DOI: 10.15514/ISPRAS-2026-38(6)-40



# Предметно-ориентированное моделирование радиотехнических измерительных стендов

[1] *Н.М. Фаст, ORCID: 0009-0001-1831-1216 <fast.nm@yandex.ru>*
[2] *А.С. Кривоногов, ORCID: 0009-0003-5960-6867 <askr87@yandex.ru>*
[1] *Д.В. Кознов, ORCID: 0000-0003-2632-3193 <d.koznov@spbu.ru>*

[1] *Санкт-Петербургский государственный университет,*
*Россия, 199034, Санкт-Петербург, Университетская наб., д. 7-9.*

[2] *ООО «Концентр»,*
*Россия, 199106, г. Санкт-Петербург, пр-кт Средний В.О., д. 99/18 литера А.*

**Аннотация.** Предметно-ориентированное моделирование широко применяется на практике с середины 2000-х годов как конструктивная альтернатива модельно-ориентированным языкам и инструментам общего назначения (например, UML и средствам его поддержки). Оказывается, что на практике востребованы небольшие визуальные языки, однако глубоко встроенные в программную среду и бизнес-процессы конкретных организаций. В свою очередь, развитие технологий программирования, и, в частности, графических библиотек и библиотек работы с данными, сделало возможным быстрое создание необходимых инструментов силами целевых компаний и коллективов. В работе представлена визуальная среда для моделирования и управления стендами тестирования радиотехнического оборудования. Создан компактный предметно-ориентированный язык моделирования, включающий чёткую исполняемую семантику и механизм расширения, реализован программный инструмент. Также выполнена апробация инструмента для разработки измерительных стендов радиостанции класса DMR (Digital Mobile Radio, цифровая подвижная радиосвязь). Выполнены измерения по трудоёмкости создания измерительных стендов конкретных систем: так, для типовой системы, состоящей из 6 устройств среднего класса сложности, время создания соответствующих тестовых стендов составило 20 человеко-дней, что позволило в общей сложности сэкономить более 130 человеко-дней на тестировании этой системы.

**Ключевые слова:** предметно-ориентированное моделирование; радиотехнические системы; радиотехнические стенды; управление измерениями

**Для цитирования:** Фаст Н.М., Кривоногов А.С., Кознов Д.В. Предметно-ориентированное моделирование радиотехнических измерительных стендов. Труды ИСП РАН, 2026, том 38, вып. 6, часть 2, стр. 317–332. DOI: 10.15514/ISPRAS-2025-38(6)-40.





# Domain-Specific Modeling for Measuring Stands

[1] *N.M. Fast, ORCID: 0009-0001-1831-1216 <fast.nm@yandex.ru>*
[2] *A.S. Krivonogov, ORCID: 0009-0003-5960-6867 askr87@yandex.ru*
[1] *D.V. Koznov, ORCID: 0000-0003-2632-3193 d.koznov@spbu.ru*

[1] *Saint Petersburg State University,*
*7-9, Universitetskaya Embankment, Saint Petersburg, 199034, Russia.*

[2] *LLC «Kontsentr»,*
*Letter A, Building 99/18, Sredny Prospekt V.O., Saint Petersburg, 199106, Russia.*

**Abstract.** Domain-specific modeling (DSM) has been widely used in practice since the mid-2000s as a constructive alternative to general-purpose model-oriented languages and tools (such as UML and its supporting tools). It turns out that small visual languages are in demand in practice and should be deeply embedded in the software environment and business processes of specific organizations. On the other hand, the development of programming technologies, particularly graphical libraries and data libraries, has made it possible to quickly develop the necessary tools by target companies and teams. The paper presents a visual environment for modeling and managing radio equipment testing stands. A compact domain-specific modeling language has been designed, including a clear executable semantics and an extension mechanism, and a software tool has been implemented. The tool has also been tested for developing DMR-class radio station measurement stands.

**Keywords:** domain-specific modeling; radio engineering systems; radio engineering stands; measurement control.

**For citation:** Fast N.M., Krivonogov A.S., Koznov D.V. Domain-Specific Modeling for Measuring Stands. Trudy ISP RAN/Proc. ISP RAS, 2026, vol. 38, issue 6, part 2, pp. 317-332 (in Russian). DOI: 10.15514/ISPRAS-2026-38(6)-40.

## *1. Введение*

В середине 2000-х годов стало понятно, что универсальные языки моделирования ПО, в частности, UML, не способны закрыть все потребности индустрии. Слишком сильно отличались предметные области и производственная культура программных проектов. Одновременно средства разработки графических инструментов, например, Eclipse Modeling Project [1], сделали большой рывок, и стало возможно создавать целевые инструменты в этой области не только силами специализированных организаций и команд, но и в рамках обычных компаний по разработке ПО, специализирующихся в телекоммуникациях, розничной торговле и пр. [2]. Таким образом, появилось предметно-ориентированное моделирование (Domain-Specific Modeling, DSM) – подход, направленный на решение задач по обработке больших объёмов информации в рамках создания и сопровождения ПО, а также в других смежных областях, с помощью семиотически точных визуальных спецификаций (моделей), основанных на чертёжном проектировании, поддержанных соответствующим программным инструментарием [2-3]. Таким образом, в рамках DSM-подхода для отдельной предметной области создаются новый язык моделирования и соответствующий программный инструмент моделирования.

Радиотехнические измерительные стенды являются неотъемлемой частью процесса разработки, массового производства и эксплуатации радиотехнических программно-аппаратных систем, используясь для проверки систем на соответствие требованиям заказчика, для выявления критичных ошибок и т.д. Создание таких стендов и сам процесс тестирования являются трудоёмкой деятельностью, во многом определяющей время выхода новой системы в промышленное производство.

Имеющиеся в этой области инструменты не могут удовлетворить нужды российских разработчиков радиотехнических программно-аппаратных систем. Система LabVIEW [4] –

графическая среда, ориентированная на управление измерительным оборудованием, сбор, обработку и анализ программно-аппаратных данных, громоздка и тяжеловесна, требует больших вычислительных ресурсов, направлена на оборудование производителей, не использующееся на российском рынке, и, наконец, не имеет сертификации на территории РФ. Отечественные инструменты, например, METLAB и ЭМИКА [5-6] – также применяются в данной области, однако METLAB ориентирован на автоматизацию метрологических и поверочных процедур, но слабо поддерживает гибкое построение алгоритмов управления измерительным оборудованием, а ЭМИКА, предназначаясь для автоматизации измерений, зависит от протокола SCPI [7] и не позволяет работать с устройствами собственного производства и/или использующими проприетарные протоколы управления. Имеется также ряд решений в смежных областях, которые можно использовать для решения данной задачи: MATLAB/Simulink [8], Keysight VEE [9], R&S CMWrun и R&S InstrumentView [10-11] для удалённого управления приборами, однако эти решения ориентированы на использование оборудования конкретного производителя, имеют ограниченные возможности адаптации и интеграции в разнородные аппаратно-программные комплексы и требуют значительных усилий для настройки под специфические задачи радиотехнических измерительных стендов.

Для преодоления вышеуказанных проблем в ООО «Концентр» был инициирован научно-производственный проект по созданию DSM-решения, нацеленного на визуальное моделирование и управление радиотехническими измерительными стендами для выпускаемых компанией радиотехнических систем. Решение было призвано понизить затраты на тестирование и доводку радиотехнических устройств перед их выпуском в производство.

В работе предложен язык визуального моделирования, основанный на метафоре «чёрного ящика с портами» (общепринятая метафора в области систем реального времени): модули с портами и соединениями для абстракции измеряемого оборудования, а также реализации алгоритмов по обработке полученных измерений. Исполняемая семантика языка задана в виде дискретного (потактового) выполнения модели. Язык также снабжён средствами расширения, позволяя в текстовом формате описывать новые модули, соответствующие дополнительным элементам оборудования и алгоритмам обработки данных, а также задавать элементы пользовательского интерфейса для работы с параметрами этих модулей.

Следует особо подчеркнуть, что несмотря на долгое развитие DSM-подхода [2], успешный опыт его практического применения актуален и обладает научной новизной из-за отсутствия пока ещё практики массового применения подхода в индустрии, а также в виду значительных отличий целевых предметных областей. В частности, отсутствуют работы по применению DSM в сфере создания инструментария визуализации для управления промышленными стендами. При этом важно отметить, что, несмотря на наличие таких многофункциональных сред как LabVIEW [4], в которых, на первый взгляд, «есть всё необходимое» для решения данной задачи (необходимые абстракции, механизмы визуализации пр.), предметно-ориентированный подход позволяет реализовать более эффективное целевое решение, включая как возможности, имеющиеся в общих системах, так и разные детали, на первый взгляд несущественные, но становящиеся всё более значимыми по мере развития и распространения новой технологии – поддержка проприетарных протоколов и нестандартного оборудования, отсутствие лишней функциональности, возможность создания целевого программно-аппаратного комплекса, тиражируемого для производства в РФ и так далее. Зачастую «вырезка» нужной функциональности из большого универсального продукта, а также доводка его под конкретные нужды и дальнейшая эволюция оказываются затруднёнными или вообще невозможными. Причём затраты и проблемы возрастают по мере осуществления проекта, в то время как в начале имеется возможность быстро получить убедительный результат, что часто и сбивает с толку. Таким образом, если стоит цель получить промышленный инструмент, а не инновационный прототип, то разработку целевых визуальных решений целесообразно осуществлять на базе стандартных библиотек, а не в





рамках большой многофункциональной среды. Данное исследование является подтверждением этого тезиса, всё ещё неочевидного для многих как в индустрии, так и в университетско-академическом сообществе. По большому счёту, именно в этом заключается суть предметно-ориентированного подхода к визуальному моделированию, т.к. всегда есть гипотетическая возможность использовать готовые универсальные языки моделирования, а также соответствующие средства их программной поддержки [2].

Статья организована следующим образом. В разделе 2 описаны существующие инструменты и подходы к измерению аппаратуры. В разделе 3 дано описание предложенного языка моделирования. В разделе 4 описан созданный программный инструмент. Апробация предложенного решения описана в разделе 5. В разделе 6 представлено заключение.

## *2. Сравнение с аналогами*

Существует значительное количество языков моделирования, предназначенных для спецификации оборудования и поддерживающих метафору чёрного ящика с портами. Можно упомянуть язык SDL [12], ориентированный на проектирование сложных телекоммуникационных систем и имеющий хорошо разработанные средства структурной декомпозиции (системы, блоки, процессы, сервисы, каналы и их типы, интерфейсы, сигналы и т.д.). Однако в нашем случае не требуется поддержка многоуровневых спецификаций в духе сетевого стандарта ISO/OSI, следовательно, наш язык может быть многократно проще. Более компактный язык моделирования систем реального времени представлен в рамках методологии ROOM [13]. В этой методологии порты и интерфейсы более явно выделены, чем в SDL, и мы использовали за основу именно этот язык моделирования. Диаграммы компонент в UML [14] использовались нами в том смысле, что наши модули, имея порты, имеют в то же время и параметры (атрибуты), которые используются исключительно внутри модуля и не видны снаружи – все коммуникации моделей осуществляются через порты. Ряд черт, в частности разъёмы и регионы, мы взяли из стандарта SysML [15], предназначенного для инженерного проектирования.

Исполняемая семантика нашего языка наследована из MATLAB Simulink [8], но в упрощённом виде. В Simulink существенную роль играет согласованность временных характеристик сигналов на портах [16]. При соединении блоков требуется согласование их временных параметров, в противном случае среда либо автоматически вводит преобразования скоростей (rate transition) [17], либо сигнализирует об ошибке. В предлагаемом языке такая сложность отсутствует – порты не имеют собственной тактовой частоты, и вопросы синхронизации сигналов не рассматриваются.

В целом, наш язык, подобно другим DSM-языкам, не несёт каких-то ярких индивидуальных черт, являясь вырезкой и комбинацией возможностей существующих языков. Однако создание подходящей конфигурации языка моделирования, отражающей реалии выбранной предметной области, является сложной инженерной деятельностью [1, 2] и представляет научно-практический интерес.

Лидером в области средств автоматизации измерений является компания National Instruments, а её программная среда LabVIEW [4] – один из наиболее распространённых в мире инструментов для построения автоматизированных измерительных систем. Однако среда LabVIEW предъявляет значительные требования к вычислительным ресурсам, особенно при работе с крупными проектами. Кроме того, программирование в LabVIEW осуществляется на уровне относительно небольших (гранулярных) функциональных блоков, из которых разработчику приходится вручную формировать сложные алгоритмы измерений. При создании собственных измерительных стендов это приводит к необходимости конфигурирования большого количества низкоуровневых элементов, что делает разработку громоздкой и трудоёмкой. Особенно сложной задачей является создание драйверов для нового оборудования: несмотря на наличие стандартных механизмов взаимодействия с

приборами (например, через интерфейсы VISA [18] и GPIB [19]), разработка собственных драйверов требует глубокого понимания протоколов обмена и особенностей платформы. Кроме того, LabVIEW является проприетарным программным обеспечением с высокой стоимостью лицензий и ограниченной совместимостью между версиями, что может осложнять сопровождение и долговременную поддержку измерительных проектов.

В РФ имеются такие системы, как METLAB и ЭМИКА [5, 6]. METLAB предназначен главным образом для автоматизации метрологических и поверочных процедур, позволяя проводить измерения и поддерживая полный цикл калибровочных работ. Однако METLAB ориентирован на использование в лаборатории (выдачу свидетельств о поверке) и не предлагает гибкого визуального конструирования алгоритмов управления промышленным оборудованием. ЭМИКА представляет собой кроссплатформенную low-code-среду программирования для автоматизации измерений и испытаний. ЭМИКА позволяет объединять различные измерительные приборы в единый сценарий измерений, упрощая создание тестовых программ и протоколов. При этом создание драйверов оборудования в системе в значительной степени ориентировано на использование стандарта SCPI (Standard Commands for Programmable Instruments, стандартизированный набор текстовых команд для дистанционного управления измерительным оборудованием через соответствующие каналы связи) [7], поэтому при работе с приборами, не поддерживающими данный протокол или использующими нестандартные интерфейсы управления, разработка драйверов в среде ЭМИКА оказывается затруднительной. В совокупности оба отечественных решения – METLAB и ЭМИКА – пока пока не достигли уровня универсальности и зрелости, характерного для коммерческих систем типа LabVIEW.

MATLAB Simulink [8] представляет собой среду моделирования, широко применяемую для разработки и анализа систем обработки сигналов, в том числе радиотехнических систем. При использовании дополнительных инструментов и интерфейсов (например, Instrument Control Toolbox [20]) MATLAB Simulink может применяться и для автоматизации измерений, включая управление оборудованием и обработку экспериментальных данных. Тем не менее, такие возможности носят вспомогательный характер и требуют значительной настройки, поскольку изначально система не ориентирована на построение полноценных измерительных стендов.

Keysight VEE [9] – это графическая среда разработки, предназначенная для создания приложений для автоматизации измерений и тестирования оборудования, преимущественно, компании Keysight. Эта среда позволяет создавать алгоритмы управления приборами в виде визуальных блок-схем и обладает встроенной поддержкой взаимодействия с измерительным оборудованием. Система позволяет ускорить разработку тестовых приложений, однако её применение, во многом, ограничено экосистемой поддерживаемых устройств, а возможности расширения и интеграции со сложными и разнородными аппаратными комплексами невелики.

Rohde & Schwarz CMWrun [10] представляет собой специализированную программную платформу для автоматизации тестирования беспроводных устройств, в первую очередь оборудования компании Rohde & Schwarz. Эта система обеспечивает выполнение сценариев испытаний, управление измерительными приборами и сбор результатов, что делает её эффективным инструментом для задач сертификационного и производственного тестирования. В то же время система ориентирована на конкретные классы задач и оборудование, что снижает её универсальность при использовании в иных областях.

Rohde & Schwarz InstrumentView [11] – это программный инструмент для удалённого доступа и управления измерительными приборами, обеспечивающий визуализацию их интерфейсов и параметров в реальном времени. Инструмент упрощает работу с тестируемым оборудованием, позволяя централизованно контролировать устройства, однако не предназначен для построения сложных измерений и автоматизированных испытательных





сценариев, выступая, скорее, вспомогательным средством в составе более крупных систем автоматизации.

Таким образом, очевидна необходимость разработки отечественного гибкого предметно-ориентированного инструмента для моделирования и управления радиоизмерительными стендами, основанного на визуальном моделировании.

## *3. Описание визуального языка*

Описание предложенного предметно-ориентированного решения следует методологии предметно-ориентированного моделирования, описанной в [2]. Соответственно, были выделены следующие измерения решения: визуальный язык, включая синтаксис, исполняемую семантику и средства расширения; инструмент моделирования; экспериментальная оценка трудоёмкости разработки и использования инструмента.

Предлагаемый язык моделирования предназначен для спецификации структуры радиотехнического стенда, создаваемого для измерений и тестирования конкретного радиотехнического устройства, включая выбор и настройку необходимых программных компонентов, обеспечивающих автоматизацию измерений. Важно отметить, что язык используется и как средство спецификации стенда, и как средство управления процессом измерения при тестировании целевого устройства.

### 3.1 Описание основных конструкций

Рассмотрим основные элементы визуального языка, используемого для проектирования моделей радиотехнических стендов – модуль, порт и соединение.

*Модуль* является основным элементом языка и создаваемой с его помощью визуальной модели. Он может соответствовать как физическому устройству (радиостанции, анализатору спектра, блоку питания или сетевому коммутатору и т.д.), так и программному компоненту. В случае физического устройства модуль выполняет роль программной оболочки, обеспечивающей управление работой оборудования со стороны стенда. В случае программных компонентов модуль инкапсулирует алгоритмы обработки данных, вычисления статистических характеристик или протоколирования результатов измерений. Объединение аппаратных и программных модулей в рамках одной мета-сущности позволяет унифицировано создавать тестовые стенды, объединяя задачи управления тестируемым оборудованием и программную обработку получаемых от оборудования сигналов.

Модуль обладает параметрами (включая элементы графического интерфейса для отображения параметров), портами, поведением, а также программной реализацией. Следует отметить, что у того или иного прибора, представленного модулем, могут быть сотни параметров, определяющих режим работы прибора и требующих правильной настройки для обеспечения корректного выполнения прибором целевого измерения. Не все они важны при построении измерительных стендов, однако и в этом случае количество параметров может исчисляться десятками. Для удобства их группировки используется специальная конструкция под названием *регион*. Спецификация модуля выполняется в текстовом режиме (см. раздел 3.3).

На диаграмме модуль изображается в виде прямоугольника, над которым располагается надпись с его названием – пример представлен на рис. 1.

*Порт* предназначен для организации взаимодействия между модулями. Порты бывают двух типов: физические и виртуальные. *Физические порты* используются для описания схемы коммутации реального оборудования и соответствуют физическим интерфейсам устройства, которое представляет модуль. Физический порт обозначается в виде закрашенного квадрата или треугольника. Квадратом обозначается двунаправленное соединение, например, Ethernet-соединение, через которое устройство получает команды и передаёт ответы.

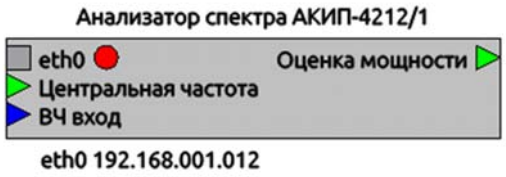



*Рис. 1. Пример модуля.*
*Fig. 1. Example of a module.*

Треугольником обозначается однонаправленное соединение, например, ВЧ-соединение между выходом генератора сигналов и входом анализатора спектра. Цвет обозначения указывает на тип интерфейса: например, Ethernet-порты отображаются серым цветом, ВЧ-порты, соответствующие высокочастотным разъёмам радиотехнической аппаратуры, – синим, а COM-порты – фиолетовым (см. рис. 1).

*Виртуальные порты* предназначены для организации информационного обмена между модулями, например, с целью автоматизированного изменения параметров модуля в соответствии с заданным алгоритмом. Данные порты отображаются в виде треугольников зелёного цвета. Направление треугольника определяет тип порта: если его вершина направлена наружу от модуля, порт является выходным, а если внутрь – то входным. Так, на рис. 1 порт «Центральная частота» является входным, а порт «Оценка мощности» – выходным.

*Соединение* представляет собой элемент визуального языка, предназначенный для установления связи между портами различных модулей. Соединение отображается в виде линии, соединяющей два порта. Цвет линии соответствует типу соединённых портов, что позволяет визуально различать типы связей между элементами модели. Соединять разрешается только порты одинакового типа. Если пользователь пытается соединить порты разных типов, редактор моделей блокирует такую операцию, предотвращая создание некорректной конфигурации стенда.

Для соединений между физическими портами дополнительно указывается тип используемых *разъёмов*. Эта информация отображается в виде подписи над линией соединения и позволяет явно указать типы интерфейсных разъёмов на обоих концах кабеля. Например, может быть указано соединение вида N to N, что означает использование кабеля с разъёмами N-типа на обоих концах, или N to BNC, что соответствует соединению между ВЧ-разъёмом N-типа и разъёмом BNC, который часто используется, например, в осциллографах. Указание типов разъёмов упрощает монтаж измерительного стенда по визуальной модели. Пример физического соединения представлен на рис. 2.

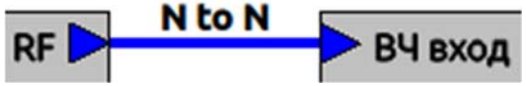



*Рис. 2. Пример соединения.*
*Fig. 2. Example of a connector.*

На рис. 3 представлен пример визуальной модели радиотехнического стенда, предназначенного для измерения характеристик фильтра нижних частот. Для выполнения измерений используются модули «Генератор сигналов» и «Анализатор спектра», осуществляющие управление соответствующими измерительными приборами.

Для подключения приборов к управляющему компьютеру по интерфейсу Ethernet используется модуль «Компьютер». Отображение значений мощности сигнала на выходе ФНЧ, измеренных анализатором спектра, выполняется с помощью модуля «Дисплей».





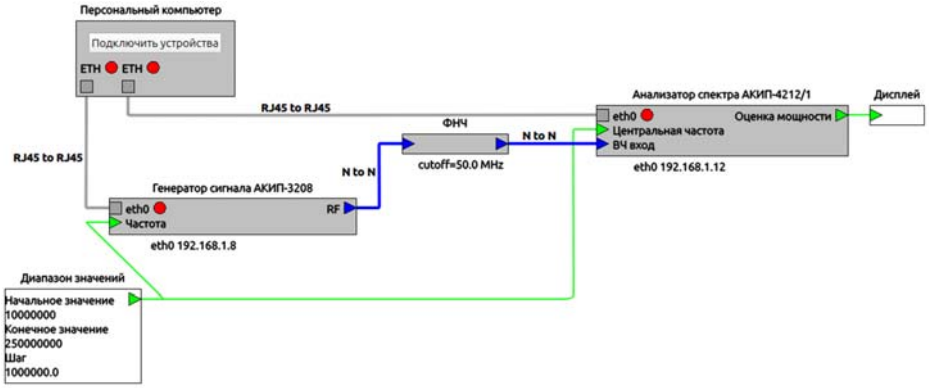



*Рис. 3. Визуальная модель стенда для измерения характеристики фильтра нижних частот.*
*Fig. 3. Visual model of a test bench for measuring the frequency of a low-pass filter.*

Модуль «Диапазон значений» осуществляет перебор частот в интервале от 10 до 250 МГц с шагом 1 МГц. Данный модуль используется для управления частотой формируемого генератором сигнала, а также центральной частотой анализатора спектра, то есть частотой, на которой производится измерение мощности сигнала.

## 3.2 Исполняемая семантика языка

Исполняемая семантика языка определяет порядок функционирования и правила взаимодействия модулей при тестировании. В предлагаемом языке используется дискретная (потактовая) модель исполнения, в рамках которой исполнение модели представляет собой последовательность тактов. В течение одного такта для каждого модуля последовательно выполняется этап **FirstStep**, причём переход к следующему модулю осуществляется после завершения данного этапа для предыдущего; после выполнения этого этапа для всех модулей аналогичным образом выполняется этап **SecondStep**. По завершении указанных этапов система выполняет ожидание в течение заданного пользователем интервала времени, необходимого для реакции управляемого оборудования на программные воздействия. Таким образом, один такт соответствует полному проходу по всем модулям модели. На каждом такте для всех модулей последовательно выполняются следующие шаги.

1) **Вычисление выходных значений** – осуществляется расчёт значений выходных портов модуля и передача этих значений на входные порты связанных с ним модулей.
2) **Обработка входных данных** – производится обработка значений, поступивших на входные порты модуля, а также обновление его внутренних параметров.
3) **Синхронизационная задержка** – выполняется ожидание в течение заданного пользователем интервала времени, необходимого для того, чтобы физические устройства успели отреагировать на программное изменение своих параметров.

Процесс тестирования системы выполняется в течение заданного пользователем количества тактов. Однако при необходимости тестирование может быть приостановлено пользователем (поставлено на паузу) или завершено досрочно. В случае возникновения нештатной ситуации, не поддающейся обработке, исполнение модели принудительно завершается.

Спецификация модуля выполняется в виде текстового описания, которое задаёт порты, параметры, графический интерфейс и, наконец, поведение модуля на каждом из этапов его работы. Подробнее текстовое описание рассматривается в следующем разделе, здесь остановимся на той его части, которая описывает поведение модуля. В поведенческой части

задаются действия, которые выполняются модулем на разных этапах его жизненного цикла. Выделяются следующие этапы жизненного цикла модуля (иначе говоря, Step-функции), описанные ниже.

- **PostCreationStep** – действия, выполняемые непосредственно после создания экземпляра модуля.
- **PreModellingStep** и **PostModellingStep** – действия, выполняемые соответственно до начала и после завершения сеанса тестирования (как правило, используются для инициализации и завершения работы, включая перевод оборудования в безопасное состояние).
- **FirstStep** – действия для формирования выходных значений модуля (например, отправка SCPI-команд-запросов измерительному прибору).
- **SecondStep** – действия по обработке входных данных и изменению параметров (например, отправка SCPI-команд управления).

Действия бывают следующих видов: SCPI-команда и Python-функция.

Пример фрагмента текстового описания модуля для параметра «Частота» приведён в листинге 1. В строках 1–2 определяется модульная единица с именем *Frequency* и типом *параметр*. В системе предусмотрена встроенная поддержка SCPI-команд, для задания которых используется специализированная структура (см. строки 3-7 листинга 1). В этих строках описывается этап **SecondStep**, отвечающий за обработку входных воздействий. Здесь описано управление параметром посредством SCPI-команд: поле *Type* со значением *Function* указывает на исполняемое действие, а поле *FunctionType* со значением *SCPIWrite* – на отправку команды устройству. Поле *Type* (строка 4) определяет тип описываемого объекта и используется интерпретатором текстового описания; для описания действий применяется значение *Function*.

```
1   "Frequency": {
2     "Type": "ParameterUnit",
3     "SecondStep": {
4       "Type": "Function",
5       "FunctionType": "SCPIWrite",
6       "CMD": ["FREQ %d\n", "FREQ %d %s\n"]
7     },
8     "Widget": {
9       "Type": "Float",
10      "Name": "Frequency",
11      "Label": "Частота",
12      "Description": "Частота формируемого сигнала",
13      "Units": {
14        "Hz": 1, "kHz": 1000, "MHz": 1000000, "GHz": 1000000000
15      },
16      "MinValue": 9000.0,
17      "MaxValue": 2100000000.0
18    },
19    "GraphicsWidget": {
20      "Type": "VirtualPort",
21      "LifeTime": "Variable",
22      "Direction": "Input"
23    }
24  }
```

*Листинг 1. Текстовое описание модульной единицы.*
*Listing 1. Text description of a modular unit.*





Далее указывается поле FunctionType (строка 5), принимающее значения SCPIWrite, SCPIRead или UserFunction. Значения SCPIWrite и SCPIRead используются соответственно для отправки команд и выполнения запросов, предполагающих получение ответа от устройства. Сама команда задаётся в поле *CMD*, допускающем указание нескольких вариантов одной команды с различными наборами аргументов. В строке 6 приведены два варианта команды: первый предполагает задание только численного значения частоты с использованием единицы измерения по умолчанию, второй – явное указание единицы измерения. Например, один вариант команды может предусматривать задание только численного значения параметра без явного указания единицы измерения (с использованием значения по умолчанию), тогда как другой вариант требует явного указания единицы измерения. Для реализации действий, не связанных с использованием SCPI-интерфейса (например, при работе с проприетарными протоколами или при выполнении некоторых вычислений), предусмотрен механизм пользовательских Python-функций. Обязательным требованием является наличие параметра *module*, который должен быть первым аргументом функции и обеспечивает доступ к объекту модуля. Для использования такой функции в текстовом описании в поле *FunctionType* указывается значение *UserFunction*, а сама функция задаётся в поле *Function*, заменяющем поле *CMD*.

## 3.3 Механизм расширения

Стандартный набор модулей может быть расширен. Это достигается с помощью механизма текстового описания модуля. Новый модуль, описанный пользователем, появляется в палитре графического редактора и становится доступен для использования при разработке нового стенда.

Текстовое описание представляет собой иерархическую структуру данных, используемую для определения состава, поведения и интерфейса модуля. Описание задаёт модуль как композицию вложенных модульных единиц, каждая из которых отвечает за отдельный аспект его функциональности.

Модуль является модульной единицей верхнего уровня и определяется набором атрибутов, включающих имя (Name), путь к программной реализации (Package) и тип (Type). Тип модуля используется для разделения модулей, управляющих физическим оборудованием, и чисто программных модулей. Кроме того, модуль может содержать определения портов, этапов выполнения (Step-функций из раздела 3.2), а также список вложенных модульных единиц (Units), формирующих его структуру и поведение.

В рамках текстового описания выделяются следующие основные типы модульных единиц.

- **Параметр (ParameterUnit)** – используется для задания параметров модуля. Параметры могут быть как управляемыми (например, частота генератора), так и вычисляемыми (например, положение и амплитуда пика спектра). Параметр может иметь графический интерфейс для ввода значений, а также виртуальный порт, позволяющий программно задавать или считывать его значение.
- **Порт (PhysicalPort, VirtualPort)** – предназначен для организации взаимодействия модуля с другими модулями и физическим оборудованием. Физические порты используются для визуализации и документирования разъёмов оборудования, а также для отображения схемы коммутации в модели стенда. Виртуальные порты применяются для передачи данных между модулями в процессе автоматизированного управления измерительным стендом.
- **Группа (GroupUnit)** – служит для логической и визуальной группировки модульных единиц. С её помощью можно структурировать графический интерфейс (например, объединять связанные параметры в отдельные разделы), а также реализовывать поведение, требующее координированного взаимодействия нескольких элементов.

- **Транспорт (TransportUnit) – предназначен** для описания интерфейсов взаимодействия с физическим оборудованием через управляющий компьютер. Используется для задания типа интерфейса и его конфигурации.
- **Step-функции, описанные в предыдущем разделе, могут быть определены** как на уровне модуля, так и на уровне отдельных модульных единиц.

На листинге 1 строки 8-18 описывают графический интерфейс параметра. На основе данных, заданных в секции *Widget*, формируется элемент ввода типа «числовое поле со счётчиком» (spinbox) с ограниченным диапазоном значений (*MinValue* и *MaxValue*). Поле *Units* задаёт набор допустимых единиц измерения и соответствующие коэффициенты пересчёта, что позволяет дополнительно сформировать выпадающий список (combobox) для выбора единицы измерения. Поля *Label* и *Description* определяют соответственно подпись параметра в интерфейсе и текст всплывающей подсказки.

В строках 19-22 листинга 1 задаётся альтернативный способ задания параметра – через входной виртуальный порт. Секция *GraphicsWidget* с типом *VirtualPort* и направлением *Input* определяет порт, через который значение параметра может передаваться программно. Параметр *LifeTime* со значением *Variable* указывает на возможность динамического отображения порта в интерфейсе. В этом случае пользователю предоставляется возможность переключения между ручным вводом значения через графический интерфейс и программным управлением через порт: при активации соответствующего элемента управления (см. рис. 4) ручной ввод блокируется и отображается виртуальный порт, в противном случае используется только графический интерфейс без отображения порта.

## 4. Инструмент моделирования

### 4.1 Описание функциональности

Разработанный инструмент моделирования поддерживает следующие функции:

- визуальную спецификацию измерительных стендов на основе модульного подхода, включая пользовательские программные модули;
- настройку параметров аппаратуры, программных модулей и тестируемых устройств;
- динамическое управление параметрами и выполнение измерений с возможностью подключения средств анализа данных;
- протоколирование, визуализацию и сохранение результатов измерений.

Создание стенда заключается в формировании модели из модулей, соединённых через порты, с последующей настройкой параметров модулей. Пользователь может задавать параметры аппаратуры и программных компонентов как вручную через графический интерфейс, так и программно, через виртуальные порты. Пример представлен на рис. 4 – управление значениями параметров «Частота» и «Смещение по уровню» выполняется с помощью одноимённых виртуальных портов. Виртуальные порты обеспечивают динамическое изменение параметров в процессе выполнения модели, что особенно важно при реализации сложных измерительных сценариев.

Созданная модель является не только статическим описанием стенда, но и средством управления процессом измерений, а также инструментом обработки и визуализации. Пользователь может подключать готовые или самостоятельно разработанные модули для анализа, обработки и отображения данных. Эти модули позволяют строить графики и другие наглядные представления результатов прямо в процессе измерений. Графические окна открываются и обновляются параллельно выполнению измерений (см. рис. 5), благодаря чему инженер может в реальном времени отслеживать изменение параметров и оперативно контролировать ход эксперимента или производственного теста.





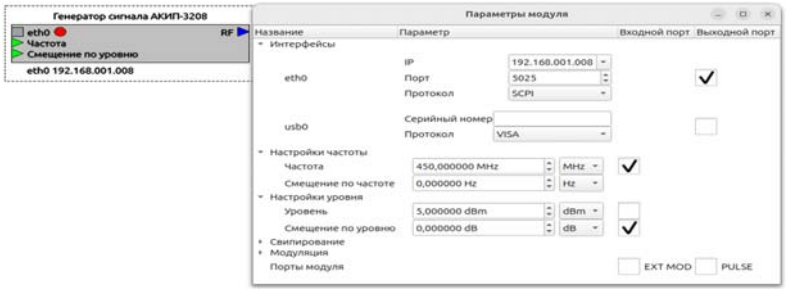



*Рис. 4. Модуль «Генератор сигнала АКИП-3208» и окно настроек его параметров.*
*Fig. 4. Module AKIP-3208 Signal Generator and the window for configuring its parameters.*

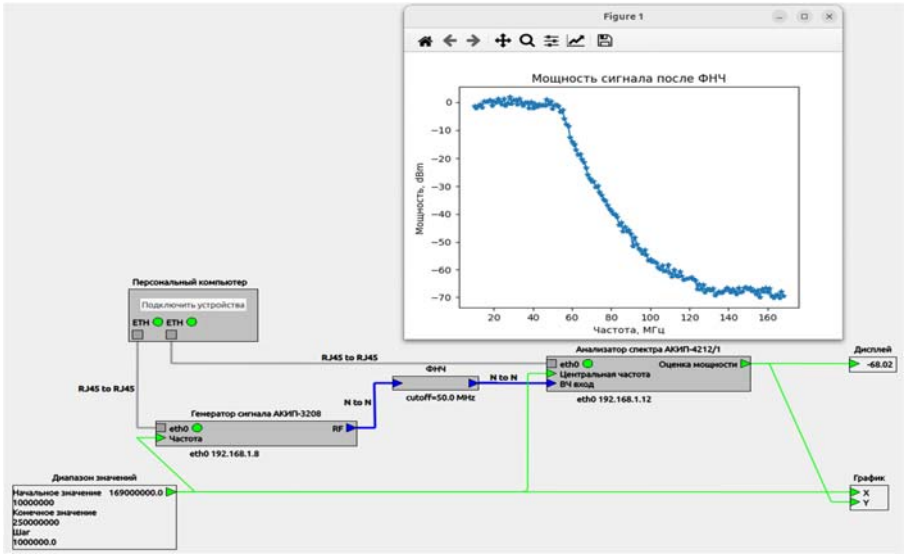



*Рис. 5. Визуализация результатов измерений.*
*Fig. 5. Visualization of measurement results.*

Результаты измерений извлекаются через виртуальные порты модулей, после чего могут сохраняться в файл и/или передаваться в подсистему протоколирования. После завершения эксперимента/теста формируется отчёт, включающий значения параметров, результаты измерений, а также метаинформацию (время проведения, используемую модель стенда и комментарии пользователя). Отчёт может быть автоматически преобразован в HTML-документ для последующего анализа и передачи.

Для обеспечения переносимости моделей реализована возможность сохранения и загрузки стендов в формате JSON, что позволяет использовать заранее подготовленные конфигурации, в том числе при передаче решений с этапа разработки на производство.

### 4.2 Разработка и трудозатраты

Программная реализация системы выполнена на языке Python. Графический интерфейс реализован с использованием библиотеки PySide2 [21], представляющей собой

перенесённую на Python версию популярной технологии Qt, которая позволяет создавать кроссплатформенные пользовательские интерфейсы. Для построения визуальных моделей используется библиотека Qt Graphics View Framework, в рамках которой сцена представляется объектом QGraphicsScene, а элементы модели – графическими объектами. Отображение модулей реализовано на основе класса QGraphicsWidget, обеспечивающего возможность компоновки и обработки событий. Для организации внутренней структуры модулей применяется QGraphicsLayout, текстовые элементы отображаются с помощью QGraphicsSimpleTextItem, а графические примитивы (например, обозначения портов) – с использованием QGraphicsPolygonItem. Соединения между модулями реализованы посредством расширения класса QAbstractGraphicsShapeItem, что позволяет задавать их геометрию, внешний вид и поведение при взаимодействии с пользователем. Генерация графического интерфейса модулей основана на активном использовании элементов Qt Widgets (QLabel, QSpinBox, QCheckBox, QComboBox и некоторые другие). Взаимодействие с измерительным оборудованием реализовано на основе стандартных сетевых и последовательных интерфейсов с использованием библиотек Socket [22], pySerial [23] и PyUSB [24]. Управление приборами осуществляется посредством SCPI-команд [7], однако предусмотрена возможность реализации других протоколов.

Трудоёмкость разработки решения составила порядка одного человеко-года, что было достигнуто за счёт продуманной архитектуры решения, а также использования библиотеки PySide2/Qt.

## 5. Апробация

Апробация разработанного решения проводилась посредством его внедрения в производственном подразделении организации. Целью внедрения являлась автоматизация поверки набора параметров производимой организацией радиостанции класса DMR (Digital Mobile Radio, цифровая подвижная радиосвязь).

### 5.1 Описание тестируемых систем

Для этого был реализован набор измерительных стендов. Данные стенды были нацелены на измерение таких параметров радиостанции, как выходная мощность передатчика, отклонение несущей частоты передатчика и некоторых других.

В рамках апробации были реализованы модули для автоматизированного управления стендами для следующих измерительных приборов (стандартный SCPI-интерфейс): генераторов высокочастотных/низкочастотных сигналов, анализатора спектра, осциллографа. Также был разработан модуль для управления тестируемой радиостанцией по проприетарному протоколу с контролем следующих параметров: переключение между аналоговым и цифровым режимами, включение/отключение передатчика, задание несущей частоты и выбор мощности передатчика.

### 5.2 Трудозатраты

Сопровождение и расширение системы имеют приемлемую трудоёмкость. Добавление новых модулей для поддержки измерительного оборудования с SCPI-интерфейсом, например анализатора спектра в комплектации с дополнительными опциями, занимает порядка трёх человеко-дней (речь идёт о подготовленном специалисте). В случае более простых устройств (например, ВЧ-генераторов) или при реализации только базового набора команд, трудоёмкость снижается до одного человеко-дня. Трудоёмкость разработки программных модулей также оказывается приемлемой, поскольку существует большое количество готовых алгоритмов обработки данных, поэтому требуется их адаптация к решаемой задаче и интеграция со средствами визуализации (около одного человеко-дня для подготовленного специалиста).





Трудоёмкость использования всего решения может быть оценена следующим образом. В рамках апробации для создания решения, включавшего 7 измерительных стендов, управлявшего 6 единицами оборудования (тестируемой радиостанцией, двумя генераторами, анализатором спектра, осциллографом и источником питания) и проверявшего 7 параметров целевой радиостанции были потрачены следующие ресурсы: на разработку новых модулей – 15 человеко-дней; на проектирование и отладку всех стендов – 5 человеко-дней. Таким образом, общая трудоёмкость внедрения составила порядка 20 человеко-дней. Ручная настройка стенда существенно менее затратна, однако трудоёмкость эксплуатации очень велика. По нашим оценкам, удалось сэкономить не менее 132 человеко-дней на тестировании данной системы.

## 6. Заключение

Приемлемые затраты на разработку и эксплуатацию созданного решения подтверждают эффективность применения DSM-подхода для решения поставленной задачи. В частности, использование диаграмм, по отзывам производственных инженеров, существенно упрощает проведение тестовых экспериментов. Однако мы понимаем, что требуются дополнительные эксперименты как в части анализа трудоёмкости предложенного решения, так и в части оценки затрат на создание новых измерительных стендов. В частности, необходимы данные о сложности обучения сторонних специалистов по разработке модулей и оценка затрат настроечных работ силами таких специалистов. Интересным также является более тщательное сопоставление затрат и целевых возможностей выбранного подхода по сравнению с реализацией этой же задачи средствами LabVIEW.

Дальнейшее развитие представленного DSM-решения целесообразно направить в сторону повышения удобства разработки и надёжности эксплуатации. В частности, требуется развитие механизмов отладки моделей стендов, включая возможность пошагового выполнения сценариев и мониторинга данных, передаваемых между модулями через виртуальные порты. Кроме того, важным направлением является разработка средств обеспечения безопасности при работе с оборудованием. Речь идёт о внедрении механизмов контроля, предотвращающих потенциально опасные режимы работы, например, подачу сигналов с недопустимыми уровнями мощности на вход измерительных приборов.

Отметим, что данный проект в настоящее время активно развивается в направлении создания программно-аппаратного комплекса, предназначенного для использования на радиотехнических производствах РФ, в целях сокращения сроков передачи новых изделий в промышленное производство.

В целом, можно констатировать, что методология предметно-ориентированного моделирования [2, 26-27] оказалась успешно применимой в контексте разработки средств визуального моделирования измерительных стендов радиотехнических систем. Дальнейшее развитие представленного подхода может осуществляться в сторону повышенной модульности [27], а также интеграции радиотехнических систем с информационными системами [28].

## Список литературы / References

[1]. Eclipse Modeling Project. Available at: https://eclipse.dev/modeling/, accessed 13.10.2026.
[2]. Кознов Д.В. Методология и инструментарий предметно-ориентированного моделирования. Дисс. на соиск. научн. степени д-ра техн. наук. СПбГУ. 2016.
[3]. Kelly S. Domain-Specific Modeling: Enabling Full Code Generation. John Wiley & Sons, 2008. 340 p.
[4]. LabVIEW overview. Available at: https://www.ni.com/en/shop/labview.html, accessed 13.10.2026.
[5]. Программный комплекс автоматизации измерений METLAB. Available at: https://dipaul.ru/catalog/element/metlab/, accessed 13.10.2026.
[6]. Low-code редактор ЭМИКА для автоматизации измерений. Available at: https://emika.tech/about/, accessed 13.10.2026.

[7]. Standard Commands for Programmable Instruments. Available at: https://www.ivifoundation.org/downloads/SCPI/scpi-99.pdf, accessed 13.10.2026
[8]. MATLAB Simulink overview. Available at: https://www.mathworks.com/products/simulink.html, accessed 13.10.2026.
[9]. Keysight VEE. Available at: https://www.keysight.com/zz/en/lib/software-detail/computer-software/keysight-vee-pro-software-1525226.html, accessed 13.10.2026.
[10]. Rohde & Schwarz CMWrun. Available at: https://www.rohde-schwarz.com/cz/products/test-and-measurement/wireless-device-testing-software/rs-cmwrun-sequencer-software_63493-199809.html, accessed 13.10.2026.
[11]. Rohde & Schwarz InstrumentView overview. Available at: https://www.rohde-schwarz.com/tw/products/test-and-measurement/analysis-remote-control-software/rs-instrumentview-analysis-software_63493-718976.html, accessed 13.10.2026.
[12]. ITU-T, Recommendation Z.100: Specification and Description Language (SDL). Geneva, Switzerland: International Telecommunication Union.
[13]. Selic B. Real-Time Object-Oriented Modeling / Selic B., Gullekson G., Ward P.T. John Wiley & Sons, Inc., 1994.
[14]. Object Management Group. Unified Modeling Language (OMG UML), Version 2.5. 2013.
[15]. Object Management Group. OMG SysML, Version 1.7. 2024.
[16]. Simulink Documentation: Sample Time. Available at: https://www.mathworks.com/help/simulink/ug/what-is-sample-time.html, accessed 13.10.2026.
[17]. Simulink Documentation: Rate Transitions. Available at: https://www.mathworks.com/help/simulink/slref/ratetransition.html, accessed 13.10.2026.
[18]. VISA (Virtual Instrument Software Architecture) Specification. Available at: https://www.ivifoundation.org/downloads/VISA/vpp43_2024-01-04.pdf, accessed 13.10.2026.
[19]. IEEE 488.1-2003. Standard for Higher Performance Protocol for the Standard Digital Interface for Programmable Instrumentation. New York: Institute of Electrical and Electronics Engineers, 2003, vol. XIII, 123 p.
[20]. Instrument Control Toolbox: Control test and measurement instruments and communicate with computer peripherals. Available at: https://www.mathworks.com/products/instrument.html, accessed 13.10.2026.
[21]. PySide2 (Qt for Python) Documentation. Available at: https://doc.qt.io/qtforpython-5/, accessed 13.10.2026.
[22]. Socket – Low-level networking interface (Python Standard Library). Available at: https://docs.python.org/3/library/socket.html, accessed 13.10.2026.
[23]. pySerial Documentation. Available at: https://pyserial.readthedocs.io/en/latest/, accessed 13.10.2026.
[24]. PyUSB documentation. Available at: https://pyusb.github.io/pyusb/, accessed 13.10.2026.
[25]. PyVISA: Control your instruments with Python. Available at: https://pyvisa.readthedocs.io/en/latest/, accessed 13.10.2026.
[26]. Кознов Д.В., Ольхович Л.Б. Визуальные языки проектов. Системное программирование. 2005, т. 1, с. 148-167.
[27]. Кознов Д.В. Визуальное моделирование компонентного программного обеспечения. Автореферат диссертации на соискание ученой степени кандидата физико-математических наук / Санкт-Петербургский гос. ун-т. Санкт-Петербург, 2000.
[28]. Терехов А.Н., Кознов Д.В., Романовский К.Ю., Долгов П., Иванов А.Н. Объектно-ориентированная методология разработки информационных систем и программного обеспечения систем реального времени // Объектно-ориентированное визуальное моделирование. Санкт-Петербург, 1999, с. 4-20.

## *Информация об авторах / Information about authors*

Никита Михайлович ФАСТ – магистр кафедры системного программирования Санкт-Петербургского государственного университета; сфера его научных интересов: программная инженерия, модельно-ориентированная разработка, цифровая обработка сигналов.

Nikita Mikhailovich FAST – master's student of St. Petersburg State University (SPbSU). His research interests include software engineering, model-based development, and digital signal processing.





Александр Сергеевич КРИВОНОГОВ – ведущий инженер ООО «Концентр», сфера его научных интересов включает цифровую связь, цифровую обработку сигналов, помехоустойчивое кодирование.

Alexander Sergeevich KRIVONOGOV – senior engineer at Kontsentr. His research interests are as follows: digital communications, digital signal processing, error-correcting coding.

Дмитрий Владимирович КОЗНОВ – доктор технических наук, профессор кафедры системного программирования Санкт-Петербургского государственного университета, сфера его научных интересов включает программную инженерию, модельно-ориентированную разработку программного обеспечения (включая предметно-ориентированные языки), анализ данных, телекоммуникации.

Dmitry Vladimirovich KOZNOV – Dr. Sci. (Tech.), Associate Professor at St. Petersburg State University (SPbSU). Research interests: software engineering, model-driven software development (including programming DSL), data analysis, telecommunications.