ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ т. 10, №2(41), с. 121–153

ББК 3812
УДК 004.75:004.41

Е. В. Биряльцев, М. Р. Галимов, Д. Е. Демидов, А. М. Елизаров

Платформенный подход к выполнению исследовательских и проектных работ с использованием высокопроизводительных вычислений

Аннотация. Проанализированы предпосылки и обоснована актуальность создания открытой интернет-платформы, объединяющей технологии больших данных, высокопроизводительных вычислений и многосторонних интернет-платформ. Задуманная как экосистема для разработки и использования прикладного программного обеспечения (в том числе в области проектирования и научных исследований), платформа призвана сократить затраты и повысить качество разработки программного обеспечения для решения аналитических задач, возникающих на промышленных предприятиях, в научно-исследовательских организациях, у государственных органов и частных лиц. Представлен работающий прототип названной платформы, функционирующий с использованием суперкомпьютерных технологий и системы виртуализации рабочих столов.

Ключевые слова и фразы: интернет-платформа, программная платформа, облачные сервисы, прикладное программное обеспечение, разработка программного обеспечения, автоматизация бизнес-процессов.

Введение

Традиционно программное обеспечение (ПО) для выполнения специфических отраслевых задач строилось по принципу разработки автоматизированных рабочих мест (АРМ), в которых группировались однотипные функции. Каждый АРМ обладал собственными местом

Работа выполнена при частичной финансовой поддержке РФФИ в рамках научных проектов № 18-07-00964 и № 18-47-160010 и Академии наук Республики Татарстан.

10.25209/2079-3316-2019-10-2-121-153

хранения данных и интерфейсом. Легальный доступ к АРМ осуществлялся путем приобретения лицензий, как правило, на основе количества пользователей, использующих АРМ (именованных или одновременно работающих). Группа АРМ одной компании могла образовывать систему, работающую над единым полем данных, для АРМ различных производителей взаимодействие осуществлялось через интеграционные процедуры, которые в основном представляют собой выгрузку/загрузку данных в специализированных форматах. Доработка или модификация программного обеспечения осуществлялись в рамках плановых релизов с типичной периодичностью 1 год. Ряд производителей (например, партнерская программа Ocean компании Schlumberger [?1]) допускал расширение своего программного обеспечения модулями сторонних разработчиков под собственным контролем. АРМ содержал, как правило, большое количество специальных функций, для освоения которых необходимо было проходить специализированные курсы, организуемые разработчиком. Стоимость одной лицензии на специализированный АРМ составляет от нескольких десятков до нескольких сотен тысяч долларов США.

Сложившиеся архитектура и бизнес-модели ориентированы на крупные корпорации, которые могут себе позволить значительные вложения в программное обеспечение и персонал, а также на относительную неизменность функций и методов предметной области. Подразумевалась также относительная однородность предприятий отрасли, незначительные вариации между которыми покрывались различиями в АРМ от нескольких крупных производителей программного обеспечения. В качестве примера можно привести нефтегазовую отрасль, в которой основными потребителями программного обеспечения являются транснациональные нефтегазовые компании, а рынок ПО делят между собой несколько основных крупных производителей: Schlumberger, CGG, ROXAR, Paradigm [?2, ?3]. Очевидно, что данная практика формирования отраслевого программного обеспечения характерна для индустриального общества. Переход к постиндустриальной экономике требует решения ряда новых задач, плохо коррелирующих с вышеприведенными архитектурой и бизнес-моделью, в частности, возникли и расширяются:

- Гиперсегментация рынков, вплоть до индивидуализации требований отдельных потребителей отраслевого ПО: насыщение рынков вынуждает отраслевых игроков специфицировать рыночные ниши и сосредотачивать свою деятельность на узких направлениях, в связи с чем общеотраслевое ПО становится, с одной

стороны, избыточным, с другой стороны, недостаточным; индивидуальная же доработка и сопровождение крупных программных комплексов слишком дороги.

- Возрастание роли межотраслевых и сквозных проектов, подразумевающих использование в рамках одного проекта функционала из различных предметных областей, что требует построения систем передачи информации между крупными системами со сложными базами данных, интеграция которых становится значительной проблемой.
- Интеграция разработок инновационных компаний, которые сосредотачиваются, как правило, на решении узкого спектра задач: разработанный программный продукт требует интеграции с уже имеющимися программными комплексами, что достигается в настоящее время отраслевой продажей стартапа крупной компании; инерционность данного процесса, а также то, что крупные компании, зачастую покупают стартапы, чтобы избежать конкуренции с собственной, менее совершенной разработкой, становятся тормозом технического прогресса.
- Активное использование в промышленности численных моделей объектов и процессов, в том числе, мультифизических, требующих адаптации алгоритмов к специфике предметной области, размерности задачи, точности и адекватности решения.
- Появление парадигмы бизнеса, управляемого данными (Data Driven Approach), подразумевающей исследование накопленных данных непосредственно в производственном цикле, построение численных моделей на основе этих исследований и немедленное их применение для получения конкурентных преимуществ.

Таким образом, современный отраслевой проект предполагает индивидуальный набор исследований и построение на их основе специализированных моделей, а также разработку оптимального технического решения с использованием таких моделей. Программное обеспечение, поддерживающее такой процесс, должно строиться динамически по мере продвижения проекта. На каждом этапе должна иметься возможность применить различные методы анализа, моделирования и синтеза, в том числе, в различных предметных областях. Очевидно, что решение этой задачи на основе отраслевых АРМ крайне затруднено по технологическим, квалификационным и финансовым причинам даже для крупных компаний, а для малого и среднего бизнеса оно практически невозможно, так как издержки на поиск подходящего ПО, поиск или подготовку персонала, согласование комплексирования и доработок разнородного ПО очень велики, даже при предоставлении

производителями программного обеспечения на условиях аренды (SaaS модель) или в форме бесплатных демонстрационных версий. Отдельной проблемой являются учет использования и регистрация возникающей в процессе комплексных проектов интеллектуальной собственности, такой, как программные модули и наборы данных.

Отметим, что большинство отмеченных выше издержек относится к так называемым транзакционным [?4]. Транзакционные издержки играли незначительную роль по сравнению с трансформационными издержками собственно на создание продукта в доиндустриальном обществе, в связи с чем индустриальные технологии были направлены на минимизацию трансформационных издержек. Благодаря развитию индустриальных технологий трансформационные издержки сократились в десятки раз, а транзакционные издержки стали составлять значительную часть стоимости продукта. Особенно это заметно на сложных многосторонних рынках, а также рынках с асимметрией информации, когда участники рыночных отношений совершенно по-разному проинформированы о свойствах интересующего их продукта. Современные отраслевые проекты, включающие исследования, проектирование и разработку и являющиеся зачастую сквозными и междисциплинарными, представляют собой очень сложный объект для сделки, транзакционные издержки в котором составляют подавляющую долю.

Концепция цифровой экономики, выдвинутая Доном Тапскоттом [?5], является ответом на возрастающую долю транзакционных издержек в стоимости конечного продукта. В рамках данной концепции разработаны и находятся на различной стадии практического применения как общие методы, такие, как открытые инновации [?6], многосторонние интернет-платформы [?7], технологии распределенного реестра, технологии гибкого управления проектом, так и специфические методы для разработки программного обеспечения: облачные технологии, микросервисная архитектура, непрерывная доставка (continuous delivery), графовые модели и другие.

В настоящей статье мы описываем построенный на основе методов цифровой экономики подход к разработке архитектуры программного обеспечения и бизнес-модели выполнения сложных комплексных отраслевых проектов, минимизирующий выявленные транзакционные издержки. Также представлен программный прототип, созданный на основе излагаемых далее подходов и принципов.

1. Подходы

Повышение модифицируемости программного обеспечения. Решение вычислительно емких прикладных проблем, для которых заранее неизвестен алгоритм решения, требует быстрой модификации программного обеспечения. Задача снижения затрат на модификацию ПО хорошо известна и связана с ускоряющимися изменениями требований бизнеса, совершенствованием бизнес-процессов и интеграцией информационных систем. Основным методом решения этой задачи является применение микросервисной архитектуры, графовых моделей вычислений и скриптовых языков реализации (более подробно такая программная архитектура описана в [?8]), позволяющих изменять программный код в процессе работы. Наиболее интенсивно названные методы применяются в таких областях, как извлечение знаний из данных и интеграция информационных систем. Ясно, что мы можем рассматривать любой прикладной проект как набор операций по извлечению и преобразованию данных и их анализу. Поэтому модели реализации программного обеспечения для DataMining и ExtractTransformLoad применимы к организации ПО для выполнения проектов в целом.

Снижение коммуникационных издержек. Коммуникации между субъектами различных отношений являются областью деятельности, активно оптимизируемой в настоящее время методами интернет-платформ [?7]. Наиболее интенсивно развиваются интернет-платформы для оптимизации простых коммуникаций в процессах общения по интересам (социальные сети), различных закупок (интернет-магазины), заказа транспорта (Uber, Яндекс-такси), доступа к видеоматериалам (Netflix, Youtube) и ряде других нишевых видов деятельности. В последнее время начинают развиваться сети, направленные на решение бизнес-задач — установления деловых контактов (LinedIn) и совместной разработки ПО (GitHub), однако они сокращают издержки только какой-то одной из сторон коммуникативной деятельности. Представляется целесообразным интегрировать поиск партнеров и ресурсов и координацию деятельности в процессе выполнения проектов на одной платформе и совмещать их с возможностью непосредственного выполнения программных модулей из коммуникационной среды. Такой подход исключает необходимость загрузки-выгрузки программного обеспечения и данных, используемых в проекте, передачи их между пользователями средствами стороннего программного обеспечения и позволяет контролировать ход реализации проекта по фактическим стадиям обработки данных, находящихся внутри платформы.

Минимизация затрат на использование интеллектуальной собственности. Серьезной проблемой, тормозящей внедрение новых исследовательских методов в прикладных отраслях и ограничивающей комплексирование уже известных методов, является сложность в защите и учете использования интеллектуальной собственности. Защита последней традиционными патентами с оформлением индивидуальных договоров на ее использование с каждым пользователем делает неприемлемым время, требуемое для доступа к информационным ресурсам (ИР), а комплексирование отдельных методов в комплексные автоматизированные рабочие места (АРМы) и системы приводит к неприемлемой стоимости их применения в исследовательском процессе. Решение названной проблемы видится сегодня в широком применении открытых лицензий [?6], позволяющих пользоваться интеллектуальной собственностью на основе некоторых общих правил, без заключения индивидуальных договоров, в том числе, за счет повременной аренды ИР, известной для программного обеспечения как SaaS.

Другая проблема управления ИР — учет использования отдельных модулей — снимается методами распределенных реестров [?9] (блокчейнов), позволяющих фиксировать любые транзакции без доверенных центров регистрации. Важно, что снижение затрат на верификацию использования интеллектуальной собственности позволяет учитывать микроиспользование отдельных модулей и делает нецелесообразным комплексирование прикладных программных решений в крупные пакеты прикладных программ. Учет в распределенных реестрах использования всеми участниками платформы интеллектуальной собственности дает возможность организовать биллинг и взаиморасчеты, которым доверяют все участники.

2. Аналоги и прототипы

Разрабатываемая платформа должна предоставлять пользователям большое количество разнонаправленных функциональных возможностей. Краткий перечень таких возможностей включает:

- графовое представление алгоритма,
- широкие возможности по динамическому управлению алгоритмами и графом обработки,
- широкие возможности по визуализации данных,
- возможность конструирования специализированных графических интерфейсов для управления алгоритмами,
- поддержку коллективной работы над проектами,

- поддержку распределенных и высокопроизводительных вычислений,
- поддержку версионности данных и вычислений,
- поддержку возможности обработки и хранения больших объемов данных,
- наличие возможностей по управлению вычислительной инфраструктурой,
- учёт использования различных алгоритмов и вычислительных ресурсов.

В настоящий момент полноценная программная платформа, которая уже обладала бы всеми необходимыми свойствами, на рынке программного обеспечения отсутствует. В то же время ряд компаний предлагает программные продукты, которые реализуют отдельные из названных выше функций платформы. Поэтому такие продукты можно считать возможными прототипами. Так как необходимых к реализации функций требуется много, то и потенциальных прототипов также соответственно имеется достаточно большое количество. Такие прототипы можно условно разбить на следующие группы:

- программные средства анализа больших данных и организации научных исследований,
- инструменты программной организации процессов,
- инструменты моделирования бизнес-процессов,
- интеграционное программное обеспечение,
- открытые интернет-платформы для разработчиков программного обеспечения.

2.1. Программные средства анализа больших данных и организации научных исследований

Возможно, прототипами, наиболее близкими к предлагаемой нами платформе, являются программные продукты, реализующие среды для аналитики, работы с большими данными и построения моделей машинного обучения. К такому программному обеспечению, так же, как в нашем случае, предъявляются требования максимальной гибкости и вариативности реализуемых алгоритмов, а также возможности модификации процессов обработки данных конечными пользователями. Для выполнения указанных требований программное обеспечение представляется в виде набора алгоритмов обработки данных, средств графической визуализации входных данных и результатов вычислений, а также визуального инструмента построения необходимых графов

обработки. В настоящий момент известно большое количество подобных программных средств: Orange, RapidMiner Studio [?10, ?11], Knime Analytics Platform [?12, ?13], EasyMorph, AdvancedMiner. Названные программы позволяют решать такие задачи анализа, как подготовка данных (preprocessing), отбор признаков, кластеризация, классификация, регрессионный анализ и ряд других, т. е. сосредоточены в основном на математической и статистической обработке. Такие программы обеспечивают интегрированную среду для проведения научных исследований, позволяют реализовать полный цикл анализа данных, включающий чтение данных из различных источников, их преобразование и фильтрацию, собственно анализ, визуализацию и экспорт. К сожалению, названная перспективная группа программного обеспечения обладает рядом недостатков с точки зрения концепции платформы, обсуждаемой нами:

- отсутствуют или недостаточно развиты возможности контроля версионности данных и графов обработки;
- взаимодействие пользователя с системой ведется на основе прямой работы с графом обработки, нет возможности конструирования специализированного пользовательского интерфейса;
- поддержка распределенных вычислений или размещения в облачной среде в основном ограничена коммерческими версиями программ;
- вопросы организации хранения обрабатываемых данных вынесены за рамки функциональных возможностей систем;
- отсутствуют или недостаточны возможности организации коллективного взаимодействия при совместной работе над графом обработки.

2.2. Инструменты программной организации процессов

Одной из важных особенностей платформы является использование графов обработки или последовательностей задач. При этом необходимо, чтобы данная характеристика отражалась не только во внешнем представлении системы пользователю в виде отображения графических диаграмм, но и была внутренне присуща самой платформе, т. е. была заложена в ее базовые программное обеспечение, архитектуру и принципы разработки.

В настоящий момент активно развиваются программные средства разработки (библиотеки), предназначенные для низкоуровневой реализации последовательностей обработки данных, которые позволяют

осуществлять разработку, планирование и мониторинг рабочих процессов, представленных в виде графов. К таким программным средствам относятся Apache Airflow, Luigi, Nextflow и многие другие.

Данный класс программных средств не является полноценным программным продуктом и предназначен для использования профессиональными программистами, вследствие чего описание алгоритма обработки производится или непосредственно средствами соответствующих языков программирования, или в специализированных файлах, что делает невозможным или сильно затрудняет их использование и модификацию конечными непрофессиональными пользователями. Кроме того, присутствует еще целый ряд недостатков:

- отсутствие возможности ручного графического построения графа обработки, который генерируется на основе специализированного кода;
- разграничение этапов разработки и исполнения графов обработки.

2.3. Инструменты моделирования бизнес-процессов

Принципы разбиения длительных и сложных процессов на составляющие их компоненты также активно используются при моделировании бизнес-процессов (BPM). В настоящее время существуют хорошо развитое методологическое обеспечение, а также соответствующие языки моделирования и средства разработки программного обеспечения.

Достаточно широко используемыми программными продуктами являются Activiti BPM Platform, Camunda. Аналогичные программные инструменты предоставляются и производителями интеграционного программного обеспечения в продуктах JBoss, Websphere и ряде других. Эти программные продукты предлагаются и как средства разработки схем процессов, и как среды их исполнения и мониторинга; также имеется возможность разработки простых интерфейсов взаимодействия пользователя с системой.

Описанная группа программных продуктов также выглядела крайне перспективно в качестве основного прототипа разрабатываемой платформы, однако после выполнения программных экспериментов выявился следующий ряд недостатков:

- отсутствует поддержка высокопроизводительных вычислений;
- ограничены возможности оперативного изменения графа обработки во время его исполнения;
- отсутствует возможность многовариантной обработки одного набора данных в отдельных узлах процесса;

- отсутствуют возможности сложной визуализации научных данных.

2.4. Интеграционное программное обеспечение

Хорошо известен другой класс программного обеспечения, в котором используются принципы декомпозиции длительных последовательностей на составные части и представления их в виде графов при разработке и мониторинге — это интеграционное программное обеспечение (EAI, ESB). Данный класс программ осуществляет организацию и управление потоками данных между информационными системами предприятия и широко использует для этого представление в виде графов.

К широко известному программному обеспечению данного типа можно отнести следующие продукты: Apache ServiceMix, Mule, JBossESB, IBM WebSphere, WSO2, Apache NIFI.

Кроме универсальных интеграционных решений можно отметить наличие специализированных решений для нефтегазового сектора, например, платформу Whereoil [?14] компании Kadmi, декларирующей обеспечение информационной поддержки бизнес-процессов, связанных с управлением большим объемом данных в нефтегазовом секторе. Названный продукт дополнительно позволяет создавать представления бизнес-процессов, связывать их с непосредственными информационными ресурсами (системами моделирования) и организовывать сквозное проектное управление.

Такие программные продукты предназначены для выполнения процедур интеграции и управления потоками данных и фактически не поддерживают актуальные задачи комплексной математической обработки данных, исследования и визуализации результатов. Такие задачи частично поддержаны в упомянутом продукте Kadmi, но скудность доступной информации по этому проприетарному продукту не позволяет провести его оценку.

2.5. Открытые интернет-платформы для разработчиков программного обеспечения

В качестве прототипов создаваемой платформы также можно рассматривать инструментальные средства, которые переносят в интернет-среду инфраструктуру для разработки программного обеспечения. Исторически первыми такими программами были платформы, обеспечивающие версионное хранение исходных кодов проектов (Github, Gitlab и

Bitbacket). В настоящий момент они представляют собой комплексные сервисы по разработке и управлению программными проектами и кроме хранения кода обеспечивают командную работу, документирование проектов, интеграцию со средствами непрерывной сборки и тестирования приложений. Существуют интернет-платформы, которые позволяют создавать целые ландшафты разработки и тестирования, т. е. совокупность прикладных специализированных программных серверов, на которых размещаются программы, разработанные пользователями (Platform as a Service). Так, например, компания Salesforce предлагает облачную PaaS-платформу Heroku [?15, ?16], которая позволяет полностью перенести разработку ПО и среду его выполнения в облака. Компания Amazon дает возможность быстро создавать необходимую виртуальную программно-аппаратную инфраструктуру, в том числе, высокопроизводительную. Эта компания предоставляет средства для облачной разработки программного обеспечения, в том числе, сервис AWS Step Functions [?17] для организации координации компонентов распределенных приложений и микросервисов в виде графических схем рабочих потоков. Компания SAP предлагает платформу HANA [?18] для решения задач распределенного хранения данных, разработки инструментов их обработки, а также исполняемую среду для предоставления данных пользователю. Существует инструмент SAP Data Hub [?19], который позволяет организовать подключение к множеству источников данных и графически сформировать графы выборки и обработки данных.

Описанный класс программных продуктов может считаться прототипом разрабатываемой нами платформы в части предоставления широкого функционала по управлению вычислительной инфраструктурой и средствами организации процесса разработки, но в то же время имеет следующие существенные недостатки:

- эти продукты ориентированы на сообщество профессиональных программистов;
- ПО предназначено преимущественно для разработки приложений с веб-интерфейсом;
- они ориентированы на разработку и эксплуатацию программного обеспечения, а не на анализ и обработку данных;
- наиболее полнофункциональные платформы (продукты от компаний SAP, Amazon и Salesforce) не имеют открытого постоянного бесплатного доступа или являются очень дорогими решениями.

Крайне интересно выглядят два новых проекта Pallium.network [?20] и SingularityNET [?21], которые ставят своей целью построение рас-

пределенной в интернете коммерческой сети для решения задач, связанных с машинным обучением. Предполагается, что сторонние пользователи смогут подключать свои программные решения в качестве отдельных узлов такой сети и затем предлагать их для коммерческого использования остальным участникам. Для защиты интеллектуальной собственности и проведения оплаты потребляемых ресурсов предлагается использовать технологии блокчейн. Однако такие программные продукты находятся в стадии разработки, и ориентироваться на них можно в основном в методологическом плане.

Итак, несмотря на значительное многообразие программных продуктов, каждая из представленных выше групп ПО обладает рядом существенных недостатков, которые препятствуют их непосредственному использованию при реализации платформы согласно нашей концепции. Таким образом, хотя первоначально у нас было желание доработать под требования платформы уже существующее программное обеспечение и тем самым уменьшить сроки разработки полнофункциональной версии платформы, впоследствии пришлось отказаться от такого решения и приступить к разработке нового программного продукта, прототип которого представлен ниже.

3. Предлагаемое решение

Предлагаемое нами решение основано на перечисленных выше подходах к формированию архитектуры программных средств и бизнес-архитектуры, а также принципах управления интеллектуальной собственностью.

Программная основа. Ею является интернет-платформа, состоящая из центрального сервера приложений (репозитория), обеспечивающего хранение алгоритмов и их компонент, регистрацию и сопровождение пользователей, коммуникации между ними, а также учет использования ими алгоритмов и их компонент при решении прикладных задач.

Использование алгоритмов и их компонент происходит в исполняемой среде (движок), загружаемой при присоединении к платформе либо на машину пользователя, либо на виртуальную машину в облачном кластере. Движок обеспечивает создание, модификацию и использование (в том числе, из множества стандартных) тех алгоритмов, которые решают конкретные прикладные задачи. Взаимодействие пользователя с движком и репозиторием осуществляется через веб-интерфейс или толстый клиент на локальной или виртуальной машинах.

Пользовательская и административная части. На платформе действуют три основных типа пользователей:

- эксперты, разрабатывающие новые типы алгоритмов в рамках научных и прикладных грантов, выполнения заказных работ или в инициативном порядке; разработанные алгоритмы снабжаются идентифицирующими информацией и описанием и депонируются в репозиторий платформы;
- специалисты, обеспечивающие выбор и адаптацию имеющихся алгоритмов к новым или изменившимся старым задачам;
- исполнители, решающие конкретные прикладные задачи на основе алгоритмов, разработанных экспертами и специалистами; взаимодействие исполнителей с системой происходит аналогично работе с обычным программным обеспечением.

Лицензионная политика и взаимодействие участников. Лицензионная политика в области интеллектуальной собственности основана на авторском праве и осуществляется на основе присоединения к лицензионной политике платформы (договор присоединения). В таком договоре должны быть прописаны обязательства пользователя по соблюдению авторских прав третьих лиц — участников платформы (в том числе, имущественных прав при использовании сторонних объектов интеллектуальной собственности), а также обязательства платформы по соблюдению прав присоединившегося пользователя.

Оплата использования алгоритмов производится на основе авторского права. Ведущим механизмом является роялти, однако для эксклюзивных алгоритмов возможен механизм авторского заказа с выплатой паушального платежа. Для модифицированных алгоритмов применяются механизмы распределения авторского вознаграждения производных произведений. Для разработок, выполненных в рамках должностных инструкций, действуют механизмы служебных вознаграждений.

4. Преимущества предлагаемого решения

Мы прогнозируем резкое качественное снижения сроков и затрат на модификацию функционала ПО. Когда профессиональные пользователи получают возможность самостоятельно модифицировать алгоритмы обработки данных и управляющий алгоритмами графический интерфейс, отпадает необходимость привлечения профессиональных программистов. Повышенные возможности модифицируемости программы позволяют приступить к опытно-промышленной эксплуатации системы в начале её разработки, а процессы разработки и опытно-промышленной эксплуатации сливаются в единый итерационный процесс.

Выявление программно-технических и алгоритмических проблем на ранних стадиях повышает качество решения.

В качестве примера приведем наш личный опыт в разработке программного обеспечения. Одним из последних наших проектов была разработка программной системы обработки и анализа данных с помощью специализированного запатентованного геофизического метода. Особенностью метода является необходимость его тщательной настройки под конкретные геологические условия проведения исследований. Нами в течение полутора лет проводилась разработка необходимого программного обеспечения. К сожалению, результат нельзя было признать удовлетворительным. Несмотря на то, что продукт был создан и обладал широким функционалом, его внедрение в эксплуатацию постоянно задерживалось, так как требовались постоянные корректировки алгоритмов. При этом процесс реализация программистами прикладных алгоритмов всегда был достаточно длителен, так как требовалось время на понимание задачи, анализ алгоритма и его реализацию на языках программирования, которые плохо подходят для научных алгоритмов. В итоге было принято решение разработать новый продукт с нуля на основе принципов, описанных выше. Как результат, через полгода новая программа (альфа-версия описанного ниже прототипа) уже находилась в промышленной эксплуатации. Это означало минимум троекратное увеличение скорости внедрения продукта при соответственном снижении затрат на услуги профессиональных разработчиков.

Предлагаемое решение обладает дополнительными преимуществами, перечисленными ниже.

Контроль выполнения работ. Объект DAG (направленный ациклический граф), реализующий некоторый бизнес-процесс (выполнение аналитического исследования, разработка проекта, составление отчета), является одновременно и сетевым графиком работ. По мере заполнения промежуточных структур данных можно судить о ходе выполнения работ, прогнозировать сроки выполнения и степень загрузки исполнителей. В отличие от классической программной архитектуры DAG-архитектура не требует дополнительных усилий по интеграции с системами управления проектами. Задачи управления проектами и ресурсами могут быть встроены в платформу как дополнительный сервис.

Верификация сложных результатов. Выполненные аналитическое исследование, отчет или проект в предлагаемом подходе представляют

собой DAG, включающие в себя исходные, окончательные и промежуточные данные, а также алгоритмы их получения. При маркировании в ходе выполнения работ алгоритмов и данных с помощью блокчейн-технологии обеспечивается гарантия соответствия полученных результатов исходным данным и примененным алгоритмам. Возможен контрольный прогон DAG от исходных данных к результату для его подтверждения.

Самодокументируемость. Сложные информационные системы с большим количеством функций, распределенных по автоматизированным рабочим местам, чрезвычайно сложны в освоении и эксплуатации. DAG представляет собой цепочку выполняемых бизнес-функций для конкретного проекта, что делает его самоочевидным для специалиста.

Интеграция прикладных исследований. Разработка новой математической модели или бизнес-функции с помощью сторонних экспертов является в настоящее время процессом с большими транзакционными издержками, а практическое её внедрение занимает много времени. Интернет-платформа даёт возможность резко ускорить процесс и приблизить прикладные исследования к решению практических задач. Организовать прикладное исследование в ней означает сформулировать требования к результату и обеспечить исполнителю доступ к тестовым данным. Результатом прикладного исследования является DAG, демонстрирующий на тестовых данных достигнутые результаты и готовый к использованию реальных данных.

Готовность к облачной технологии. Предлагаемая архитектура опирается на подходы, разработанные для распределенных вычислений, и полностью готова к эксплуатации на публичных или частных кластерах.

5. Прототип

Согласно описанной выше концепции был разработан прототип платформы, общую архитектуру которого можно представить в следующем виде (Рисунок **??**):

- регистрирующий узел (central reg node);
- управляющий кластер (control node);
- вычислительный кластер (compute cluster);
- клиентские приложения (cloudview).

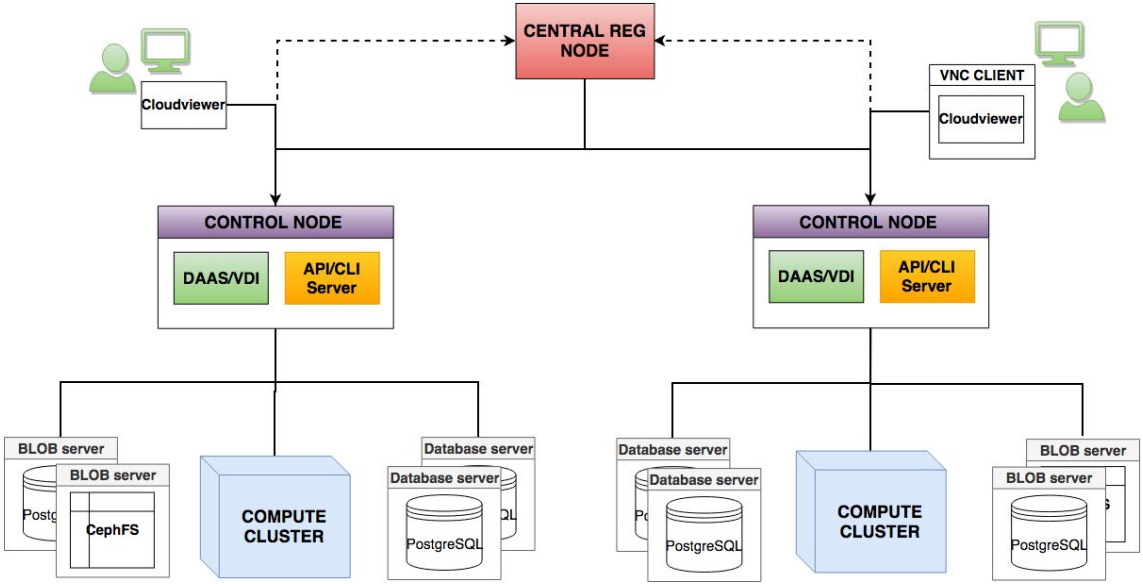

Рисунок 1. Общая архитектура платформы

0

Регистрационный узел — это точка регистрации различных управляющих и вычислительных кластеров для обеспечения интеграционных процедур и создания единой точки доступа. Кроме того, регистрационный узел обеспечивает учет пользователей для всех кластеров, подключенных к данной платформе, а также размещение для всех проектов общедоступных объектов.

Управляющий кластер — это программно-аппаратная инфраструктура, реализующая основные сервисы платформы и предоставляющая к ним удаленный доступ в интернет-среде. Исполнение ресурсоемких программ производится на подключаемых вычислительных кластерах.

Вычислительный кластер — это высокопроизводительная программно-аппаратная инфраструктура, предназначенная для выполнения ресурсоемких пользовательских программ (скриптов). Вычислительные кластеры могут быть как интегрированными в управляющие кластеры, так и являться кластерами общего пользования, входящими в общедоступные вычислительные сети.

Клиентские приложения — это набор программного обеспечения, который обеспечивает взаимодействие пользователя с сервисами управляющего кластера платформы. Клиентские приложения могут подключаться к любым управляющим кластерам при наличии прав у соответствующих пользователей. Для поиска доступных управляющих кластеров может использоваться регистрационный узел.

Архитектура платформы позволяет проводить гибкую настройку в зависимости от потребностей пользователей и создавать на ее основе как закрытые площадки внутри компаний, так и открытые — в интернете. Технически возможна интеграция различных разрозненных площадок в одну, что часто требуется при проведении сложных комплексных работ с множеством исполнителей.

Рассмотрим подробнее архитектуру управляющего кластера — основного элемента платформы. Он состоит из:

- центрального сервера приложения;
- реляционного хранилища;
- объектного хранилища;
- сервисов.

Центральный сервер приложений является единой точкой доступа к сервисам, предоставляемым данным управляющим кластером, а также обеспечивает:

- API&CLI для работы сервисов;

- среду размещения/регистрации сервисов платформы;
- организационное взаимодействие пользователей с платформой.

Реляционное хранилище предназначено для хранения служебной информации платформы, структуры и метаинформации графов обработки, хранения информации сервисов, а также непосредственного хранения данных небольших объемов. Объектное хранилище предназначено для хранения пакетов данных больших размеров.

Непосредственное хранение осуществляется в специализированном key-value архиве, в качестве которого выступает база данных PostgreSQL 10. Данные хранятся в виде байтовых массивов (bytea) в записях таблиц. Для снижения нагрузки на единичные таблицы данные распределяются по группе таблиц согласно первым символам хеша (ключа) хранения. Предусмотрена возможность разбиения данных на блоки определенного размера и хранения в виде цепочки таких блоков.

Сервисы — это программные модули, развернутые на центральном сервере приложений и реализующие различные функциональные возможности управляющего кластера платформы. Сервисы взаимодействуют с основными компонентами управляющих и вычислительных кластеров платформы программным способом и предоставляют пользователю возможность оперировать функционалом платформы.

Таким образом, управляющий кластер позволяет хранить и обрабатывать данные различных типов и различных размеров, а также обеспечивать работу группы пользователей.

Важной особенностью платформы является возможность проведения вычислительно нагруженных и ресурсоемких работ по моделированию и обработке данных различных исследований. Для реализации этого принципа управляющий кластер прототипа был интегрирован с высокопроизводительной GPU-системой и системой удаленного VDI-доступа, программно-аппаратная архитектура которых описана в [?22]. Для проведения распределенных вычислений поддерживается система управления общими ресурсами и выполнения групп задач SLURM, а для выполнения локальных задачи используется система управления NQ.

В рамках работ по формированию прототипа был реализован ряд клиентских приложений, которые обеспечивают:

- администрирование платформы;
- функционирование личного кабинета пользователя;
- управление проектом (редактор и менеджер проекта);

- коллективную работу на основе проведения дискуссий и обмена графами обработки и данными на форумах и галереях платформы.

Все клиентские приложения разработаны в виде desktop-клиента с использованием языка программирования Python 3 и технологии PyQt 5. В связи с этим для обеспечения возможности работы с данными больших размеров основная деятельность пользователей происходит удаленно на площадке VDI-кластера. Начаты исследования по разработке веб-клиента для организации взаимодействия с сервисами платформы.

Для возможности расширения функциональности клиентских приложений сторонними пользователями предусмотрена система плагинов. Плагин позволяет изменять как визуальное представление приложения пользователю, так и расширять его производственные возможности.

Разработанный прототип внедрен в опытно-промышленную эксплуатацию в одной из геофизических компаний России и активно ею используется при реализации текущих проектов в нефтегазовой отрасли.

6. Преимущества прототипа

Представленный выше прототип может рассматриваться только как промежуточный вариант платформы. В то же время он уже имеет ряд преимуществ перед существующими программными продуктами, описанными в разделе «Аналоги и прототипы».

Прототип представляет собой *первую* реализацию глубокой интеграции (синтеза) в едином программном комплексе разносторонних средств коллективной разработки и эксплуатации алгоритмов анализа и обработки данных, которые в представленном варианте ранее предлагались пользователям в виде отдельных программных продуктов. В отличие от существующих аналитических платформ прототип не ограничивает пользователей в выборе языка программирования при разработке скриптов алгоритмов: если в прототипе отсутствует встроенная поддержка какого-либо языка, то пользователю достаточно самостоятельно реализовать процесс загрузки/выгрузки данных в скрипт в обменный интерфейс в виде файлового каталога со стандартизированным размещением входных и выходных данных.

Другой особенностью прототипа, которая отличает его от существующих систем анализа данных типа Knime или Rapidminer,

является возможность разработки специализированного графического интерфейса (автоматизированного рабочего места пользователя или АРМ) над узлами графа обработки для сокрытия деталей реализации алгоритмов и процессов от конечных пользователей и упрощения его взаимодействия с программой. В общем виде АРМ представляет собой набор форм ввода/вывода и визуализации параметров и данных, связанных с конкретными узлами графа обработки.

Графический интерфейс, отображаемый пользователю, определяется конкретным состоянием графа обработки. Для создания АРМ должен использоваться специальный конструктор, который позволяет объединить панели ввода и визуализации параметров и данных, взятых с различных узлов, в новые формы представления. В прототипе обычный пользователь может создать такой АРМ как для отдельного узла (рис. **??**), так и для группы узлов и всего графа.

Таким образом, пользователь может работать уже не с графом обработки, а с привычным всем интерфейсом в виде набора управляющих элементов и элементов визуализации. Это позволяет составлять стандартные шаблоны (процессы) обработки данных в виде специализированных АРМ и передавать их в эксплуатацию менее квалифицированным пользователям. В то же время профессиональный пользователь имеет возможность перейти в графовый режим и оперативно изменить необходимые ему алгоритмы.

К достоинствам прототипа можно отнести также возможность гибкого управления версиями алгоритмов и их параметров, так как для каждого конкретного процесса исполнения можно выбрать их различные сочетания. При этом каждый имевшийся в прошлом процесс исполнения алгоритма пользователь имеет возможность проанализировать в независимости от того, какие изменения алгоритма или данных произошли после этого.

Все процессы исполнения алгоритмов связаны в единую последовательность согласно графа обработки, и пользователь может в любой момент получить полную картину проведенных исследований (**??**) и при необходимости начать их заново с любого интересующего его момента (отметим, что пока существует ряд ограничений, связанных с историей изменения самого графа, но они должны быть преодолены в следующих версиях). Такая возможность ретроспективного анализа самого процесса исследований бывает крайне полезной как в научных, так и в бизнес-задачах и ранее не встречалась нами в других изученных платформах.

Рисунок 2. Пример АРМ

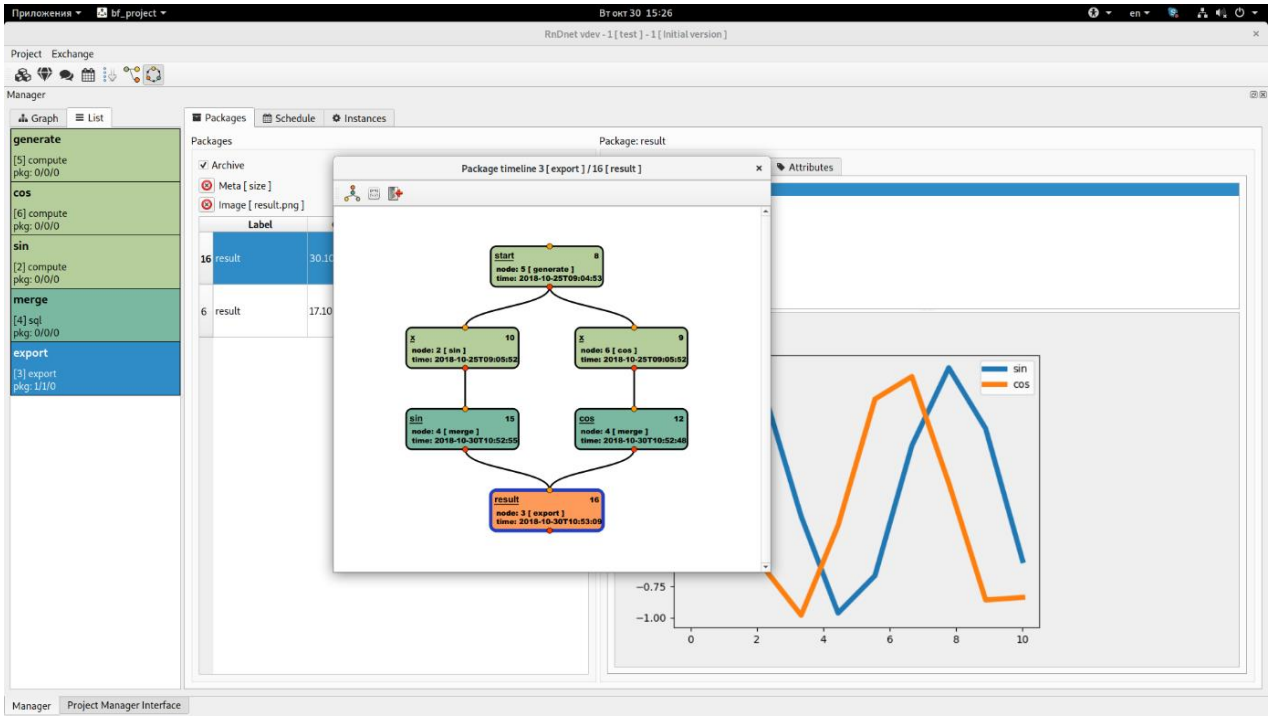

Рисунок 3. Пример отображения истории обработки

В основных аналогах создаваемой платформы, проанализированных выше, поддержка коллективной работы вынесена в отдельные платные версии программ, что сильно ограничивает возможности их применения в сложных научных или производственных процессах с участием большого количества участников.

Описанный прототип изначально был предназначен для коллективной работы над проектами, в том числе, в ситуациях, когда данные и вычислительные ресурсы географически разнесены. Технически каждый алгоритм (блок графа) может выполняться на различных вычислительных кластерах или компьютерах пользователей. Более того, предполагается, что различные экземпляры прототипа платформы, установленные у разных пользователей или организаций, могут объединяться в единую сеть и производить как обмен алгоритмами, так и передавать данные из одного графа обработки в другой, для чего реализованы специализированные модули экспорта/импорта.

В проектах Pallium.network и SingularityNET запланирована похожая функциональность. Однако в них предполагается построение единой вычислительной сети, а созданный нами прототип программной платформы позволяет строить группы независимых вычислительных сетей.

Встроенный в прототип форум даёт перспективную функциональную возможность (ранее не встречаемая авторами) обмена частями графа обработки вместе с необходимыми данными между различными пользователями. Необходимые части графа и данные могут загружаться в сообщения форума (Рис. **??**) непосредственно из одного проекта и выгружаться в другой произвольный проект. Это позволяет пользователям вести производственные или научные дискуссии и сразу обмениваться частями алгоритмов, функционирование которых можно быстро проверить, выгрузив их в новый проект. Описанная функциональная возможность позволит во многих случаях решить известную службам технической поддержки проблему воспроизведения ошибки, так как пользователь сможет передать не только описание проблемы, но и непосредственно данные и часть графа, которая к ней приводят.

В настоящий момент времени на начальной стадии разработки находятся возможности проектного управления в рамках прототипа платформы. Запланирована реализация возможности определения конкретных исполнителей по каждому узлу (разработчиков алгоритмов),

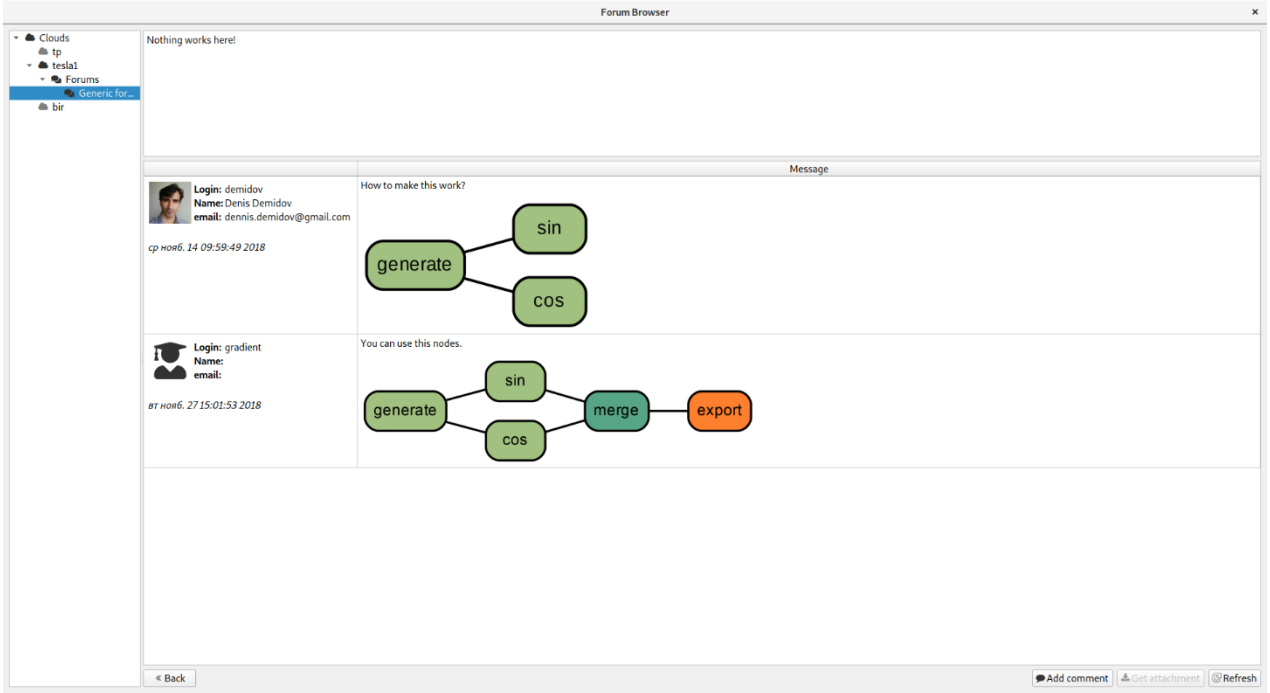

Рисунок 4. Форум с вложенными в сообщения графами

сроков начала и завершения разработки (Рис. ??). Таким образом, появится возможность контроля процесса исследования или обработки данных со стороны менеджмента компании, реализующей соответствующее исследование. В дальнейшем предполагается организация дополнительного информирования менеджера проекта о состоянии проекта: о разработанных узлах, успешно протестированных узлах и т. д. Это позволит повысить управляемость и открытость проводимых исследований. Также в прототипе пользователь может сгенерировать отчет по исследованиям, выбрав и визуализировав необходимые данные и введя дополнительную информацию (Рис. ??).

Описанные наиболее интересные возможности созданного прототипа программной платформы по отдельности хорошо известны, но в описанном сочетании в известных программных продуктах ранее не встречались.

7. Проблемы и направления дальнейшего развития

Прототип, представленный в настоящей статье, показал принципиальную реализуемость предложенного подхода, однако его прикладное применение требует решения ряда технологических вопросов, в том числе, решения проблемы интероперабельности, выбора стратегии работы с версиями в многопользовательском окружении, проблемы обеспечения защищенности данных и алгоритмов от несанкционированного доступа, а также повышения эргономичности (usability) его интерфейса пользователя.

Интероперабельность мы понимаем в данном случае в смысле необходимости настройки программного и системного окружения вычислительных узлов для возможности выполнения скриптов (программ) пользователей. Сейчас данный аспект не автоматизирован, и пользователю требуется заранее сообщать системным администраторам о необходимых ему настройках или самостоятельно готовить соответствующее виртуальное окружение Python или Docker контейнер.

Очевидно, что при массовом использовании платформы с большим количеством пользователей потребуется автоматизация данного процесса с целью его упрощения.

Примеры решения данной проблемы известны. Например, на платформе Heroku пользователь организует свою вычислительную инфраструктуру, самостоятельно выбирая необходимое ПО в специальном конструкторе, а в AWS каждый исполняемый скрипт снабжается дополнительным файлом с описанием необходимой ему инфраструктуры.

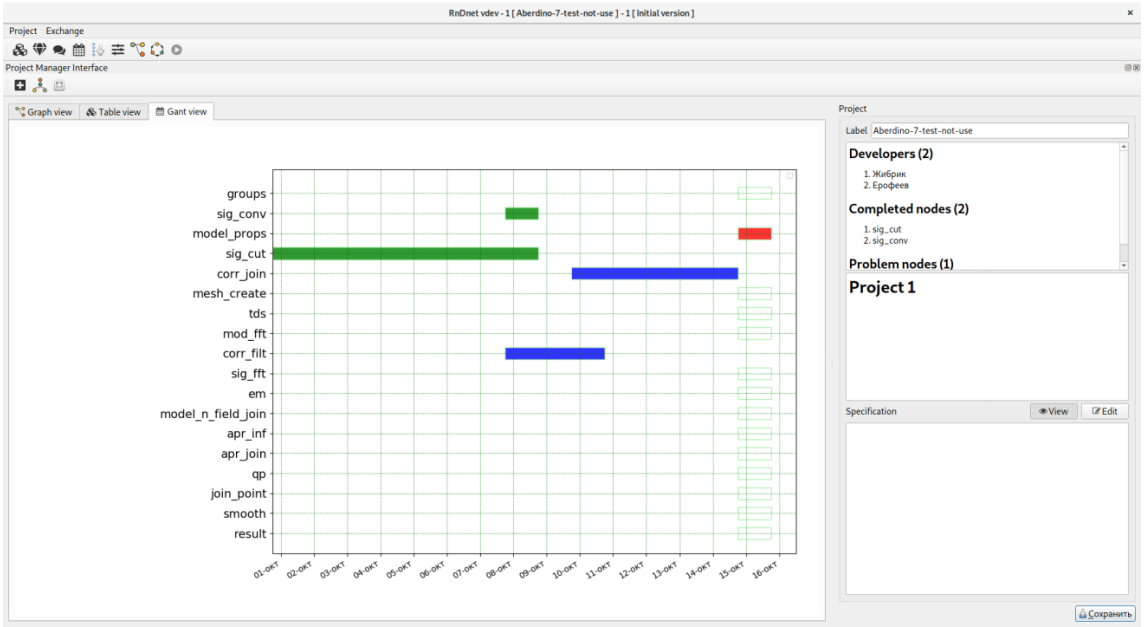

Рисунок 5. Диаграмма Ганта

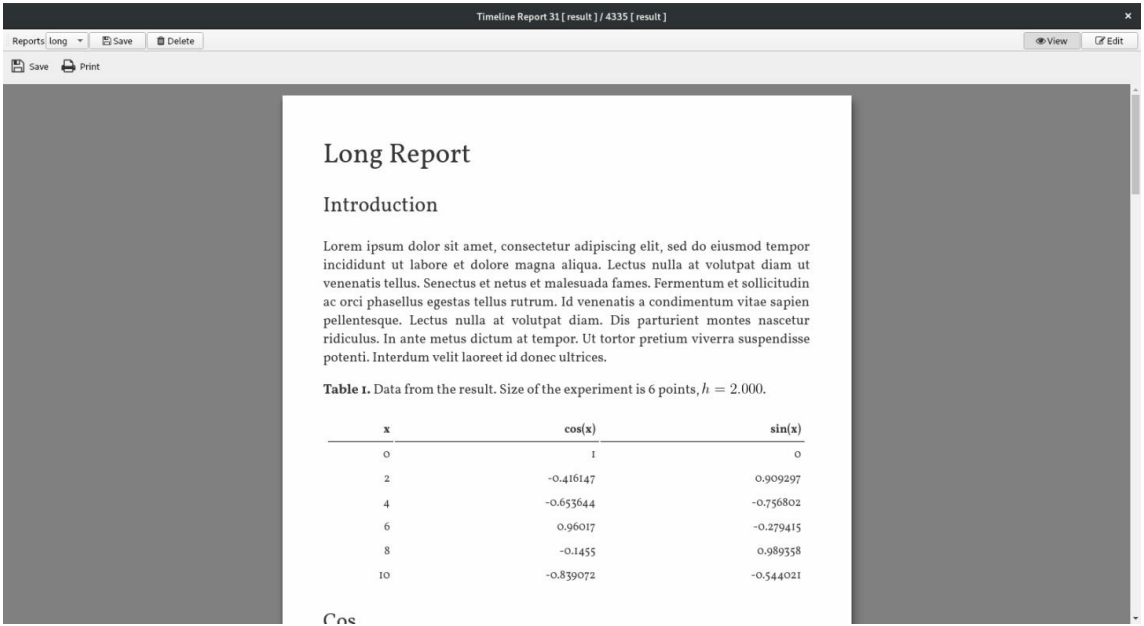

Рисунок 6. Пример отчета

Выбор между этими двумя стратегиями является ближайшей нашей задачей.

Одним из недостатков прототипа является недостаточная проработка вопросов, возникающих при многопользовательской работе в рамках одного проекта. Система работает в режиме единого потока данных и представления графа обработки для всех пользователей, то есть изменения, производимые одним пользователем, сразу же становятся доступными всем остальным. Такое прямолинейное поведение может предоставлять определенные проблемы в ряде случаев, например, когда необходимо быстро поэкспериментировать с последовательностью узлов обработки, не нарушая при этом текущего бизнес-процесса. Сейчас для этого необходимо производить клонирование проекта и данных, что может быть не всегда удобно. Альтернативным решением является использование версий графа обработки и потоков данных. Данное решение исследовалось, и была создана версия прототипа с поддержкой подобной функциональности, но решение еще не было протестировано в достаточной мере и поэтому в данной статье не рассматривается.

Также не решенными до конца являются вопросы защиты проприетарных алгоритмов обработки данных, представленных в виде скриптов на интерпретируемых языках программирования, от их несанкционированного копирования и использования. Известные к настоящему моменту способы программной защиты не дают полной гарантии сохранности. Возможным решением являются обеспечение физической недоступности алгоритма для пользователя за счет организации вычислений на изолированных вычислительных узлах и отказ от локального исполнения таких алгоритмов в интерактивном режиме. Существенным препятствием для облачных вычислений с конфиденциальными данными вообще является возможность утечки расчетных данных. Данная проблема сейчас отрефлексирована в виде концепции безопасных вычислений, принципиально исключающих возможность утечки, однако методы безопасных вычислений находятся в самом начале разработки и пока не могут использоваться в промышленных целях.

Другая проблема связана с построением сложного интерфейса пользователя для организации АРМ для управления графом непрофессиональным пользователем. Как показала практика использования прототипа, функциональность имеющихся графическим компонент не всегда устраивает конечных пользователей, и для удобства работы им требуется разработка сложных графических компонент со специализированными возможностями. Имеющийся сегодня прототип позволяет

разрабатывать новые компоненты, но это снова требует привлечения профессиональных программистов, что в общем случае снижает преимущества в скорости разработки платформы. Таким образом, одной из ближайших задач является разработка более широкого спектра графических компонент управления и визуализации, а также создание специализированного конструктора, который бы позволил пользователю создавать новые компоненты самостоятельно.

Мы также не рассматривали вопрос организационных и финансовых схем работы на предлагаемой платформе. Из самого платформенного подхода вытекает серьезное изменение структуры транзакционных издержек наукоемких проектов, что неизбежно повлечет за собой изменение ролей участников процесса и форм их взаимодействия, в том числе, финансовых взаимоотношений. Весьма вероятно, что для оптимизации организационных и финансовых отношений можно будет применить современные финансовые технологии (FinTech), в том числе, распределенный реестр, смарт-контракты и криптовалюту. Возможно, методом тотальной фиксации действий всех участников взаимоотношений на платформе с последующим анализом пользовательской активности методами data mining можно будет решить и проблему несанкционированного доступа, вынеся её с технического уровня на организационно-юридический.

Ответы на поставленные вопросы мы надеемся получить не только теоретически, но и в процессе экспериментальных внедрений разработанной платформы в различные отрасли.

Заключение

Проблемы и задачи, связанные с решением прикладных вычислительных проблем, описанные, поставленные и обсуждаемые выше, могут быть решены путем использования комплексной программной платформы, включающей в себя инструменты построения программных средств на основе графово-модульной архитектуры, и коммуникационной платформы для участников, позволяющей выполнять вычислительные графы, создавать средства регистрации интеллектуальной собственности и биллинга её использования с помощью методов распределенного реестра.

Предлагаемое нами решение может быть использовано для разработки программного обеспечения для разных отраслей промышленности и государственного управления в рамках развития парадигмы цифровой экономики. В частности, достаточно перспективным является

разработка на основе описанной цифровой платформы ситуационных центров глав регионов Российской Федерации [?23]. Такая платформа позволит не только решить ряд технических и организационных проблем создания и развития этих ситуационных центров, но и сформировать основу для совершенно нового типа взаимодействия между государством, бизнесом и обычными гражданами при решении задач, стоящих перед обществом.

Благодарности. Авторы благодарны ЗАО «Градиент» за огромную помощь в научно-исследовательских и практических технологических работах, оказанную при создании программного комплекса.

Список литературы

[1] *Ocean software development framework* (access date 11.02.2019). ↑

[2] *Маркетинговое исследование рынка программного обеспечения для геофизического сервиса в нефтегазовой сфере*, O2Consulting, M., 2014 (дата обращения 11.02.2019). ↑

[3] Б. Г. Левин. *Программная платформа для решения инженерных задач по разведке и разработке нефтегазовых месторождений «Геоплат Про»* (дата обращения 11.02.2019). ↑

[4] R. H. Coase. “The nature of the firm”, *Economica*, **4**:16 (1937), pp. 386–405. ↑

[5] D. Tapscott. *The digital economy: promise and peril in the age of networked intelligence*, McGraw-Hill, 1995, 342 pp. ↑

[6] H. W. Chesbrough. *Open innovation: the new imperative for creating and profiting from technology*, Harvard Business School Publishing, Cambridge, MA, 2003, 227 pp. ↑

[7] G. G. Parker, M. W. Van Alstyne, S. P. Choudary. *Platform revolution: how networked markets are transforming the economy and how to make them work for you*, W. W. Norton & Company, 2016, 352 pp. ↑

[8] E. V. Biryal’tsev, M. R. Galimov, A. M. Elizarov. “Workflow-based internet platform for mass supercomputing”, *Lobachevskii Journal of Mathematics*, **39**:5 (2018), pp. 647–654. ↑

[9] S. Nakamoto. *Bitcoin: a peer-to-peer electronic cash system.* ↑

[10] RapidMiner (access date 11.02.2019). ↑

[11] Z. Prekopcsák, G. Makrai, T. Henk, C. Gáspár-Papanek. “Radoop: analyzing Big Data with RapidMiner and Hadoop”, *Proceedings of the 2nd RapidMiner Community Meeting and Conference*, RCOMM 2011, 2011, pp. 1–12. ↑

[12] KNIME Analytics Platform (access date 11.02.2019). ↑

[13] W. A. Warr. “Scientific workflow systems: Pipeline Pilot and KNIME”, *Journal of Computer-aided Molecular Design*, **26**:7 (2012), pp. 801–804. ↑

[14] Whereoil (access date 11.02.2019). ↑

[15] Heroku (access date 11.02.2019). ↑

[16] N. Middleton, R. Schneeman. *Heroku: up and running: effortless application deployment and scaling*, O'Reilly Media, Inc., 2013, 100 pp. ↑

[17] AWS Step Functions (access date 11.02.2019). ↑

[18] V. Sikka, F. Färber, A. Goel, W. Lehner. "SAP HANA: the evolution from a modern main-memory data platform to an enterprise application platform", *Proceedings of the VLDB Endowment*, **6**:11 (2013), pp. 1184–1185. ↑

[19] SAP Data Hub (access date 11.02.2019). ↑

[20] Pallium Computing Network Concept (access date 11.02.2019). ↑

[21] B. Goertzel et al. *SingularityNET: a decentralized, open market and inter-network for AIs*, 2017 (access date 11.02.2019). ↑

[22] А. А. Беляева, Е. В. Биряльцев, М. Р. Галимов, Д. Е. Демидов, А. М. Елизаров, О. Н. Жибрик. «Кластерная архитектура программно-технических средств организации высокопроизводительных систем для нефтегазовой промышленности», *Программные системы: теория и приложения*, **8**:1 (2017), с. 151–171. ↑

[23] Е. В. Биряльцев, Р. Н. Минниханов. «Ситуационный центр главы региона Российской федерации в парадигме цифровой экономики», *Современные проблемы безопасности жизнедеятельности: интеллектуальные транспортные системы и ситуационные центры*, Материалы V Международной научно-практической конференции. Т. II, ред. Р. Н. Минниханов, Центр инновационных технологий, Казань, 2018, с. 3–11. ↑

Поступила в редакцию 18.12.2018
Переработана 24.04.2019
Опубликована 27.06.2019

Рекомендовал к публикации *д.ф.-м.н. С. М. Абрамов*

Пример ссылки на эту публикацию:

Е. В. Биряльцев, М. Р. Галимов, Д. Е. Демидов, А. М. Елизаров. «Платформенный подход к выполнению исследовательских и проектных работ с использованием высокопроизводительных вычислений». *Программные системы: теория и приложения*, 2019, **10**:2(41), с. 121–153.

10.25209/2079-3316-2019-10-2-121-153

http://psta.psiras.ru/read/psta2019_2_121-153.pdf

Эта же статья по-английски: 10.25209/2079-3316-2019-10-2-93-119

Об авторах:

Евгений Васильевич Биряльцев

Специалист в области специализированных информационных систем, к. т. н., автор более 50 публикаций, в том числе 3 свидетельств о регистрации программ, 2 изобретений. Генеральный директор компании ООО «Градиент технолоджи» (резидент Сколково). Заведующий Центром цифровых технологий Института перспективных исследований Академии наук Республики Татарстан.

0000-0002-5193-8627
e-mail: Igenbir@yandex.ru

Марат Разифович Галимов

Специалист в области разработки программного обеспечения для нефтегазовой отрасли, к. т. н., заместитель директора ООО «Градиент технолоджи» (резидент Сколково).

0000-0002-4997-6878
e-mail: glmvmrt@gmail.com

Денис Евгеньевич Демидов

Специалист в области высокопроизводительных вычислений с использованием технологий GPGPU, к. ф.-м. н, с. н. с КазО МСЦ РАН – филиала ФГУ ФНЦ НИИСИ РАН, с. н. с. ВШ ИТИС КФУ.

0000-0002-5794-5326
e-mail: dennis.demidov@gmail.com

Александр Михайлович Елизаров

Д. ф.-м. н., профессор Казанского (Приволжского) федерального университета, заслуженный деятель науки Республики Татарстан, директор КазО МСЦ РАН — филиала ФГУ ФНЦ НИИСИ РАН, член Американского математического общества (AMS), Немецкого общества математиков и механиков (GAMM) и Международного общества по индустриальной и прикладной математике (SIAM). Автор более 300 публикаций, в том числе 12 монографий.

0000-0003-2546-6897
e-mail: amelizarov@gmail.com

Sample citation of this publication:

Eugeniy V. Biryaltsev, Marat R. Galimov, Denis E. Demidov, Aleksandr M. Elizarov. “The platform approach to research and development using high-performance computing”. *Program Systems: Theory and Applications*, 2019, **10**:2(41), pp. 121–153. (*In Russian*).

10.25209/2079-3316-2019-10-2-121-153

http://psta.psiras.ru/read/psta2019_2_121-153.pdf

The same article in English: 10.25209/2079-3316-2019-10-2-93-119