Санкт-Петербургский государственный университет

Математическое обеспечение и администрирование информационных систем
Системное программирование

Платонова Мария Викторовна

Графическое моделирование на базе Eclipse в системах управления

Бакалаврская работа

Научный руководитель:
д. ф.-м. н., профессор Терехов А. Н.

Рецензент:
генеральный директор ООО "Рэйдикс" Федоров А. Р.

Санкт-Петербург
2019

SAINT-PETERSBURG STATE UNIVERSITY

Software and Administration of Information Systems
Department of Software Engineering

Mariia Platonova

Graphical modeling of control systems based on Eclipse technologies

Bachelor's Thesis

Scientific supervisor:
professor Andrey Terekhov

Reviewer:
CEO at Raidix LLC Andrey Fedorov

Saint-Petersburg
2019

Оглавление

Введение

Прогресс в сфере информационных технологий оказал огромное влияние на развитие экономики [13, 16]. Современные веяния являются ключевым фактором формирования цифровой среды, с помощью которой экономика переходит на новый цифровой уровень [14, 18].

Только пару лет как Россия встала на путь цифровизации [17, 19], а уже показывает хорошие темпы развития и даже имеет перспективы занять лидирующие позиции в рейтинге цифровых экономик мира [1]. Более того, в связи с увеличением зависимости всей экономики от использования необходимого программного обеспечения, встает вопрос о национальной безопасности и технологической независимости страны. В связи с этим, необходимо наличие собственных программных продуктов, конкурентоспособных на международном уровне [17].

В рамках утвержденной программы цифровой экономики представлено множество различных задач. Одной из них является автоматизация производства [12]. В данной области выделяют различные направления деятельности, к примеру, представление производственных процессов в графическом виде, что подразумевает описание некой совокупности действий в виде одной или нескольких диаграмм. Для представления процессов в нужном виде используется международный стандарт BPMN (Business Process Model and Notation) [6].

Такие диаграммы доступны для понимания всеми заинтересованными лицами, и кроме того, упрощают дальнейшую работу с процессами, так как именно визуализация способствует улучшению понимания и осмысления необходимых сведений [15, 10].

Применение визуализации процессов не ограничивается лишь удобным и простым способом восприятия информации, потому как зачастую процессы подразумевают возможность генерации исполняемой программы.

Соответственно, создание решения для визуализации процессов с последующей генераций кода является актуальной задачей.

Разным типам процессов соответствуют разные генераторы кода,

поэтому также важно учитывать особенности процессов и требования к совместимости кода разных систем. Вдобавок, генератор может накладывать некие ограничения на диаграмму, например, на создание, модификацию или удаление ее объектов или на переходы между ее состояниями.

Особенности операционных систем могут значительно повлиять на разработку, так как не все системы могут обладать необходимыми свойствами. Реализация отдельно под каждую систему сильно замедляет и усложняет процесс. Таким образом, использование среды разработки, совместимой с многими системами, не только упростит реализацию, но и расширит возможности применения.

Подводя итог, данная работы основана на разработке кроссплатформенного приложения для визуализации производственных процессов и генерации соответствующего им кода.

1. Постановка задачи

Целью данной работы является разработка plug-in приложения, которое на основе визуализации процессов управления некоторой системы, генерирует код в необходимом виде.

Для достижения этой цели были поставлены следующие задачи.

1. Сделать обзор предметной области.
2. Сделать обзор существующих решений.
3. Реализовать графический редактор.
4. Разработать логическую модель, соответствующую графической.
5. Реализовать генератор кода для управления гибким автоматическим производством (ГАП).

2. Обзор

2.1. BPMN

Стандарт Business Process Model and Notation (BPMN) [6] — это система условных обозначений, предназначенных для моделирования бизнес-процессов.

Каждый процесс может быть описан одной или несколькими диаграммами, состоящими из графических элементов.

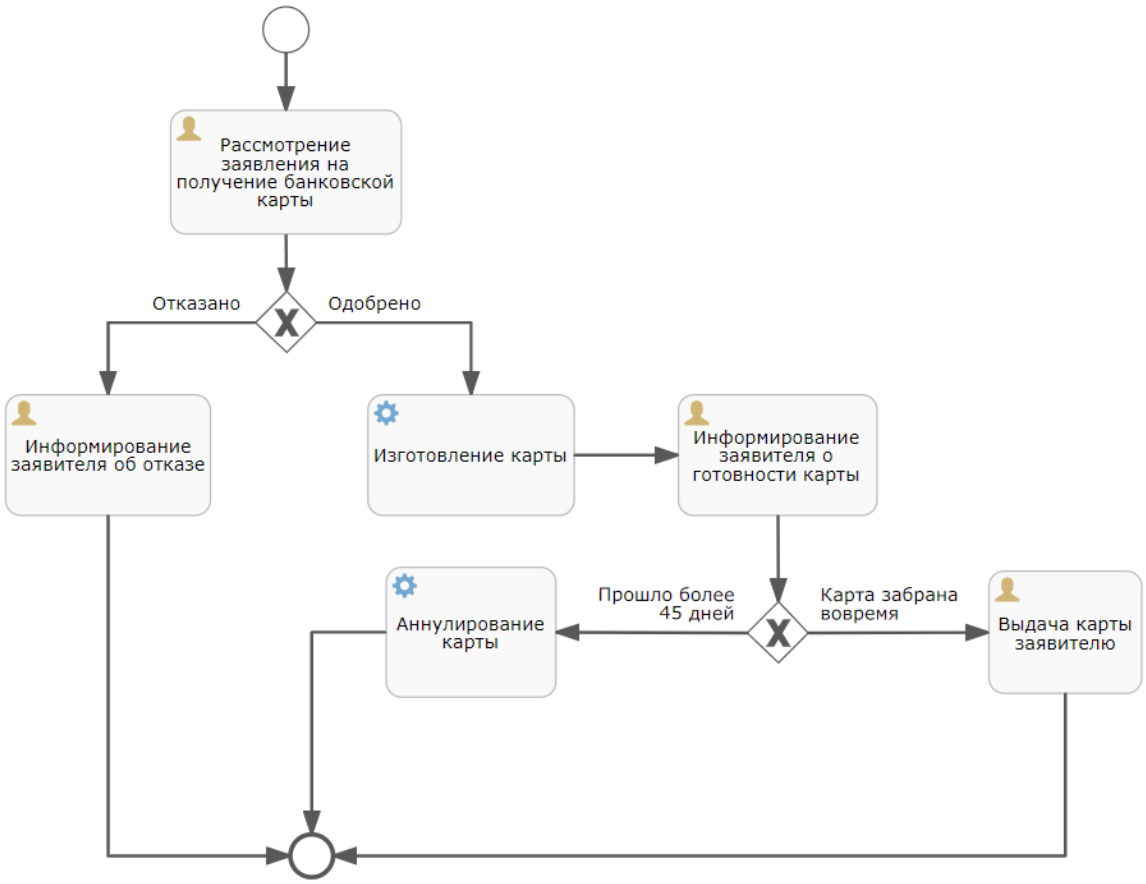

Рис. 1: Пример системы управления на языке BPMN

Выделяют основные категории элементов:

1. объекты потока (Flow Objects);
2. данные (Data);

3. зоны ответственности (Swimlanes);

4. соединяющие элементы (Connecting Objects);

5. артефакты (Artifacts).

С помощью BPMN можно реализовывать самые различные процессы, например, на рис.1 представлен процесс управления на примере выдачи банковской карты.

2.2. ГАП

Последние годы на фабриках становится все больше станков с ЧПУ [11], что значительно повышает автоматизацию производства и позволяет гарантировать качество результата.

Полная автоматизация таких процессов возможна с использованием в дополнение к станкам с ЧПУ складских роботов (далее робот), например, робот-тележка, робот-манипулятор и другие. Их основная задача — это передача необходимых материалов и инструментов от одного станка к другому.

Система, состоящая из таких станков и роботов, образует гибкое автоматическое производство.

Возникает вопрос о том, как программировать и контролировать работу такой системы. Важно не только задать всю последовательность действий, но и описать реакцию на сбойные ситуации, так как, например, из-за случайного мусора или неровностей поверхности робот может сбиться с пути или упасть, что способно вызвать нарушение корректной работы системы или даже поломку перевозимого груза.

Работа ГАП, включая работу станков с ЧПУ и роботов, может быть описана в виде BPMN диаграмм, например, рис.2. Кроме того, таким способом удобно задавать в том числе и ветки отказов роботов, что позволит быстро реагировать на проблемы и восстанавливать работу производства.

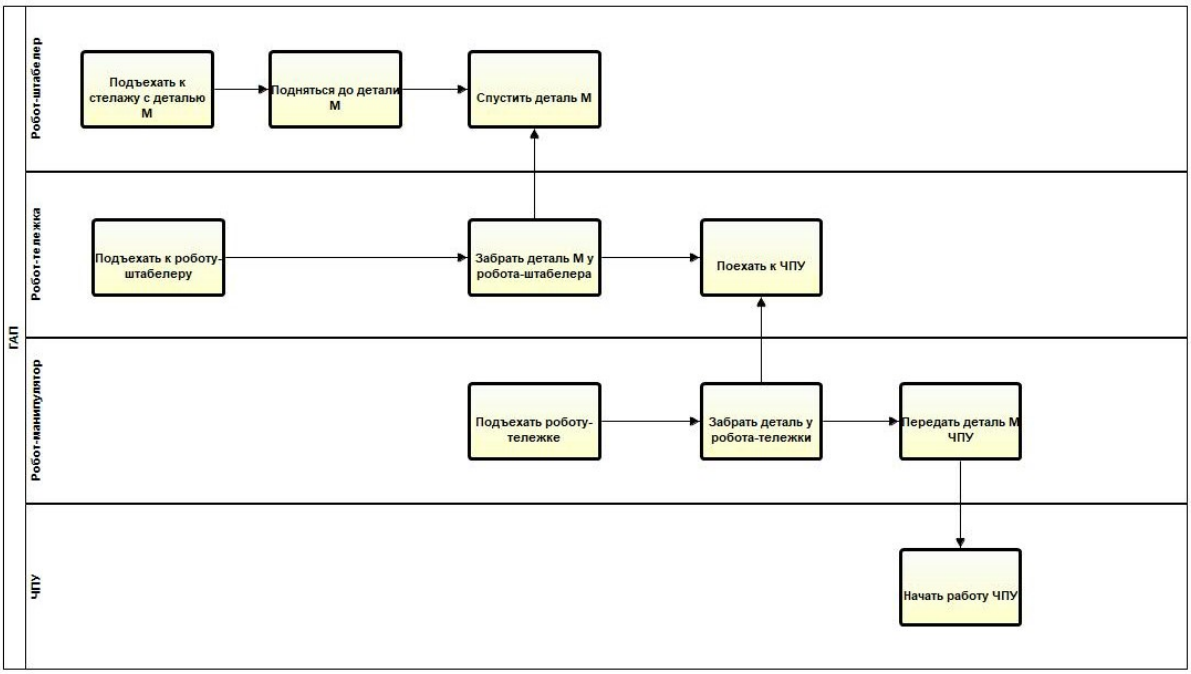

Рис. 2: Пример ГАП на языке BPMN

2.3. Среда разработки

Основываясь на предыдущем опыте можно заметить, что использование мало распространенного, сложного или имеющего много ограничений инструментария может привести к проблемам с развитием и поддержкой продукта, а также при сравнении его с аналогами.

В данной работе используется популярная среда разработки Eclipse — бесплатная, кроссплатформенная система, постоянно развивающаяся и, кроме того, обладающая большим сообществом разработчиков, что облегчает решение возникающих проблем.

2.4. Графический редактор

Основной задачей графического редактора в данной работе является построение диаграмм с использованием нотации BPMN, при этом подразумевается, что возможности редактора не ограничиваются лишь визуальным представлением процессов, с его помощью также можно обеспечить хранение логической модели, описываемой в виде графа.

Полностью реализовать графический редактор отнюдь не просто, а так как существуют различные варианты готовых решений, нам показалось разумным выбрать подходящую версию существующего редактора и доработать ее.

Существует достаточно примеров реализации BPMN редакторов как Eclipse plug-in приложений.

Среди них можно выделить BPMN2 Modeler [5], Activiti-Designer [4] и Papyrus BPMN [8]. Они являются одними из самых используемых и часто обновляемых. Более того, эти приложения находятся под лицензией, позволяющей использовать их для собственной разработки, но только лицензия Activiti-Designer позволяет не предоставлять открытый доступ к измененному коду. В связи с этим, для данной работы был выбран именно Activiti-Designer.

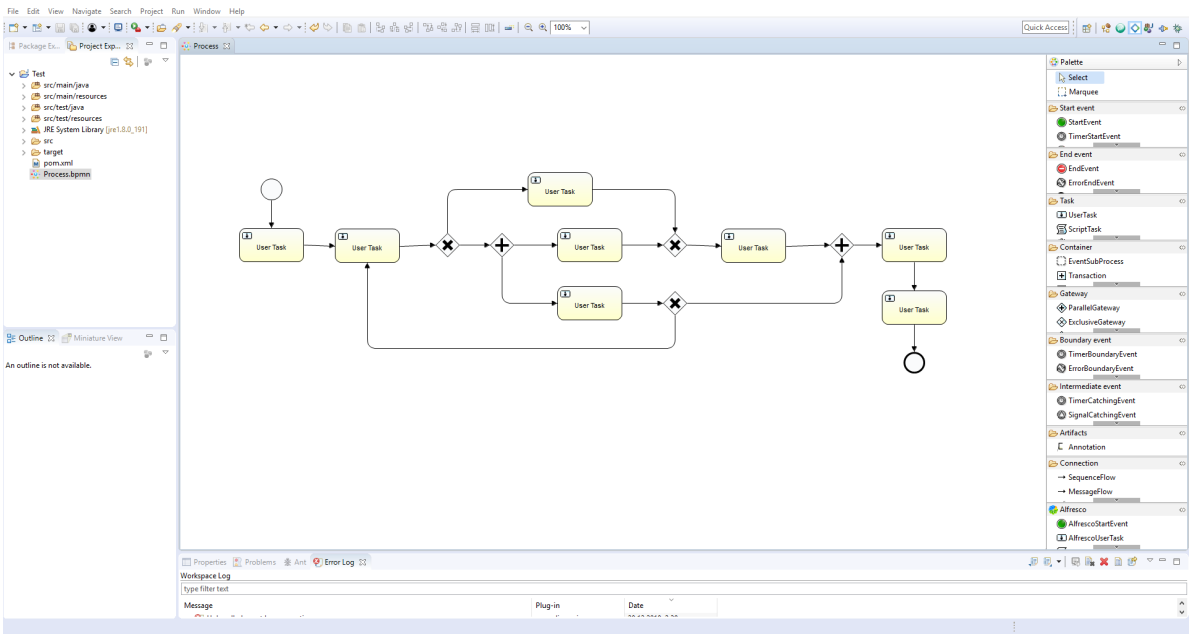

Рис. 3: Activiti-Designer

Приложение является частью системы Activiti [3], предназначенной для моделирования бизнес-процессов в виде BPMN диаграмм, с открытым исходным кодом, распространяемой в рамках лицензии Apache.

Activiti-Designer (далее Activiti) обладает удобным графическим интерфейсом и содержит все необходимые для работы с BPMN средства.

Activiti состоит из холста и панели элементов, на которой представлено 42 элемента, разделенных на 9 категорий: Start Event, End Event, Task, Container, Gateway, Boundary Event, Intermediate Event, Artifacts и Connection. Более того, редактор позволяет выравнивать элементы на диаграммах, а также сохранять полученные диаграммы как изображения. На рис.3 представлен интерфейс плагина и пример описанного с его помощью процесса.

Заметим, что Activiti обладает не всеми желаемыми и необходимыми функциями, поэтому было принято решение о расширении возможностей редактора.

2.5. Существующие решения

QReal:BP [9]

QReal:BP — это metaCASE технология для инструментария QReal.

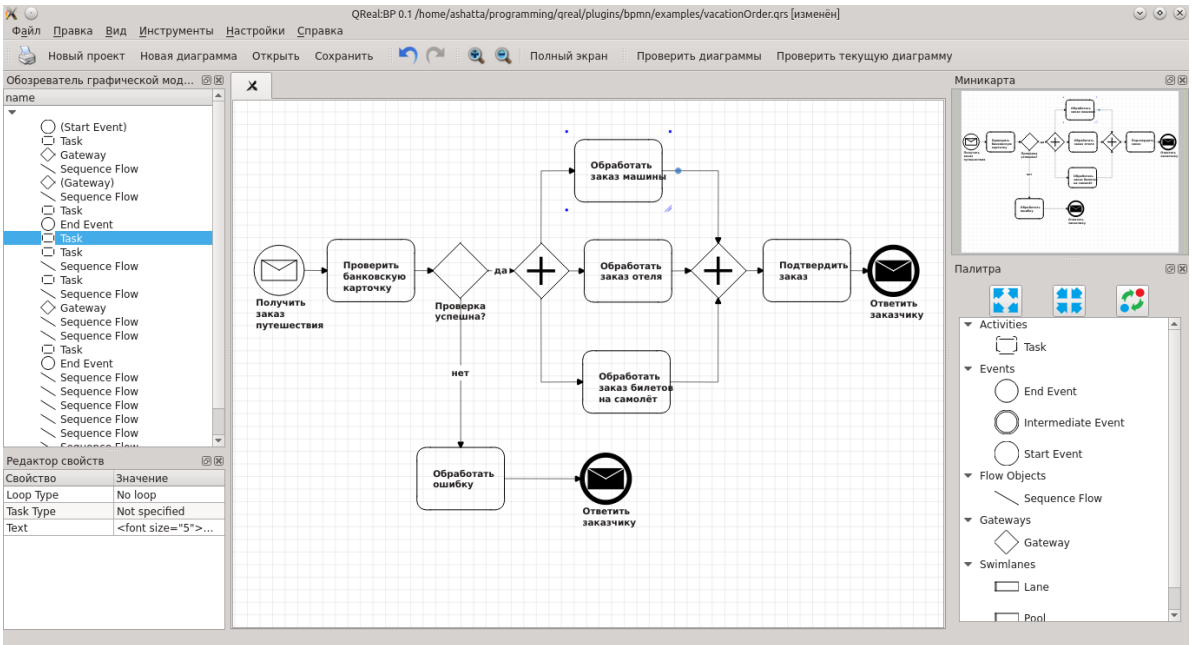

Рис. 4: QReal:BP

MetaCASE подразумевает гибкое описание новой технологии, с последующей генерацией кода. Этот способ разумно использовать при создании множества технологий, а не одной.

QReal:BP позволяет создавать и редактировать сложные BPMN-диаграммы. Более того, по диаграмме можно автоматически сгенерировать исполняемую программу, состоящую из вызовов подпрограмм для каждой операции.

Одной из основных причин, почему это средство не стало широко использоваться, является малая распространенность технологии QReal, что вызывало сложности при сопровождении продукта.

Общий вид интерфейса QReal:BP приведен на рис.4.

Acceleo [2]

Acceleo — это генератор кода с открытым исходным кодом на базе Eclipse, представляющий собой plug-in приложение, реализующее стандарт MOFM2T (MOF Model to Text Transformation Language) [7]. Хотя сам стандарт не обновлялся с 2008 года, Acceleo успешно используется во многих проектах.

Acceleo, используемое совместно с графическим редактором BPMN, может использоваться для генерации кода по BPMN диаграммам.

Однако Acceleo распространяется в рамках лицензии Eclipse Public License, что подразумевает возможность доступа третьим лицам к измененному коду.

В данном обзоре рассматриваются технологии, которые позволяют каким-либо образом достигнуть поставленную цель. И подводя итог, учитывая такие факторы как лицензионные ограничения или несоответствие предпочитаемой среде, с помощью существующих технологий заданная цель не достигается.

3. Доработка графического редактора

Рассмотрим подробнее основные функции, добавленные в графический редактор Activiti.

Быстрый доступ

Важной доработкой приложения стало добавление функции ”быстрого доступа”, что позволяет отображать на холсте элементы, не перемещая их из панели элементов.

Пользователю предоставляется набор комбинаций ”CTRL+M CTRL +N”, где M — это номер категории, где расположен элемент, а N — это номер элемента внутри категории. Таким образом, применяя какую-либо комбинацию клавиш из данного набора, соответствующий элемент добавляется на диаграмму в то место, где находится курсор мыши.

Видимость элементов

Функция ”видимость элементов” также является значимым расширением возможностей редактора. Она позволяет пользователю скрывать элементы контейнера, что применимо в случае громоздких диаграмм и высокого уровня вложенности.

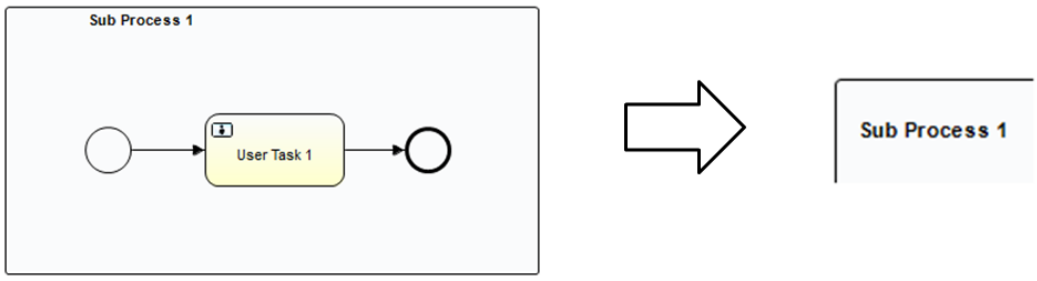

Рис. 5: Пример использования функции ”видимость элементов”

Пользователю предоставляется две функции: minimize и maximize, одна из которых добавляется в контекстное меню контейнера, в зависи-

мости от того, скрыты его элементы или нет. Таким образом, применяя функцию minimize, все элементы контейнера скрываются, а сам контейнер представляется в виде небольшого прямоугольника, сохраняя при этом все внешние связи, которые присутствовали в его полноразмерном варианте, а при применении функции maximize контейнер принимает свой изначальный вид.

На рис.5 представлен пример использования данной функции.

4. Описание логической модели

Визуальное, то есть создаваемое пользователем представление бизнес-процесса, принято называть графической моделью, а внутреннее представление в виде, например, графа специального вида, необходимое для генерации отчетов, исполняемых программ и т.д., — логической моделью.

В данной работе одной из основных задач является выбор представления логической модели. Удобный и быстрый доступ к данным значительно упрощает последующие действия, связанные с генерацией исполняемой программы и для разбора и анализа рассматриваемой модели.

Основной причиной хранения информации в виде ориентированных графов (далее граф) являлась простота преобразования диаграмм пользователей в графы и, кроме того, на их ребрах и вершинах удобно хранить какую-либо дополнительную информацию.

Вдобавок, из-за отсутствия каких-либо ограничений на диаграммы, возрастает сложность работы с графами. Например, множество циклов и петель мешают определять начальные и конечные вершины описанного процесса, также это усложняет анализ переходов между элементами, поэтому мы приняли некоторые ограничения, направленные на повышение структурности моделей.

Учитывая особенности требуемого графа, была выбрана определенная модель его представления. Рассмотрим подробнее полученную структуру.

Базовый тип

Каждый элемент графа имеет уникальный идентификатор (далее id) и дополнительную информацию об элементе (далее info). Id задается при создании элемента и записывается в info. Кроме того, в info хранится имя, тип и множество других компонентов элемента. Невозможно не заметить важность этих данных, и именно поэтому id и info легли в основу базового класса — BaseItem.

Типы данных в графе

Граф состоит из элементов трех типов: Container — контейнер, Node — узел, Edge — ребро. Соответствующие им классы предоставляют всю необходимую информацию об элементах и их связях с другими элементами графа. Заметим, что хранение всей информации может быть избыточным и неэффективным, поэтому в качестве связей между элементами используется только id.

Важно пояснить, что такое контейнер и для чего он нужен. Контейнер предназначен для хранения любых элементов графа, в нем может лежать сколько угодно других контейнеров. Контейнер может также использоваться в качестве вершины.

Каждый элемент диаграммы содержит информацию о контейнере, в котором он находится (далее родительский контейнер), причем ребро может иметь два таких контейнера, поскольку его начало и конец могут находиться в разных контейнерах. Базовым контейнером является холст диаграммы, что означает наличие родительского контейнера для любого элемента графа.

Представление графа в памяти

За представление графа отвечают отдельные классы — NodeUtil, EdgeUtil и ContainerUtil. Эти классы является статическими, что позволяет не создавать отдельный экземпляр при каждом применении, а использовать один и тот же для всех обращений.

Каждый класс отвечает за актуальный список соответствующих элементов. Модификация классов происходит при любых изменениях диаграммы, например, при добавлении, изменении или удалении элемента.

5. Реализация генератора кода

Исполняемые программы разных типов производственных процессов могут существенно различаться по структуре и конечному результату. В связи с этим, генератор кода реализуется только для процессов, соответствующим ГАП системе. Каждый такой процесс представляется в виде диаграммы, по которой строится логическая модель (далее модель) с последующей генерацией исполняемого кода.

Основным этапом, предшествующим кодогенерации, является анализ и разбор модели. Этот промежуточный этап необходим для выявления начальных и конечных узлов, циклов, петель и т.д. Данные, полученные на этом этапе, являются входными для генератора кода, поэтому безусловно полезно иметь возможность их визуализировать, так как благодаря этому могут быть обнаружены различные неточности и ошибки, упущенные во время построения диаграммы.

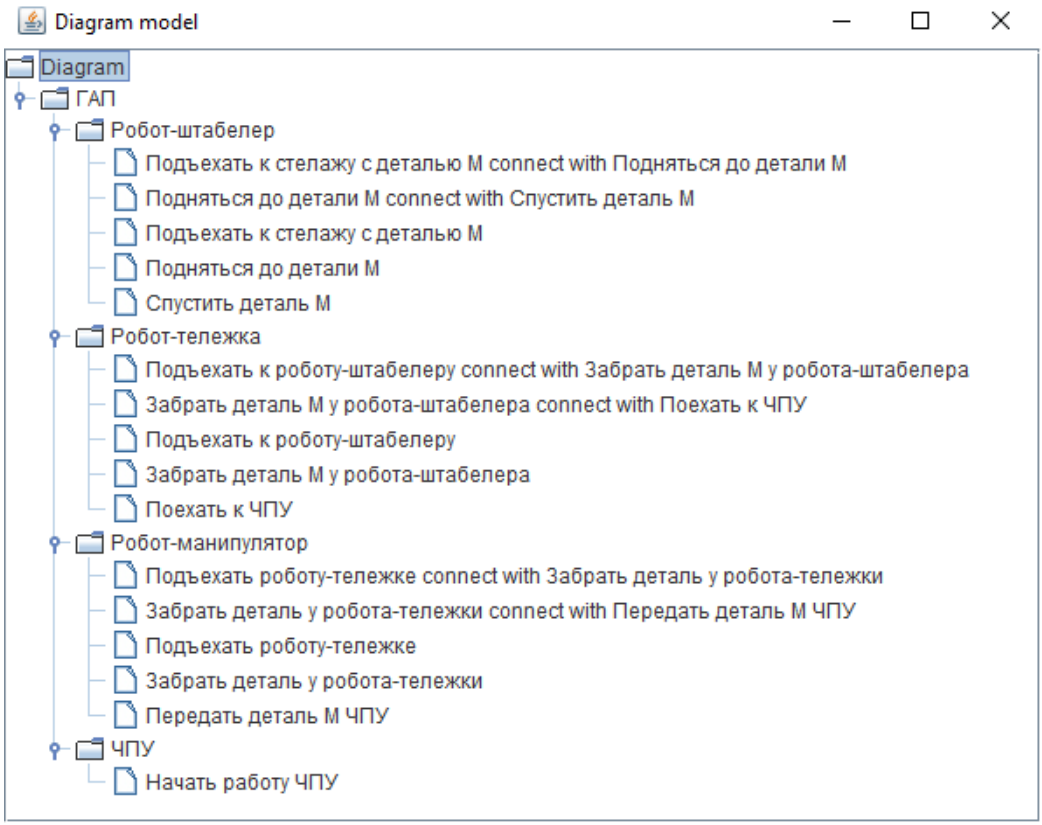

Рис. 6: Пример визуализации модели с помощью функции ”Show model”

Пользователю предоставляется функция showModel, которая вызывается нажатием кнопки ”Show model”, расположенной в основном меню. Заметим, что эту функцию можно применить для любой диаграммы, даже если дальнейшая кодогенерация невозможна. На рис.6 приведен пример визуализации модели диаграммы с рис.2 с использованием этой функции.

Таким образом, применяя эту функцию для какой-либо диаграммы, открывается отдельное окно, где все ее элементы отображаются в конкретном виде. Важно напомнить, что ребенок узла — это узел, являющийся конечным для какого-либо ребра, выходящего из него, и что контейнер может также выступать в качестве узла, то есть может как иметь детей, так и быть ребенком.

Для каждого контейнера, включая базовый, в окне отображаются все соответствующие ему элементы: узлы и их дети. Связи между элементами представляются в определенном виде: элемент1 connect with элемент2.

Имея такое представление логической модели, задача генерации кода становится простой – линейный обход с генерацией нужного фрагмента кода по каждому действию.

6. Заключение

В рамках данной работы было реализовано plug-in приложение на базе Eclipse, позволяющее генерировать код в требуемом виде, основываясь на визуализации процессов управления.

Более детальное описание полученных результатов представлено ниже.

1. Произведен обзор предметной области.
2. Произведен обзор существующих решений.
3. Доработан графический редактор.
4. Разработана логическая модель.
5. Реализована визуализация модели, по которой генерируется код для управления ГАП.

Список литературы

[1] *Digital Planet 2017: How Competitiveness and Trust in Digital Economies Vary Across the World* / The Fletcher School, Tufts University ; Executor: Chakravorti Bhaskar, Chaturvedi Ravi S. : 2017. — Access mode: https://sites.tufts.edu/digitalplanet/files/2017/05/Digital_Planet_2017_FINAL.pdf.

[2] *Acceleo.* — Access mode: https://www.eclipse.org/acceleo/documentation/.

[3] *Activiti & Activiti Cloud* / Activiti Team : 2017. — Access mode: https://activiti.gitbook.io/activiti-7-developers-guide.

[4] *Activiti-Designer* / Activiti Team. — Access mode: https://github.com/Activiti/Activiti-Designer.

[5] *BPMN2 Modeler* / Red Hat. — Access mode: https://www.eclipse.org/bpmn2-modeler/.

[6] *Business Process Model And Notation* / Object Management Group (OMG) : 2011. — Access mode: https://www.omg.org/spec/BPMN/2.0/.

[7] *MOF Model to Text Transformation Language* / Object Management Group (OMG) : 2008. — Access mode: https://www.omg.org/spec/MOFM2T/1.0/.

[8] *Papyrus BPMN* / Papyrus. — Access mode: https://marketplace.eclipse.org/content/papyrus-bpmn.

[9] *QReal:BP* / QReal : 2013. — Access mode: http://qreal.ru/static.php?link=QRealBP.

[10] А.А. Афанасьев. *Технология визуализации данных как инструмент совершенствования процесса поддержки принятия решений* // Инженерный вестник Дона. — 2014. — № 4-1(31). —

Режим доступа: http://ivdon.ru/uploads/article/pdf/IVD_66_Afanasyev.pdf_da5aca6ae5.pdf.

[11] А.А. Ловыгин, Л.В Теверовский. *Основы числового программного управления* // Современный станок с ЧПУ и CAD/CAM система. — 2015. — С. 10–24. — Режим доступа: http://planetacam.ru/college/learn/.

[12] В.А. Плотников. *Цифровизация производства: теоретическая сущность и перспективы развития в российской экономике* // Известия СПбГЭУ. — 2018. — № 4(112). — С. 16–24. — Режим доступа: https://elibrary.ru/item.asp?id=35304372.

[13] В.Е. Удовик, А.В. Селютин. *Информационная революция и становление информационного общества* // Известия Московского государственного технического университета МАМИ. — 2011. — № 2(12). — С. 263–267. — Режим доступа: https://elibrary.ru/item.asp?id=17100420.

[14] Е.А. Нестеренко, А.С. Козлова. *Направления развития цифровой экономики и цифровых технологий в России* // Экономическая безопасность и качество. — 2018. — № 2(31). — С. 9–14. — Режим доступа: http://www.seun.ru/content/nauka/5/1/doc/Economical_security_2_31_2018.pdf.

[15] Е.В. Полякова. *Применение способов и методов визуального мышления в современном образовании* // Известия ЮФУ. Технические науки. — 2012. — № 10(135). — С. 120–124. — Режим доступа: http://izv-tn.tti.sfedu.ru/wp-content/uploads/2012/10/19.pdf.

[16] И.И. Лонский. *Информатизация и эволюция общества* // Перспективы Науки и Образования. — 2015. — № 2(14). — С. 29–35. — Режим доступа: https://pnojournal.files.wordpress.com/2015/01/pdf_150204.pdf.

[17] Распоряжение Правительства Российской Федерации от 28 июля 2017г. № 1632-р : *Цифровая экономика Российской Федерации.* — Режим доступа: http://static.government.ru/media/files/9gFM4FHj4PsB79I5v7yLVuPgu4bvR7M0.pdf.

[18] Т.М. Хусяинов. *Информационная революция и трансформация занятости* // Наука. Мысль. — 2017. — № 1-3. — С. 76–79. — Режим доступа: http://wwenews.esrae.ru/pdf/2017/%201-3/673.pdf.

[19] Указ Президента Российской Федерации от 09 мая 2017г. № 203 : *О Стратегии развития информационного общества в Российской Федерации на 2017 – 2030 годы.* — Режим доступа: http://static.kremlin.ru/media/acts/files/0001201705100002.pdf.