

Санкт-Петербургский государственный университет

Программная инженерия

Гонта Ксения Алексеевна

Реализация редактора форм фигур как  
визуального языка на DSM-платформе  
QReal

Курсовая работа

Научный руководитель:  
ст. преп. Литвинов Ю. В.

Санкт-Петербург  
2016

# Оглавление

<b>Введение</b>	<b>3</b>
<b>1. Обзор</b>	<b>6</b>
1.1. Система QReal . . . . .	6
1.2. Технология Qt Quick . . . . .	8
1.3. Интеграция QtQuick 1.0 в QReal . . . . .	8
1.4. Выводы . . . . .	9
<b>2. Реализация</b>	<b>10</b>
<b>3. Редактор интерфейсов пультов управления роботами</b>	<b>13</b>
<b>Заключение</b>	<b>14</b>
<b>Список литературы</b>	<b>15</b>

# Введение

Модельно-ориентированный подход (domain-specific approach) в разработке программного обеспечения позволяет представлять программы в виде набора моделей, чаще всего визуальных. Для описания системы целесообразным становится использование визуальных языков моделирования. Это облегчает процесс разработки системы и её понимания. Однако за простотой использования скрывается сложность поддержки: необходимо поддерживать различные редакторы, генераторы и т.д.

Существует две группы систем визуального моделирования — универсальные и предметно-ориентированные. Первый подход не даёт существенного выигрыша в производительности труда разработчика, так как он оперирует теми же терминами, какими бы оперировал при текстовом программировании.

Поэтому часто более выгодным оказывается предметно-ориентированный подход. Разработанные под определённую программную область инструменты дают возможность полной генерации кода на текстовом языке, при этом оставаясь понятным экспертам в данной области, возможно, не владеющими навыками программирования. Поскольку создание визуальных языков с нуля является сложным процессом, появляется необходимость в инструментах, которые будут позволять в короткие сроки создавать новые языки и поддерживать модели на них. Такие инструменты получили название DSM-платформ<sup>1</sup>. Одной из таких платформ является платформа QReal[1][5], разрабатываемая на кафедре системного программирования СПбГУ.

При создании визуального языка необходимо задать абстрактный и конкретный синтаксис. Абстрактный синтаксис задаёт элементы, их свойства и возможные связи между ними. Конкретный синтаксис — внешний вид элементов.

Современные DSM-платформы умеют задавать абстрактный синтаксис элементов, но не все умеют задавать конкретный синтаксис ви-

---

<sup>1</sup>DSM — domain-specific modeling

зуальными средствами. Некоторые позволяют выбирать из готовых форм, другие — задавать описания форм фигур на текстовых языках. На платформе QReal внешний вид элемента задаётся при помощи встроенного редактора форм. Редактор форм фигур представляет собой объёмный код на C++, который необходимо поддерживать. К тому же он частично реализует заново функциональность DSM-платформы, такую как редактирование элементов на сцене, отмена, повтор действий и т.д.

Однако визуальный язык редактора внешнего вида элементов может быть задан при помощи самой DSM-платформы QReal как ещё один предметно-ориентированный визуальный язык. Преимуществом такого подхода является возможность использовать функциональность редакторов визуальных языков DSM-платформ. Не будет необходимости решать проблемы, которые и так решены в визуальных редакторах.

Создание такого языка оказывается не настолько простым, каким кажется на первый взгляд. Обычно в DSM-платформах не возникает необходимость изменять визуальное представление элементов при изменении их свойств, поэтому они не обладают подобной функциональностью. Таким образом теряется наглядность проектирования языков, потому что внешний вид элементов при проектировании будет отличаться от того, как они будут выглядеть в сгенерированном редакторе.

Целью данной работы было создание редактора форм фигур, как ещё одного визуального языка, средствами DSM-платформы QReal. Новый язык должен отвечать требованиям динамической изменемости внешнего вида элементов при изменении их свойств.

Для достижения поставленной цели было необходимо решить следующие задачи:

- рассмотреть существующие средства работы с графическим представлением элементов Qt и QReal;
- создать новый визуальный язык редактора форм фигур элементов;
- написать генератор внешнего вида элементов для метамодели;

- провести апробацию языка созданием редактора пультов управления роботами.

# 1. Обзор

## 1.1. Система QReal

QReal — DSM-платформа, предназначенная для быстрой разработки новых визуальных языков. Она разрабатывается на кафедре системного программирования СПбГУ. Архитектурно платформа QReal устроена так. Есть абстрактное ядро, которое реализует общую для всех редакторов функциональность. Ядро не содержит специализированного кода для каждого визуального редактора. Весь специфический для каждого языка код автоматически генерируется из метамодели в подключаемый модуль или пишется вручную. Общий вид представлен на рисунке 1.

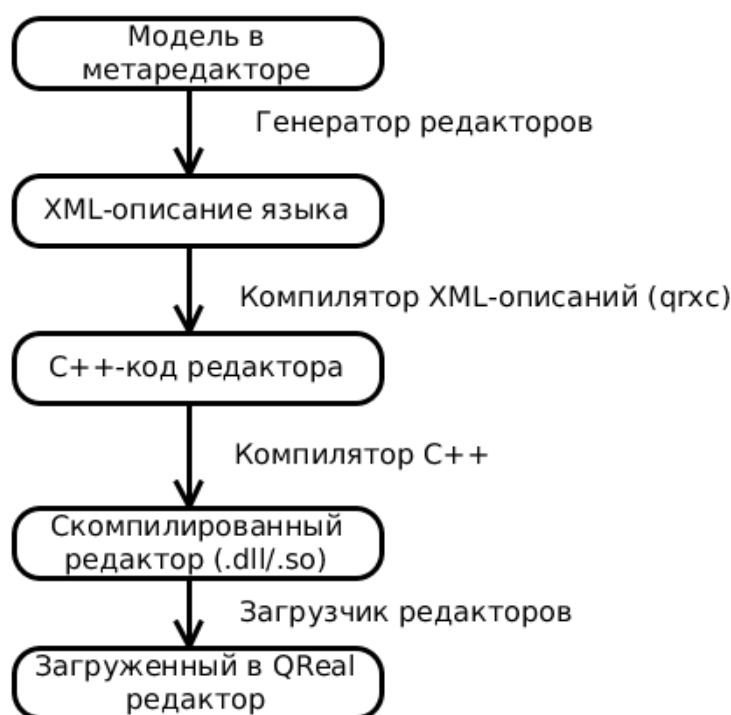


Рис. 1: Архитектура QReal.

Процесс создания новых визуальных языков на DSM-платформе QReal следующий. Основными компонентами языка являются элементы и связи между этими элементами. При помощи метаредактора, являющегося частью DSM-платформы QReal, задаётся множество таких

элементов со всеми их свойствами. Таким образом задаётся абстрактный синтаксис языка. В то же время нужно задать внешний вид элементов, который будет описывать конкретный синтаксис языка.

Созданная таким образом метамодель генерируется в XML файл с полным описанием языка. По этому файлу генерируется код редактора на C++, который компилируется вместе с исходным кодом платформы QReal и впоследствии может быть динамически подключён к ядру системы. Схематически этот процесс представлен на рисунке 2.

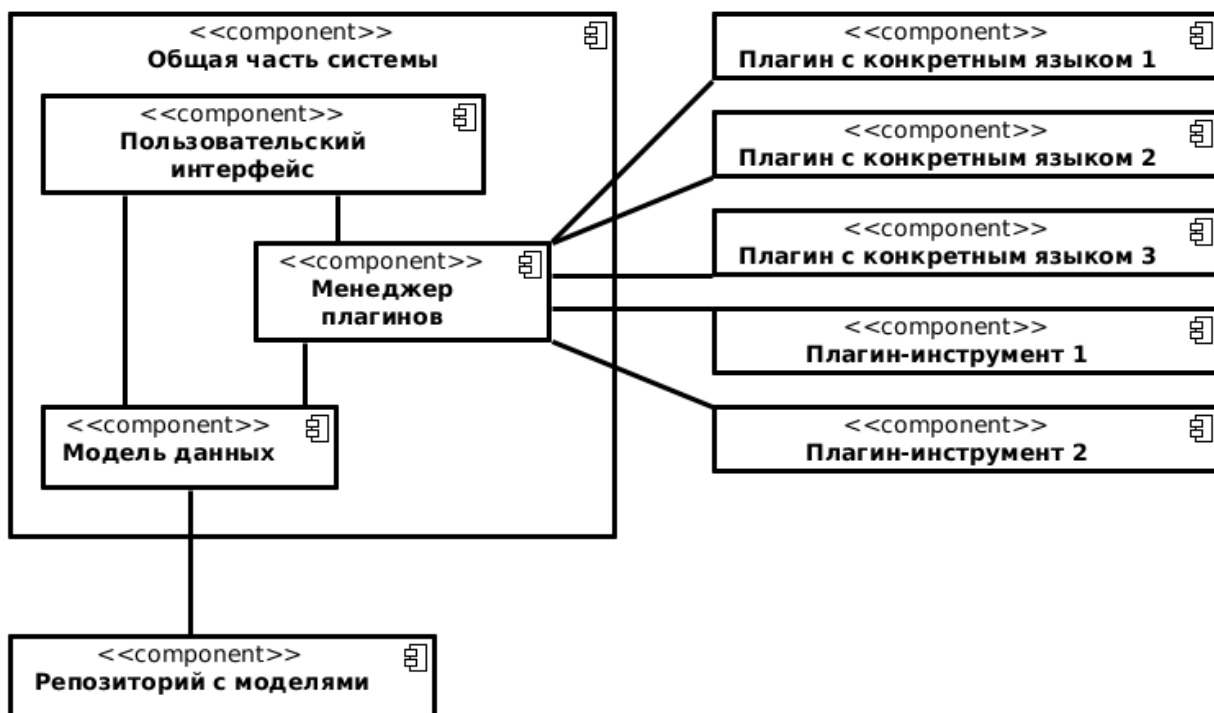


Рис. 2: Процесс создания визуального редактора в QReal.

На момент начала работы для описания внешнего вида элементов использовался внутренний формат данных SDF. SDF был создан в качестве надстройки над форматом векторной графики SVG, так как последний не позволял совершать некоторые действия, такие как задание портов и динамическая подгрузка надписей — возможность изменять текст надписи на элементе в соответствии с логической моделью. Для создания редактора форм фигур как визуального языка на платформе QReal функциональности формата SDF также было недостаточно. Так, например, для создания красного прямоугольника необходимо было вы-

тянуть на сцену сущность "Прямоугольник" и изменить в свойствах цвет на красный. При этом генерировался красный прямоугольник, однако при проектировании не было возможности увидеть будущий вид элемента, так как элементы на сцене не реагировали на изменение свойств. Такая статичность никак не могла быть использована для создания редактора внешнего представления элементов.

## 1.2. Технология Qt Quick

При рассмотрении сред разработки пользовательских интерфейсов было обращено внимание на технологию Qt Quick, которая в последнее время активно используется для разработки интерфейсов мобильных приложений. Основной идеей технологии является представление интерфейса в виде визуального иерархического дерева элементов, а также предоставление богатого набора переходов и анимаций, которые существенно облегчают процесс разработки. В Qt Quick используется основанный на JavaScript язык QML, позволяющий определять иерархию объектов, их внешний вид и свойства. Отличительной особенностью технологии является предоставление множества способов обмена данными между C++ и QML, что и обеспечивает необходимое изменение внешнего вида элементов при изменении их свойств.

На данный момент существует две версии QtQuick: QtQuick 1.0[2] и QtQuick 2.0[3]. Во второй версии были улучшены используемые библиотеки, добавлены новые элементы, а также появилась поддержка API OpenGL 2.0.

## 1.3. Интеграция QtQuick 1.0 в QReal

В прошлом году в качестве дипломной работы[4] уже была предпринята попытка внедрения QtQuick 1.0 в платформу QReal. Код тогда не удалось своевременно включить в главную ветку проекта, а из-за постоянно изменяющегося кода платформы сейчас это сделать практически невозможно. Ключевые этапы этой дипломной работы представлены ниже.



Для начала работы QtQuick необходимо было встроить в инфраструктуру графической сцены. Для этого потребовалось механизмы создания визуальных представлений, изменения их свойств и записи полученных результатов в метамоделю. В первой версии Qt Quick было недостаточное количество компонентов, поэтому в рамках диплома были дописаны недостающие элементы управления. Некоторые элементы были реализованы средствами C++, другие — средствами QML.

Так как ранее визуальные представления элементов были заданы при помощи формата SDF, то было принято решение перевести внешний вид элементов в формат QML. Количество разработанных ранее языков и их элементов было большим, поэтому, ввиду схожести SDF и QML был разработан генератор для перевода из одного формата в другой. После получения графического вида элементов также были прописаны их связи с логической частью модели.

По полученному представлению необходимо было сохранять внешний вид элемента. Для этого был реализован генератор, разбирающий присутствующие на сцене элементы с их положением и сохраняющий представление элемента в XML-формате.

## **1.4. Выводы**

Технология Qt Quick предоставляет функциональность, необходимую для решения поставленной задачи. Работа прошлого года показала возможность встраивания Qt Quick 1.0 в QReal, но также показала недостаточную мощь первой версии технологии. Было решено внедрить Qt Quick 2.0 и апробировать полученные результаты.

## 2. Реализация

Реализация базируется на работе Дмитрия Мордвинова, интегрировавшего технологию QtQuick в ядро графической подсистемы QReal. Целью моей курсовой работы было используя эту технологию создавать язык визуального редактора форм фигур на базе платформы QReal.

Благодаря QtQuick появилась возможность описания внешнего вида элементов при помощи QML-файлов. К тому же стали доступны стандартные элементы библиотеки компонентов QtQuick 2.0, среди которых в том числе есть такие интерактивные элементы управления как кнопка, флажок и т.д.

Так как ранее элементы были описаны на языке SDF, был осуществлён перевод описания элементов с SDF-формата на QML. Переиспользовать автоматический конвертер, написанный в прошлом году Богданом Клепачем, не удалось, так как тот генератор переводил элементы в сущности, присутствовавшие в QtQuick 1.0, а не QtQuick 2.0. Таким образом перевод осуществлялся вручную, на данный момент такая работа проделана для метаредактора и редактора диаграмм блоков.

Появилась возможность редактирования внешнего вида элементов изменением свойств элемента на палитре. На рисунке 3 можно увидеть прямоугольник до изменения его свойств с палитры, а на рисунке 4 — после.

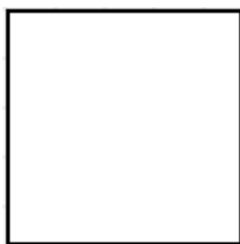


Рис. 3: До изменения свойств.

Был создан язык редактора форм фигур. Элементы пользовательского управления, комбинирования их в формы. При создании был сделан упор на удобство пользователя, лаконичность и простоту. В таблице



Рис. 4: После изменения свойств.

1 приведены некоторые элементы полученного языка вместе с перечислением тех свойств элементов, которые можно изменять.

Процесс задания внешнего вида элемента стал следующим. Воспользовавшись метаредактором разработчик задаёт множество элементов, из которых будет состоять его будущий язык. По двойному щелчку по элементу открывается редактор форм фигур для текущего элемента.

Для сохранения полученного вида в XML-файл был написан генератор из нового языка. Для начала необходимо обозначить общий размер фигуры. Так как одна фигура может быть составлена из нескольких простых, причём мы не можем гарантировать их визуальную вложенность друг в друга, то необходимо посчитать, какого размера будет достаточно для расположения всех элементов. Для этого высчитываются точки, содержащие все элементы, это берётся за контур основного элемента. Все остальные элементы располагаются относительно верхнего левого угла основного элемента. Для каждого графического элемента сцены вызывается своя функция генерации.




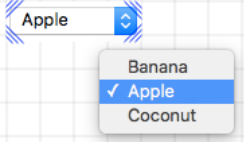



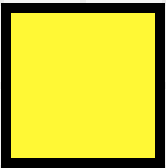
	Элемент	Изменяемые свойства
Button		ширина, высота, текст
Check Box		ширина, высота, текст
Radio Button		ширина, высота, текст
Combo Box		ширина, высота, варианты выпадающего списка
Label		ширина, высота, текст
Progress Bar		ширина, высота, процент загрузки
Switch		ширина, высота
Rectangle		ширина, высота, цвет бортика, ширина бортика

Таблица 1: Описание изменяемых свойств некоторых элементов языка

### 3. Редактор интерфейсов пультов управления роботами

На данный момент робототехнические конструкторы набирают всё большую популярность в образовательном процессе. Это удобный способ обучения конструированию, программированию, отработки алгоритмов движения и компьютерного зрения. Модели для обучения могут быть совершенно разнообразными: с лампочками, моторами, хватом, видеокамерой и т.д. Было бы очень удобно иметь свой пульт управления под каждую модель. Поэтому необходим визуальный язык, который позволит быстро выбрать подходящие кнопки, расположить их на сцене и сгенерировать по ним готовое приложение. Генерация самого кода — это большая работа, не входящая в курсовую, поэтому в этой работе задача была ограничено написанием языка. На рисунке 4 можно увидеть пример получившегося интерфейса.

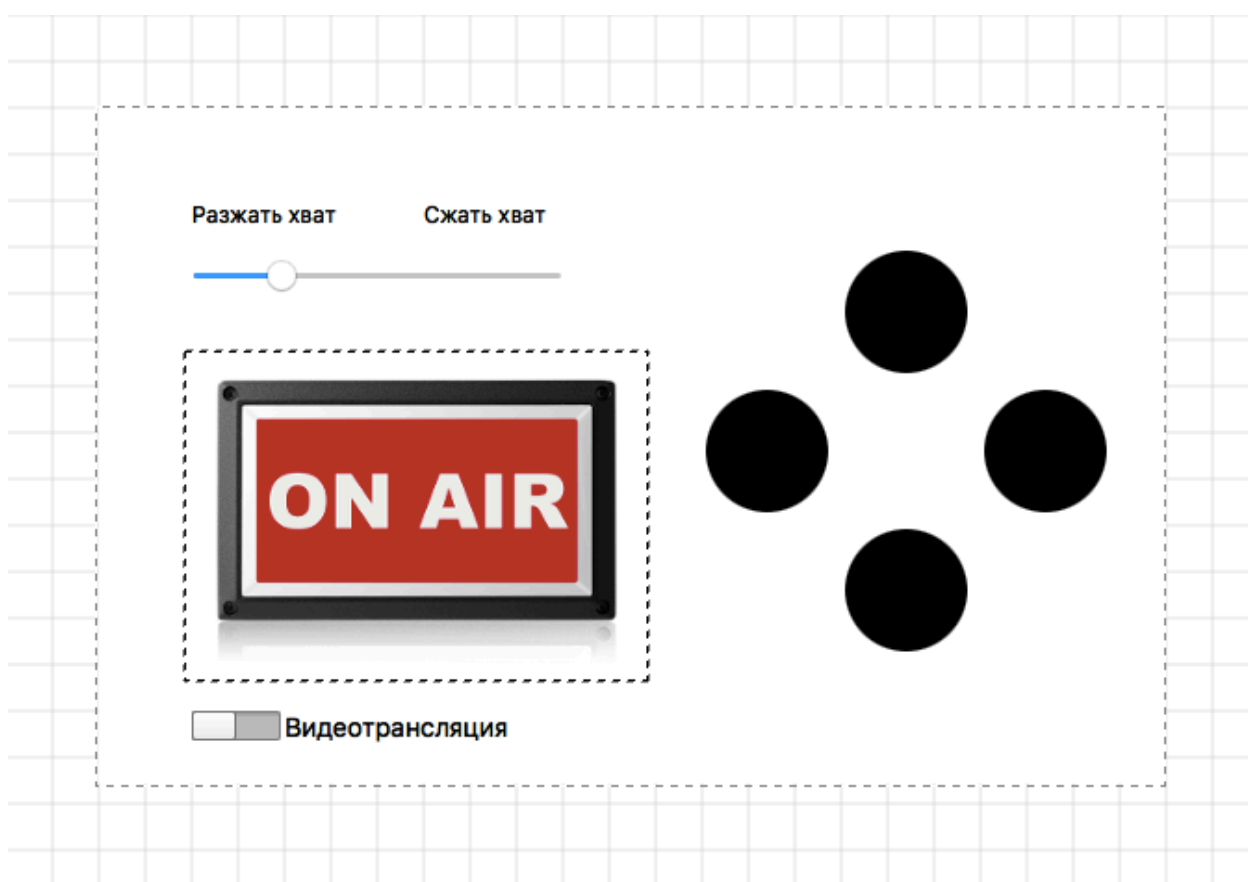


Рис. 5: Пример интерфейса пульта управл.

## Заключение

В процессе работы были изучены современные технологии разработки пользовательских интерфейсов. Была изучена архитектура платформы QReal. Были созданы:

- редактор форм фигур;
- генератор из полученного редактора;
- в качестве апробации был создан язык создания пультов управления роботами.

Таким образом появилась возможность создания из простых примитивов более сложных конструкций и использования их самих в дальнейшем в качестве исходных элементов для проектирования.

## Список литературы

- [1] A. Kuzenkova A. Deripaska T. Bryksin Y. Litvinov V. Polyakov.
- [2] QtQuick 1.0.
- [3] QtQuick 2.0.
- [4] Б.В Клепач. Развитие графической подсистемы DSM-платформы QReal. — 2015. — URL: [http://se.math.spbu.ru/SE/diploma/2015/bse/Клепач\\_Bogdan\\_Valentinovich-text.pdf](http://se.math.spbu.ru/SE/diploma/2015/bse/Клепач_Bogdan_Valentinovich-text.pdf).
- [5] Терехов А.Н. Брыксин Т.А. Литвинов Ю.В.