Правительство Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Санкт-Петербургский государственный университет»

Кафедра системного программирования

Когутич Денис Александрович

Поддержка работы с подпрограммами в DSM-платформе

Курсовая работа

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

Санкт-Петербург
2016

Оглавление

Введение

Визуальное программирование[1] – способ создания программы для ЭВМ путём манипулирования графическими объектами вместо написания её текста. Визуальное программирование часто представляют как следующий этап развития текстовых языков программирования.

Визуальные языки позволяют существенно сократить затраты на разработку программного обеспечения и снизить требуемые знания и умения программистов. Особенно интересны предметно-ориентированные визуальные языки – языки, специально созданные для узкой предметной области или класса задач. Разумеется, создавать инструментальные средства для таких языков "с нуля" было бы неоправданно трудозатратно, поэтому существуют специальные инструменты быстрой разработки таких языков и инструментов для них – редакторов, генераторов кода, средств проверки ограничений и т.д. Такие инструменты называются DSM[2]-платформами [1].

На кафедре системного программирования СПбГУ разрабатывается DSM-платформа QReal[3]. Данная платформа позволяет автоматически генерировать код произвольных визуальных редакторов по описаниям их метамоделей, также в ней реализовано так называемое метамоделирование "на лету", которое позволяет быстро создавать визуальные языки в процессе редактирования диаграмм на этих языках. На базе проекта QReal создана среда обучения основам программирования и кибернетики QReal:Robots (TRIK Studio), которую использует ряд российских школ [2].

Подпрограммы в визуальных языках, как и функции в текстовых, играют очень важную роль. С их помощью можно создавать сложные программы без нагромождения блоков на главной диаграмме,

---

[1] Визуальное программирование, url: https://ru.wikipedia.org/wiki/Визуальное_программирование (дата обращения 07.12.2015)

[2] DSM --- Domain-Specific Modeling

[3] Домашняя страница, исходные коды QReal, URL: https://github.com/qreal/qreal (дата обращения: 28.03.2016г)

переиспользовать написанный “код”, делать рекурсию. Более того, с их грамотным использованием программы приятнее воспринимаются, своеобразная читаемость “кода”. Также визуальные языки активно используются в качестве обучения программированию, работа с подпрограммами в процессе такого обучения благоприятно отразится на учениках и даст представление о структурном программировании, что в дальнейшем сильно упростит знакомство с текстовыми языками, такими как C++, Java и т.д.

На данный момент в проекте QReal реализована поддержка работы с подпрограммами [3]. Однако реализована она частично: нет возможности передавать параметры в функции (единственный способ – через глобальные переменные), изменять внешний вид блока вызова создаваемой пользователем подпрограммы и т.д. Поэтому целью данной работы стала полноценная поддержка подпрограмм в данной платформе.

1. Задачи

В рамках данной курсовой работы ставились следующие задачи:

- реализовать возможность передачи параметров в подпрограммы;
- обеспечить работу параметров в режиме интерпретации для языка программирования роботов;
- обеспечить возможность изменения внешнего вида блока вызова подпрограммы;
- создать редактор свойств для редактирования параметров, имени и картинки подпрограммы.

2. Обзор предметной области

В DSM-платформах реализация подпрограмм сложна из-за того, что подпрограммы проявляют свойства как конкретных блоков, так и элементов языка, то есть типов блоков. Подпрограмма представляет собой, как правило,

отдельную диаграмму, на которой средствами языка (возможно, с использованием других подпрограмм) описана часть решения задачи, которую можно переиспользовать. Подпрограмма в визуальном языке является аналогом функции в текстовых языках, она может иметь аргументы, параметры со значениями по умолчанию или не иметь их вовсе. Как и в текстовых языках, формальные параметры подпрограммы характеризуются парой: имя и тип, а фактические – конкретные значения, ставящиеся в соответствие формальным. Для переиспользования подпрограммы в языке обычно есть отдельный элемент "Вызов подпрограммы", который добавляется на другие диаграммы и служит для обозначения места, где диаграмма вызывает подпрограмму. Блок "Вызов подпрограммы" является элементом визуального языка, но каждая подпрограмма решает свою задачу, так что было бы естественно дать пользователю возможность выбрать внешний вид этого блока для разных подпрограмм, иметь палитру подпрограмм, откуда можно было бы помещать блоки “Вызов подпрограммы” на диаграмму, так что вызовы разных подпрограмм ведут себя очень похоже на стандартные элементы языка, такие как блоки "Моторы вперёд" или "Задержка по таймеру".

3. Обзор существующих решений

В существующих аналогах QReal работа с подпрограммами реализована лишь на базовом уровне. Так, в известной платформе MetaEdit+ есть отношение "Explosion", которое позволяет связать элемент диаграммы с другой диаграммой, в которую этот элемент раскрывается, так что при двойном клике мышью по элементу откроется диаграмма [4]. В наиболее известной в научном сообществе DSM-платформе Eclipse Graphical Modeling Project (точнее, в подпроекте Sirius) прямая поддержка подпрограмм отсутствует, требуемое поведение приходится реализовывать вручную [5].

4. Обзор инструментария

4.1. QReal

QReal имеет плагинную архитектуру, плагины подключаются к общей части. Они создают кнопки, пункты меню и т.д., устанавливают свои обработчики и передают в пользовательский интерфейс с указанием куда их добавить. Поскольку код плагинов вынужден использовать код из общей части, QReal был разбит на несколько модулей. На диаграмме (рис. 1) показаны компоненты, собираемые в отдельные бинарники. Стрелки – зависимости по сборке.

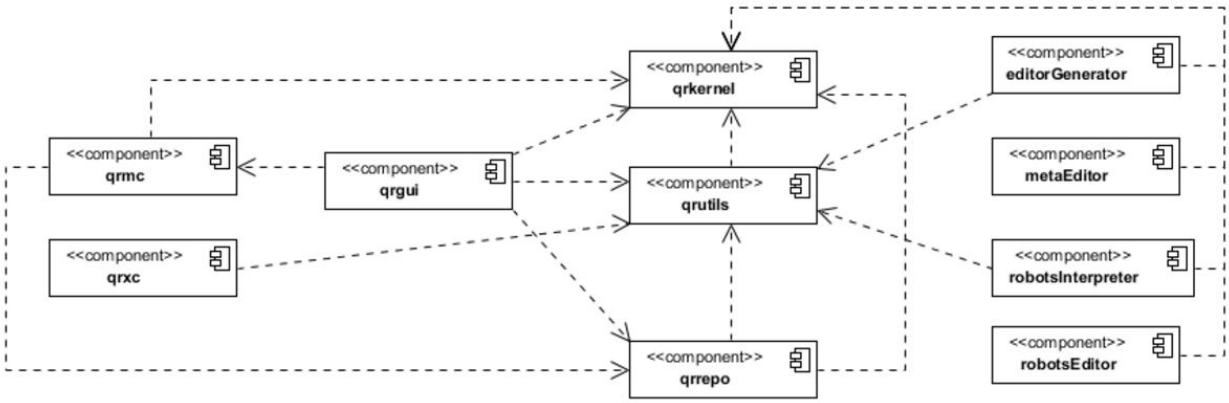

рис. 1. Диаграмма компонентов QReal [6]

- **qrkernel** содержит классы, используемые повсюду в системе, такие как Id и Exception.
- **qrutils** содержит код, который может быть полезен в нескольких компонентах. Например, вспомогательные классы для работы с xml или записи в файлы.
- **qrrepo** – репозиторий. Плагины обычно получают доступ к нему не напрямую, а через модели qrgui.
- **qrgui** – "основная программа", пользовательский интерфейс и система управления плагинами. qrgui грузит плагины динамически, qrgui и плагины тесно связаны, причём в обе стороны.

- **editorGenerator**, **metaEditor**, **robotsInterpreter**, **robotsEditor** и т.д. – плагины. Часть из них – плагины-редакторы, собираемые из метамоделей, часть – плагины-инструменты, рукописные. Плагины-редакторы – редакторы визуальных языков. Плагины-инструменты предоставляют некоторую функциональность, регистрируемую в qrgui и вызываемую пользователем.
- **qrxc**, **qrmc** – компиляторы метамоделей. Первый генерирует код на C++ по описанию синтаксиса языка в .xml-файле, второй – по сохранённой метамодели.

(подробнее см. [6]).

4.2. Понятие эксплозии

Визуальные языки в современных DSM-платформах, как правило, задаются с помощью метамоделей, то есть моделей синтаксиса языка. Они описывают, какие элементы могут находиться на диаграмме, какие свойства есть у этих элементов, и как элементы могут быть связаны друг с другом (см. [7]). В QReal при добавлении блока "Подпрограмма" на сцену появляется новое окно, где надо ввести имя подпрограммы, после чего создаётся новая диаграмма, и блок подпрограммы связывается с ней так, что по двойному нажатию мыши эта новая диаграмма открывается, и там можно рисовать её реализацию. Внутри это устроено так: в метамодели узел Subprogram связан отношением "explodes to" (эксплозией) с узлом SubprogramDiagram. Target – цель эксплозии, это SubprogramDiagram, source – источник эксплозии, он же узел Subprogram. У одной цели эксплозии может быть несколько источников.

4.3. TRIK Studio

TRIK Studio — среда визуального и текстового программирования образовательных конструкторов роботов. TRIK Studio появилась как развитие проекта кафедры системного программирования СПбГУ

QReal:Robots. Программа на визуальном языке (визуальная диаграмма) может быть исполнена в трех режимах:

- отладка на симуляторе;
- отладка на компьютере с посылкой пакетов на робота по одному из физических каналов (USB, Bluetooth, Wi-Fi);
- режим генерации кода на текстовом языке с последующим автономным исполнением его на роботе.

В режиме отладки на симуляторе диаграмма интерпретируется на двумерной имитационной модели робота. Отладка на компьютере с посылкой команд роботу (режим интерпретации в терминах среды) удобна для отслеживания поведения программы на целевом устройстве в реальном времени. Режим генерации кода позволяет перейти от визуального представления программы к текстовому.

Интерпретатор преобразует визуальную диаграмму в последовательность команд на целевом устройстве (рис. 2).

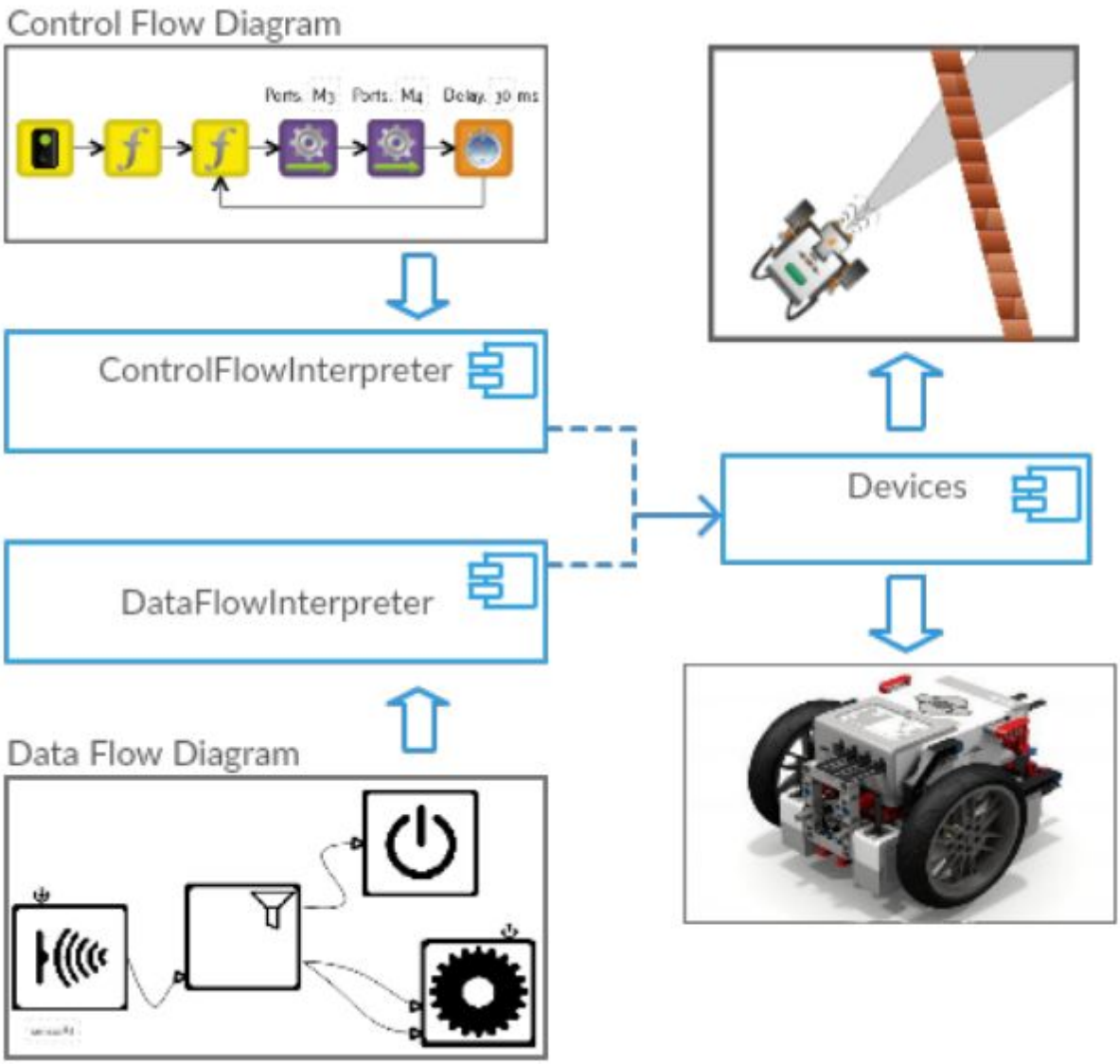

рис. 2. Общий принцип работы систем интерпретации [8]

Интерпретатор трассирует поток управления диаграммы в том порядке, как он задан стрелками. Если на каком-то шаге интерпретации повстречался блок распараллеливания, интерпретатор запускает новые потоки исполнения с соответствующих точек, создавая для каждого свой стек вызовов. При достижении любого завершающего блока со стека снимается верхний фрейм, и исполнение продолжается с соответствующей точки на предыдущей диаграмме. Если при снятии очередного фрейма стек вызовов стал пустым, текущий поток исполнения считается завершенным. Таким образом,

программа исполняется по мере того, как она «открывается» интерпретатору (подробнее см. [8]).

5. Реализация

Сходные вещи с нашими целями можно сделать с помощью интерпретации метамоделей и внесения изменений в метаязык на лету (см. [7]), но поскольку в QReal основной режим – это режим генерации редакторов, мы пошли другим путём. Было принято решение ввести понятие “динамические свойства” для каждой подпрограммы, интерпретируемые ядром системы. Они представляют собой XML-документ и используются для хранения дополнительной информации (картинка подпрограммы, описание параметров и т.д.), см. примеры ниже. Эти свойства хранятся в репозитории, но являются скрытыми от пользователя, то есть не отображаются в стандартном редакторе свойств QReal и на сцене. Цели эксплозии добавлены скрытые “динамические свойства” "shape" и "labels", в которых хранятся картинка и описание формальных параметров подпрограммы, соответственно. При создании эксплозии, то есть после того как пользователь добавил на сцену блок "Подпрограмма", в "shape" сохраняется стандартная картинка блока “Вызов подпрограммы”, которую впоследствии можно изменить. В редактор диаграмм QReal был добавлен редактор свойств (рис. 4) с удобным пользовательским интерфейсом. В этом редакторе можно изменить имя подпрограммы, установить картинку для блока вызова подпрограммы и изменить/добавить/удалить параметры. Для открытия редактора достаточно кликнуть правой кнопкой мыши на блок вызов подпрограммы на сцене или в пользовательской палитре и выбрать соответствующий пункт меню (рис. 3).

Пример свойства shape:

```
<picture sizex="50" sizey="50">
        <image x1="0" x2="50" y2="50"
        name="subprogramImages/subprogramBackgrounds/2.svg\" y1="0"/>
        <image x1="0" x2="50" y2="50"
        name="subprogramImages/subprogram1.svg\" y1="0"/>
</picture>
```

Пример свойства labels:

```
<labels>
        <label x="40" y="60" value="123" type="int"
        textBinded="{0c167ea6-f187-4e14-89c3-dc58b1202393}"/>
        <label x="-30" y="60" text="x"/>
        <label x="40" y="90" value="false" type="bool"
        textBinded="{f1654990-0599-4f74-a928-8c03bf2117cb}"/>
        <label x="-30" y="90" text="y"/>
        <label x="40" y="120" value="'hello,world'" type="string"
        textBinded="{11aa79cd-cedf-4da9-b16b-0e4e85b28d80}"/>
        <label x="-30" y="120" text="z"/>
</labels>
```

Пример свойства dynamicProperties:

```
<properties>
        <property value="123" text="x" type="int"
        textBinded="{0c167ea6-f187-4e14-89c3-dc58b1202393}"/>
        <property value="false" text="y" type="bool"
        textBinded="{f1654990-0599-4f74-a928-8c03bf2117cb}"/>
        <property value="'hello,world'" text="z" type="string"
        textBinded="{11aa79cd-cedf-4da9-b16b-0e4e85b28d80}"/>
</properties>
```

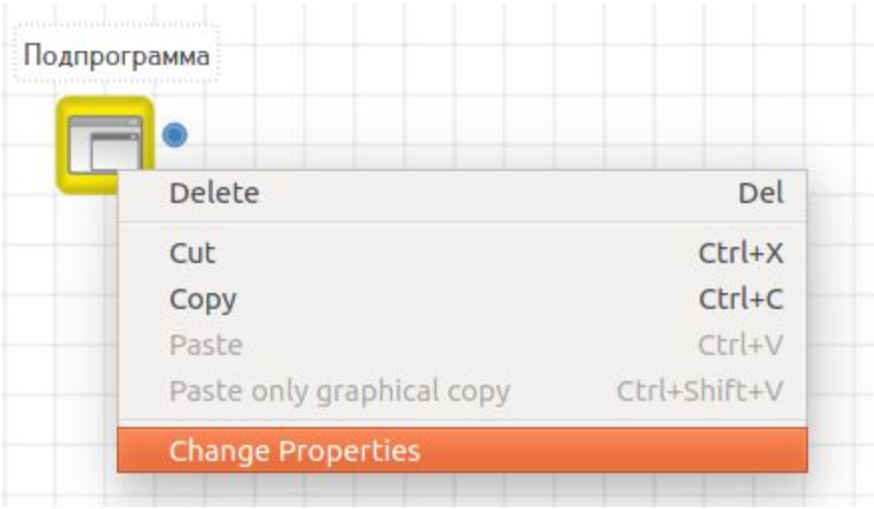

рис. 3. Запуск редактора свойств со сцены

В редакторе есть поле для указания имени подпрограммы, есть таблица для перечисления параметров, состоящая из трёх столбцов: Имя, Тип, Значение. Обязательными для заполнения являются поля Name и Type, значение может быть пустым. Указанные значения являются значениями "по умолчанию". В редакторе можно выбрать картинку и подложку для неё из "стандартного" набора. Коллекцию можно легко расширять своими, для этого достаточно загрузить их в соответствующую папку с проектом.

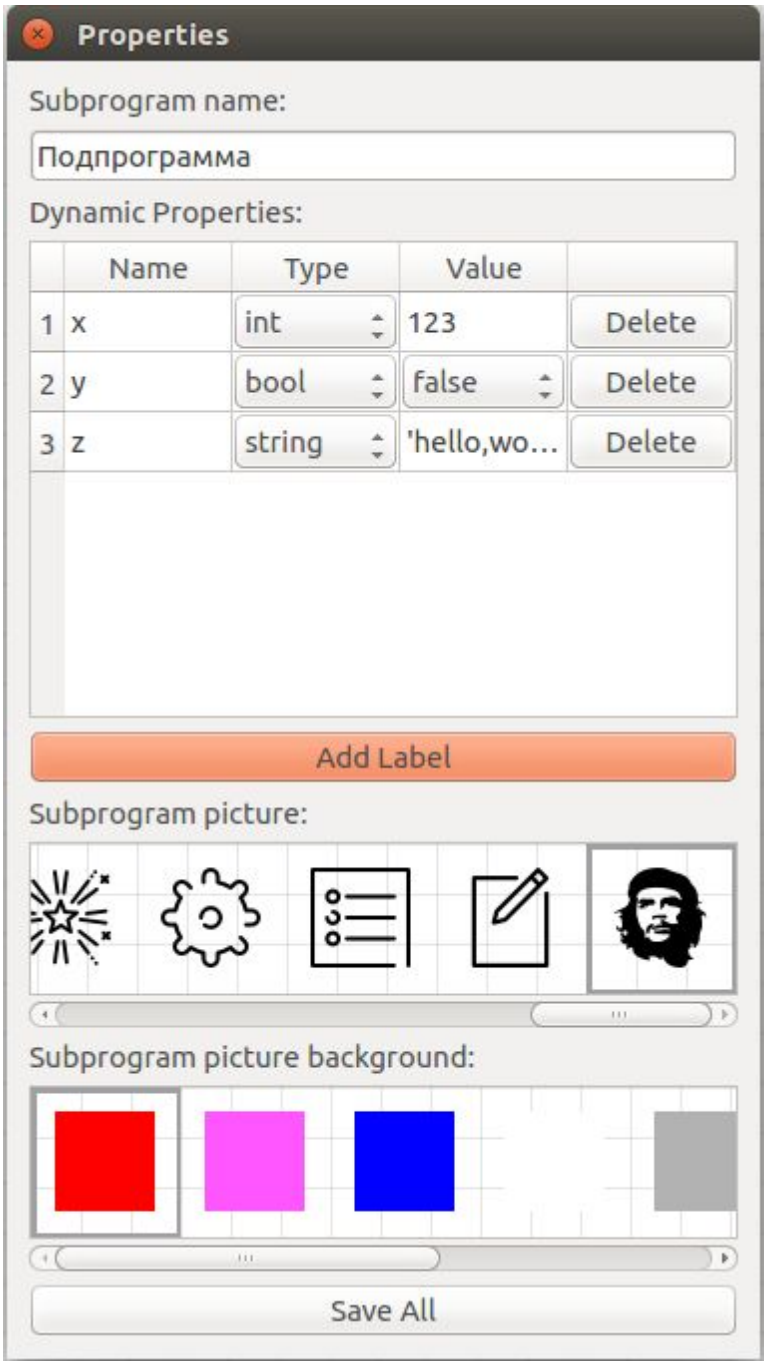

рис. 4. Редактор свойств

При изменении картинки в редакторе свойств и сохранении результата, она сохраняется в свойстве "shape" цели эксплозии и обновляется у всех источников, относящихся к данной цели (рис. 5). При изменении параметров поведение аналогично, но заметим, что параметры устроены сложнее. Дело в том, что, как и в текстовых языках, есть формальные параметры, характеризующиеся именем и типом, и фактические, которые ставят в

соответствие формальным параметрам значения. Формальные параметры – свойства цели эксплозии, фактические – свойства источника. Таким образом, "Вызов подпрограммы" должен получать из блока "Подпрограмма" описание формальных параметров и отображать их в редакторе свойств и на сцене, давая возможность указывать им значения. Значения могут быть разными для каждого блока "Вызов подпрограммы", но формальные параметры для всех них одинаковые, и при модификации формальных параметров в блоке "Подпрограмма" все блоки "Вызов подпрограммы" обновляются. Для того, чтобы хранить фактические параметры в блоке "Вызов подпрограммы", у каждого источника существует скрытое свойство "dynamicProperties".

Фактические параметры подпрограмм отображаются в виде текстовых меток блока "Вызов подпрограммы". Метки, отвечающие за значения, установлены в режим редактирования, что позволяет изменять значения аргументов прямо со сцены.

рис. 5. Результат сохранения свойств

В QReal существует стандартный редактор свойств, в котором отображаются свойства блоков, описанные в метамодели. В нём можно изменять их значения, после чего обновлённые значения тут же отображаются на сцене в текстовых метках блока. При реализации подпрограмм это было учтено и параметры подпрограмм отображаются в стандартном редакторе (рис. 6). Причём существует совместимость, то есть при изменении в стандартном редакторе значения обновляются на сцене и наоборот.

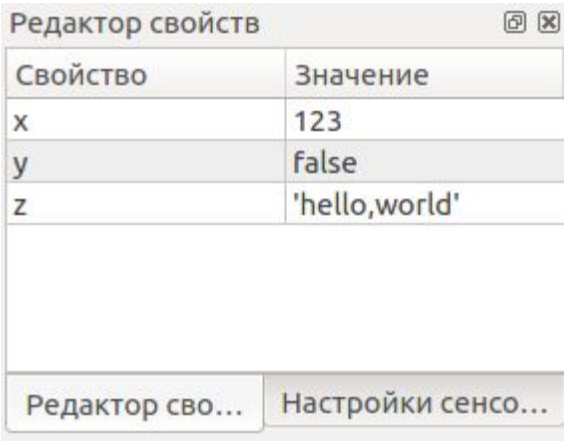

рис. 6. Отображение параметров в стандартном редакторе

Помимо сцены, картинка подпрограммы обновляется и в пользовательской палитре (рис. 7).

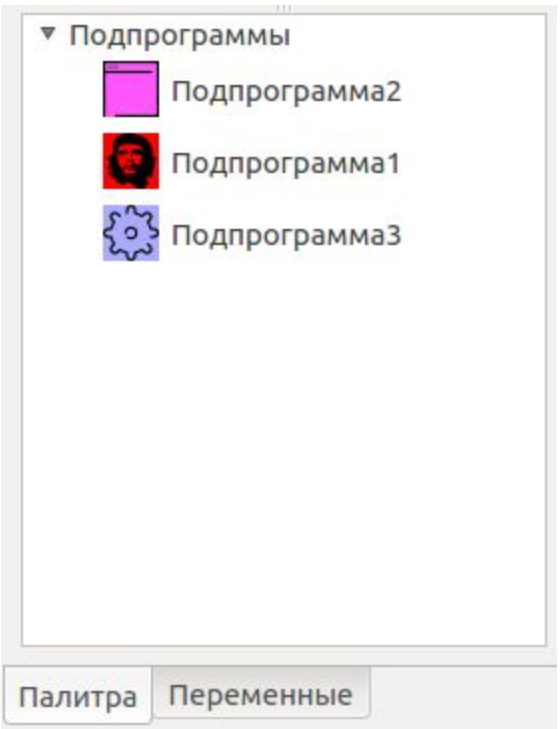

рис. 7. Обновление в пользовательской палитре

В Trik Studio, в режиме интерпретации, был расширен стек вызовов. Теперь, помимо текущего исполняемого блока, в нём хранится ещё и контекст. Контекст хранит в себе набор пар имя-значение, то есть в нем описаны текущие переменные программы и их значения. Это сделано для того, чтобы

переменные подпрограмм находились в “локальном” контексте. После отрабатывания подпрограммы, восстанавливается старый контекст, хранящийся в стеке. Таким образом исключается возможность перезаписи значения глобальной переменной, если её имя совпадает с именем некоторого аргумента подпрограммы.

6. Итоги

- Добавлена возможность передачи формальных и фактических параметров в подпрограмму
- Добавлена возможность изменять внешний вид блока «Вызов подпрограммы»
- Создан редактор свойств подпрограмм

Также была поддержана работа параметров в Trik Studio, в режиме интерпретации. Теперь появилась возможность создавать рекурсивные программы как следствие реализации передачи параметров.

Литература

[1] Литвинов Ю.В., Методы и средства разработки графических предметно-ориентированных языков. //Автореферат диссертации на соискание учёной степени кандидата технических наук
URL: http://www.math.spbu.ru/ru/mmeh/AspDok/pub/2016/litvinov.pdf
(дата обращения 13.05.2016)

[2] Литвинов Ю.В., Кириленко Я.А., TRIK Studio: среда обучения программированию с применением роботов // V Всероссийская конференция «Современное технологическое обучение: от компьютера к роботу» (сборник тезисов), СПб., ЗАО «Полиграфическое предприятие № 3», 2015, С. 5-7 (URL: https://robofinist.ru/uploads/2015/Thesis_2015.pdf

(дата обращения 13.05.2016))

[3] Нефёдов Е. А., Переиспользование кода в визуальных языках программирования. //Курсовая работа. (URL: https://raw.githubusercontent.com/wiki/qreal/qreal/345_Nefedov_report.pdf (дата обращения 13.05.2016))

[4] MetaEdit+ (URL: http://www.metacase.com/) (дата обращения 13.05.2016))

[5] Sirius (URL: https://eclipse.org/sirius/) (дата обращения 13.05.2016))

[6] Литвинов Ю.В., Плагины. //Welcome to QReal! URL: https://github.com/qreal/qreal/wiki/Плагины (дата обращения 16.12.2015)

[7] Птахина А.И., Эволюция языков при метамоделировании «на лету» в DSM-платформе QReal // Список-2014: Материалы всероссийской научной конференции по проблемам информатики. 23–25 апреля 2014 г. Санкт-Петербург. — СПб.: ВВМ, 2014, С. 18-27. (URL:http://spisok.math.spbu.ru/2014/txt/SPISOK-2014.pdf (дата обращения: 16.12.2015г))

[8] Мордвинов Д.А., Техническое введение в TRIK Studio — визуальную среду программирования роботов
URL:
https://github.com/qreal/articles/blob/master/2016-TRIKStudioTechnicalIntroduction/TRIKStudioTechnicalIntroduction.pdf
(дата обращения 21.05.2016)