Интеграционная архитектура адаптируемого распределённого программного обеспечения для решения задач газоснабжения

Леонов Дмитрий Геннадьевич
доцент, д.т.н.,
РГУ нефти и газа (НИУ) имени И.М. Губкина
119991, Москва, Ленинский пр-т., д. 65, к. 1
dl@asugubkin.ru

Папилина Татьяна Михайловна
к.т.н.,
РГУ нефти и газа (НИУ) имени И.М. Губкина
119991, Москва, Ленинский пр-т., д. 65, к. 1
papilina.tm@asugubkin.ru

Степанкина Ольга Александровна
РГУ нефти и газа (НИУ) имени И.М. Губкина
119991, Москва, Ленинский пр-т., д. 65, к. 1
olga@asugubkin.ru

Аннотация: Рассматриваются вопросы построения распределённого программного обеспечения для диспетчерского управления системами газоснабжения на основе открытой интеграционной платформы, позволяющей объединить разнородные программные комплексы. Описывается применение этого подхода при решении задачи краткосрочного прогнозирования газопотребления на основе распределённой системы сбора информации о технологических характеристиках и актуальных значений спроса газа.

Ключевые слова: интеграционные платформы, диспетчерское управление, системы газоснабжения, прогнозирование газопотребления.

The Integration Architecture of Adaptable Distributed Software for Gas Supply Tasks Solving

Dmitry Leonov
Doctor of Engineering Sciences,
National University of Oil and Gas «Gubkin University»
65 Leninsky pr-t, Moscow, 119991
dl@asugubkin.ru

Tatiana Papilina
Candidate of Engineering Sciences,
National University of Oil and Gas «Gubkin University»
65 Leninsky pr-t, Moscow, 119991
papilina.tm@asugubkin.ru

Olga Stepankina
National University of Oil and Gas «Gubkin University»
65 Leninsky pr-t, Moscow, 119991
olga@asugubkin.ru

Abstract: The research concerns with the problems of distributed dispatching control software for gas supply systems development on the base of open integration platform allowing the consolidation of heterogeneous software. The application of this approach to the task of short-range gas consumption forecasting on the base of distributed system of technological and consumption parameters acquisition is discussed.

Keywords: integration platforms, dispatching control, gas supply systems, gas consumption forecasting.

Введение

Прогнозирование объёмов потребления газа используют в процессе решения различных задач, например, при планировании поставок газа, расчёте режимов работы газотранспортной сети, проектировании новой или модернизации существующей газотранспортной системы (ГТС).

Различают несколько видов прогнозов: сверхкраткосрочный — по часам на несколько суток, краткосрочный — по суткам на неделю, среднесрочный — по неделям на месяц или квартал, долгосрочный — по месяцам на год, и перспективный — по годам на несколько лет. Прогнозы выполняются на различных иерархических уровнях управления системой. Краткосрочный прогноз необходим в оперативном автоматизированном диспетчерском управлении на уровнях диспетчерской службы газотранспортного общества и газораспределительной организации. Величины потребностей в газе используются для рационального перераспределения потоков газа между потребителями в течение суток, недели, месяца. Среднесрочные прогнозы позволяют заранее выявить и подготовиться к периоду повышенного потребления газа. Долгосрочные и перспективные прогнозы используют при планировании и анализе тенденций развития системы газоснабжения на верхнем уровень диспетчерского управления, там же используются агрегированные показатели регионального потребления.

Имеется большое количество работ, посвящённых оперативному, или краткосрочному, прогнозированию газопотребления, что говорит об актуальности задачи, обоснованные прогнозы являются эффективным инструментом планирования и управления. Но унифицированное решение задачи отсутствует, что может быть обусловлено уникальностью газотранспортных объектов: различиями в структуре, истории развития, уровнях автоматизации и телемеханизации.

1 Анализ подходов к прогнозированию газопотребления

1.1 Особенности и проблемы подготовки данных

Анализ процессов потребления газа проводят на основе статистики объёмов газопоставок, при этом в качестве потребителей могут выступать как отдельные объекты, например, конкретное предприятие, так и агрегированные, например, город, включая предприятия и население. В зависимости от уровня, на котором решается задача прогнозирования газопотребления, в качестве агрегированного потребителя могут рассматривать множество потребителей, получающих газ от одной газораспределительной станции (ГРС).

Особенностью задачи прогнозирования спроса на газ потребителями часто является приравнивание его к газопоставкам, что может привести к ошибочным выводам о сезонных закономерностях, если в результате аварии и/или недостаточной пропускной способности газопровода в период пиковых нагрузок не все потребности в газе удовлетворяются. В ситуации, когда невозможно полностью удовлетворить потребности, имеется регламент, в соответствии с которым отдельным потребителям ограничивают поставки газа.

Исходная информация представляется временными рядами объёмов поставленного газа с отдельными пропусками, и в ряде случаев значительно зашумлена. Источниками шумов являются погрешности и отказы измерительного оборудования, утечки, потери газа при ремонте и пр. При сведении данных об отданном предприятием магистрального транспорта газе и полученном конкретными потребителями последние два дня каждого месяца содержат не значения объёмов реальных поставок, а балансовые значения [1].

Анализ данных показал наличие нестационарности из-за периодического влияния различного характера на газопотребление. Факторы, оказывающие такое влияние, можно разделить на три группы: хронологические, метеорологические и организационные. К хронологическим факторам относятся: внутрисуточная неравномерность — смена дня и ночи, внутринедельная — чередование рабочих, выходных и праздничных дней, сезонная — смена времён года. Наиболее значимым метеорологическим фактором является температура окружающей среды, изменение которой отражается на объёмах потребления газа, поступающего на нужды отопления и вентиляции промышленных предприятий и жилых помещений. Организационные факторы изменения в структуре потребления связаны с подключением новых потребителей, реконструкцией предприятий, приводящей к смене технологии производства, переходами на альтернативные виды топлива, принудительным ограничением потребителей, проведением ремонтных работ и др.

Кластеризация потребителей газа Нижегородской области [2] позволила выявить, что классы различаются по сезонной составляющей, то есть в отдельный кластер попали энергоёмкие потребители, для которых объёмы поставок в течение года существенно не меняются, и был свой кластер для потребителей, нуждающихся в газе только в холодный период времени. Помимо крайних случаев можно выделить одну или несколько промежуточных групп потребителей.

1.2 Статистические методы

При построении модели прогнозирования газопотребления должно быть учтено влияние на газопотребление температуры окружающего воздуха и хронологических факторов, к которым относят день недели и сезон. Дни недели обычно разделяют на группы, и для каждой группы строят свою модель. Сезонная неравномерность газопотребления, по существу, учитывается путём введения в модель зависимости от температуры воздуха.

В качестве примера рассматривались данные нескольких потребителей «Газпром трансгаз Москва» за 2011-2013 годы, для которых было проведено прогнозирование объёмов потребления газа различными методами.

Среди статистических методов прогнозирования временных рядов удачной оказалась линейная многофакторная регрессионная модель «с памятью» [3], по сути представляющая собой авторегрессию:

$$Q(t) = aT_t + bQ_{t-1} + cQ_{t-2} + cQ_{t-3} + \varepsilon_t\,, \tag{1}$$

где a, b, c, d — коэффициенты регрессии, ε_t — свободный член, T_t — значение температуры окружающей среды на момент прогноза, $Q(t)$, Q_{t-1}, Q_{t-2}, Q_{t-3} — значения поставки газа за предшествующие три дня. Три дня были выбраны для конкретного потребителя в результате перебора различных вариантов по количеству дней. Выбран вариант с наибольшим значением регрессионной статистики R^2:

$$R^2 = 1 - \frac{\sigma^2}{\sigma_y^2}\,, \tag{2}$$

где σ^2 — условная по входным факторам дисперсия, σ_y^2 — условная дисперсия выходной переменной. При равенстве или очень близких значениях R^2 выбирается вариант с меньшим значением ошибки $\hat{\sigma}$:

$$\hat{\sigma} = \frac{\sigma}{\sqrt{n}}\,, \tag{3}$$

где n — объем выборки, σ — среднеквадратическое отклонение.

Отдельно были рассчитаны коэффициенты для рабочих и для выходных дней и проведён прогноз на неделю. Средняя ошибка для прогноза не превысила 3%.

Средняя ошибка прогноза по ARIMA-модели Бокса-Дженкинса составила 5%.

1.3 Нейронные сети

Современным и эффективным методом прогнозирования является прогнозирование с помощью искусственной нейронной сети. Нейронная сеть представляет модель параллельно работающих элементов — нейронов, соединённых определённым образом, подобным функциональной организации биологических нейронных сетей нервных клеток живых организмов.

Часть нейронов сети помечена как входы сети, а некоторые нейроны — как выходы сети. Задавая числа на входе сети, на выходе получают отклик. То есть входной вектор преобразуется в выходной. Такое преобразование задаётся весами связей сети, которые настраиваются в процессе обучения. Таким образом, выявляются зависимости между входными и выходными векторами, что делает сеть способной к прогнозированию.

На вход искусственного нейрона поступают сигналы (рисунок 1). Каждый входной сигнал умножается на соответствующий ему вес W_i. Все произведения суммируются, определяя уровень активации нейрона. Далее полученный сигнал поступает для преобразования в функцию активации, а затем на вход нейронов следующего слоя. В качестве функции активации выбирается функция, приводящая сигнал нейрона к определённому диапазону, в нашем случае — сигмоида. Сигналы, поступающие на вход нейронной сети, должны быть нормализованы, для это-

го значения исходного ряда пересчитываются как отношение разности значения с минимальным к размаху, таким образом, исходный диапазон значений ряда приводится к диапазону от 0 до 1.

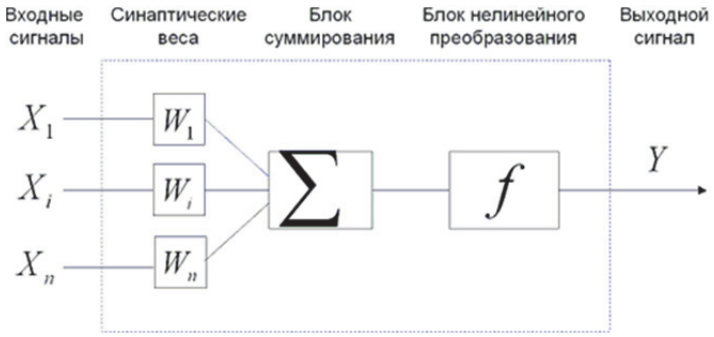

Рисунок 1 – Модель искусственного нейрона

На рисунке 2 представлена модель многослойной сети с 1 скрытым слоем и обратным распространением ошибки.

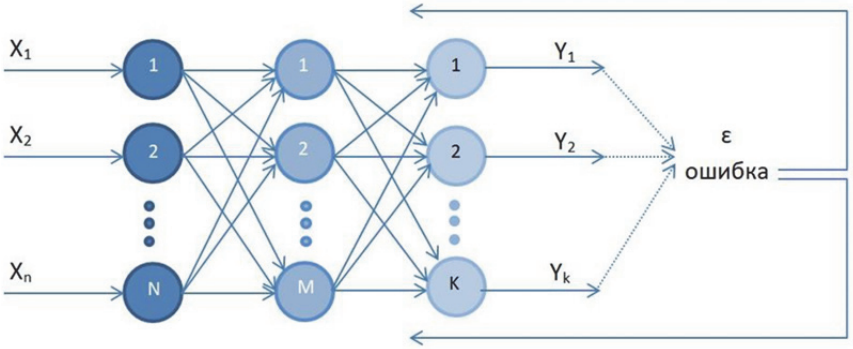

Рисунок 2 – Модель трёхслойной нейронной сети с обратным распространением ошибки

Алгоритм обучения состоит в следующем: инициализация весовых коэффициентов W случайными значениями из интервала (0; 1); нейроны входного слоя принимают значения и передают их на входы нейронов скрытого слоя; нейроны скрытого передают на входы нейронов выходного слоя значения функции активации от взвешенной суммы входных значений, то же делают нейроны выходного слоя. Далее работает алгоритм обратного распространения ошибки: вычисляется ошибка как произведение разности вычисленного значения и эталонного выходного значения и производной функции активации для входного значения взвешенной суммы. Произведение полученной величины и параметра сети α ($\alpha \in [0; 1]$), отвечающего за скорость обучения, даёт величину для корректировки смещения взвешенной суммы, а то же произведение со значением выходов нейрона даёт величины для корректировки весовых коэффициентов W. Аналогично считаются ошибки и поправки для значений и коэффициентов скрытого слоя. Далее пересчитываются все выходы с учётом поправок к весовым коэффициентам. Если сумма разностей расчётных значений выходного слоя и эталонных значений меньше заданной величины ошибки $Q = \sum^{k}(Ti - Yi) < \varepsilon < 1$, или выполнено заданное количество циклов обучения, процесс обучения прекращается.

Средняя ошибка прогноза с помощью трёхслойной нейронной сети составила 3.59%.

Точность прогноза повышается с увеличением количества скрытых слоёв, при этом резко возрастает время обучения сети и расчёта прогнозных значений.

1.4 Оценка применимости фрактальных моделей

В качестве аппарата для предпрогнозного анализа может использоваться метод нормированного размаха, или RS-анализ, позволяющий сделать ряд выводов о характеристиках ряда и возможности его прогнозирования. Применение метода позволяет определить направления динамики изменений в условиях сильной зашумлённости и выделить периодическую и непериодическую циклическую составляющую.

Одной из вычисляемых характеристик является показатель Хёрста H — величина, которая уменьшается, если задержка между двумя одинаковыми парами значений временного ряда увеличивается.

Для $(x_n, y_n) = (ln(n), ln(R/s(n)))$ строят регрессионную зависимость $y_n = H x_n + C$.

Последовательности, для которых $H > 0.5$, сохраняют тенденцию, т. е. возрастание в прошлом наиболее вероятно приведёт к возрастанию в последующем, и наоборот. При значении 0,5 явной тенденции не прослеживается, а при $H < 0.5$ тенденция ряда стремится смениться на противоположную.

Для имеющихся данных показатель Хёрста варьируется от 0.79 до 0.82. Это означает, что исследуемые ряды содержат сохраняющуюся тенденцию к увеличению среднего уровня и дисперсии ряда. На рисунке 3 представлена зависимость *ln(R/s(n))* от *ln(n)*.

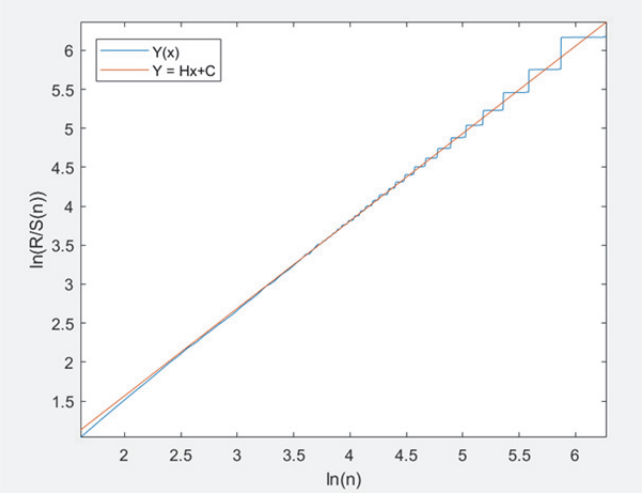

Рисунок 3 – Вид зависимости *ln(R/s(n))* от *ln(n)*

Также значение показателя Хёрста, лежащее в интервале (0.5, 1), говорит о том, что ряд прогнозируем.

Показатель Хёрста связан с фрактальной размерностью Хаусдорфа следующей формулой: $D = 2 - H$. Для исследуемых рядов $D \cong 1.2$. Так как $1 < D < 1.5$, это позволяет говорить о том, что ряд может содержать как детерминированную ($D = 1$), так и случайную составляющую ($D = 1.5$).

2 Организация сбора и обработки данных на основе открытой интеграционной платформы

Эффективная работа математических моделей, возникающих при описании задач прогнозирования газопотребления, требует постоянной актуализации фактических исходных данных. Решение задач прогнозирования объёмов потребляемого газа на современном уровне развития информационных технологий требует создания специализированной информационной системы, основанной на распределённом доступе к информационным и вычислительным ресурсам (например, взаимодействующей с геоинформационными системами для получения прогнозных значений температур) и способной войти в состав АСДУ как для сбора информации со SCADA-систем, так и для передачи обработанных данных для принятия решений на верхние уровни управления.

Различная степень автоматизации потребителей, использующих различные методы хранения и протоколы передачи данных, а также требования учёта реального спроса, отличающегося от фактических поставок, обусловливают необходимость построения гибкой архитектуры информационной системы, обеспечивающей эффективную интеграцию данных из разнородных источников.

Также следует учесть, что наметившаяся тенденция перехода российских государственных и крупных предприятий, в том числе ПАО «Газпром», на открытые решения (в частности, Astra Linux) в рамках политики импортозамещения в сочетании с накопленным парком Windows-систем приводит к необходимости проектирования информационных систем, способных адаптироваться к различным условиям функционирования.

2.1 Основные подходы к построению интегрированных систем

Развитие управления технологическими процессами соответствовало росту сложности принимаемых управленческих решений: от базовых систем автоматического регулирования к комплексным системам, обеспечивающим решение задач во всей цепочке управления, от технологических процессов до принятия организационно-экономических решений и сопровождающимся повышением степени использования информационных технологий (рисунок 4) [4].

Бурное развитие вычислительной техники и неоднородное внедрение ИТ привело к накоплению большого количества так называемого унаследованного программного обеспечения и связанной с этим необходимостью решения ряда технических проблем, обусловленных закрытостью результатов работы отдельных программных продуктов, несовместимостью используемых форматов и протоколов передачи данных, непереносимостью и плохой масштабируемостью программного обеспечения [5].

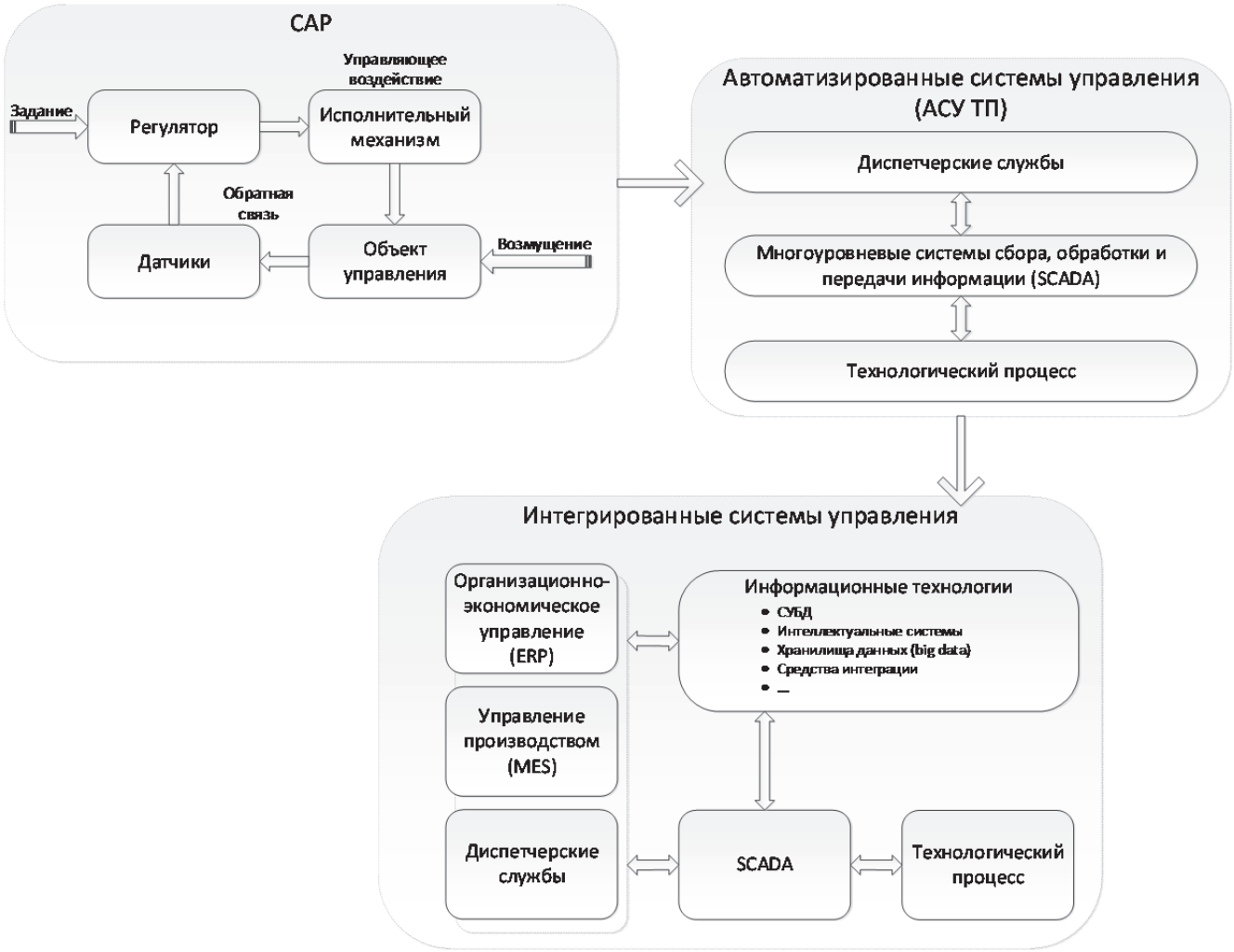

Рисунок 4 – Эволюция автоматизации управления технологическими процессами

Одновременно с этим процессом происходило развитие технических средств обеспечения интеграции. На ранних этапах они сводились к поддержке интеграции на уровне данных (файл-серверная, предполагающая организацию общего доступа к файлам с данными, а также совместное использование одной либо нескольких реплицируемых СУБД) и поддержке интеграции на уровне сетевых протоколов общего назначения (как правило, клиент-серверная). Недостатком подобных решений является слабая масштабируемость, обусловленная резким снижением управляемости по мере усложнения структуры системы.

2.1.1 Объектная интеграция

Возросшая популярность объектно-ориентированного подхода в 1990-х годах привела к созданию ряда конкурирующих архитектур, которые рассматривали интеграционные решения в контексте взаимодействия компонентов объектно-ориентированных программных систем.

Обобщённая архитектура брокера объектных запросов CORBA (Common Object Request Broker Architecture) — предложенная консорциумом OMG (Object Management Group) архитектура, предназначенная для интеграции гетерогенных систем, реализованных с помощью различных языков программирования и работающих на различных узлах сети предприятия. Основу программного кода составляют объекты, спецификация взаимодействия которых с внешним миром задаётся на языке описания интерфейсов, для которого определены правила отображения в синтаксические конструкции ряда популярных языков программирования.

Модель распределённого компонентного объекта DCOM (Distributed Component Object Model) была разработана компанией Microsoft и стала развитием компонентой модели COM, составляющей основу ряда операционных систем семейства Windows. Применение технологии COM к задачам автоматизации технологических процессов привело к созданию семейства программных технологий OPC (Open Platform Communications), в настоящее время ставшего основным стандартом взаимодействия устройств со SCADA-системами.

Enterprise JavaBeans (EJB) — распределённая компонентная модель, предназначенная для разработки распределённых переносимых Java-приложений. Взаимодействие модулей обеспечивается механизмом вызова удалённых

методов Java RMI (Java Remote Method Invocation), обеспечивающего локальную и сетевую передачу Java-объектов в сериализованном виде.

Перечисленные решения имеют ряд архитектурных и технологических ограничений по применению в качестве основы построения корпоративных интегрированных систем.

Архитектурные ограничения обусловлены всё ещё низким уровнем абстрагирования, обусловленным технологическими особенностями реализации приложений, а не спецификой бизнес-процессов, в рамках которых эти приложения предполагается использовать. Взаимодействие «точка-точка» между объектами, характерное для этих технологий, приводит к снижению управляемости и не гарантирует динамическое изменение связей при добавлении новых компонентов в процессе функционирования.

Технологические ограничения связаны с тем, что все перечисленные решения, будучи разработанными до массового распространения интернет-технологий, рассчитывают на надёжность соединений и предполагают использование специализированных бинарных сетевых протоколов. Более поздние реализации CORBA, DCOM и EJB включают расширения, позволяющие организовать передачу данных поверх стандартного протокола HTTP, но эти модификации не получили широкого распространения.

2.1.2 Сервисная интеграция

Эволюционным развитием идей, на которых была основана архитектура CORBA, стала разработка сервис-ориентированной архитектуры (Service-oriented architecture, SOA). В SOA все функции определяются в виде независимых сервисов с вызываемыми интерфейсами, а обращение к этим сервисам в определённой последовательности соответствует реализации того или иного бизнес-процесса. Существенным моментом является нейтральность интерфейсов к специфике реализации сервисов, приводящая к взаимозаменяемости сервисов и обеспечивающая относительную лёгкость адаптации системы к изменениям в реализации сервисов и в составе задач.

Хотя SOA является лишь набором архитектурных принципов, не оговаривающим технические детали, основную популярность ей принесла практическая реализация на основе web-сервисов, построение, публикация и использование которых формализовано группой открытых спецификаций, основанных на расширяемом языке разметки XML, таких как SOAP, WSDL и UDDI. В дальнейшем помимо SOAP стал широко использоваться более простой с точки зрения реализации, менее ресурсоёмкий и открытый для использования различных форматов данных протокол REST.

Одним из наиболее популярных подходов к построению SOA-систем, особенно в случаях, требующих асинхронного взаимодействия, стало использование сервисной шины предприятия (Enterprise service bus, ESB). Сервисная шина обеспечивает унифицированный механизм отправки запросов и получения результатов работы сервисов, концентрируя обмен сообщениями между сервисами в единой точке. Подключение существующего сервиса к сервисной шине реализуется с помощью написания адаптера, базовый набор которых уже входит в состав готовых продуктов. Популярные реализации сервисной шины включают такие пакеты как IBM Integration Bus, Microsoft BizTalk, Oracle Enterprise Service Bus, SAP Process Integration.

Переход на SOA является сложным процессом, требующим существенных изменений как в информационной инфраструктуре, так и во взаимосвязях между ИТ и бизнес-процессами. Ряд попыток перехода на SOA потерпел неудачу, что на некоторое время привело к снижению интереса к этой методологии. Его возрождение в последние годы связано с ростом популярности облачных технологий в сочетании с осознанием необходимости соблюдения эволюционного принципа создания сложных систем.

Основными недостатками реализации SOA на основе web-сервисов являются сложность реализации асинхронной связи между приложениями и неудовлетворительное быстродействие для передачи больших объёмов данных [6]. В связи с этим ряд ведущих производителей программного обеспечения предпочитает использовать традиционные средства организации взаимодействия, постепенно добавляя ограниченную поддержку новых стандартов. В частности, Honeywell UniSim Design Suite включает набор специализированных модулей интеграции наиболее распространённых продуктов сторонних производителей, а в качестве универсального средства доступа ориентируясь на более традиционную технологию OLE Automation. В продуктах Schlumberger Software Integrated Solutions используется открытая платформа IAM, также включающая отдельные модули для сторонних решений и средства импорта/экспорта файлов MS Excel.

Тем не менее, и для задач, требующих высокопроизводительного взаимодействия, применимы основные архитектурные принципы SOA, поддержанные более эффективной реализацией на основе брокеров сообщений, которые играют в данном случае роль облегчённой сервисной шины.

Примерами таких инструментов являются библиотека ZeroMQ, обеспечивающая низкоуровневую реализацию механизма передачи сообщений, и брокер сообщений RabbitMQ, использующий протокол AMQP. Преимуществом RabbitMQ является поддержка гарантированной доставки сообщений и кластеризация с дублированием очередей сообщений в целях повышения надёжности. Элементы сервисной архитектуры также были добавлены в CORBA 3.0 и Microsoft WCF.

Последовательное развитие этого подхода привело к созданию архитектуры микросервисов, идея которой заключается в использовании небольших специализированных модулей в сочетании с облегчёнными протоколами.

2.2 Методы разработки адаптируемого программного обеспечения

Под адаптируемостью (или портируемостью) программного обеспечения понимается возможность его адаптации (портирования) для работы в среде, отличающейся от той, для которой она разрабатывалась, по возможности с максимальным сохранением пользовательских свойств. Частным случаем такой адаптации является кроссплатформенная (или платформонезависимая) разработка, которая изначально ведётся с использованием инструментальных средств, обеспечивающих переносимость на том или ином уровне.

При разработке распределённой информационной системы, функционирующей в гетерогенном окружении, необходимо обеспечить адаптируемость для входящих в её состав клиентских приложений и серверных компонентов, а также используемой транспортной подсистемы.

Можно выделить следующие основные подходы к разработке адаптируемого программного обеспечения [7]:

- применение технологий тонких клиентов в качестве основного подхода к переносимости интерфейса приложений, первоначально разработанного под определённую систему; хорошо сочетается с использованием сервис-ориентированной архитектуры, обеспечивающей взаимодействие удалённых компонентов;
- применение кроссплатформенных библиотек, таких как Qt, wxWidgets и т.п.; обеспечивает максимальную производительность и функциональность, требуя перестроения и тестирования работоспособности приложений для каждой целевой платформы;
- использование решений, ориентированных на замкнутую среду выполнения, таких как виртуальная Java-машина (JVM) либо кроссплатформенную реализацию .NET с открытым кодом .NET Core; обеспечивает более высокую производительность и лучшую функциональность по сравнению с тонкими клиентами без необходимости перестроения приложений.

Адаптируемость транспортной подсистемы (т.е. брокера сообщений) подразумевает варианты миграции на решения, отличающиеся возможностями масштабирования, производительности и требований к ресурсам с сохранением при этом совместимости с существующей реализацией на уровне прикладного кода.

Можно выделить три стратегии миграции с одного брокера сообщений на другой. Первая предусматривает разработку вспомогательного транслятора в виде независимого сетевого модуля, принимающего сообщения от одного брокера и транслирующего их в сообщения другого. Реализация такого транслятора принципиально не отличается от разработки трансляторов для поддержки унаследованных приложений. Основной недостаток — увеличение накладных расходов за счёт последовательного использования двух брокеров. Основное достоинство — отсутствие необходимости модифицировать прикладной код.

Вторая стратегия ориентирована на автоматизированное генерирование каркасного кода на основе предварительно подготовленной модели [8]. Достоинством данного подхода является отсутствие накладных расходов, но модификация существующих приложений может быть значительно затруднена.

Последний вариант — разработка обобщающего программного интерфейса, реализующего трансляцию терминов и адресов с помощью вспомогательных функций. Объединяет преимущества первых двух подходов, поскольку накладные расходы сводятся к локальному преобразованию данных, а требуемые изменения в коде можно свести к минимуму.

2.3 Открытая интеграционная платформа как основа построения распределённых информационных систем в АСДУ ЕСГ

Проводимые в течение ряда лет в РГУ нефти и газа (НИУ) имени И.М. Губкина исследования в области разработки распределённых программно-вычислительных комплексов (ПВК) стали основой создания открытой интеграционной платформы (ОИП), предназначенной для организации взаимодействия разнородных программных продуктов на основе принципов сервис-ориентированной архитектуры [9].

ОИП стандартизирует формат и протоколы обмена данными с помощью открытого программного интерфейса (Application Programming Interface, API). Она является распределённой гетерогенной компьютерной средой, ключевые составляющие которой — сетевое ядро и реализация прикладного программного интерфейса, обеспечивающие подключение и взаимодействие разнородных клиентов (рисунок 5).

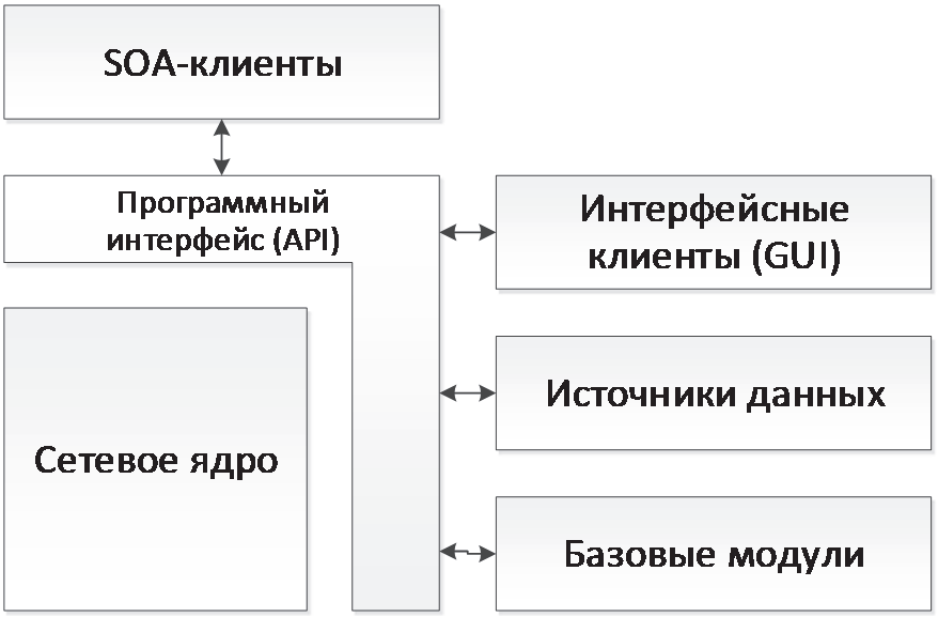

Рисунок 5 – ОИП: укрупнённая архитектура

При подключении к ОИП клиенты регистрируются в одной из следующих ролей:

- базовые модули, обеспечивающие основную расчётную функциональность и способные обслуживать запросы клиентских приложений, в роли которых выступают интерфейсные клиенты и источники данных;
- интерфейсные клиенты (как «тонкие», так и «толстые»), отвечающие за взаимодействие с пользователем;
- сторонние приложения и источники данных, обращающиеся к базовым модулям в роли клиентских приложений для выполнения отдельных задач.

Сетевое ядро ОИП отвечает за подключение и взаимодействие программных модулей на различных уровнях абстрагирования, предъявляющих различные требования к степени их интеграции и квалификации разработчиков:

- непосредственное взаимодействие с низкоуровневой реализацией механизма передачи сообщений;
- высокоуровневое взаимодействие на основе передачи данных по протоколу REST в формате JSON;
- применение общего низкоуровневого сетевого API для взаимодействия с популярными брокерами сообщений (ZeroMQ, RabbitMQ) и использующим библиотеку Boost.Asio брокером AsgardMQ.

В общем взаимодействии принимают участие следующие основные компоненты (рисунок 6):

- брокер сообщений (AsgardMQ, ZeroMQ либо RabbitMQ вместе с соответствующим транслятором при необходимости организации взаимодействия с другой реализацией механизма передачи сообщений);
- диспетчеры модулей, отвечающие за отслеживание активных подключений и распределение задач;
- функциональное ядро, обеспечивающее решение прикладных задач;
- подсистема взаимодействия с web-клиентами (интерфейс WMI);
- web-клиенты WAPP (Web Applications);
- сторонние и «толстые» клиенты (Legacy Applications, LAPP), взаимодействие которых с ОИП требует создания дополнительных адаптеров, также управляемых диспетчерами модулей и схем;
- клиенты RAPP (REST Applications), взаимодействующие с ОИП с помощью прикладного интерфейса и являющиеся элементами SOA-системы верхнего уровня.

Программный интерфейс ОИП обеспечивает взаимодействие подключённых к сетевому ядру модулей в терминах прикладных задач.

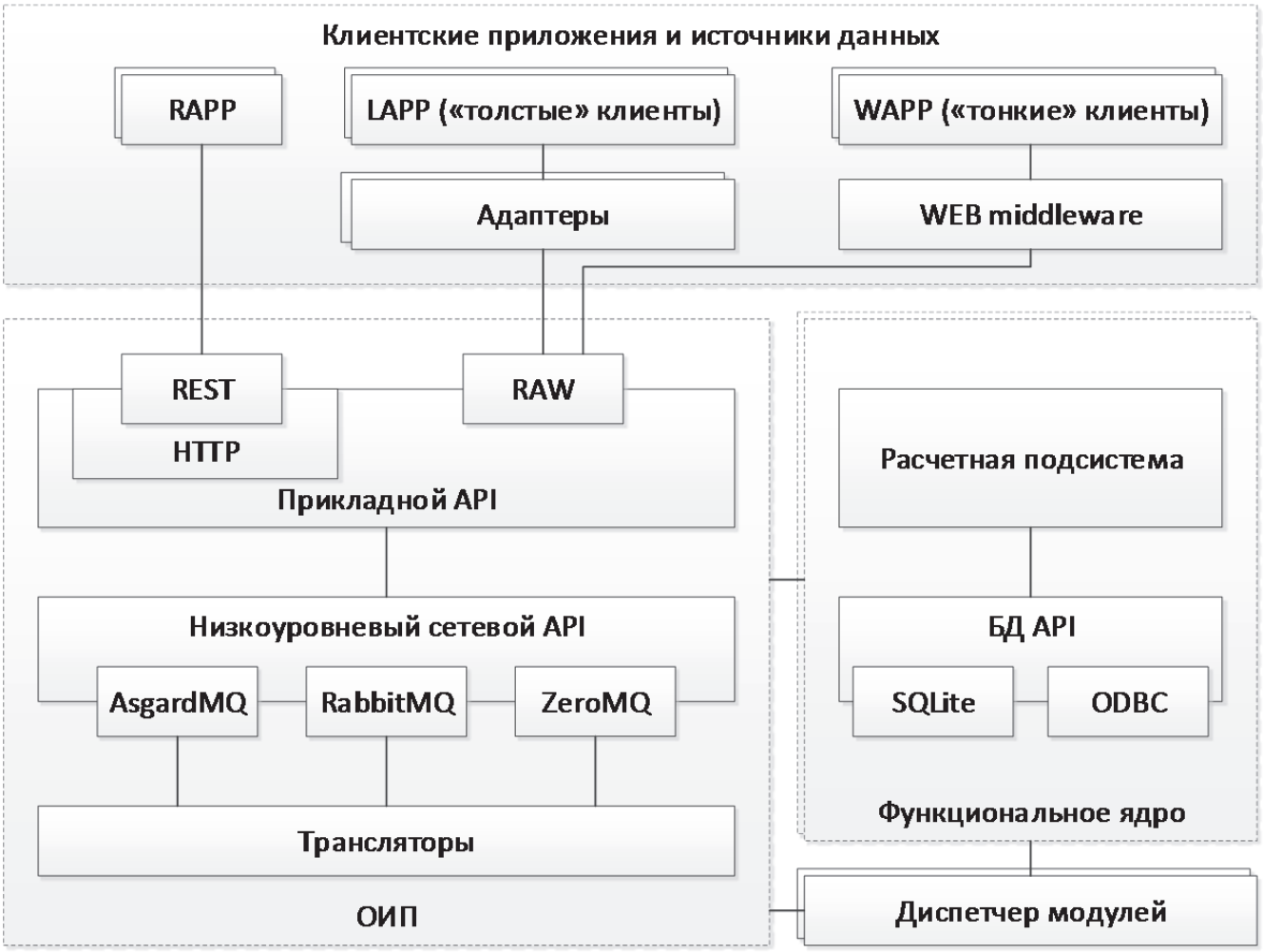

Рисунок 6 – Взаимодействие компонентов распределённой информационной системы на основе ОИП

2.4 Организация работы с помощью тонких клиентов и средств пакетной подготовки данных

Работа с информационной системой предусматривает два основных сценария онлайн-взаимодействия, а также использование модуля пакетной подготовки данных в условиях отсутствия прямой связи с информационной системой.

Приложение администратора, основные функции:

- управление пользователями;
- проверка поступающих данных, их отбраковка либо одобрение;
- проведение расчётов.

Приложение пользователя, основные функции:

- внесение новых данных, в том числе в пакетном режиме;
- проведение расчётов по оперативным данным;
- проведение расчётов/получение результатов по одобренным данным.

2.4.1 Средства пакетной подготовки данных

Модуль пакетной подготовки данных обеспечивают следующую функциональность:

- импорт данных из SCADA и других источников (текстовые файлы, Excel и т.п.);
- первичный контроль и табличное редактирование;
- преобразование данных в формат, пригодный для пакетной передачи в web-приложение;
- архивацию данных для передачи по сторонним каналам для последующего проведения расчётов.

С учётом требований к переносимости и функционированию в среде Astra Linux было принято решение о разработке данного модуля с помощью кроссплатформенной библиотеки Qt.

2.4.2 Подсистема взаимодействия с web-клиентами

Подсистема взаимодействия с web-клиентами реализована на основе серверного web-приложения, которое является связующим звеном между функциональным ядром и интерфейсом пользователя, обеспечивая преобразование клиентских данных в общий для всей системы формат.

В целях совместимости с операционной системой Astra Linux, а также для повторного использования существующих наработок [10] разработка серверной части web-приложения основана на платформе с открытым исходным кодом .Net Core.

Ядро web-приложения содержит информацию о запущенных расчётных сессиях и подключениях тонких клиентов. Под тонким клиентом в данном случае понимается одностраничное web-приложение (SPA), в любой момент времени ему может соответствовать не более одной расчётной сессии. Несколько тонких клиентов могут работать в рамках одной расчётной сессии. Связь между расчётными комплексами и тонкими клиентами – один-ко-многим.

Тонкие клиенты разрабатываются на базе Angular — платформы с открытым исходным кодом. Преимуществом такого подхода является не только кроссплатформенность, но и возможность быстрой разработки специфических тонких клиентов под нужды даже небольших групп пользователей за счёт использования написанных ранее компонентов и гибкой настройки вёрстки и стилей отображения.

Для автоматического отображения технологических схем используется преобразование исходных данных, получаемых от расчётного комплекса в xml-формате, в векторную графику (svg) посредством xslt-шаблона. В основе всех используемых форматов лежит текст, что значительно облегчает тестирование отдельных модулей.

Взаимодействие тонких клиентов с сервером происходит по защищённому протоколу https.

На стороне web-приложения входящие сообщения от тонкого клиента попадают в очередь сообщений и далее доставляются обработчику, список которых хранится в контроллере web-приложения. Если запрос может быть обработан приложением самостоятельно, то ответ формируется сразу, в противном случае он преобразуется в формат, понятный брокеру сообщений, и переадресуется функциональному ядру.

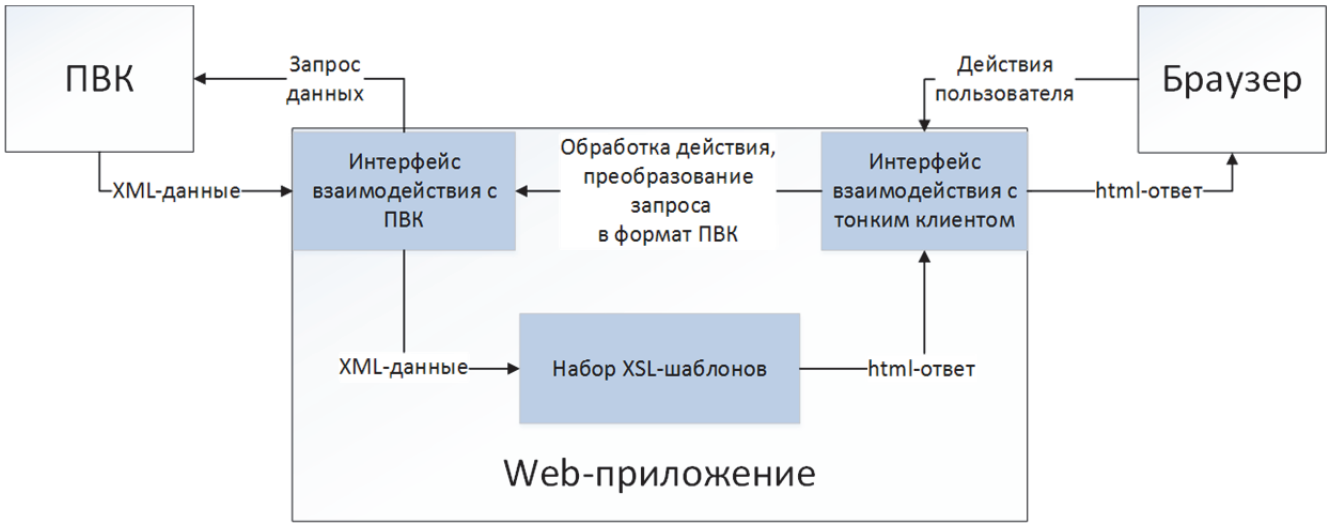

Рисунок 7 – Механизм взаимодействия компонентов web-приложения

Таким образом, в функции web-приложения входит:

- обеспечение работы тонких клиентов по протоколу https;
- контроль запущенных сессий;
- контроль доступа пользователей к задачам;
- первичная обработка пользовательских запросов;
- трансляция запроса соответствующему адресату с помощью брокера сообщений;
- преобразование полученной информации в html-формат.

2.4.3 Подсистема взаимодействия с SOA-клиентами

Подсистема взаимодействия с SOA-клиентами решает задачи, в целом совпадающие с задачами подсистемы взаимодействий с web-клиентами. Отличия включают:

- ориентированность на взаимодействие со сторонними источниками данных, а не пользователями;
- большее разнообразие обрабатываемых запросов: если web-клиенты обрабатывают фиксированное подмножество запросов, связанных с отображением схемы и передачей действий пользователей, SOA-клиенты должны обеспечить реализацию полного программного интерфейса ОИП, а также все его последующие модификации;

- отсутствие самостоятельной сложной обработки: в задачи SOA-транслятора входит лишь трансляция полученных данных из XML/JSON в формат сообщений сетевого брокера и обратно.

Поэтому вместо использования полноценного сервера приложений в сочетании с web-приложением подсистема взаимодействия с SOA-клиентами была реализована в виде базового HTTP-сервера на основе библиотеки Boost.Asio.

Транслятор допускает два варианта использования: в режиме RAW и режиме API. В первом случае в обработку http-запросов встраивается лишь трансляция запросов в формат на основе применяемого в клиентской библиотеке AsgardMQ механизма сериализации. Работа в режиме API обеспечивает передачу именованных параметров в произвольном порядке и взаимодействие с запущенными модулями на более высоком уровне абстрагирования, не требующем явного указания адресатов сообщений. Взаимодействие осуществляется в рамках сессии, идентификатор которой SOA-клиенты получают первом подключении и в дальнейшем указывают в каждом отправляемом сообщении.

Заключение

Качество решения задачи прогнозирования газопотребления зависит как от применяемых методов, так и от качества исходных данных. При отсутствии жёстких ограничений на время и вычислительные ресурсы более высокую точность прогнозов дают нейросетевые методы, хотя при удачном выборе авторегрессионной модели точность прогнозирования сопоставима с результатами прогнозирования нейросетью при меньших вычислительных затратах.

Решение задачи прогнозирования газопотребления требует достоверной информации, собираемой на разных уровнях диспетчерского управления. Предложенная архитектура, основанная на применении открытой интеграционной платформы, обеспечивает как механизм подключения к разнородным источникам данных, так и решение задачи доступа к собранной и обработанной информации в условиях гетерогенного окружения.

Список использованной литературы

[1] Сухарев М.Г., Самойлов Р.В. Анализ и управление стационарными и нестационарными режимами транспорта газа. – М.: Издательский центр РГУ нефти и газа (НИУ) имени И. М. Губкина, 2016. – 399 с.

[2] Григорьев Л.И., Степанкина О.А. Проблема оценки газопотребления в диспетчерском управлении транспортом газа; особенности и модели // Тр. конф. «Проблемы управления и автоматизации технологических процессов и производств». – Уфа: Уфимский государственный нефтяной технический университет, 2010. – С. 48-53.

[3] Степанкина О.А., Абрамов А.С. Системные основы постановки и решения задач прогнозирования для автоматизированного диспетчерского управления процессами нефтегазового производства // Автоматизация, телемеханизация и связь в нефтяной промышленности. – 2015. – № 12. – С. 19-23.

[4] Шмаль Г.И., Григорьев Л.И., Кершенбаум В.Я., Леонов Д.Г. Цифровая экономика нефтяного производства // Нефтяное хозяйство. – 2019. – № 1. – С. 100-103.

[5] Папилина Т.М., Леонов Д.Г. Преодоление архитектурных ограничений программно-вычислительных комплексов в автоматизированной системе диспетчерского управления // Neftegaz.RU. – 2016. – № 1-2. – С. 14-18.

[6] Серрано Н., Эрнантес Х., Галлардо Г. Сервисы, архитектура и унаследованные системы // Открытые системы. СУБД. – 2014. – № 8. – С. 20-22.

[7] Леонов Д.Г. Методы, модели и технологии разработки и интеграции распределенных гетерогенных программно-вычислительных комплексов в транспорте газа. – М.: Издательский центр РГУ нефти и газа (НИУ) имени И. М. Губкина, 2017. – 196 с.

[8] Леонов Д.Г. Применение сетей Петри к построению адаптируемого распределенного прикладного программного обеспечения // Автоматизация, телемеханизация и связь в нефтяной промышленности. – 2017. – № 1. – С. 5-11.

[9] Леонов Д.Г. Построение гетерогенных распределённых программно-вычислительных комплексов на основе открытой интеграционной платформы // Труды Российского государственного университета нефти и газа имени И. М. Губкина. – 2017. – № 2. – С. 125-135.

[10] Папилина Т.М. Интеграция существующих программных комплексов при построении распределенной системы поддержки принятия диспетчерских решений // Автоматизация в промышленности. – 2014. – № 5. – С. 15-18.