**Проект ООТРПО**

**Дайджест по второму этапу**

**Участники проекта**: гр. 5303 Допира В, Бочкарев И, Ильянов В., гр. 5304 Павлов Д

**Выбранный контейнер**: многодольный граф

В дайджесте представлены:

* План итерации.
* Запросы заинтересованных лиц.
* Прототип интерфейса.
* Архитектурный план:
  + Диаграмма вариантов использования
  + Диаграмма последовательностей
  + Диаграмма состояний
  + Диаграмма компонентов
  + Диаграмма классов.
* Ручное тестирование приложения.

Система автоматизированной генерации расписания

План итераций

Версия 1.0

**История Ревизий**

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Описание** | **Автор** |
| 30.04.2020 | 1.0 | Первоначальная версия | Допира Валерия |

**1. Введение**

**1.1 Цель**

Цель документа – описание плана итерации проекта. В данной итерации проводится уточнение требований к ПО.

**1.2 Определения и сокращения**

Представлены в артефакте Глоссарий.

**2. План**

График сдачи каждой задачи проекта представлен ниже (см. табл.1).

Таблица 1 – График проекта

|  |  |  |  |
| --- | --- | --- | --- |
| ***Фаза*** | ***Задача*** | ***Окончание работы*** | ***Исполнители*** |
| Уточнение | План итерации | 30.04.2020 | Допира Валерия |
| Запросы заинтересованных лиц | 04.05.2020 | Бочкарев Иван |
| Прототип интерфейса | 04.05.2020 | Павлов Данила, Ильянов Вячеслав |
| Архитектурный план | 04.05.2020 | Допира Валерия, Бочкарев Иван, Павлов Данила, Ильянов Вячеслав |
| Диаграмма вариантов использования | 14.05.2020 | Бочкарев Иван, Павлов Данила |
| Диаграмма классов | 14.05.2020 | Допира Валерия |
| Диаграмма компонентов | 14.05.2020 | Бочкарев Иван |
| Диаграмма последовательностей | 14.05.2020 | Бочкарев Иван |
| Диаграмма состояний | 14.05.2020 | Ильянов Вячеслав |
| Ручное тестирование приложения | 14.05.2020 | Допира Валерия |

**3. Нагрузка исполнителей**

Количество часов в неделю, которые исполнители тратят на проект распределено равномерно между исполнителями на протяжении всего проекта. Роли распределены в соответствии с планом разработки программного обеспечения. Задачи для каждого участника:

* Допира Валерия

Таблица 2 – Нагрузка на исполнителя 1

|  |  |  |  |
| --- | --- | --- | --- |
| ***Задача*** | ***Окончание работы*** | ***Затраченное время*** | ***Исполнители*** |
| План итерации | 30.04.2020 | 1 день | Допира Валерия |
| Архитектурный план | 04.05.2020 | 7 дней | Допира Валерия, Бочкарев Иван, Павлов Данила, Ильянов Вячеслав |
| Диаграмма классов | 14.05.2020 | 2 дня | Допира Валерия |
| Ручное тестирование приложения | 14.05.2020 | 5 дней | Допира Валерия |

* Бочкарев Иван

Таблица 3 – Нагрузка на исполнителя 2

|  |  |  |  |
| --- | --- | --- | --- |
| ***Задача*** | ***Окончание работы*** | ***Затраченное время*** | ***Исполнители*** |
| Запросы заинтересованных лиц | 04.05.2020 | 2 дня | Бочкарев Иван |
| Архитектурный план | 04.05.2020 | 7 дней | Допира Валерия, Бочкарев Иван, Павлов Данила, Ильянов Вячеслав |
| Диаграмма вариантов использования | 14.05.2020 | 2 дня | Бочкарев Иван, Павлов Данила |
| Диаграмма компонентов | 14.05.2020 | 1 день | Бочкарев Иван |
| Диаграмма последовательностей | 14.05.2020 | 1 день | Бочкарев Иван |

* Ильянов Вячеслав

Таблица 4 – Нагрузка на исполнителя 3

|  |  |  |  |
| --- | --- | --- | --- |
| ***Задача*** | ***Окончание работы*** | ***Затраченное время*** | ***Исполнители*** |
| Прототип интерфейса | 04.05.2020 | 1 день | Павлов Данила, Ильянов Вячеслав |
| Архитектурный план | 04.05.2020 | 7 дней | Допира Валерия, Бочкарев Иван, Павлов Данила, Ильянов Вячеслав |
| Диаграмма состояний | 14.05.2020 | 1 день | Ильянов Вячеслав |

* Павлов Данила

Таблица 5 – Нагрузка на исполнителя 4

|  |  |  |  |
| --- | --- | --- | --- |
| ***Задача*** | ***Окончание работы*** | ***Затраченное время*** | ***Исполнители*** |
| Прототип интерфейса | 04.05.2020 | 1 день | Павлов Данила, Ильянов Вячеслав |
| Архитектурный план | 04.05.2020 | 7 дней | Допира Валерия, Бочкарев Иван, Павлов Данила, Ильянов Вячеслав |
| Диаграмма вариантов использования | 14.05.2020 | 2 дня | Бочкарев Иван, Павлов Данила |

Система автоматизированной генерации расписания

Запросы заинтересованных лиц

Версия 1.0

**История Ревизий**

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Описание** | **Автор** |
| 04.05.2020 | 1.0 | Первоначальная версия | Бочкарев Иван |

**1. Введение**

* 1. **Цель**

Целью является определение, фиксация и анализ запросов к разрабатываемому проекту от заинтересованных лиц.

* 1. **Контекст**

Документ содержит описание запросов заинтересованных лиц к различным аспектам разрабатываемого программного продукта, таких как описание проблем, потенциальных пользователей, возможностей, надежности, производительности и поддержки.

* 1. **Определения, акронимы и сокращения**

См. Глоссарий.

1. **Определение заинтересованных лиц**

В данном разделе описываются пользователи системы составления учебного расписания. Система подразумевает только один тип пользователей – это составители расписания.

**2.1 Состояние рынка**

Предоставляемый продукт является системой автоматизированного составления расписания.

Аналоги данного ПО на данный момент либо слабо развиты, либо предоставляют только косвенный функционал, к примеру, такой как простое представление расписания в электронном виде.

В сравнении с аналогами наш продукт имеет много схожих возможностей, а именно возможность постоянно пополнять данные по учебным дисциплинам, учебным группам, учебным помещениям, учебным неделям, времени начала лекций.

Но отличительной чертой, а также особенностью нашего продукта, является наличие уникального функционала по генерации оптимального расписания. В этом аспекте продукт сильно выигрывает.

**2.2 Описание заинтересованных лиц**

Таблица 6 – Описание заинтересованных лиц

|  |  |
| --- | --- |
| ***Заинтересованное лицо*** | ***Роль*** |
| IT Executive | Следит за ходом разработки проекта |
| Пользователи системы | В качестве пользователя системой выступает человек отвечающий за составление расписания. Пользователь имеет возможность добавлять и удалять элементы расписания (дисциплина, время, помещение, группы учащихся) |
| Заказчик | Представляет интересы образовательной организации, заинтересованной нашем продукте. |

**2.3 Описание пользователей**

Таблица 7 – Описание пользователей

|  |  |
| --- | --- |
| ***Имя*** | ***Описание*** |
| Пользователи системы | Редактирование и просмотр данных: учебные дисциплины, учебные группы, учебные помещения, учебные недели, время начала лекций, на основе которых генерируется расписание, которое можно просматривать в виде таблиц или многодольного графа |

**2.4 Окружение пользователя**

Для нормальной работы с ПО пользователю достаточно иметь обычный стационарный компьютер с установленной системой по генерации расписания.

Отличительной чертой является то, что системе не требуется выход в интернет, что означает программой можно пользоваться когда и где угодно.

**2.5 Ключевые потребности заинтересованных лиц**

Таблица 8 – Ключевые потребности

|  |  |  |  |
| --- | --- | --- | --- |
| ***Потребность*** | ***Приоритет*** | ***Текущее решение*** | ***Предлагаемое решение*** |
| Автоматизированное  составление расписания | Высокий | Многие пользователи вынуждены составлять расписание вручную | Система предлагает на основе загружаемых данных генерировать учебное расписание |

**3. Набор основных возможностей**

В таблице в этом разделе указаны основные возможности системы генерирования расписания с точки зрения преимуществ и возможностей.

Таблица 9 – Возможности

|  |  |
| --- | --- |
| ***Преимущество для пользователей*** | ***Поддерживающие функции*** |
| Представление расписания в виде электронной таблицы | Система выводит составленное расписание в виде электронной таблицы |
| Представление расписания в виде многодольного графа | Система выводит составленное расписание в виде многодольного графа |
| Фильтрация расписания, представленное в виде многодольного графа | Система позволяет отфильтровать многодольный граф по: учебной дисциплине, времени начала занятий, учебным помещениям, учебным группам |
| Сохранение расписания в файл и загрузка расписания из файла | Система позволяет сохранять исходные и выходные данные в файл, который потом можно будет загрузить при новом сеансе работы с системой. |
| Легкость освоения продукта | Отсутствие авторизации и интуитивно понятный интерфейс |

Система автоматизированной генерации расписания

Прототип интерфейса

Версия 1.0

**История Ревизий**

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Описание** | **Автор** |
| 04.05.2020 | 1.0 | Первоначальная версия | Павлов Данила, Ильянов Вячеслав |

**1. Введение**

**1.1 Цель**

Целью является описание и визуальное представление пользовательского интерфейса приложения.

**2.** **Описание**

Пользовательский интерфейс — совокупность средств и методов, при помощи которых пользователь взаимодействует с различными устройствами и аппаратурой.Интерфейс воспринимают как то, что помогает пользователюполучить доступ к функционалу устройства.

**3. Составление расписания**

1. Пользователь загружает данные с файла, содержащий информацию по учебным дисциплинам, группам, помещениям, времени начале занятий
2. После загрузки данных пользователь нажимает на кнопку “Составить расписание”
3. Приложение по полученным данным выстраивает расписание в виде электронной таблице и многодольного графа
4. Далее пользователь может посмотреть на полученную электронную таблицу и граф с возможностью фильтрации, переключая вкладки “Расписание” и “Граф” соответственно

Также пользователь может:

1. Добавление в таблицу данных
   1. 1.Пользователь открывает таблицу, где расположена таблица с данными по учебным дисциплинам, учебным группам, учебным помещениям, время начала занятий.
   2. 2. Пользователь нажимает кнопку “добавить” с одной из колонок
   3. 3. Приложение вводит данные в таблицу
2. Удаление в таблице данных
   1. 1. Пользователь открывает таблицу, где расположена таблица с данными по учебным дисциплинам, учебным группам, учебным помещениям, время начала занятий.
   2. 2. Пользователь выбирает в таблице ячейку и нажимает кнопку “Удалить”
   3. 3. Приложение удаляет с таблицы данные, выбранные пользователем
3. Изменение в таблице данных
   1. 1. Пользователь открывает таблицу, где расположена таблица с данными по учебным дисциплинам, учебным группам, учебным помещениям, время начала занятий.
   2. 2. Пользователь находит нужную ячейку и нажимает на кнопку “Изменить”
   3. 3. Пользователь переписывает информацию в окне, которое вывело приложение
   4. 4. Приложение меняет информацию в ячейке
4. Некорректные данные в таблице данных
   1. 1.Пользователь вводит данные по учебным дисциплинам, учебным группам, учебным помещениям, времени начала занятий.
   2. 2. Программа выдает сообщение об ошибке и не дает занести данные
   3. 3. Пользователь продолжает работу

**4. Диаграмма варианта использования**

Проект поддерживает MVP (Model — View — Presenter) архитектуру.

*См. Архитектурный план*

Система автоматизированной генерации расписания

Архитектурный план

Версия 1.0

**История Ревизий**

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Описание** | **Автор** |
| 14.05.2020 | 1.0 | Первоначальная версия | Допира Валерия, Бочкарев Иван, Павлов Данила, Ильянов Вячеслав |

**1. Введение**

**1.1 Цель**

Обзор разрабатываемой системы и описание различных архитектурных решений, принятых при проектировании ПО.

**1.2 Определения, акронимы, сокращения**

*См. Глоссарий.*

**2. Размер и работа**

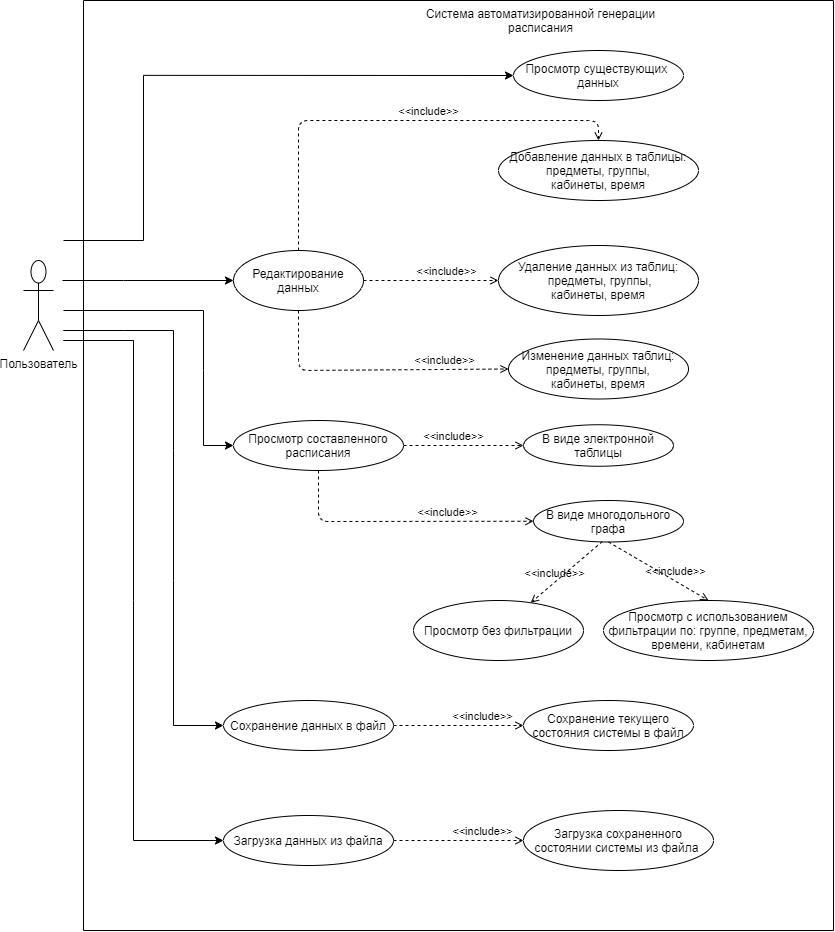
Размеры проекта ограничены учебным курсом. Временные рамки сужены, поэтому проект имеет достаточно большие размеры по сравнению со стандартными курсовыми проектами, но достаточно малые размеры относительно реальных проектов.

**3. Качество**

Целью разработки является освоение самого процесса разработки, поэтому к качеству предъявляются минимально возможные требования, с условием выполнения дополнительных функций, обеспечения надежности работы программы.

**4. Диаграмма варианта использования**

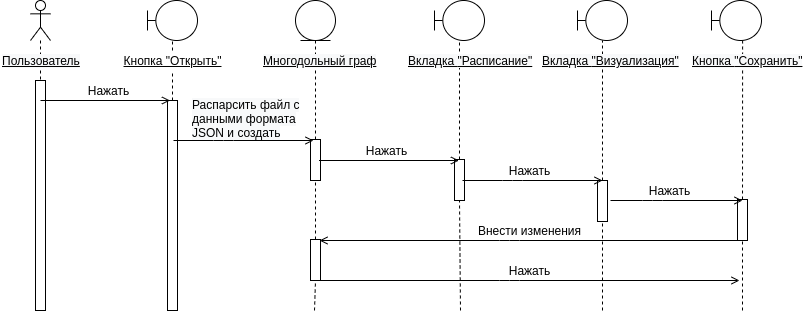
Пользователь: при первой работе с системой, необходимо добавить данные в таблицы: предметы, группы, кабинеты, время на основе которых система генерирует учебное расписание. Составленное расписание представляется в виде электронной таблицы и многодольного графа. Представление многодольного графа можно отфильтровать по: группе, предметам, времени, кабинетам. После окончания работы с системой, пользователь может сохранить полученные данные в файл, который можно использовать при следующем входе в систему (рис. 1).



*Рис. 1 Диаграмма вариантов использования*

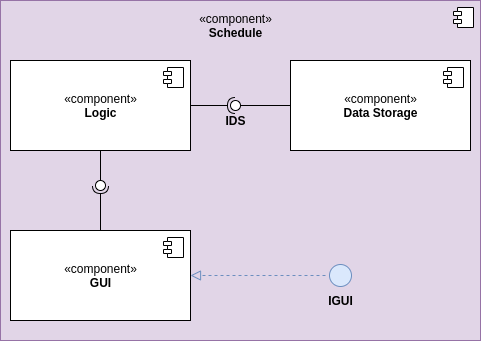
**5. Диаграмма последовательностей**

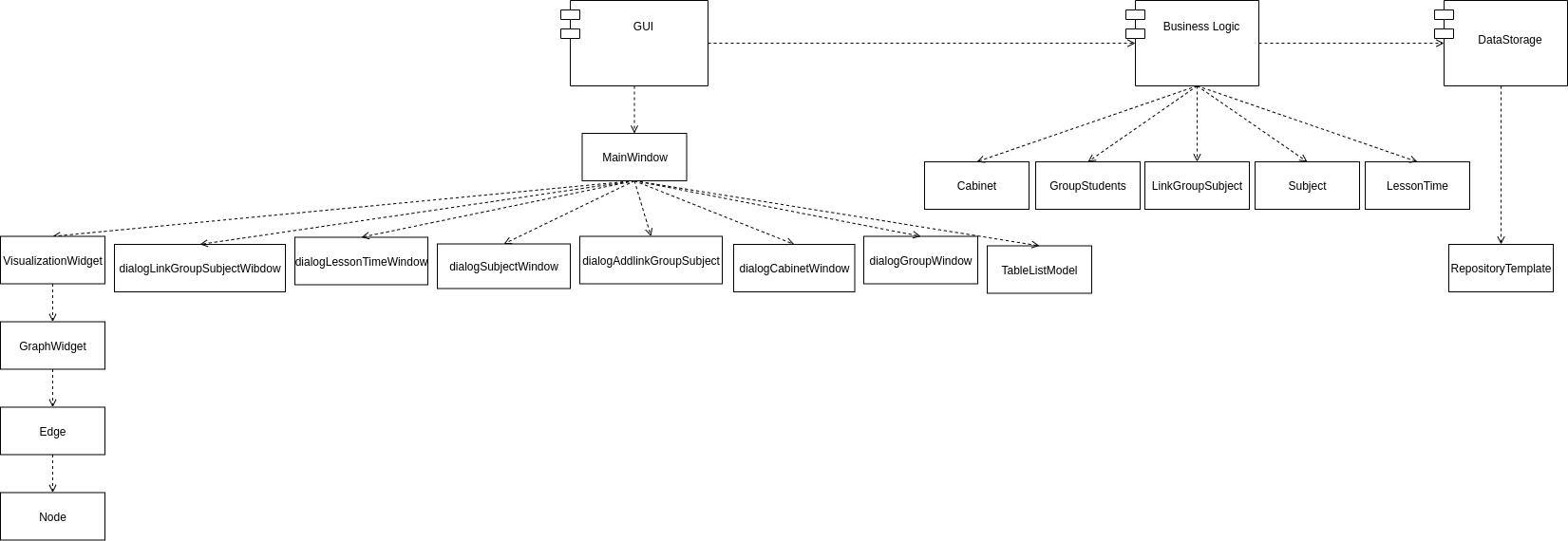
На диаграмме представлен сценарий использования программы. Все сообщения асинхронные и отображаются сплошной линией с открытой стрелкой, так как в программе поддерживается шаблон проектировании Модель-Вид-Контроллер.

*Рис. 2 Диаграмма последовательностей*

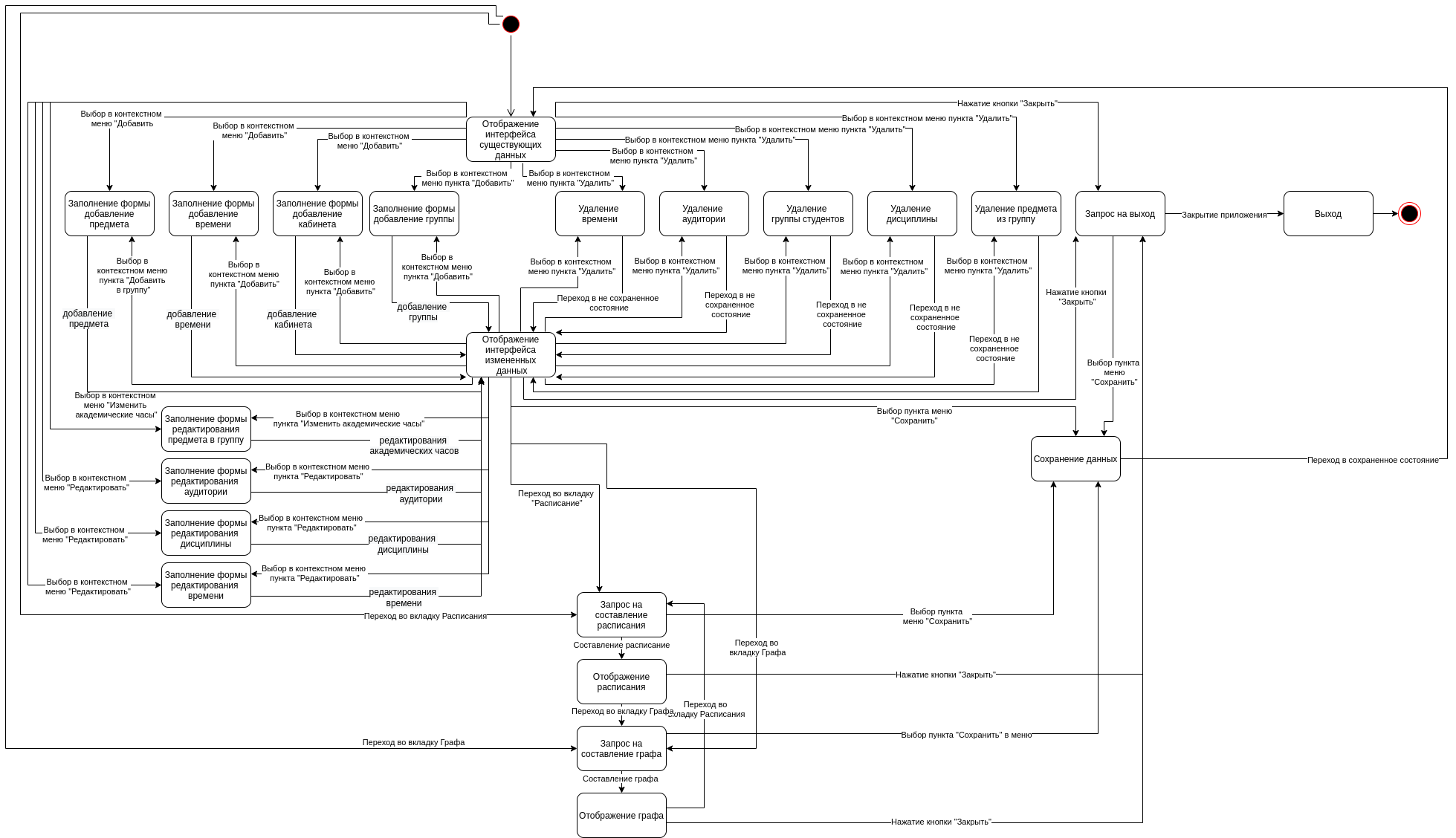
**6. Логическое представление**

Для представления архитектуры используется диаграмма компонентов. Диаграмма компонентов показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и т. п. Диаграмма компонентов представлена на рисунках 3, 4.

*Рис. 3 Диаграмма компонентов*

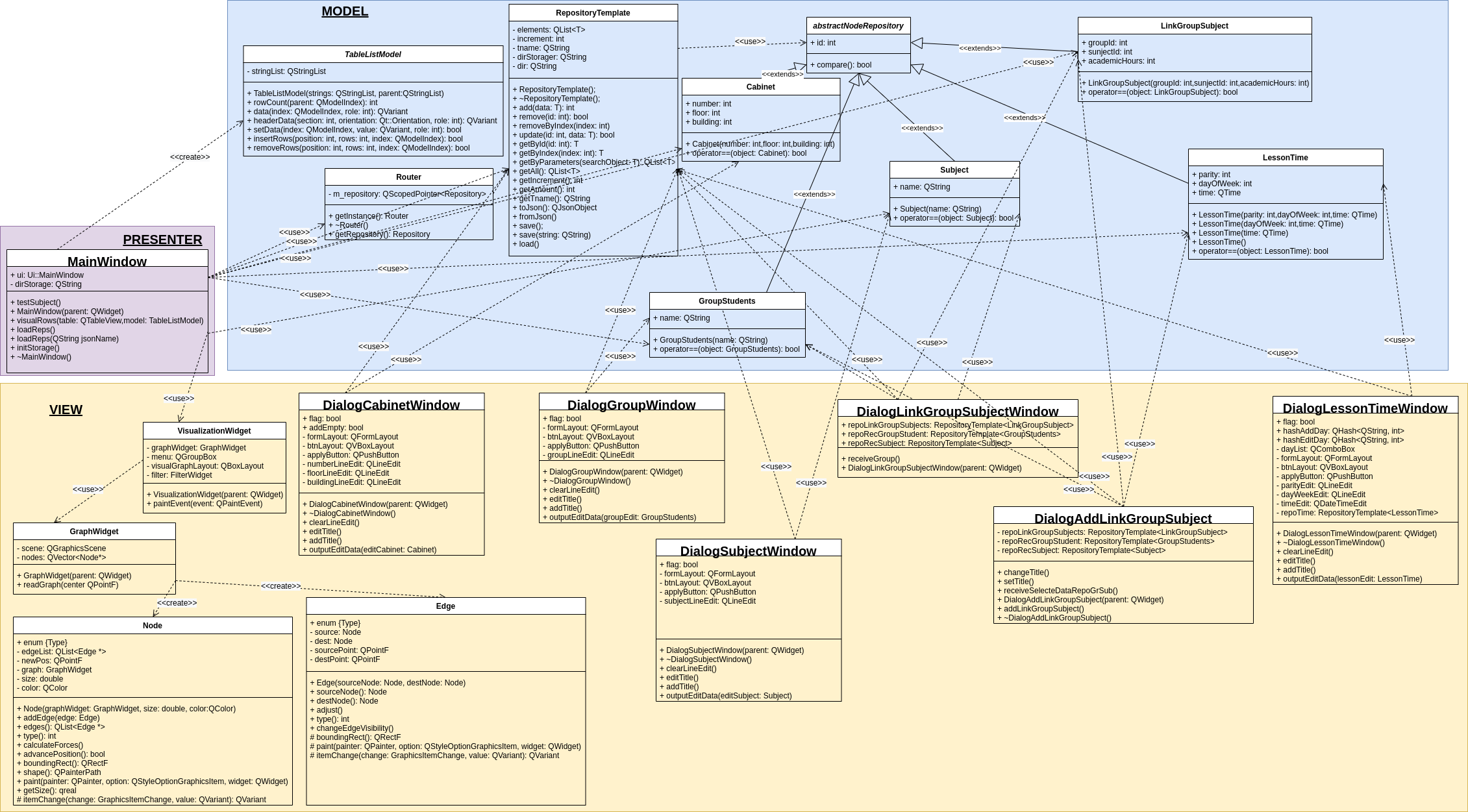
*Рис. 4 Диаграмма компонентов*

**7. Диаграмма состояний**

*Рис. 3 Диаграмма состояний*

**8. Представление структуры**

На этой схеме представлена часть диаграммы классов для отображения MVP архитектуры и связей между основными классами. Далее будут описаны и другие составляющие.

*Рис. 5 Диаграмма классов*

Выше использованы два вида связи:

1. Отношение обобщения — это наследование.

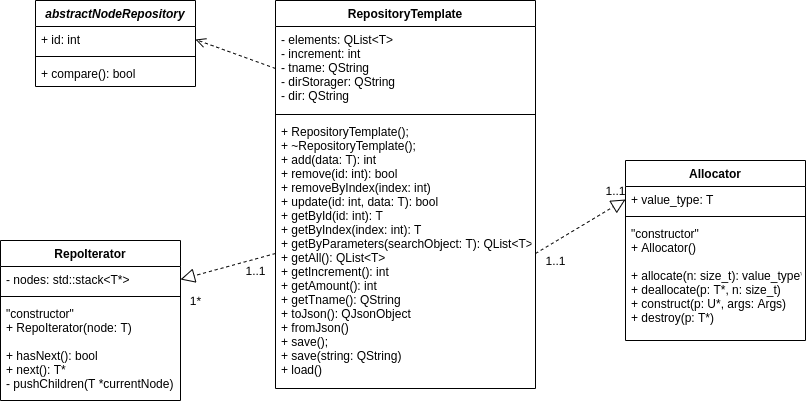
##### 2. Зависимость используется для организации диалога одного класса с другим. Классы либо используют методы и объекты другого, либо создаются внутри другого класса.

При разработке использовались следующие шаблоны проектирования.

Аллокатор является обёрткой, то есть, все экземпляры данного аллокатора являются взаимозаменяемыми, сравнительно равными и могут освободить память, выделенную любым другим экземпляром того же типа аллокатора.

Итератор относится к поведенческому типу. Представляет способ последовательного доступа к элементам множества, независимо от его внутренного устройства.

Диаграмма представлена на рисунке 2.

*Рис. 6 Аллокатор и итератор*

Система автоматизированной генерации расписания

Ручное тестирование приложения

Версия 1.0

**История Ревизий**

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Версия** | **Описание** | **Автор** |
| 14.05.2020 | 1.0 | Первоначальная версия | Допира Валерия |

**1. Введение**

**1.1 Цель**

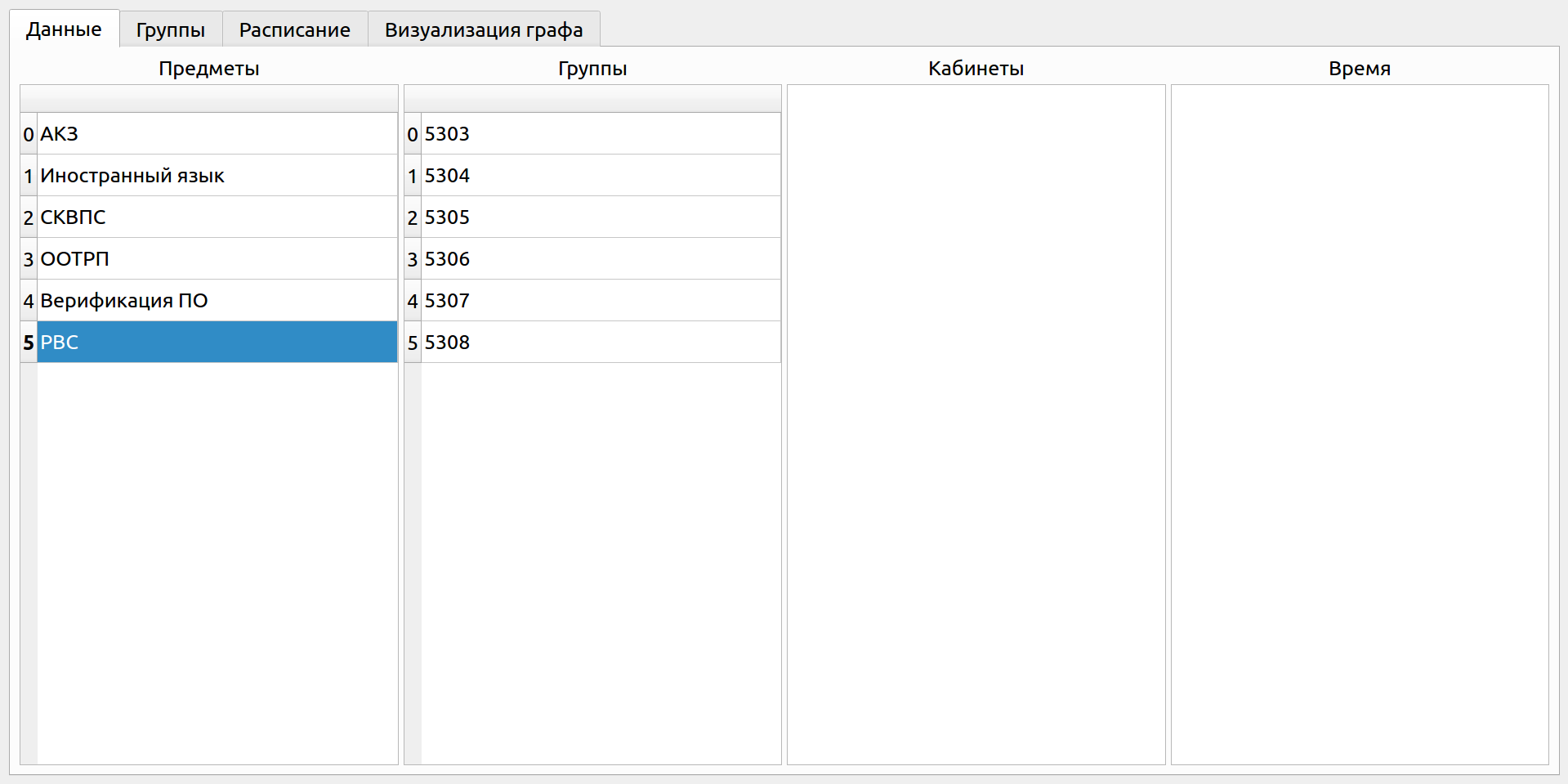
Провести ручное тестирование и демонстрацию работы разработанного продукта.

**1.2 Определения, акронимы, сокращения**

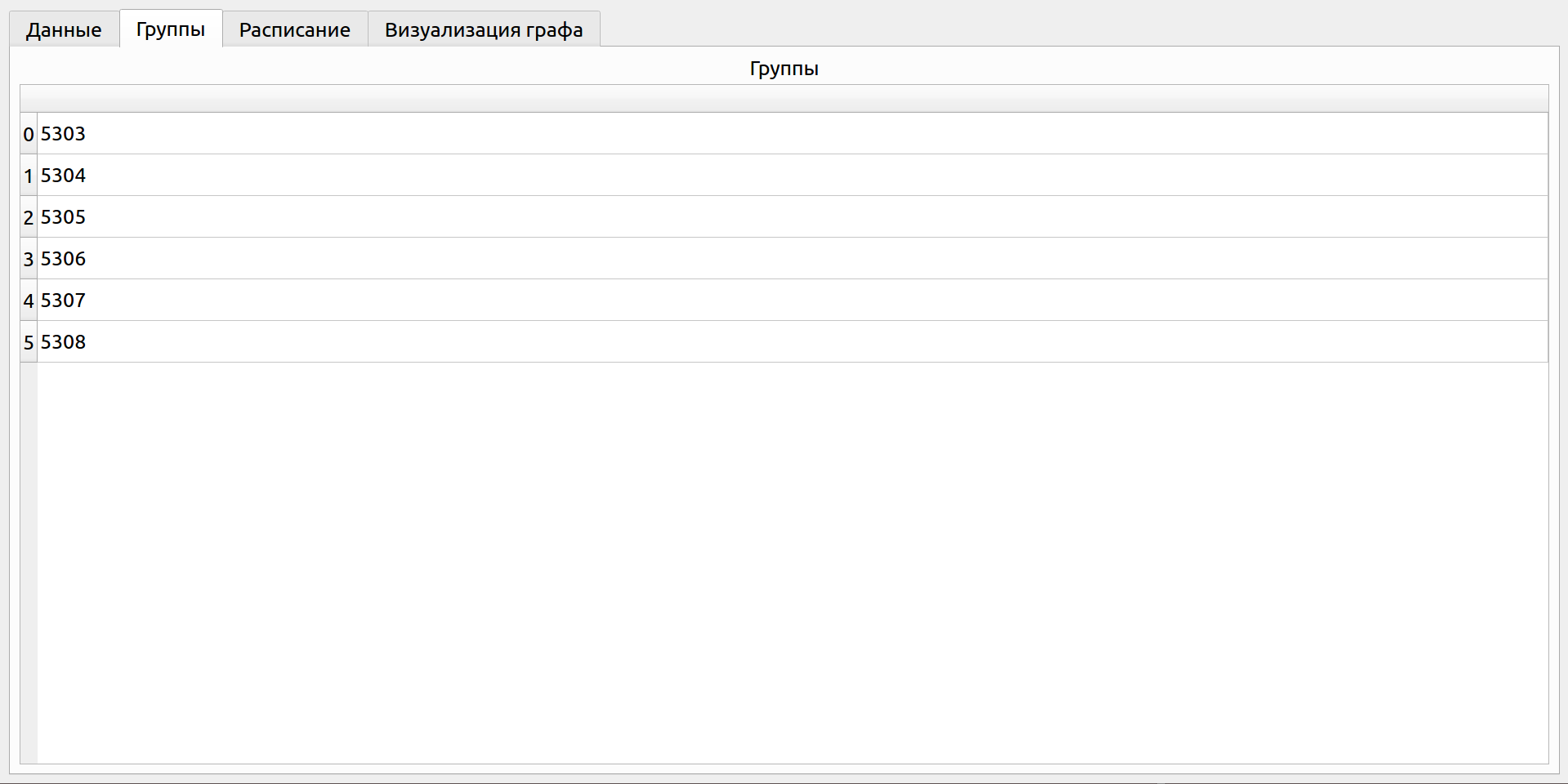
*См. Глоссарий.*

**2. Ручное тестирование приложение**

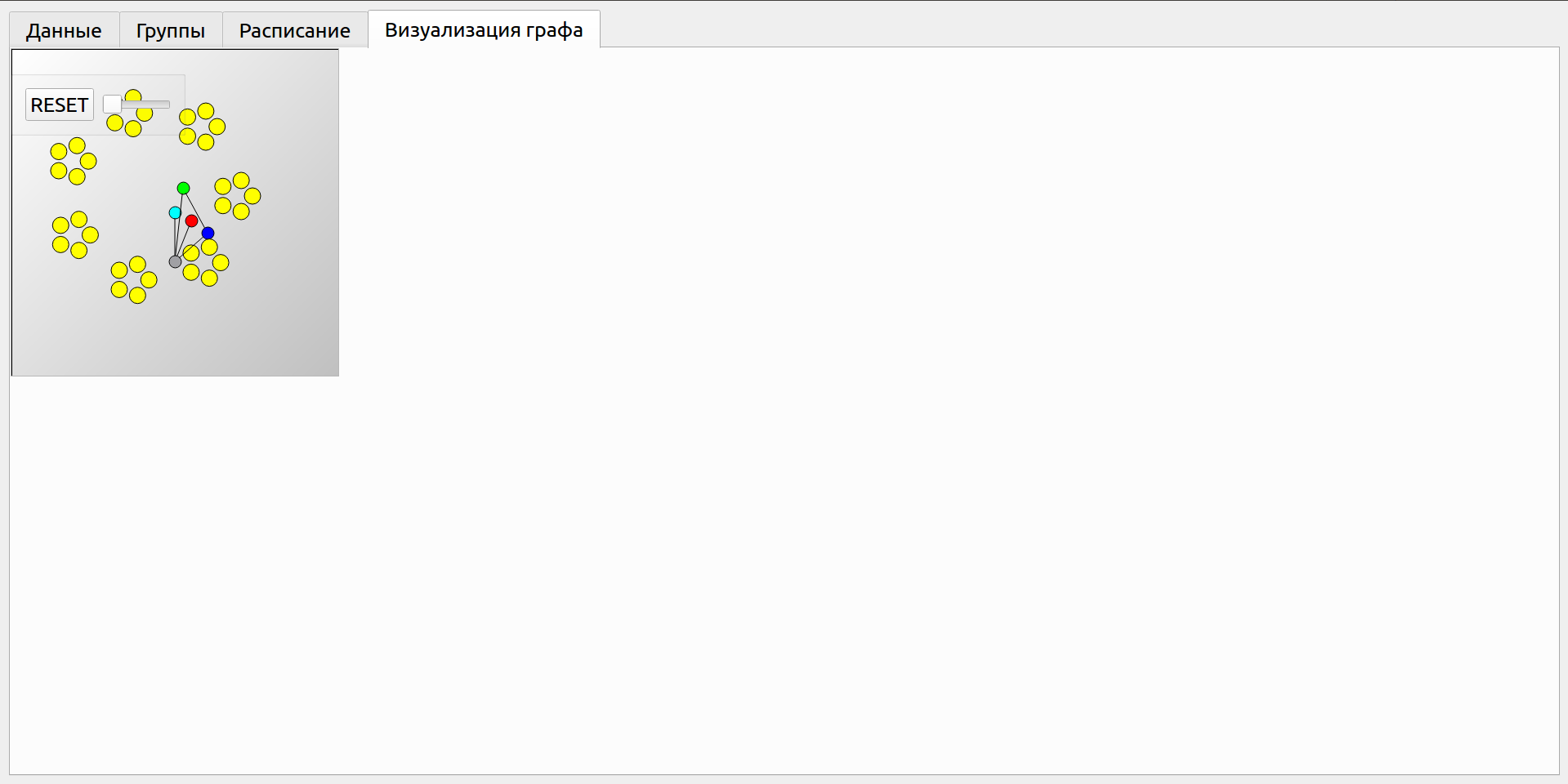
Произведено тестирование интерфейса разработанного приложения с помощью метода чёрного ящика. Приложение имеет следующий интерфейс с тестовыми данными. Главное окно:



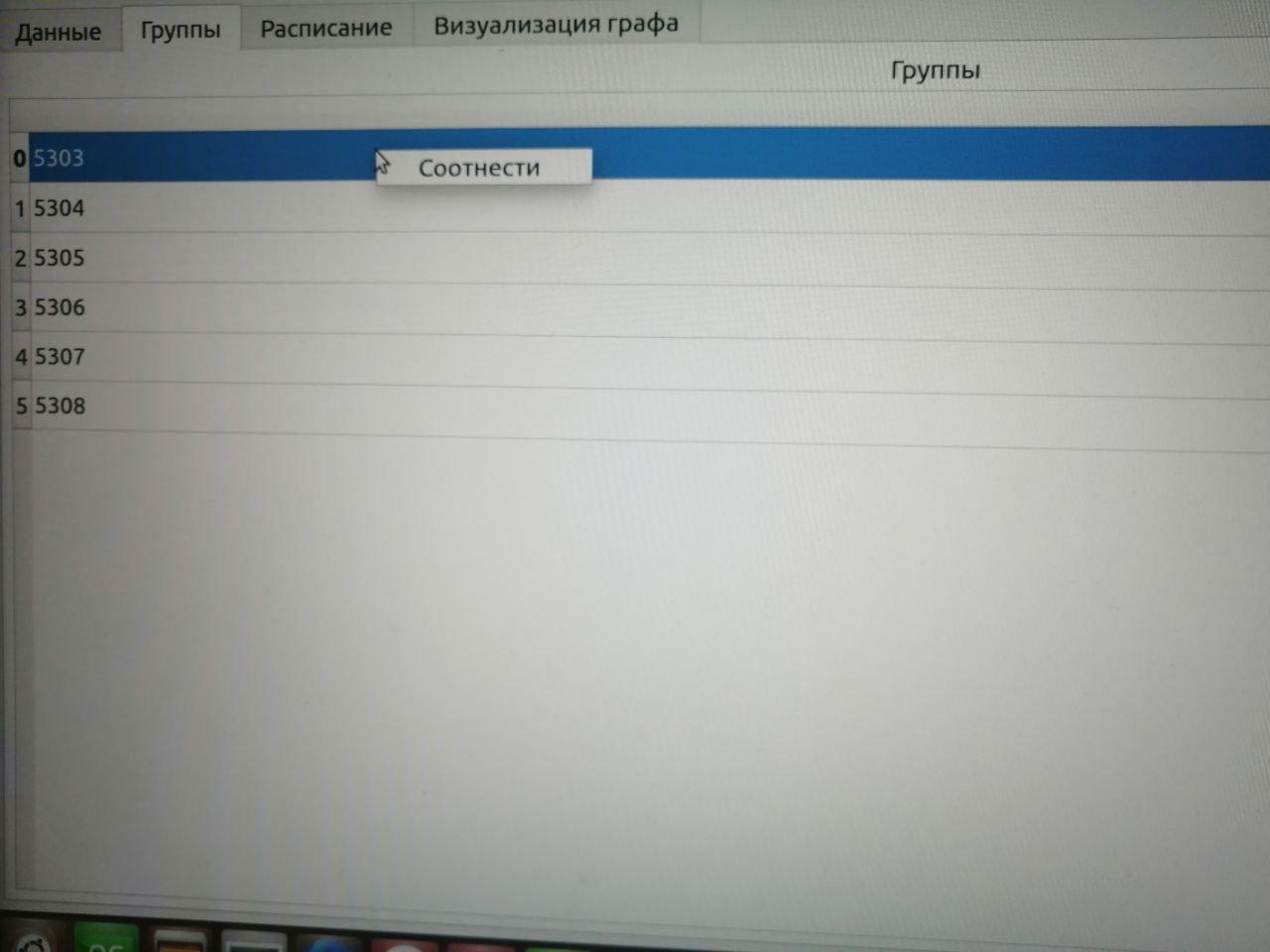
Данные групп:



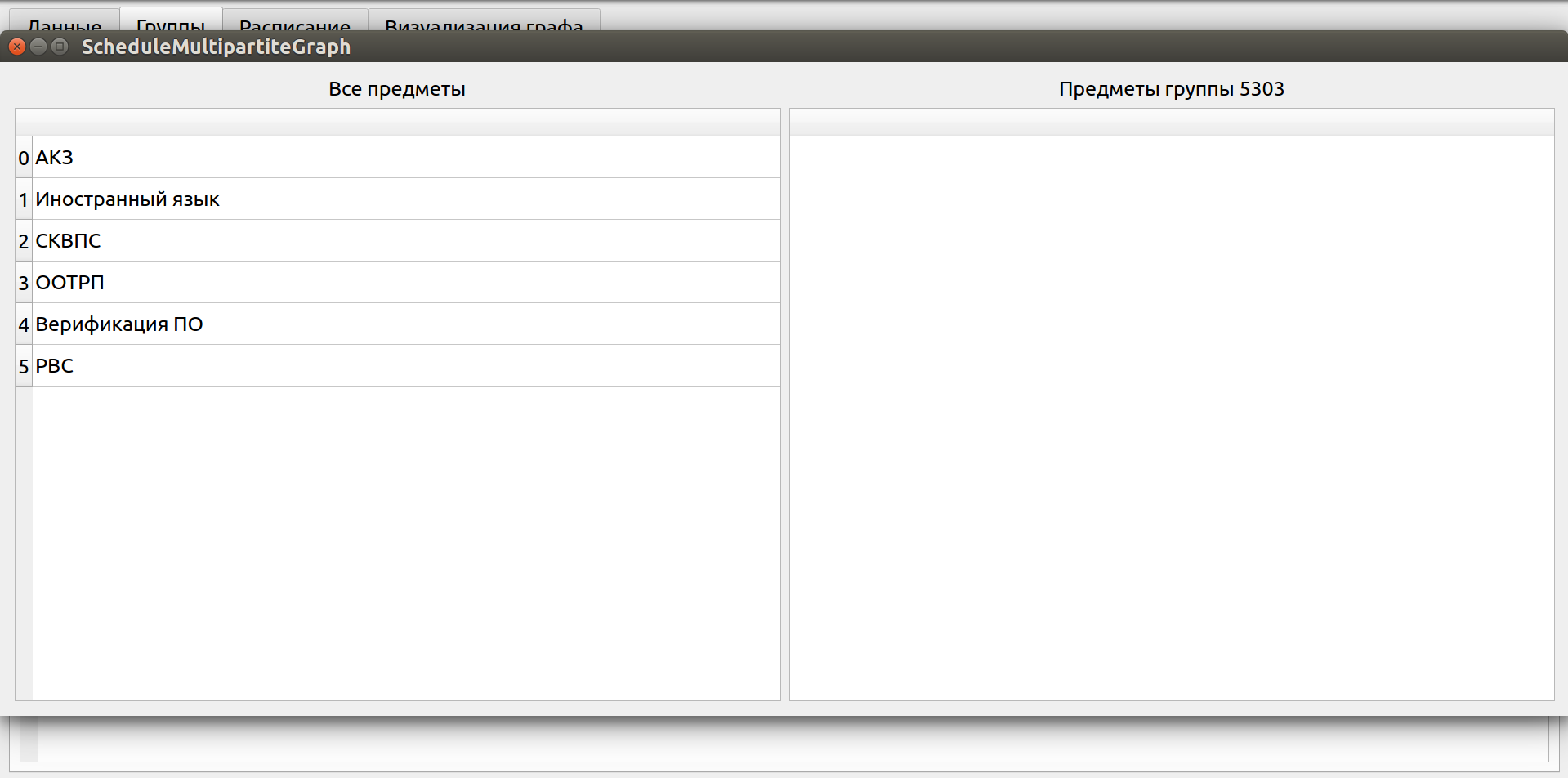
Визуализация графа:



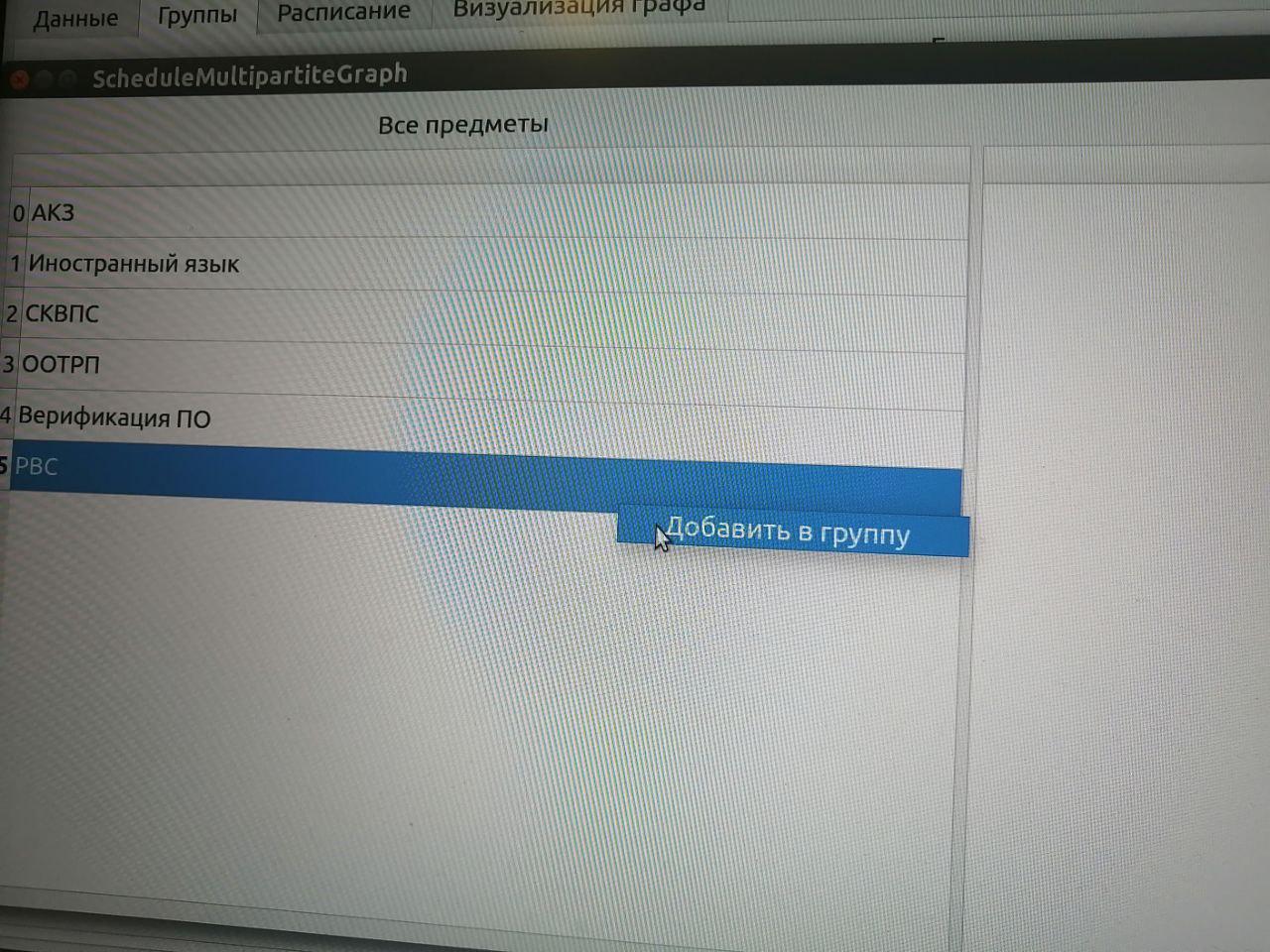
На вкладке группы, можно нажать на любую группу правой кнопкой, откроется контекстное меню: соотнести



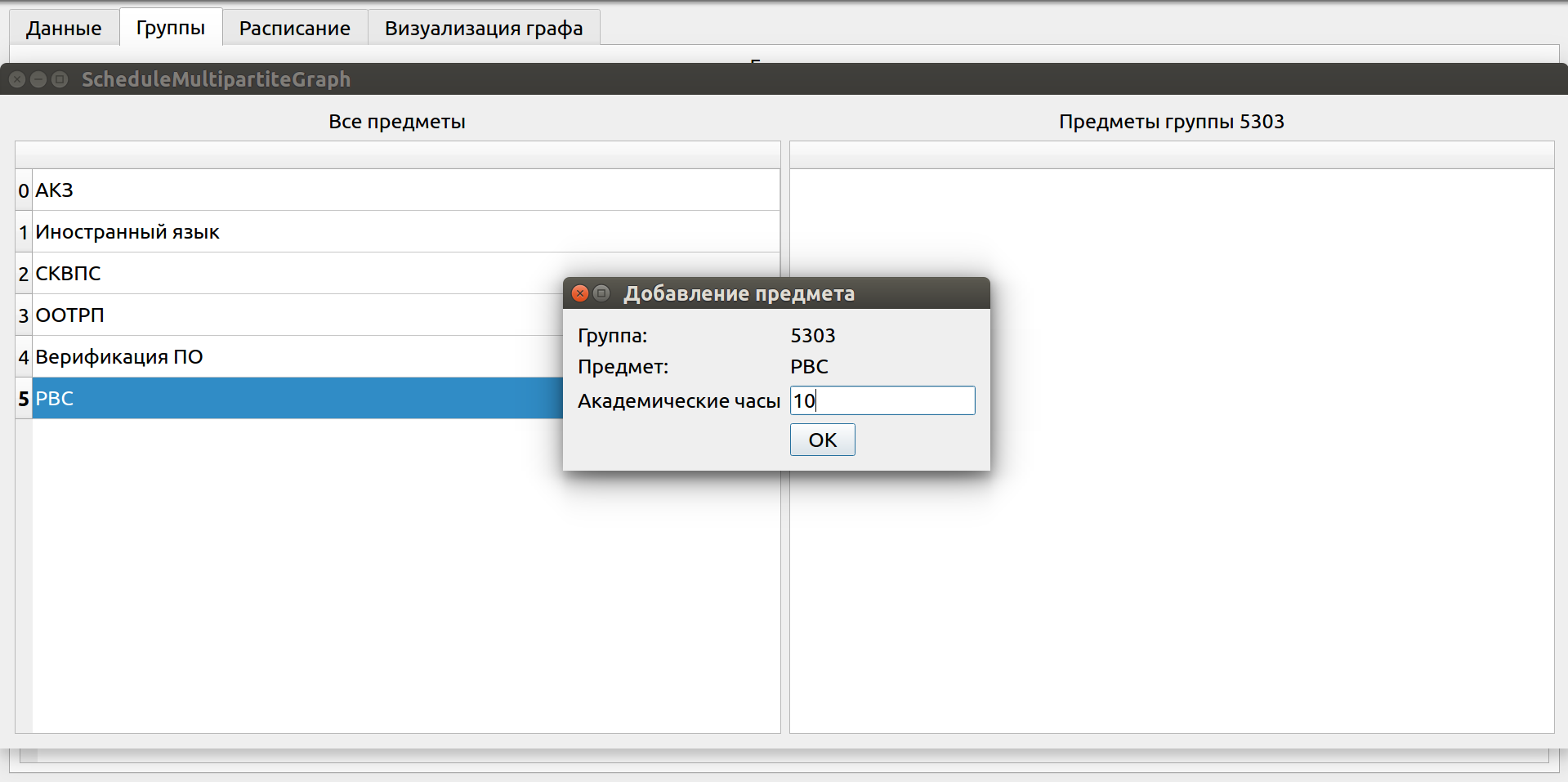
Далее откроется новое окно для добавления группировки предметов по группам с использованием контекстного меню:



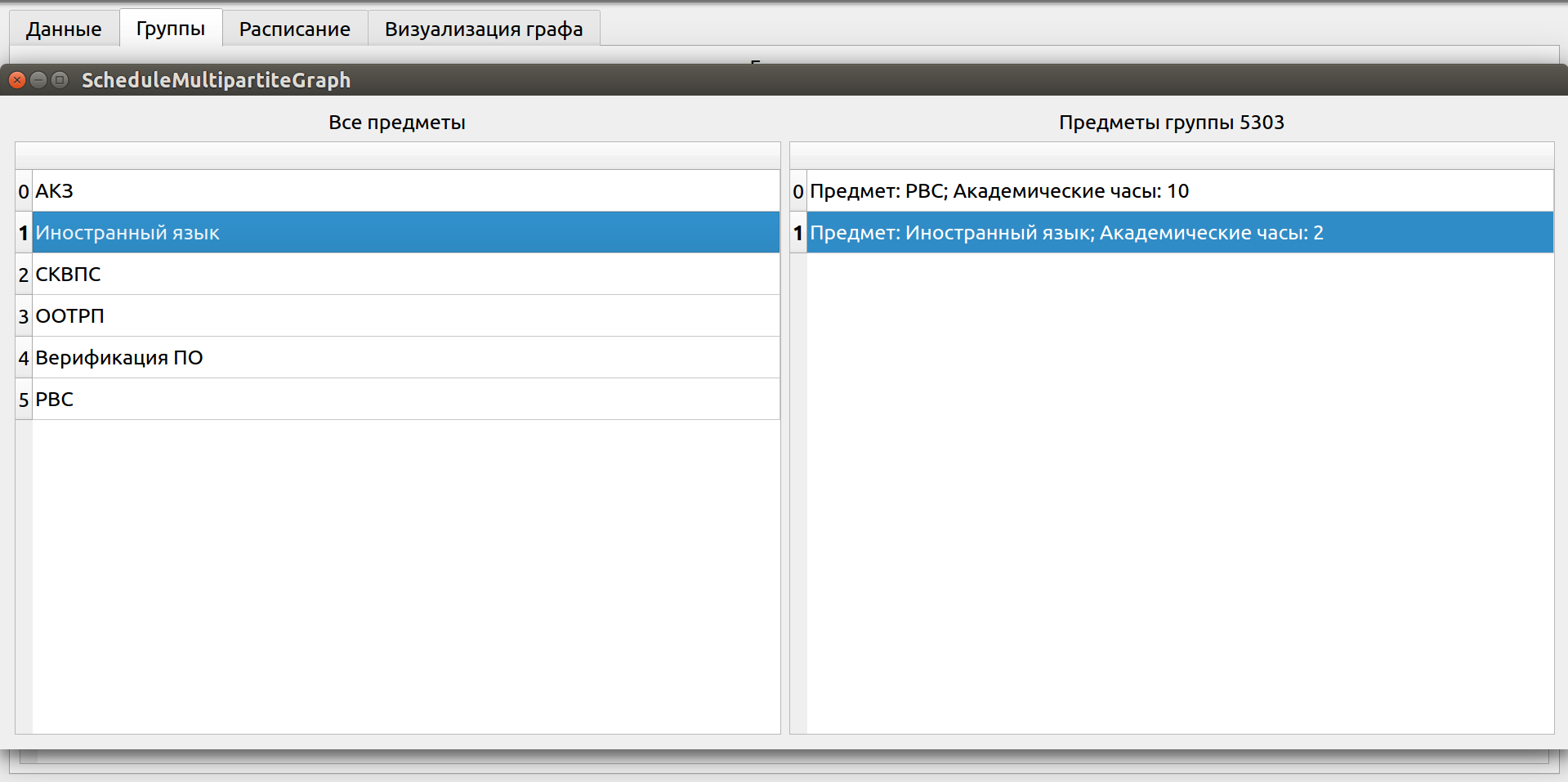
Добавить в группу:



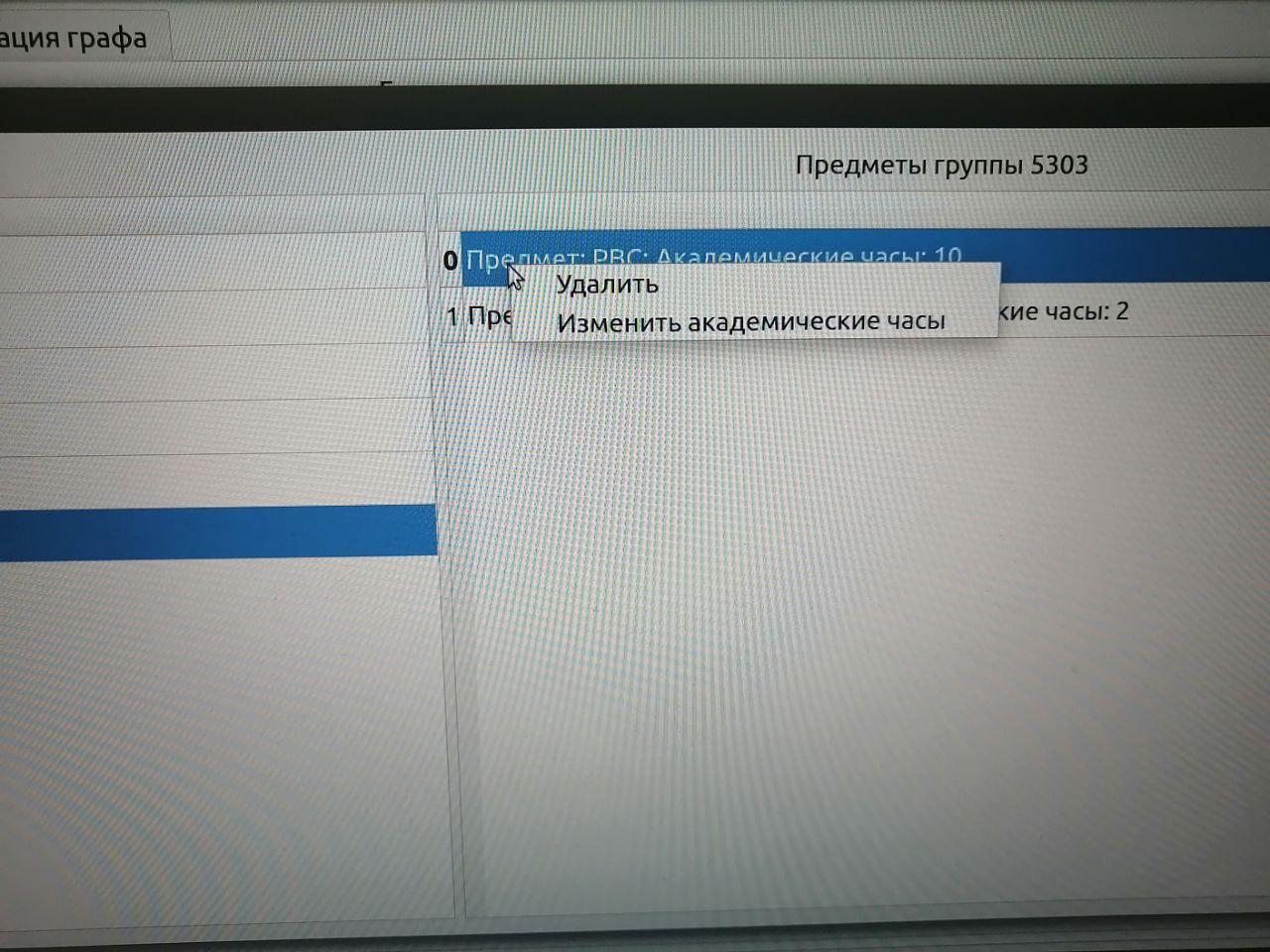
Добавление предмета:



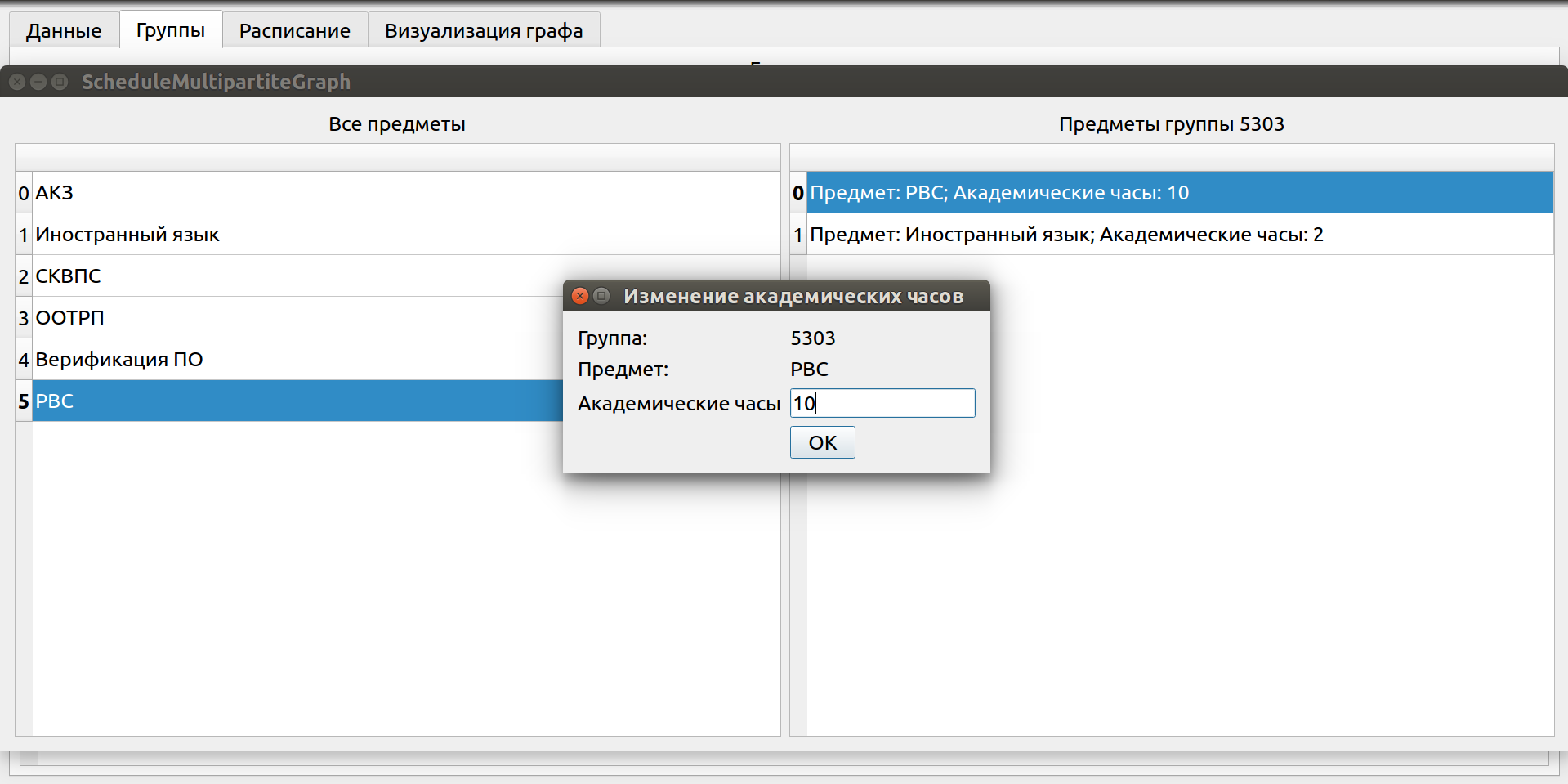
Предметы добавлены:

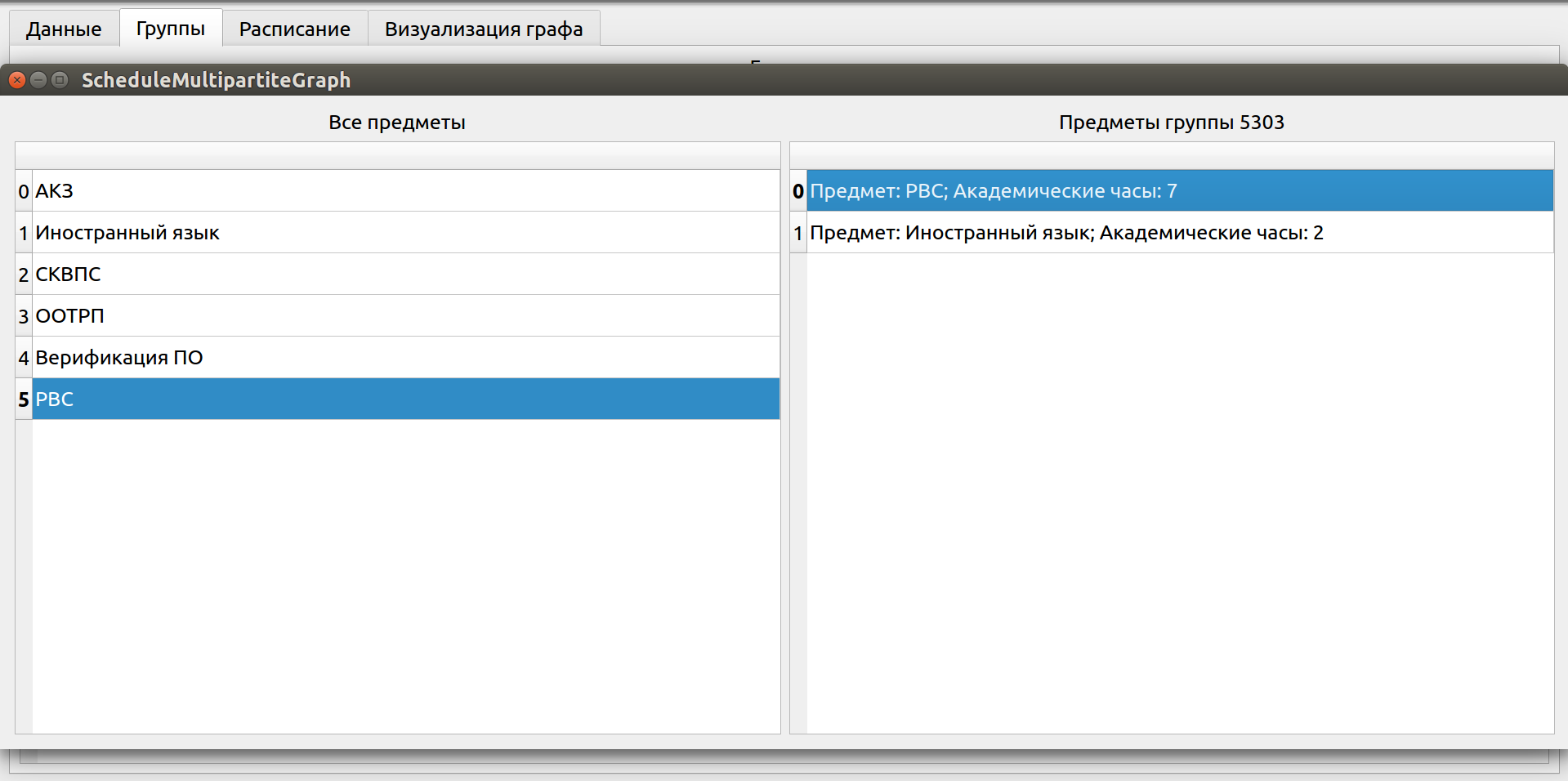


У добавленных предметов можно поменять академические часы и удалять предметы из групп с помощью контекстного меню:

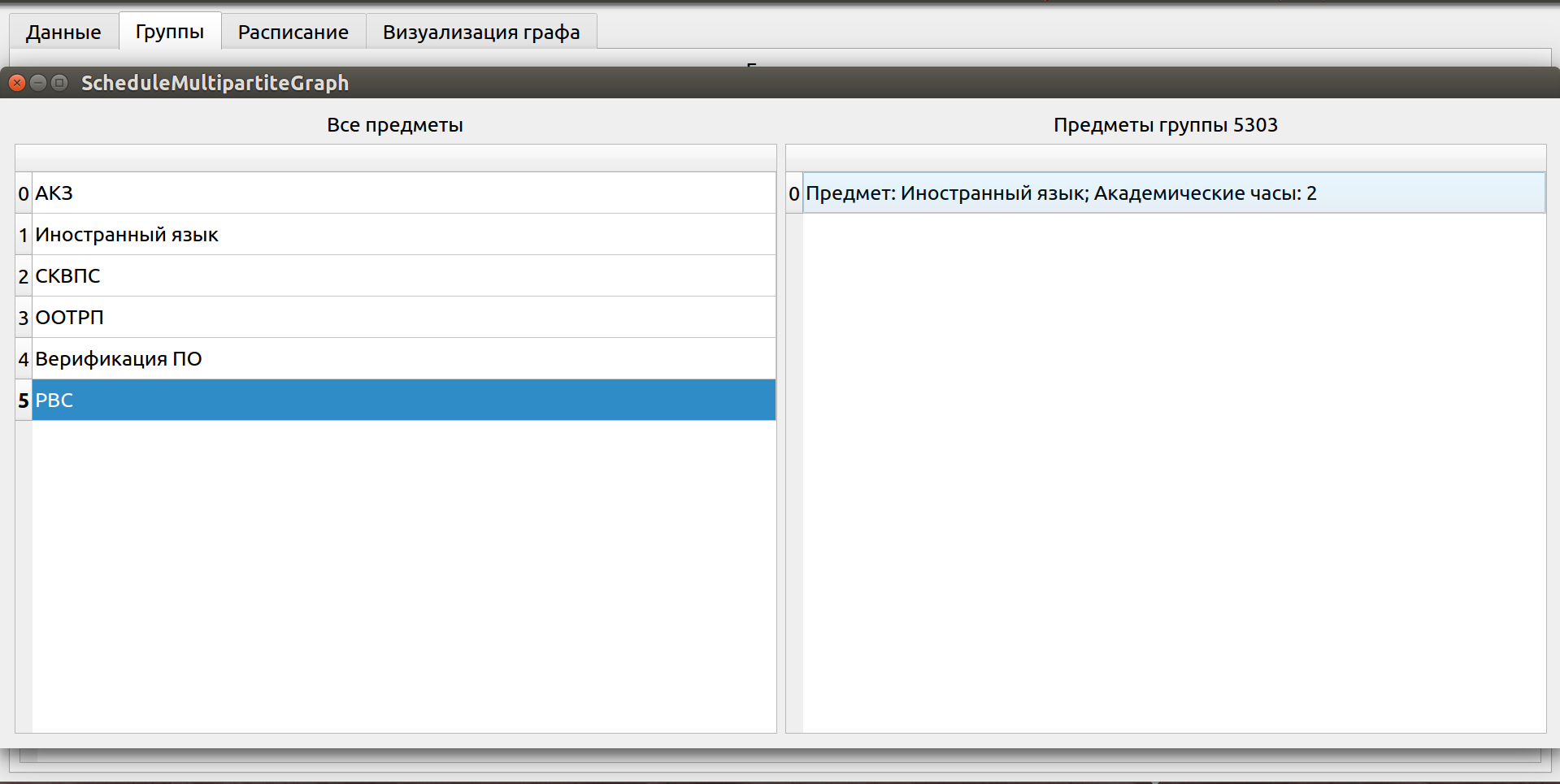


Изменить академические часы:





Удалить предмет:



Добавлено сохранение расписания в JSON — файл.

