С., Чертихина Д. С.** Практическое руководство по программированию STM-микроконтроллеров: уч. пособие. Томск: Изд-во Томского политехнического университета, 2015. 111 с.

6. **Универсальный** асинхронный приемопередатчик. URL: https://ru.wikipedia.org/wiki/Последовательный_порт

7. **Добавляем** COM-порт на ПК. URL: https://zen.yandex.ru/media/id/5cd03c8b7dea6f00b30de39b/dobavliaem-com-port-na-pk-5ec4a3e7a16e2c16b877b659

8. **Жмудь В. А., Трубин М. В., Трубин И. В.** Обмен данными между компьютером и микроконтроллером STM32F100 по последовательному интерфейсу связи RS-232//Автоматика и программная инженерия. 2015. № 1(11). С. 45 — 51. URL: http://kb-au.ru/wp-content/uploads/AaSI-2015-1-6.pdf

9. **Компания** WCH. URL: http://wch-ic.com/

10. **Terminal** 1.93b. URL: https://micro-pi.ru/terminal-1-9b-работаем-com-портом/

11. **RM0008.** Reference manual. URL: https://www.st.com/content/ccc/resource/technical/document/reference_manual/59/b9/ba/7f/11/af/43/d5/CD00171190.pdf/files/CD00171190.pdf/jcr:content/translations/en.CD00171190.pdf

12. **AN2606.** Application note. URL: https://www.st.com/resource/en/application_note/cd00167594-stm32-microcontroller-system-memory-boot-mode-stmicroelectronics.pdf

13. **Flash** Loader Demonstrator. URL: https://www.st.com/en/development-tools/flasher-stm32.html

14. **Пакетный** файл. URL: https://ru.wikipedia.org/wiki/Пакетный_файл

15. **UM0462**. User manual. URL: https://www.st.com/resource/en/user_manual/cd00171488-stm32-and-stm8-flash-loader-demonstrator-stmicroelectronics.pdf

16. **STM32F103x8.** Datasheet. URL: https://www.st.com/resource/en/datasheet/stm32f103c8.pdf

17. **Гальваническая** развязка. URL: https://ru.wikipedia.org/wiki/Гальваническая_развязка

18. **ADUM** 3160. URL: https://www.analog.com/en/products/adum3160.html

Programming of STM32F10x Microcontrollers Using the Built-in USART Bootloader

A. E. Bliznyuk[1], Master, bliznyuksaha@mail.ru, **V. A. Zhmud**, D. Sc., Deputy General Director[2], Chief Researcher[3], oao_nips@bk.ru, **M. V. Trubin**[1], Postgraduate Student, morkai@bk.ru, **V. G. Trubin**, Senior Lecturer[1], Director[4], trubin@ngs.ru,
[1]Novosibirsk State Technical University, Novosibirsk, 630073, Russian Federation,
[2] Novosibirsk Institute of Program Systems for Science JSC "Novosibirsk Institute of Program Systems" (NIPS), Novosibirsk, Russian Federation,
[3] Institute of Laser Physics SB RAS, Novosibirsk, Russian Federation,
[4] KB Avtomatika LLC

Corresponding author:

Zhmud Vadim A., D. Sc., Deputy General Director of the Novosibirsk Institute of Program Systems for Science JSC "Novosibirsk Institute of Program Systems" (NIPS), Chief Researcher of the Institute of Laser Physics SB RAS
E-mail: oao_nips@bk.ru

Received on December 26, 2021
Accepted on January 19, 2022

This paper is devoted to the description of the method of programming STM32F10x microcontrollers using the built-in bootloader via the USART serial interface, using the Flash Loader Demonstrator program from ST Microelectronics. To automate the programming process, the process of creating a batch file is described. Quite popular is the method of programming the microcontroller through the SWD interface using the ST-Link hardware programmer. However, due to the current shortage of microcircuits, the cost of electronic components has risen markedly, making this method of programming more expensive. Thus, at this point in time, the method of programming microcontrollers using the built-in bootloader via USART using the USB to RS-232_TTL converter is quite in demand. The article may be of interest to students and engineers.

Keywords: *microcontroller, STM32, bootloader, serial interface, USART, USB to RS-232_TTL, CH340G, Flash Loader Demonstrator, COM port, ST Microelectronics*

For citation:

Bliznyuk A. E., Zhmud V. A., Trubin M. V., Trubin V. G. Programming of STM32F10x Microcontrollers Using the Built-in USART Bootloader, *Programmnaya Ingeneria*, 2022, vol. 13, no. 3, pp. 132—141.

DOI: 10.17587/prin.13.132-141.

References

1. **STM32** 32-bit Arm Cortex MCUs, available at: https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html
2. **Universal** asynchronous receiver-transmitter, available at: https://en.wikipedia.org/wiki/Universal_asynchronous_receiver-transmitter
3. **Microcontroller** and Bootloader, available at: https://microtechnics.ru/mikrokontroller-i-bootloader-opisanie-i-princip-raboty/ (in Russian).
4. **ST-LINK/V2** in-circuit debugger/programmer for STM8 and STM32, available at: https://www.st.com/en/development-tools/st-link-v2.html
5. **Torgaev S. N., Trigub M. I., Musorov I. S., Chertikhina D. S.** *A practical guide to programming STM microcontrollers:* a tutorial, Tomsk, Publishing House of Tomsk Polytechnic University, 2015, 111 p. (in Russian).
6. **Serial** port, available at: https://en.wikipedia.org/wiki/Serial_port
7. **Dobavlyaem** COM-port na PK, available at: https://zen.yandex.ru/media/id/5cd03c8b7dea6f00b30de39b/dobavliaem-com-port-na-pk-5ec4a3e7a16e2c16b877b659 (in Russian).
8. **Zhmud V. A., Trubin M. V. Trubin I. V.** Exchange of Data between the Computer and the Microcontroller STM32F100 by Serial Communication Interface RS-232, *Avtomatika i programmnaya inzheneriya*, 2015, no. 1 (11), pp. 45—51, available at: http://jurnal.nips.ru/sites/default/files/A%26SE-1-2015-6.pdf (in Russian).
9. **WCH** forum, available at: http://wch-ic.com/
10. **Terminal** 1.93b, available at: https://micro-pi.ru/terminal-1-9b-работаем-com-портом/ (in Russian).
11. **RM0008.** Reference manual, available at: https://www.st.com/content/ccc/resource/technical/document/reference_manual/59/b9/ba/7f/11/af/43/d5/CD00171190.pdf/files/CD00171190.pdf/jcr:content/translations/en.CD00171190.pdf
12. **AN2606.** Application note, available at: https://www.st.com/resource/en/application_note/cd00167594-stm32-microcontroller-system-memory-boot-mode-stmicroelectronics.pdf
13. **Flash** Loader Demonstrator, available at: https://www.st.com/en/development-tools/flasher-stm32.html
14. **Batch** file. URL: https://en.wikipedia.org/wiki/Batch_file
15. **UM0462**. User manual, available at: https://www.st.com/resource/en/user_manual/cd00171488-stm32-and-stm8-flash-loader-demonstrator-stmicroelectronics.pdf
16. **STM32F103x8.** Datasheet, available at: https://www.st.com/resource/en/datasheet/stm32f103c8.pdf
17. **Galvanic** isolation, available at: https://en.wikipedia.org/wiki/Galvanic_isolation
18. **ADUM** 3160, available at: https://www.analog.com/en/products/adum3160.html

The Application of Neural Networks in the Construction of a Program for the Markup of Text

D. O. Zhaxybayev, darhan.03.92@mail.ru, West Kazakhstan Agrarian and Technical University named after Zhangir Khan, Uralsk, Kazakhstan

Corresponding author:

Zhaxybayev Darkhan O., Master of Pedagogical Sciences, Lecturer, West Kazakhstan Agrarian and Technical University named after Zhangir Khan, Uralsk, Kazakhstan
E-mail: darhan.03.92@mail.ru

Received on December 19, 2021
Accepted on January 19, 2022

This paper explores the use and applications of neural networks in the construction of a text markup program. This research paper describes various tasks that require textual data analysis and discusses the problems, issues, and solutions that accompany them. Interest in neural networks has increased in recent years, and they are finding applications in a wide variety of fields, such as business, medicine, engineering, geology, and physics. Neural networks have made great strides in forecasting, planning, and management. There are several reasons for this situation. Neural networks are very powerful modeling systems capable of creating complex dependencies. Neural networks can be widely used in areas such as text/speech recognition, semantic search, decision support/expert systems, inventory forecasting, data storage systems and content analysis. The object of the work is the process of functioning of neural networks and an algorithm for text markup recognition. The purpose of the scientific paper is the application of neural networks in the construction of the program for the markup of the text. In order to achieve the goal, the following tasks were put forward: a) study existing neural networks, b) choose a neural network to create a model and study its structure, c) convert input data to feed it into a neural network model. Representation and analysis of symbolic structures in neural networks seems to be an interesting and useful direction in neural network theory. In this paper, we have reviewed some neural network architectures that may merit further consideration in these circumstances. In what follows, we will focus on specific experiments in this area. Thus, this paper outlines a problem area for further research and testing.

Keywords: *neural networks, text markup, text data, neural network testing*

УДК 004.85 DOI: 10.17587/prin.13.142-147

Д. О. Жаксыбаев, магистр пед. наук, препод., darhan.03.92@mail.ru,
Западно-Казахстанский аграрно-технический университет имени Жангир хана,
Уральск, Казахстан

Применение нейронных сетей при построении программы для разметки текста

Исследуется использование нейронных сетей при построении программы для разметки текста. Описаны различные задачи, требующие анализа текстовых данных, обсуждены проблемы и решения, которые их сопровождают. В последние годы возрос интерес к нейронным сетям, они находят применение в самых разных областях, таких как бизнес, медицина, инженерия, геология и физика. С использованием нейронных сетей были достигнуты успехи в вопросах прогнозирования, планирования и управления. Причин такого эффекта несколько. Нейронные сети — очень мощные системы моделирования, способные создавать сложные зависимости. Нейронные сети могут широко использоваться в таких областях, как распознавание текстов / речи, семантический поиск, поддержка принятия решений / экспертные системы, прогнозирование запасов, системы хранения данных и анализ контента. Объектом работы является описание процесса функционирования нейронных сетей. Предметом научной работы является

алгоритм распознавания разметки текста. Целью научной статьи является изучение возможности применения нейронных сетей при построении программы для разметки текста. Для достижения описанной цели были поставлены следующие задачи: а) изучить существующие нейронные сети; б) выбрать нейронную сеть для создания модели и изучить ее строение; в) преобразовать входные данные для подачи их в модель нейронной сети. Представление и анализ символьных структур в нейронных сетях кажется интересным и полезным направлением в теории нейронных сетей. В статье проанализированы некоторые архитектуры нейронных сетей, которые, возможно, заслуживают дальнейшего рассмотрения. В будущем планируется сосредоточиться на конкретных экспериментах в этой области. Таким образом, данная работа очерчивает проблемную область для дальнейшего исследования и тестирования.

Ключевые слова: *нейронные сети, разметка текста, текстовые данные, тестирование нейронных сетей*

Introduction

In recent years there has been an increased interest in neural networks, which are successfully used in various fields — business, medicine, engineering, geology, physics. Neural networks have made great strides in solving prediction, planning, and control problems. There are several reasons for this shocking development.

1. Huge capabilities. Neural networks are a very powerful modeling system that can create complex dependencies.

2. Ease of use. Neural networks are learned through examples. The neural network user takes proxy data and then runs a learning algorithm that sees only the structure of the data. This, of course, requires the user to have heuristic knowledge of selecting and preparing the data, selecting the necessary network architecture, and interpreting the results, but must have the level of knowledge necessary to successfully use a neural network. For example, it is easier than using traditional numerical methods.

The neural network is intuitively interesting because it is based on the old biological mode of operation of the nervous system. In the future, the development of such neurobiological forms may lead to the creation of real "imaginary" computers.

Artificial neural network is based on the basic biological neural network, which is a neural network that performs certain functions. A neural network consists of neurons.

Synapses are connections through which signals from some neurons arrive at the input of others. A set of synapses is characterized by its weight. Those connected with a good weight are called excited, and those with an insufficient weight are called inhibited. An offshoot of neurons is called an axon. In an artificial neural network, the artificial neuron is a nonlinear function where the conflict is a combination of the lines of all signals. This function is called activation. The result of the activation function is then sent to the output neuron. By combining these neurons with others, it creates an artificial neural network.

Neurons perform several functions:

- reception function — synapses receive information;
- integration function — when a neuron is released, the signal carries information about all the signals combined in the neuron;
- transmission function— information is transferred from axon to synapse;
- translation function — a pulse reaching the end of the axon causes the transmitter to send a signal to the next neuron.

Applications of neural networks are diverse: message and speech recognition, semantic search, expert and decision support systems, stock price forecasting, data storage systems and content analysis.

The proliferation of computer and communication technologies has resulted in powerful systems and information flows unparalleled in the past. For example, the daily volume of records on electronic media does not allow even one person to access them immediately. The division of labor among several actors creates a problem of communication and organization between them.

At the same time, content analysis can be an important way to obtain relevant information that is important for a company to gain a competitive advantage. Many companies have established data protection departments that monitor and evaluate information about the company and its competitors. Similar methods can be used, for example, to evaluate new employees. By learning how a person behaves on Internet forums, blogs and social networks, you can get an idea of their character.

In this article we look at text markup, which is based on semantics. Semantic text markup as we understand it is a study by a computer of text that reveals the dependencies and combinations in certain words and expressions, within the text.

Obviously, it is not promising to constantly study the entire flow of information on a given topic and, incidentally, to form such a flow even without the use of automatic tools.

They can come in many forms, but we will consider the methods of these tools. Over the past two decades, methods of information production have evolved in the field of intelligent analysis and information extraction from data arrays. This is a field of study, created and developed on the scientific basis of exact sciences, such as statistics and artificial intelligence, theory of databases, etc. [1]. Originally, data mining was developed to produce highly organized data, but technological advances require the use of similar tools in unstructured data.

The presence of a large number of electronic documents stored in non-semantic tagged textual document formats indicates the importance of creating universal algorithms for automatic analysis of these documents to determine the semantics of individual document fragments.

The task is to select individual fragments from all text documents and assign these fragments to one of the predefined data types. The first information received is the

type of information for each fragment of its design, the presence of keywords or symbols, the position of other fragments in the corresponding text, etc. In general, there are no clear algorithms and rules for assigning an element to a fixed data type. The algorithm needed to solve such a problem allows decisions to be made under uncertain conditions. According to these algorithms, algorithms that perform mathematics using an artificial neural network have the ability to learn and make decisions based on the "knowledge" they acquire through hands-on learning [1].

Among the existing possibilities of using a neural network for color recognition, the use of a single-layer network consisting of an infinite number of neurons with a "backward distribution" learning algorithm is of great importance [2]. The theory of neural networks allows an infinite number of components and the number of neurons in each component, but in fact these numbers are limited by computer resources. A neural network consists of a set of neurons, each with a number of sharp spines, called spurs, and protruding axons. Each neuron is connected to synapses with input parameters of this neural network, each of which has its own weight. The process of training the neural network comes down to finding the ideal value for all synaptic embedded weights (some of which can be constant), which will provide the required response of the neural network. The ability of the network to perform operational tasks depends on how well it is trained. The quality of training depends on the duration of training and the order of replay during training, and is usually based on the appropriate ratio of training time to the acceptance of the final network performance. Thus, the ability of neural networks to make informed decisions depends more on the mapping of specific information to the network metrics on which the decision-making is based.

We consider the concept of text markup as syntactic and semantic processing of text in order to highlight and denote individual information objects.

Syntactic markup of text is one of the most complex, or morphosyntactic, markup means that in addition to the morphological information assigned to each word of the text, each sentence is also given its syntactic structure, and in the form of a dependency tree.

In order to use a neural network in text markup analysis, the following issues must be addressed:

- mapping the information coming into the neural network and ensuring that it is extracted from the file being examined;
- maintaining the structure of the neural network and the learning outcomes;
- taking steps to record the results as the neural network evolves.

Of course, there is a central module that provides an interface to all the other modules. Interaction between objects is done through a set of defined mechanisms — the programming interface. This feature makes it easy to upgrade the system by replacing a separate module for each system to update it or use it in a new environment. For example, a system may have several flexible user interface modules to work in different environments (Windows program, web interface, console program, web service), independent of other system modules. In addition, each of these modules must perform a number of specific tasks, for example: the choice of file for testing, determining the parameters of the algorithm, the report on the progress of reading algorithms, running training programs, etc. The well-known file integration module (I/O) uses a software interface to handle the file. It should provide a means to read information from the file and provide access to the file to open collection information to analyze and record results. This module can be added, for example, to Microsoft Word files and XML files. A set (or new ones) including the structure of the neural network, the description of the training result, and the collection and recording functions of the raw data are selected as parameters for the algorithm through the user interface. The test results are consistent and independent because the structure of the network depends on their number.

One example of randomly organized information is plain text. Another example (for semi-structured data [2]) is XML documents. Text is a global way of representing, collecting and transmitting information in society. Moreover, even this transformation is incapable of turning text into a relational representation without losing the semantics of the text and the relationship between entities. At the same time, a lot of information is hidden in the text, but due to the lack of structure it is impossible to evaluate it by the way it is obtained. This problem is solved by text analysis — Text Mining [2, 3].

Text Extraction — Text Mining

Let's take a look at the general practices of text mining [3]. These include general functions as well as very specific functions related to the type of data being studied. The first group includes planning and coordinating activities, the second group includes automatic annotation, recording key ideas, document navigation, trend analysis, and organization searches.

One of the most common tasks is organization (classification). The purpose of classification is to identify a set of components from a predetermined set to which a document belongs. Classification includes and other things that can be used for display.

The second task is content clustering. The purpose of this method is to automatically select identical groups of text between fixed sentences. Groups are formed based on only two identical text descriptions. No description of the group has been given.

Automatic annotation allows you to shorten notes and record their meaning. At the same time, the size of the sentences is usually adjustable by the user, allowing you to choose the amount of information you need.

If you take the main points, you can see the facts and context in the text. Often these ideas are proper nouns: names of people, names of organizations, etc.

Navigating between characters gives the user a fully coherent description of the type of text with clear themes and meaningful words. The user can select a subset of items of interest and switch between related links, for example by using automatic hyperlinks.

Trend analysis allows you to see certain patterns in text sentences. This can be used, for example, to measure how a company's interests change from one market segment to another.

When researching the analysis of organizational trends, it is necessary to examine the connections between key articles in the text.

Neural network methods

A number of strategies have been developed to solve the above problems. These tend to be the most common algorithmic methods. On the other hand, there are a number of powerful and effective solutions to some of these problems. These are neural network approaches. Neural network theory emerged and evolved as an attempt to explain the principles of the brain. Neural networks have proven to be a flexible and effective way to solve certain problems, which other methods cannot fully solve.

A neural network consists of organized neurons. Each neuron is a simple arithmetic object with many inputs and one output. Each input of the *i*-th neuron has a weight of synaptic signals w_i and x_i. Usually neurons compute weights by the weighted sum of inputs, and the results are influenced by another non-linear function *f* (neuron activation function) $y = f\left(\sum_i x_i w_i\right)$ (fig. 1).

Neurons are organized in fragments. Vector input $\mathbf{x} = \{x_i\}_0^N$ into each neuron in the cell (i. e. each neuron has N inputs). Release of all neurons at one stage serves as an input to the next stage. A neural network usually consists of several sections with feedback (repetition or perspective distribution), which exist as lessons (with or without a teacher) [5].

The use of neural networks to analyze datasets is not very common for several reasons. One of the main reasons is that the neural network is not set up to handle the data, but requires an access number. However, there are other options for this solution, and it could potentially play a key role in determining how our brains work with information. As you know, our thinking is symbolic (logical, we don't mean intuitive). There is a significant gap in our knowledge of how the brain works at a "lower" level (a good example of this mechanism is the neural network) and how our consciousness works. Perhaps someday this issue will be clarified by exploring the potential of the neural network to process text markup.

The most interesting neural network architectures suitable for processing text messages are the autonomous Kohonen map and the automatic associative memory network with repetition. Let's talk about them in more detail.

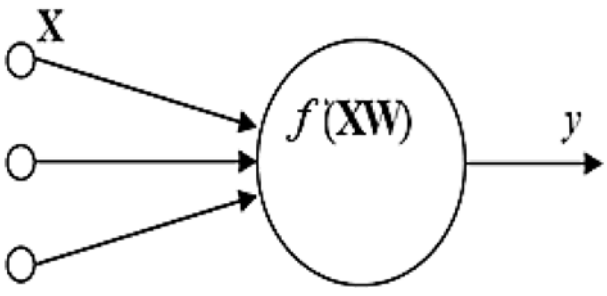


Fig. 1. Formal neuron with vector input X, vector weights W, activation function *f* and signal output *y*

Competitive Networks

The principles of operation and educational self-mapping were developed in 1982 by the Finnish scientist Teuvo Kalevi Kohonen. Kohonen's main idea is to provide information about his role in the regulation of learning neurons. According to Kohonen, a neural network consists of an input layer with the same number of neurons as the number of inputs, and a hidden (output) layer of one-dimensional (linear) or two-dimensional (rectangular) neurons. In relation to topographic maps such network is also called Kohonen map. The network architecture is shown on fig. 2 [6].

The $\|\,\mathrm{dist}\,\|$ box in this figure accepts the input vector $\mathbf{p}$ (R — number of elements in input vector) and the input weight matrix $\mathbf{IW}^{1,1}$, and produces a vector having S^1 elements. The elements are the negative of the distances between the input vector and vectors $_i\mathbf{IW}^{1,1}$ formed from the rows of the input weight matrix.

In this paradigm training is conducted without a teacher, which means that the neural performance was not comparable to the baseline at the time of the study. The training provides successive training examples to tune such a neural network. Another example identifies the most similar neuron, i.e. the neuron that is sent a smaller dot product of weight and vector as input. Such neuron is considered to be successful and adjusts the weight of neighboring neurons. The principle of learning, defined by Kohonen, provides competitive learning, taking into account the length of the "winning neuron" and neurons received as a result of recording as. $\Delta w_i^r = \beta\Lambda\left(|i - i^*|\right)\left(\mathbf{x}^r - \mathbf{w}_i\right)$, where $\Lambda(|i - i^*|)$ is the neighborhood function that determines the amount of neuron weight adjustment, $\mathbf{w}_i$ is the weight of the *i*-th neuron, and β is the learning rate.

At the end of the search process, examples of similarity clusters will be listed on the Kohonen map. Each set of neurons in the output layer was modeled to provide a form of training models in a multidimensional environment. The selected self-organizing map is included in the transformation of *N*-dimensional space into two-dimensional or one-dimensional space. The only thing to keep in mind is that such a change involves some errors.

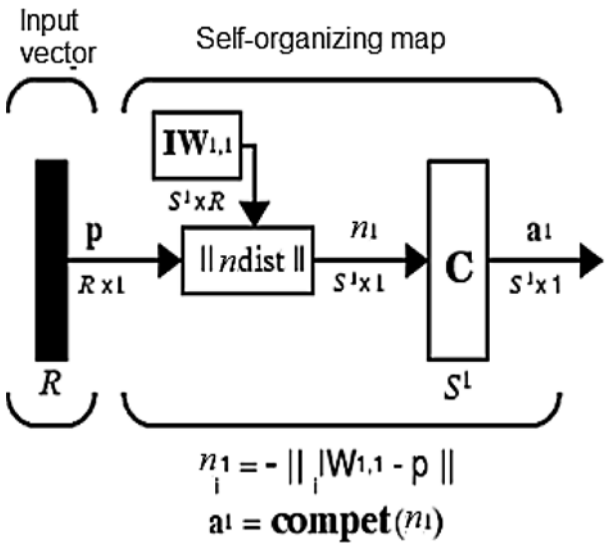


Fig. 2. Creating an independent Kohonen map

The two nearest points of the Kohonen map are closer to the *N*-dimensional input region, but not vice versa.

Based on this design, Finnish researchers have proposed solutions to the problems of coordination and movement of text in text boxes [7]. The way WEBSOMs are organized is that they present lines of full text as a two-sided map showing the distribution of text descriptions on the sides. In this case, specific documents are linked to their location on the map, and each location may contain a set of similar documents in a thematic text class. In addition, nearby areas are often associated with the closures of the text classes that make up the most important map. Places on the map are named according to contextual references. The user selects a location on the map of interest and gets a related text class with similar content. When searching for documents that contain other terms, search results can be displayed on a map that confirms the location of the documents. This allows the user to evaluate the thematic distribution of the required information.

An evolution of competitors, such as the Kohonen network, is the LVQ (Linear Vector Quantization) network. The LVQ neural network consists of two interconnected layers: the competitor construct and the conductive layer. Both areas of the LVQ neural network have a competitive neuron for each set and a linear neuron for each target class. S^1 is the number of clusters, S^2 is the number of target classes (S^1 is always greater than S^2). For example, suppose neurons 1, 2, and 3 in the competitive layer all learn subclasses of the input space that belongs to the linear layer target class 2. Then competitive neurons 1, 2, and 3 will have $\mathbf{LW}^{2,1}$ weights of 1.0 to neuron $\mathbf{n}^2$ in the linear layer, and weights of 0 to all other linear neurons. Thus, the linear neuron produces a 1 if any of the three competitive neurons (1, 2, or 3) wins the competition and outputs a 1. This is how the subclasses of the competitive layer are combined into target classes in the linear layer.

The competitive design divides the set of input vector sets **X** into classes and shows the regions between them for placing vectors $\mathbf{m}_i$. This is done by determining the distances between the *x*-value of the positions of the boundary segments and the initial value. The queue converts the input class (cluster $\mathbf{a}^1$) defined by the participant into the target user class $\mathbf{a}^2$ defined by the class. The output of all linear neurons is the corresponding $\mathbf{n}^2$, thus forming a binary vector $\mathbf{a}^2$, all elements of which are 0, except for the corresponding object class (this element is 1). Thus, the LVQ network consists of two parts: the competitive one, which

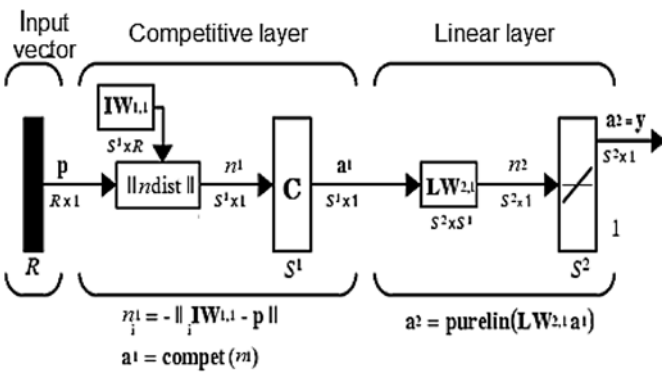


Fig. 3. LVQ network

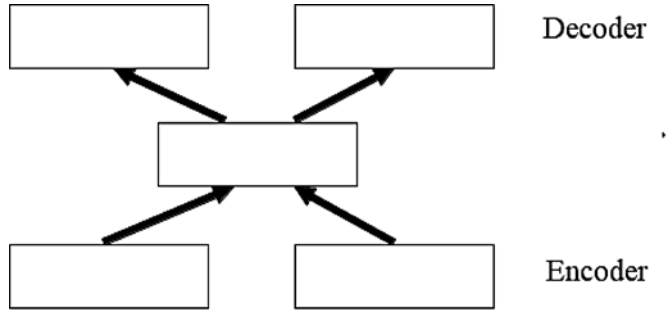


Fig. 4. RAAM binary network architecture

realizes the collection of access objects; the linear one, which connects groups of input objects with output classes. The architecture of the LVQ network is shown in fig. 3.

Recursive autoassociative memory

The third architecture we are interested in is recursive autoassociative memory (RAAM), developed by Pollack in 1990. The purpose of RAAM is to represent symbolic structures in a neural network. Symbols are stable forms of valid trees of fixed valence. In essence, a RAAM network is an automatic associative error back propagation network. The input and output fields of RAAM classify elements into fields, with each field containing the same number of elements. The number of fields is determined by the weight of the code trees, and the number of fields in the hidden area depends on the number of properties in each field. If we take a RAAM network that is trained to represent a set of poles, this network can be thought of as two automata: the first weight layer is the automaton for the production tree (called encoder) and the second layer is the dividing agent that represents the components (called decoder). A representative form of the RAAM architecture is shown in fig. 4 [7].

The representation of the structure in a RAAM network consists of the nature of the hidden event of this layer. When constructing a full tree view, RAAM creates a view for each inner hill (the bottom tree). Trees are repetitive forms, and RAAM creates its screen repeatedly, recreating the input design with a preconceived description of the components encountered at a given time.

On the Web, special indicators are represented as vectors. Orthogonal vectors are widely used. Measuring vectors representing signals shows the minimum number of elements that make up the RAAM field. Since the volume of the hidden component is smaller than the volume of the input material, the output signal is displayed in a compressed form. Thus, the degree of compression depends on the actual data set, and therefore it may be necessary to augment the vectors with zeros [8].

Hidden layer activation values form a distributed spatial representation of signals transmitted to the input network. The locations of different shapes and sizes can be determined using a self-organizing map to see the shape of their ends. It is still difficult to say what the result will be. However, we are convinced that such a function can be used, for example, to represent other structural shapes in texts.

Natasha Library for neurolinguistic programming

In our project we used a plug-in library called Natasha, written in the Python programming language. The goal of our project is to do syntactic and semantic markup of

the text. The first step in marking up a text is to find and identify all the verbs in the text. Then we mark the text semantically, i. e. we find the compatibility of verbs with adjacent words and classify these combinations.

Natasha is distributed under the MIT (Massachusetts Institute of Technology) license, i. e. open and free software license, which gives you the right to use it for your own purposes. The library itself is mostly written by two developers, natives of Russia. They have made all the source code and documentation for the library publicly available on GitHub. You can get access to the repository by following the link https://github.com/natasha/natasha.

Natasha solves the basic tasks of neurolinguistic programming (NLP) for the Russian language. These are at least the following tasks: tokenization, sentence segmentation, word embedding, morphology tagging, lemmatization, phrase normalization, syntactic analysis, Named Entity Recognition (NER) tagging, fact extraction. Natasha is not a research project; the underlying technologies are designed for use in real-world tasks. The developers have focused their attention on model size, memory usage and performance. NumPy is used to output information.

NumPy is a Python language library that adds support for large multidimensional arrays and matrices, along with a large library of high-level (and very fast) mathematical functions for operations on those arrays.

Natasha comes with an API which integrates quite a few written libraries. Using this API makes it easy to work with the module. Natasha includes the following libraries:

- Razdel — a system of lexemes of Russian sentences and words based on rules; it divides the text into words and sentences;
- navec — a set of compact pre-trained embeddings for the Russian language;
- SlovNet — modern deep learning methods for Russian NLP, compact models for Russian morphology, syntax;
- yargy — rule-based fact extraction;
- ipymarkup — neurolinguistic programming visualizations for named entity recognition and syntactic markup.

Using the Razdel library, we can break up text in sentences into individual words. It is also possible to divide text consisting of many sentences into a set of sentences (an array). For example, let's take the sentence "0.5L thermos mug (50/64 cm^3). Splitting the text into words, we get an array "['0.5', 'l', 'thermos', 'mug', '(', '50/64', 'cm^3',')']".

SlovNet is a Python library for NLP modeling based on deep learning for the Russian language. The library is integrated with other Natasha projects: Nerus, a large automatically annotated corpus, Razdel and Navec. Slovnet provides high-quality practical models for Russian NER, morphology and syntax. So, using SlovNet library we can perform morphological text parsing, breaking the whole text into words and get part of speech for each word. Also using this library we can perform syntactic parsing of a sentence [8].

Vargy uses rules and dictionaries to extract structured information from Russian text. I. e., using this library we can, for example, split the sentence "managing director Ivan Ulyanov" into an array "{ "person":{ "position": "managing director", "name":{ "first": "Ivan", "last": "Ulyanov" } } ".

Using the ipymarkup library we can find accessory words in a large volume of text. For example, we can find all the words that denote a person's identity, i. e. names and surnames. We can also find names of countries, cities, buildings, rivers, etc. [8].

In order to use Natasha in your application first you need to install the module in your system. And as we have already mentioned that the project is written in Python, so to install one must have Python at least version 3.5 and pip. First of all, download the Natasha project from GitHub and then run the command "pip install Natasha" in the root directory of the Natasha module. The installation of Natasha will also install other python libraries (dependencies) which are listed in the requirements directory.

To use the library in your application, as usual in the Python language, you need to import all the necessary Natasha library modules with the "import" statement [9].

Conclusion

Description of symbolic structures and their analysis with neural networks seems to be an interesting and useful guide in neural network theory. In our opinion, we have considered a number of neural network architectures that deserve further study in this context. Our future work focuses on specific experiments in this area. This defines the problem for further learning and practice, as well as the possibility of using other neural network modules to evaluate the displayed markup of text [6].

For citation:

Zhaxybayev D. O. The Application of Neural Networks in the Construction of a Program for the Markup of Text, *Programmnaya Ingeneria,* 2022, vol. 13, no. 3, pp. 142-147.

DOI: 10.17587/prin.13.142-147

References

1. **Chubukova I. A.** *Data Mining,* Moscow, BINOM, 2006, 382 p. (in Russian).
2. **Barsegyan A. A., Kupriyanov M. S., Stepanenko V. V., Holod I. I.** *Tehnologii analiza dannyh,* Saint Petersbirg, BHV-Peterburg, 2007, 384 p. (in Russian).
3. **Bashmakov A. I., Bashmakov I. A.** *Intellektual'nye informacionnye tehnologii,* Moscow, Izdatel'stvo MGTU im. N. E. Baumana, 2005, 304 p. (in Russian).
4. **WEBSOM —** Self-organizing Maps for Internet Exploration, available at: http://websom.hut.fi/
5. **Haykin S. O.** *Neural Networks and Learning Machines*, Pearson, 2009, 1104 p.
6. **Callan R.** *The Essence of Neural Networks*, Prentice Hall, 1998, 248 p.
7. **Barskij A. B.** *Logicheskie nejronnye seti:* Uchebnoe posobie, Moscow, Binom, 2013, 352 p. (in Russian).
8. **Galushkin A. I.** *Nejronnye seti: osnovy teorii,* Moscow, GLT, 2012. 496 p. (in Russian).
9. **Galushkin A. I., Cypkin Ja. Z.** *Nejronnye seti: istorija razvitija teorii:* Uchebnoe posobie dlja vuzov, Moscow, Al'jans, 2015, 840 p. (in Russian).

УДК 004.5 DOI: 10.17587/prin.13.148-152

Я. А. Туровский[1,2], д-р техн. наук, канд. мед. наук, доц., Yaroslav_turovsk@mail.ru,
С. В. Борзунов[2], канд. физ.-мат. наук, доц., **А. А. Вахтин**[2], канд. физ.-мат. наук, доц.
[1] Институт проблем управления РАН им. В. А. Трапезникова, Москва
[2] ФГБОУ ВО Воронежский государственный университет

Алгоритм коррекции статистического оценивания с учетом эффекта множественных сравнений на основе группировки результатов тестов[1]

Многократное применение критериев статистических сравнений приводит к значительному возрастанию вероятности ошибочного установления различий в тех случаях, когда они статистически незначимы. В настоящей статье предложен алгоритм оценки значимости различий в задаче управления дизайном медико-биологического исследования с учетом вариаций значений попарного статистического критерия. Представлен пример применения алгоритма для окулографических данных.

Ключевые слова: *множественные сравнения, оценка значимости, дизайн исследования, окулографический интерфейс*

Введение

В подавляющем большинстве исследований явлений медицинской и биологической природы в настоящее время используют многофакторные зависимости. Соответственно, при этом возникает необходимость осуществить в рамках статистической обработки более чем одно применение статистического критерия. Как следствие, возникает широко известный эффект множественных сравнений, заключающийся в повышении риска необоснованно принять альтернативную гипотезу (H_1), что, естественно, скажется на выводах исследования. Тем не менее различные методы противодействия данному эффекту основаны на пересчете уровня значимости, что в конечном итоге может, подобно поправке Бонферрони, привести к ошибочному отклонению гипотезы H_0 [1—4]. В недавно опубликованной работе [5] описан разработанный авторами алгоритм управления дизайном медико-биологического исследования, основанный на оценке доли отклонений основной гипотезы H_0 от общего числа сравнений в пространстве сравнения переменных. Использование данных не только о результатах статистического сравнения, но и об их распределении в рамках экспериментального плана исследования позволяет выделить группы реакций, в которых при сохранении мощности критерия снижается вероятность принятия ложноположительных гипотез. Действительно, если ложноположительные решения должны быть распределены равномерно по всему пространству сравнения переменных, то группировка положительных решений в каких-то отдельных областях, отличных от случайного распределения, позволяет предположить наличие истинно положительных различий между сравниваемыми группами.

Однако при таком подходе учитываются все значения сравнений при стандартном значении $p < 0{,}05$, которые и формируют группы различий на факторном плане эксперимента (здесь и далее p означает уровень значимости результата попарного сравнения согласно применяемому статистическому критерию). В то же время подобный подход не учитывает, что полученные в ходе расчета статистических критериев значения p могут быть существенно меньше 0,05. Таким образом, возникает определенная сложность в интерпретации результатов, полученных согласно работе [5]. Действительно, предположим, что значения, полученные при применении статистического критерия, оказались много меньше 0,0001, но расположены эти различия не в группе, а по отдельности. Тогда, согласно указанному алгоритму, эти различия должны быть признаны как ложноположительные, что увеличивает вероятность ошибки второго рода. Отсюда возникает очевидная необходимость учесть разные значения вероятности ошибки, полученные при применении статистических критериев.

Таким образом, целью работы была модификация алгоритма снижения вероятности ошибки второго рода в условиях эффекта множественных сравнений, основанного на оценке группировки результатов применения тестов в медико-биологических исследованиях.

[1] Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 19-07-01037 А.

Оценка значимости событий в условиях эффекта множественных сравнений

Рассмотрим задачу выделения из множества всевозможных результатов эксперимента подмножеств, выделяемых на основе дизайна эксперимента. Такие подмножества M_i (где $i = 1, 2, ..., J$) универсального множества M (содержащего результаты полного набора опытов проведенного эксперимента, иными словами M — пространство исходов анализируемого эксперимента) будем называть "блоками" или "кластерами" испытаний. Как элементы множеств M_i, так и число указанных множеств J подлежат определению. Для управления дизайном исследования наибольший интерес представляет ситуация, при которой число выявленных различий (при некотором фиксированном значении p) без учета эффекта множественных сравнений сравнимо с числом, к которому стремится их число при только ложноположительных значениях попарного статистического критерия.

Описание алгоритма

Шаг 1. В качестве начального шага предлагаемого алгоритма вычислим вероятность положительного исхода одиночного опыта p_+ в предположении равномерного распределения положительных исходов в множестве согласно формуле $p_+ = s/|M|$, где s — число выявленных в эксперименте положительных исходов.

Шаг 2. Далее определим порог B числа ложноположительных решений, воспользовавшись неравенством

$$\sum_{k=1}^{B} C_{|M|}^{k} p^k (1-p)^{|M|-k} > 1 - p_0, \qquad (1)$$

где p_0 — уровень значимости, соответствующий применяемому попарному критерию; $C_{|M|}^{k}$ — стандартное обозначение для биномиальных коэффициентов. Значение числа положительных исходов, превышающее B, трактуется как наличие в результатах исследования положительных результатов, которые не являются ложными.

С учетом информации о значении попарного критерия в i-м эксперименте $p(i)$ выполним группировки элементарных опытов в блоки M_i, $i = 1, 2, ..., J$. Элементы объединяются в блоки исходя из постановки и дизайна эксперимента. В задаче об обнаружении синхронной электрической активности областей головного мозга элементарные исходы — сравнения пар сигналов — объединяются согласно мере близости параметров той или иной серии опытов как соответствие электродов тем или иным областям головного мозга. В целях формирования очередного блока M_t, вычислим число положительных результатов s_t и вероятность в предположении равномерного распределения $p_t = s_t/|M_t|$, где в знаменателе — число элементов в новом блоке.

Шаг 3. Сравним вероятности реализации s_t исходов в серии $|M_t|$ испытаний с уровнем значимости в диапазоне $\left[\min p(t_i),\ \max p(t_i)\right]$:

$$\min_{1 \leqslant i \leqslant |M_t|} p(t_i) \leqslant C_{|M_t|}^{s_t} p_t^{s_t} (1-p_t)^{|M_t|-s_t} \leqslant \max_{1 \leqslant i \leqslant |M_t|} p(t_i). \quad (2)$$

Условие (2) играет центральную роль в рассматриваемом алгоритме и сравнивает вероятности реализации ровно s_t исходов в серии $|M_t|$ независимых испытаний с диапазоном значений критерия, учитывающим его вариацию в различных индивидуальных опытах. Критерий (2) позволяет сформировать блок M_i за счет значительно меньшего значения попарного критерия в одном или нескольких исходах, уменьшая вероятность ошибки второго рода.

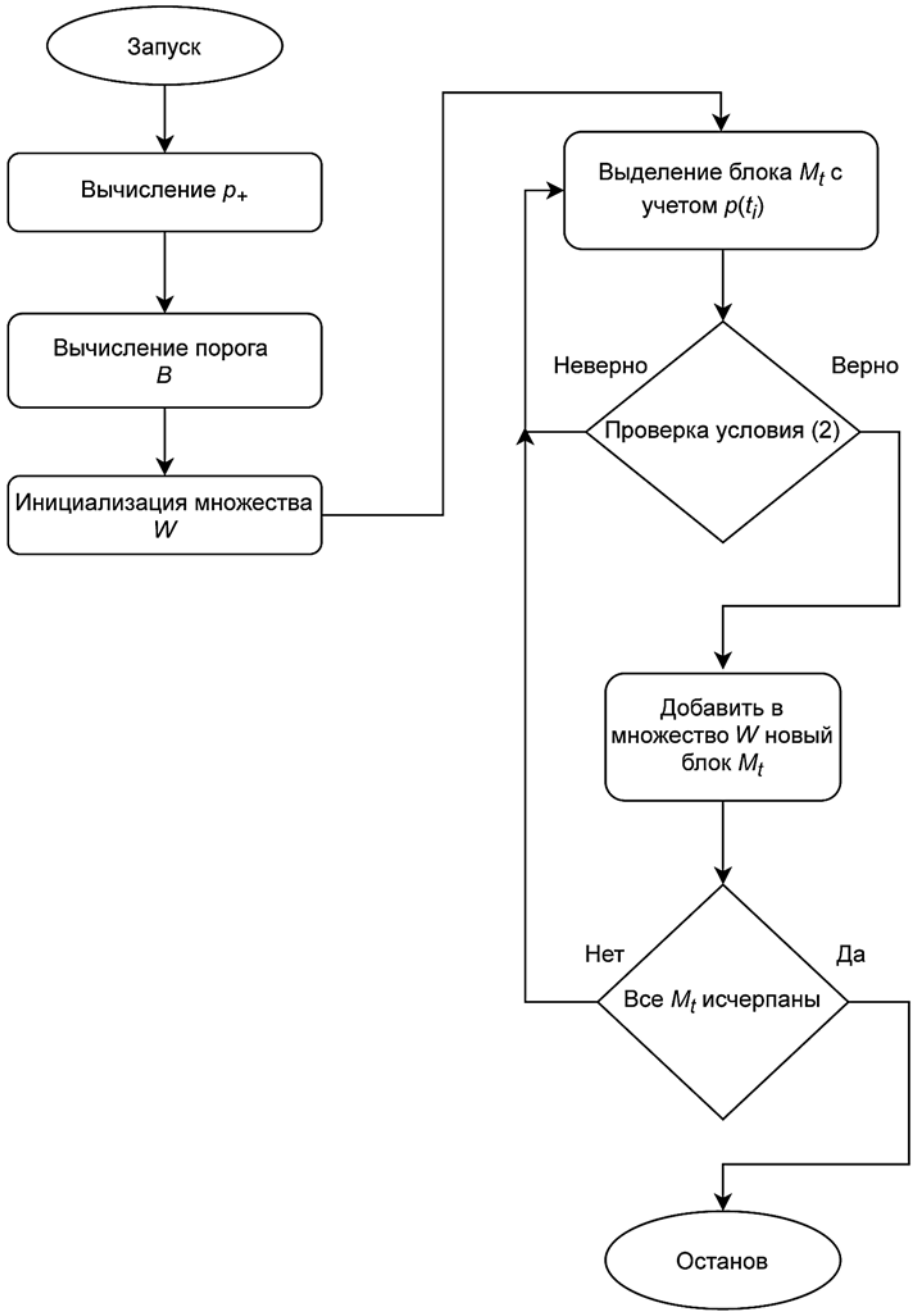
В результате получим множество блоков $W = \{M_1, M_2, ..., M_J\}$. В него сгруппированы положительные исходы, обладающие вероятностями сформировать группу, значимо большими по сравнению с равномерным распределением. Далее происходит останов алгоритма.

Подчеркнем, что критерий (2) дает возможность учитывать и сравнивать между собой как маловероятные, так и практически достоверные события. Это обстоятельство имеет большое значение, в частности, для исследований медико-биологической природы. Такой эмпирический способ учета различных, в том числе очень малых значений критерия позволяет сконструировать простой алгоритм оценки значимости событий в условиях эффекта множественных сравнений. Блок-схема алгоритма приведена на рисунке.

Рассмотрим пример практического применения алгоритма. Пусть имеется экспериментальный план, в котором по одному фактору есть восемь возможных вариантов сравнений, по другому фактору — два, по третьему фактору — четыре, т. е. $|M| = 8 \cdot 2 \cdot 4$. В этом случае следует различать возможные сравнения и возможные переменные, полученные при использовании данного факторного плана. В простейшем случае, например, одна переменная и две группы, дадут нам одно сравнение, а одна переменная в трех группах в зависимости от подходов может дать как одно, так и три сравнения. Таким образом, при наличии в трех группах одного показателя в виде, например, окулографической активности, т. е. параметров движения глаз [6], можно воспользоваться критерием Краскейла—Уоллеса [7] и в ходе одного сравнения оценить, извлечены ли все три выборки из одной генеральной совокупности. Если группа одна и та же, но показатели получены в ходе лечения, возможно использование критерия Фридмана для связанных выборок. Однако, если необходимо определить конкретные межгрупповые различия, то нужно с соответствующими поправками применять попарные сравнения. В этом случае число таких сравнений увеличиться до трех.

Пространство событий состоит из пар элементарных опытов. Его образуют сравнения вида "фактор i — фактор j — фактор k", $1 \leqslant i \leqslant 8$, $1 \leqslant j \leqslant 2$, $1 \leqslant k \leqslant 4$. Удобно рассматривать двумерный "срез" результатов, соответствующий фиксированной координате по одной из осей пространства событий, например, пусть $k = \text{const}$.

Предположим, что результаты серии попарных сравнений сформировали результаты, например, как это представлено в табл. 1. Прочерки в таблицах с



Блок-схема алгоритма

Таблица 1

Пример результатов применения попарного критерия — уровни значимости сравнений. Наблюдается формирование кластера согласно предлагаемому алгоритму

Варианты постановки серии опытов	Фактор 1-1 — фактор 2-1	Фактор 1-2 — фактор 2-2	Фактор 1-3 — фактор 2-1	Фактор 1-4 — фактор 2-2	Фактор 1-1 — фактор 2-1	Фактор 1-2 — фактор 2-2	Фактор 1-3 — фактор 2-1	Фактор 1-4 — фактор 2-2
1	—	—	—	—	0,0001	—	—	—
2	—	—	—	—	—	—	0,02	—
3	0,03	0,04	—	0,04	—	—	—	—
4	0,04	—	0,03	0,04	—	—	—	—

Таблица 2

Пример результатов применения попарного критерия — уровни значимости сравнений. Значимые различия не формируют какие-либо блоки

Варианты постановки серии опытов	Фактор 1-1 — фактор 2-1	Фактор 1-2 — фактор 2-2	Фактор 1-3 — фактор 2-1	Фактор 1-4 — фактор 2-2	Фактор 1-1 — фактор 2-1	Фактор 1-2 — фактор 2-2	Фактор 1-3 — фактор 2-1	Фактор 1-4 — фактор 2-2
1	—	—	0,04	—	—	—	0,04	0,03
2	0,02	—	—	—	—	—	—	—
3	—	—	—	0,04	—	—	—	—
4	—	—	0,01	—	—	0,03	0,01	—

примерами означают отсутствие значимого результата применения попарного критерия.

Предположим, что общее число результатов сравнений, при которых выполняется неравенство $p<0{,}05$, равно 13. Из следствия формулы Бернулли (1) согласно алгоритму следует, что вероятность выпадения от 0 до 6 ложноположительных результатов превышает 95,9 %. Значит, не менее семи из полученных результатов являются истинно положительными. Однако как и в работе [5] задачей является установление того, какие именно различия являются истинно положительными. В данном случае оценим, согласно (2), вероятность того, что кластер из восьми сравнений даст шесть истинно положительных решений, она составит $3{,}948\cdot10^{-7}$ при 95 %-ном уровне вероятности положительных исходов. В предположении равномерного распределения исходов математическое ожидание числа попаданий равно 1,40. Поскольку число положительных исходов в нашем случае $6 > 1{,}40$, переходим к следующему шагу алгоритма. Вероятность получить кластер при $p = 0{,}04$ не выше $\pi' = 1{,}057\cdot10^{-7}$, а при $p = 0{,}03$ — не ниже $\pi = 1{,}921\cdot10^{-8}$. Внутри диапазона $[\pi, \pi']$ получаем формирование кластера из опытов с положительными исходами.

Обратим внимание, что в табл. 1 значение $p = 0{,}0001$ образует кластер из одного элемента. Таким образом, при применении стандартного подхода на основе поправки Бонферрони, из всех сравнений, представленных в табл. 1, должно было бы остаться только единственное значение $p = 0{,}0001 < \frac{p_0}{64}$, что привело бы к отклонению всех остальных результатов. Поскольку вероятность того, что шесть результатов сформируют кластер составляет менее $3{,}948\cdot10^{-7}$, то даже с учетом поправки Бонферрони можно признать его существование статистически значимым.

Рассматривая весь срез данных, следует признать, что сравнение для $p = 0{,}02$ не может быть определено как ложно- или истинно положительное, и следовательно, должно быть исключено из рассмотрения.

В качестве примера, когда кластеры согласно алгоритму не формируются, можно привести результаты исходов в табл. 2. Из данных табл. 2 видно, что критерии значимости не превышают значений, отвечающих равномерному распределению ложноположительных результатов, и оснований для выделения кластеров нет, поскольку области факторного плана, где наблюдаются статистически значимые различия, не позволяют принять гипотезу, что они формируют какую-либо группу.

Заключение

Предложено обобщение алгоритма управления дизайном медико-биологического исследования, основанного на оценке доли отклонений основной гипотезы H_0 от общего числа сравнений в пространстве сравнения переменных. Принят во внимание тот факт, что результаты применения статистического критерия позволяют учитывать не только пороговые значения критерия, но и показатели его значений в критической области. Продемонстрирована работа алгоритма в двух различных случаях, когда допускается как формирование кластеров положительных решений (принятия гипотезы H_1), так и принятия единичной гипотезы H_1 при достаточно малых значениях вероятности ошибки первого рода. На основе данных о разных значениях вероятности ошибки, алгоритм поможет подготовить дизайн медико-биологического исследования, что снизит вероятности ошибок второго рода в случае большого числа множественных сравнений. Полученный эффект достигается в первую очередь за счет введения дополнительной информации в виде распределения положительных решений в пространстве исходов применения статистических критериев.

Список литературы

1. **Гланц С.** Медико-биологическая статистика. М.: Практика, 1998. 459 с.
2. **Hochberg Y., Tamhane A. C.** Multiple Comparison Procedures. New York: Wiley, 1987. 482 p.
3. **Benjamini Y., Cohen R.** Weighted false discovery rate controlling procedures for clinical trials // Biostatistics. 2017. Vol. 18, No. 1. P. 91—104. DOI: 10.1093/biostatistics/kxw030.
4. **Lawrence J.** Familywise and per-family error rates of multiple comparison procedures // Statistics in Medicine. 2019. Vol. 38, No. 19. P. 3586—3598. DOI: 10.1002/sim.8190.
5. **Туровский Я. А., Борзунов С. В., Вахтин А. А.** Алгоритм оценки результатов статистического анализа данных биомедицинской природы в условиях эффекта множественных сравнений // Программная инженерия. 2021. Т. 12, № 9. С. 470—474. DOI: 10.17587/prin.12.470-474.
6. **Suefusa K., Tanaka T.** A comparison study of visually stimulated brain-computer and eye-tracking interfaces // Journal of Neural Engineering. 2017. Vol. 14. P. 036009-1—036009-16.
7. **Рунион Р.** Справочник по непараметрической статистике: Современный подход. М.: Финансы и статистика, 1982. 198 с.

An Algorithm for Correction of Statistical Estimations Taking into Account the Effect of Multiple Comparisons based on Test Results Grouping

Ya. A. Turovsky, Yaroslav_turovsk@mail.ru, V. A. Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, 117997, Russian Federation; Voronezh State University, Voronezh, 394018, Russian Federation, **S. V. Borzunov,** sborzunov@gmail.com, Associate Professor, Voronezh State University, Voronezh, 394018, Russian Federation, **A. A. Vahtin,** alvahtin@gmail.com, Associate Professor, Voronezh State University, Voronezh, 394018, Russian Federation

Corresponding author:

Turovsky Yaroslav A., Principal Researcher, V. A. Trapeznikov Institute of Control Sciences, Russian Academy of Sciences Moscow, 117997, Russian Federation; Head of Lab, Voronezh State University, Voronezh, 394018, Russian Federation
E-mail: Yaroslav_turovsk@mail.ru

Received on December 05, 2021
Accepted on January 28, 2022

Repeated application of statistical comparison criteria leads to a significant increase of the probability of erroneous identification of differences in cases where they are statistically insignificant. An algorithm is proposed for assessing the significance of differences in the problem of managing the design of a biomedical study, taking into account variations in the values of a pairwise statistical criterion. An example of using the algorithm for oculographic data is given.

Keywords: *multiple comparisons, significance assessment, study design, eye tracking interface*

Acknowledgements: *The reported study was funded by RFBR according to the research project no. 19-07-01037 A.*

For citation:

Turovsky Ya. A., Borzunov S. V., Vahtin A. A. An Algorithm for Correction of Statistical Estimations Taking into Account the Effect of Multiple Comparisons based on Test Results Grouping, *Programmnaya Ingeneria,* 2022, vol. 13, no. 3, pp. 148—152.

DOI: 10.17587/prin.13.148-152

References

1. **Glantz S. A.** *Primer of Biostatistics*, McGraw-Hill, 2011. 320 p.
2. **Hochberg Y., Tamhane A. C.** *Multiple Comparison Procedures*, New York, Wiley, 1987, 482 p.
3. **Benjamini Y., Cohen R.** Weighted false discovery rate controlling procedures for clinical trials, *Biostatistics*, 2017, vol. 18, no. 1, pp. 91—104, DOI: 10.1093/biostatistics/kxw030.
4. **Lawrence J.** Familywise and per-family error rates of multiple comparison procedures, *Statistics in Medicine*, 2019, vol. 38, no. 19, pp. 3586—3598. DOI: 10.1002/sim.8190.
5. **Turovsky Ya. A., Borzunov S. V., Vahtin A. A.** An Algorithm for Evaluating the Results of Statistical Analysis of Biomedical Data under the Conditions of the Effect of Multiple Comparisons, *Programmnaya Ingeneria*, 2021, vol. 12, no. 9, pp. 470—474, DOI: 10.17587/prin.12.470-474 (in Russian).
6. **Suefusa K., Tanaka T.** A comparison study of visually stimulated brain-computer and eye-tracking interfaces, *Journal of Neural Engineering*, 2017, vol. 14, pp. 036009-1—036009-16, DOI: 10.1088/1741-2552/aa6086.
7. **Runyon R. P.** *Nonparametric Statistics: A Contemporary Approach*, Addison-Wesley Publishing Company, 1977, 218 p.

ООО "Издательство "Новые технологии". 107076, Москва, ул. Матросская Тишина, д. 23, стр. 2
Технический редактор *Е. М. Патрушева.* Корректор *А. В. Чугунова.*

Сдано в набор 31.01.2022 г. Подписано в печать 28.02.2022 г. Формат 60×88 1/8. Заказ PI321
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**

Рисунки к статье **А. Е. Близнюка, В. А. Жмудя, М. В. Трубина, В. Г. Трубина**
«ПРОГРАММИРОВАНИЕ МИКРОКОНТРОЛЛЕРОВ STM32F10x
С ПОМОЩЬЮ ВСТРОЕННОГО ЗАГРУЗЧИКА ПО USART»

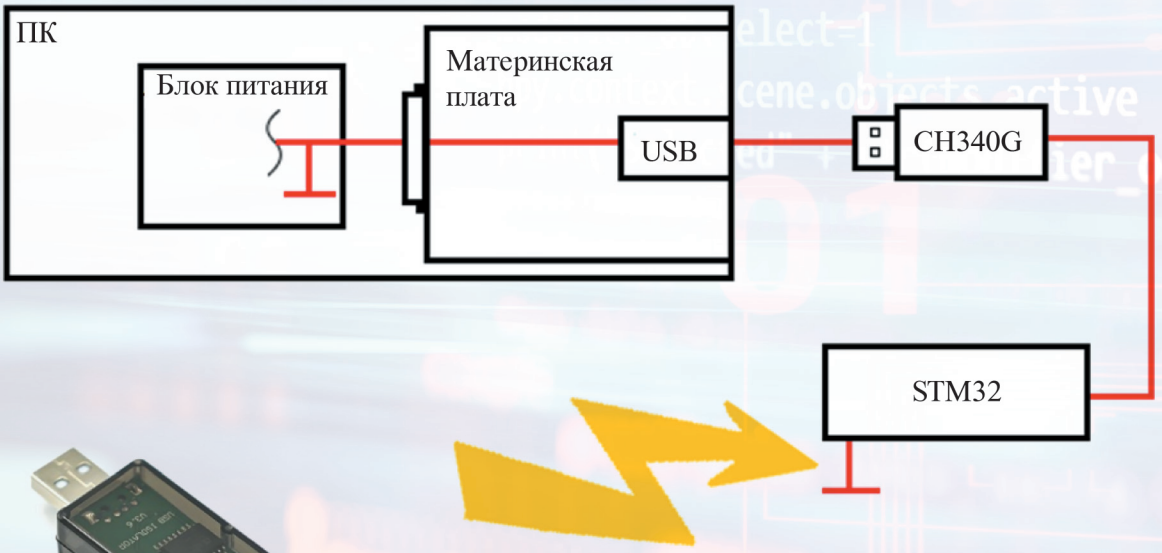


Рис. 29. Схема подключения без гальванической развязки

Рис. 30. Модуль гальванической развязки для USB на микросхеме ADUM3160

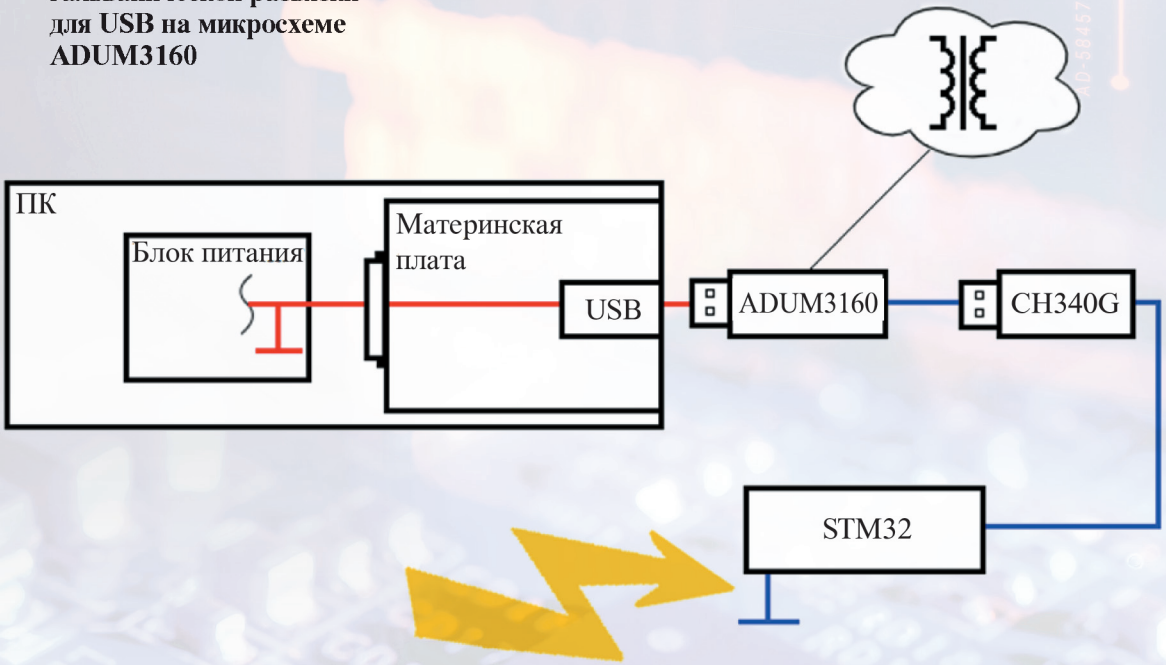


Рис. 31. Схема с использованием гальванической развязки

Издательство «НОВЫЕ ТЕХНОЛОГИИ»

выпускает *научно-технические журналы*

Теоретический и прикладной научно-технический журнал

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

В журнале освещаются состояние и тенденции развития основных направлений индустрии программного обеспечения, связанных с проектированием, конструированием, архитектурой, обеспечением качества и сопровождением жизненного цикла программного обеспечения, а также рассматриваются достижения в области создания и эксплуатации прикладных программно-информационных систем во всех областях челове ческой деятельности.

Подписной индекс по Объединенному каталогу «Пресса России» – 22765

Подписной индекс по Объединенному каталогу «Пресса России» – 72656

Ежемесячный теоретический и прикладной научно-технический журнал

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

В журнале освещаются современное состояние, тенденции и перспективы развития основных направлений в области разработки, производства и применения информационных технологий.

Междисциплинарный теоретический и прикладной научно-технический журнал

НАНО- и МИКРОСИСТЕМНАЯ ТЕХНИКА

В журнале освещаются современное состояние, тенденции и перспективы развития нано- и микросистемной техники, рассматриваются вопрос разботки и внедрения нанои микросистем в различные области науки, технологии и производства.

Подписной индекс по Объединенному каталогу «Пресса России» – 79493

Подписной индекс по Объединенному каталогу «Пресса России» – 79492

Ежемесячный теоретический и прикладной научно-технический журнал

МЕХАТРОНИКА, АВТОМАТИЗАЦИЯ, УПРАВЛЕНИЕ

В журнале освещаются достижения в области мехатроники, интегрирующей механику, электронику, автоматику и информатику в целях совершенствования технологий производства и создания техники новых поколений. Рассматриваются актуальные проблемы теории и практики автомати ческого и автоматизированного управления техническими объектами и технологическими процессами в промышленности, энергетике и на транспорте.

Научно-практический и учебно-методический журнал

БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

В журнале освещаются достижения и перспективы в области исследований, обеспечения и совершенствования защиты человека от всех видов опасностей производственной и природной среды, их контроля, мониторинга, предотвращения, ликвидации последствий аварий и катастроф, образования в сфере безопасности жизнедеятельности.

Подписной индекс по Объединенному каталогу «Пресса России» – 79963

Адрес редакции журналов для авторов и подписчиков:
107076, Москва, ул. Матросская Тишина, д. 23, стр. 2, оф. 45. Издательство "НОВЫЕ ТЕХНОЛОГИИ".
Тел.: (499) 270-16-52. E-mail: antonov@novtex.ru

ISSN 2220-3397. Программная инженерия. 2022. Том 13. № 3. 105 – 152. Подписной индекс по Объединенному каталогу "Пресса России" – 22765