|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Информационные системы и телекоммуникации»

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ***

***НА ТЕМУ:***

***«Система планирования турнирных матчей»***

Студент **ИУ3-83** **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.А.Вилкова**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель ВКР **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.М.Иванов**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Нормоконтролер **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.А.Тренин**

(Подпись, дата) (И.О.Фамилия)

*2017 г.*

# АННОТАЦИЯ

Данная работа посвящена решению проблемы планирования турниров, а именно разработке системы планирования и управления турнирными матчами.

За время работы в сфере организации спортивных мероприятий по большому теннису мною была выявлена нехватка системы планирования в рамках проведения различных этапов российских теннисных турниров. На данный момент соревнования Российского теннисного тура планируются вручную. Было решено реализовать автоматизированную систему планирования турнирных матчей в рамках выпускной квалификационной работы.

В исследовательской части работы был проведен анализ заинтересованных сторон системы, были спроектированы сценарии поведения системы, а также были выявлены ее функции и требуемые качества. В рамках исследования были проанализированы существующие приложения и сервисы управления спортивными соревнованиями с целью выявления необходимости разработки собственной системы планирования турнирных матчей.

В конструкторской части был реализован сервис планирования турнирных матчей с возможностью расширяемости системы с использованием технологии Eclipse Communication Framework и реализован пользовательский интерфейс с помощью графической платформы JavaFX с возможностью многопользовательского доступа к системе на основе платформы приложений e(fx)clipse e4.

В технологической части были описаны используемые средства работы, описан способ получения бинарной сборки системы и проведено тестирование системы, по заранее составленному плану тестирования.

В заключении описаны полученные результаты и сформулированы дальнейшие планы по усовершенствованию системы.

СОДЕРЖАНИЕ

[АННОТАЦИЯ 5](#_Toc485253136)

[ОПРЕДЕЛЕНИЯ 8](#_Toc485253137)

[ВВЕДЕНИЕ 9](#_Toc485253138)

[1. Исследовательская часть 10](#_Toc485253139)

[1.1. Анализ предметной области 10](#_Toc485253140)

[1.1.1. Олимпийская система 10](#_Toc485253141)

[1.1.2. Круговая система 12](#_Toc485253142)

[1.2. Анализ заинтересованных сторон 14](#_Toc485253143)

[1.3. Выявление функций системы 18](#_Toc485253144)

[1.4. Показатели качества системы 20](#_Toc485253145)

[1.5. Анализ существующих систем планирования 27](#_Toc485253146)

[1.5.1. Tennis Tournament Manager 27](#_Toc485253147)

[1.5.2. Table Tennis Manager 30](#_Toc485253148)

[1.5.3. Tournament Mango 31](#_Toc485253149)

[1.5.4. Сравнение существующих аналогов 32](#_Toc485253150)

[1.6. Выводы к главе 34](#_Toc485253151)

[2. Конструкторская часть 35](#_Toc485253152)

[2.1. Концептуальная архитектура 35](#_Toc485253153)

[2.2. Спецификация OSGi 37](#_Toc485253154)

[2.2.1. OSGi-сервисы 37](#_Toc485253155)

[2.2.2. Eclipse Communication Framework 38](#_Toc485253156)

[2.3. Платформа JavaFX 39](#_Toc485253157)

[2.3.1. FXDiagram 40](#_Toc485253158)

[2.4. Компонентная декомпозиция архитектуры 41](#_Toc485253159)

[2.4.1. Компонентная декомпозиция архитектуры для одного устройства 41](#_Toc485253160)

[2.4.2. Компонентная декомпозиция системы для двух устройств 42](#_Toc485253161)

[2.5. Развертывание системы на оборудовании 43](#_Toc485253162)

[2.5.1. Развертывание системы на одном устройстве 43](#_Toc485253163)

[2.5.2. Развертывание системы на двух устройствах 45](#_Toc485253164)

[2.6. Детальное проектирование 47](#_Toc485253165)

[2.6.1. Диаграммы классов 47](#_Toc485253166)

[2.6.2. Диаграмма состояний конечного автомата 52](#_Toc485253167)

[2.6.3. Диаграмма последовательностей 52](#_Toc485253168)

[2.7. Вывод к главе 53](#_Toc485253169)

[3. Технологическая часть 55](#_Toc485253170)

[3.1. Используемые средства 55](#_Toc485253171)

[3.2. Настройка Eclipse IDE 55](#_Toc485253172)

[3.3. Тестирование 58](#_Toc485253173)

[3.3.1. Планирование процесса тестирования 58](#_Toc485253174)

[3.3.2. Разработка тестовых случаев 60](#_Toc485253175)

[3.3.3. Результаты тестирования 61](#_Toc485253176)

[3.4. Вывод к главе 65](#_Toc485253177)

[ЗАКЛЮЧЕНИЕ 66](#_Toc485253178)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 67](#_Toc485253179)

[ПРИЛОЖЕНИЕ А 68](#_Toc485253180)

[ПРИЛОЖЕНИЕ Б 77](#_Toc485253181)

# ОПРЕДЕЛЕНИЯ

**Теннис** – это спортивная игра, в которой соперничают либо два игрока (singles), либо две команды, состоящие из двух игроков (doubles). Задачей теннисиста является при помощи ракетки отправлять мяч на сторону соперника так, чтобы тот не смог его отразить. Удар мяча об игровую площадку (корт) может осуществляться не более чем один раз на половине соперника, в противном случае очко считается проигранным.

**Турнирная единица (матч/встреча)** – это одно спортивное соревнование определенного названия в одной возрастной группе в одиночном (среди юношей (мужчин), либо девушек (женщин)), парном (среди юношей (мужчин), либо девушек (женщин)) или смешанном (среди юношей (мужчин) и девушек (женщин)) разрядах.

**Турнир** – состоит из одного или нескольких турнирных единиц, проходящих в одни сроки с одинаковым названием, включающие в себя турнирные единицы разных возрастных групп, разрядов и категорий.

**Регистрация на турнир** – процесс подачи заявки в установленные сроки. На данном этапе игроков можно добавлять или удалять из турнира. По окончанию регистрации игроки вносятся в базу игроков, из которой нельзя удалить игрока как по его личному желанию, так и по желанию организаторов, и турнир считается открытым.

# ВВЕДЕНИЕ

На сегодняшний день сфера спорта является популярной как среди молодежи, так и среди взрослого и даже пожилого населения. Города стараются поддерживать положительное стремление людей к занятиям физической культурой. За последние годы произошли изменения в инфраструктуре спорта. В качестве примера, в Москве количество систематически занимающихся спортом людей увеличилось на 73%, по сравнению с 2010 годом. Этому способствовали модернизация спортивных объектов и открытие новых спортивных комплексов.

С увеличением интереса к спорту увеличилось количество проводимых турниров: от маленьких соревнований, проводимых в рамках детских спортивных секций, до городских состязаний, насчитывающих сотни и даже тысячи участников. Таким образом, турниры часто требуют системы планирования соревнований для корректности их проведения.

Целью данной квалификационной работы является разработка системы планирования теннисных турнирных матчей, организацией которых занимается Федерация тенниса России вместе с отделом Российского теннисного тура. Система планирования турнирных матчей должна позволять регистрировать игроков, выбирать настройки турнира, создавать турнир, автоматически генерировать матчи на каждом этапе его проведения, вносить результаты каждой встречи, формировать в зависимости от итогов турнира рейтинг игроков для определения победителя созданного турнира.

# Исследовательская часть

В данном разделе представлен анализ предметной области и описаны виды турниров с целью дальнейшей реализации моделей турниров и алгоритмов автоматического генерирования матчей в конструкторской части работы.

Также был проведен детальный анализ заинтересованных сторон, выявлены их интересы и цели, рассмотрены возможные сценарии и ограничения для того, чтобы установить функциональные и качественные требования к системе. На основе этих показателей качества и функций была проведена сравнительная характеристика с существующими системами планирования турниров и составлено техническое задание.

## Анализ предметной области

Вид спорта определяет представление счета встречи, например, в футболе результатом игры может стать счет 2:1, а в большом теннисе матч может быть завершен со счетом 6:2, 4:6, 7:6 (3). Несмотря на различность представления счета одного матча, общие схемы проведения турниров различных видов спорта обычно совпадают. Основными системами проведения турниров является олимпийская и круговая, которые будут ниже подробно рассмотрены. В некоторых видах спорта иногда практикуют использование таких систем проведения турниров, как швейцарская система (выбывание из соревнования после двух поражений) и смешанная система, которая может комбинировать несколько систем в зависимости от этапа турнира. В данной квалификационной работе были рассмотрены только две основные системы организации турнирных матчей.

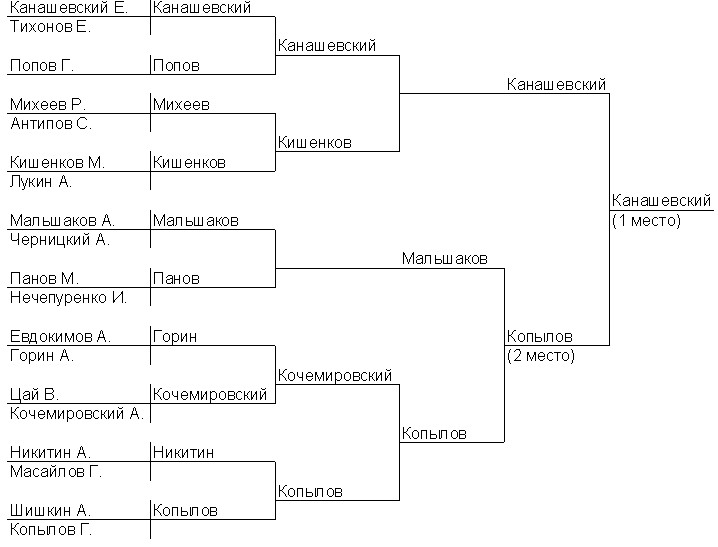
### Олимпийская система

В турнире по олимпийской системе (далее - О) игрок (команда) выбывает из соревнования при первом же поражении, а победителем становится игрок (команда), победивший(ая) во всех своих матчах. В соответствии с положением о турнире может быть проведен матч за 3-е место между игроками (командами), проигравшими в полуфинале турнира.

Турнир по олимпийской системе проводится в несколько туров. Количество туров зависит от числа игроков: для 3-4 игроков количество туров равно 2, для 5-8 игроков – 3, для 9-16 игроков – 4, для 17-32 игроков – 5, для 33-64 игроков – 6 и т.д.

Общее количество матчей при проведении турнира по олимпийской системе на единицу меньше числа игроков в таблице турнира (в случае проведения матча за 3-е место – равно числу игроков в таблице турнира).

В каждом туре между собой встречаются игроки, фамилии которых расположены в одном ряду на соседних (смежных) строках, соединенных справа вертикальной линией (рисунок 1). Победители в матчах первого тура попадают во второй тур (в таблице турнира – в следующий вертикальный ряд), победители в матчах второго тура – в третий тур и т.д.

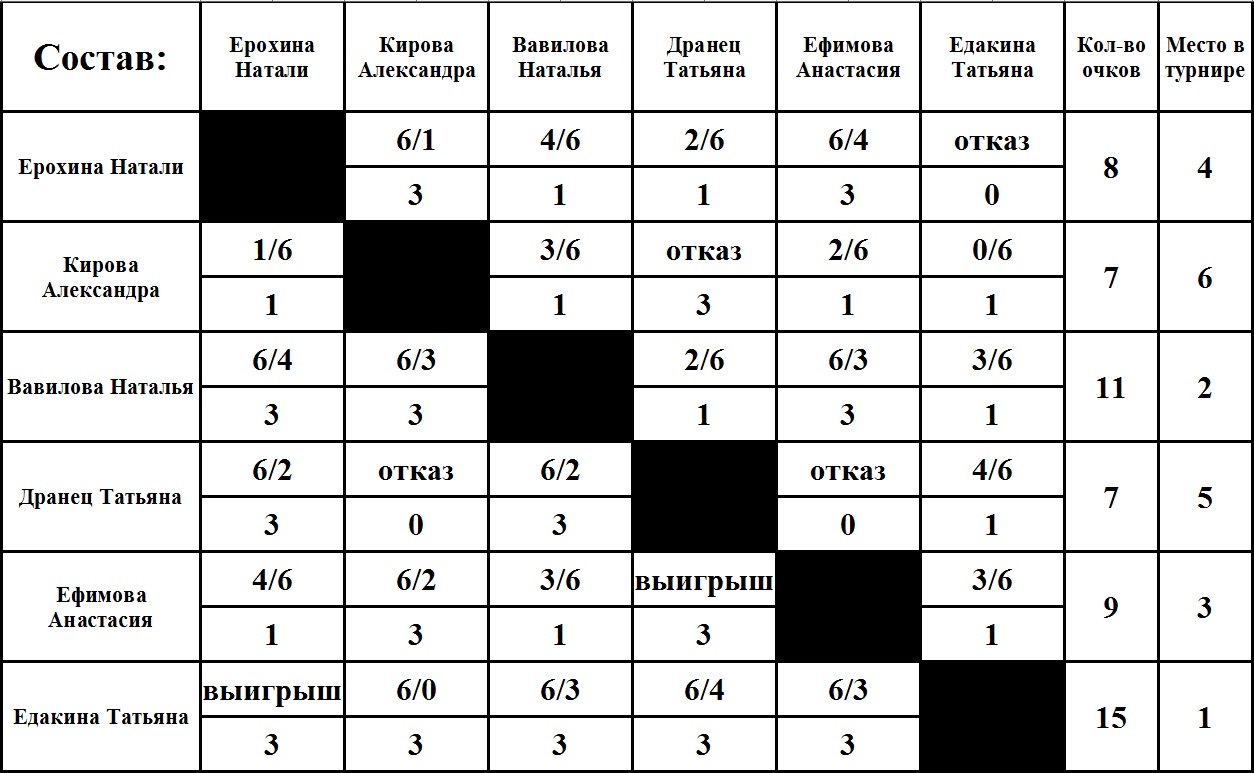


1. – Сетка матчей турнира по олимпийской системе

Тур, в котором встречаются 32 игрока, называется 1/16 финала, 16 игроков – 1/8 финала, 8 игроков – ¼ финала или четвертьфиналом, 4 игрока – ½ финала или полуфиналом, 2 игрока – финалом.

### Круговая система

Круговая система (К) применяется в турнире, в котором для определения всех мест, занятых каждым(ой) игроком (командой), проводятся матчи между всеми игроками (командами), участвующими в турнире. Места, занятые игроками (командами), определяются по количеству набранных очков. За победу в матче (личном или командном) игроку (команде) начисляется одно очко, за поражение – ноль. Сетка матчей турнира, организованного по принципу круговой системы представлена на рисунке 2.



1. – Сетка матчей турнира по олимпийской системе

В случае неявки игрока на матч или отказа от него и принятия главным судьей соответствующего решения игроку засчитывается поражение, а его противнику – победа с начислением очка, при этом в таблице турнира указывается соответствующая причина, но при подсчете дополнительных показателей в случае равенства количества очков у нескольких игроков результат матча учитывается как «6:0, 6:0». Если игрок (команда) сыграл(а) менее половины предусмотренных таблицей турнира матчей, все его (её) результаты предыдущих сыгранных матчей аннулируются только для определения места в таблице, однако все результаты сыгранных матчей этим игроком (игроками этой команды) учитываются в истории матчей.

Победителем турнира (победителем группы) считается игрок (команда), набравший(ая) наибольшее количество очков. При равенстве очков у двух игроков (команд) преимущество получает победитель матча между ними.

Очередность матчей игроков (командных встреч) определяется согласно таблице 1, в которой указывается, какие номера встречаются друг с другом в каждом туре соревнования.

1. – Очередность матчей игроков по турам

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Туры* | | | | | | |
| *1* | *2* | *3* | *4* | *5* | *6* | *7* |
| *Для 3-4 игроков (команд)* | | | | | | |
| 1-(4) | 1-3 | 1-2 |  |  |  |  |
| 2-3 | 2-(4) | 3-(4) |  |  |  |  |
| *Для 5-6 игроков (команд)* | | | | | | |
| 1-(6) | 1-5 | 1-4 | 1-3 | 1-2 |  |  |
| 2-5 | 2-3 | 2-(6) | 2-4 | 3-(6) |  |  |
| 3-4 | 4-(6) | 3-5 | 5-(6) | 4-5 |  |  |
| *Для 7-8 игроков (команд)* | | | | | | |
| 1-(8) | 1-7 | 1-6 | 1-5 | 1-4 | 1-3 | 1-2 |
| 2-7 | 2-5 | 2-3 | 2-(8) | 2-6 | 2-4 | 3-(8) |
| 3-6 | 3-4 | 4-(8) | 3-7 | 3-5 | 5-(8) | 4-7 |
| 4-5 | 6-(8) | 5-7 | 4-6 | 7-(8) | 6-7 | 5-6 |

Примечание. При четном числе игроков (команд) в таблице учитываются номера, стоящие в скобках. При нечетном числе номер, расположенный на одной строке против номера, стоящего в скобках, в соответствующем туре свободен от игры.

Общее количество матчей (командных матчей) в турнире, проводимом по круговой системе, равно

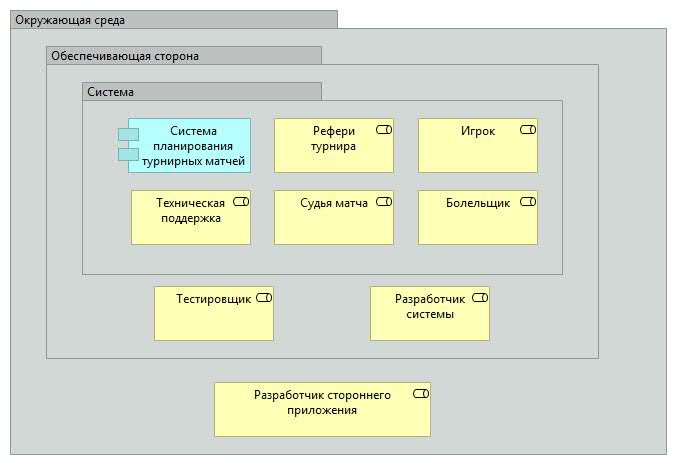
где n – число игроков (команд), участвующих в турнире.

## Анализ заинтересованных сторон

Для разрабатываемой системы был проведен анализ заинтересованных сторон и были выявлены стороны, изображенные на диаграмме заинтересованных сторон. Данная диаграмма представлена на рисунке 3.

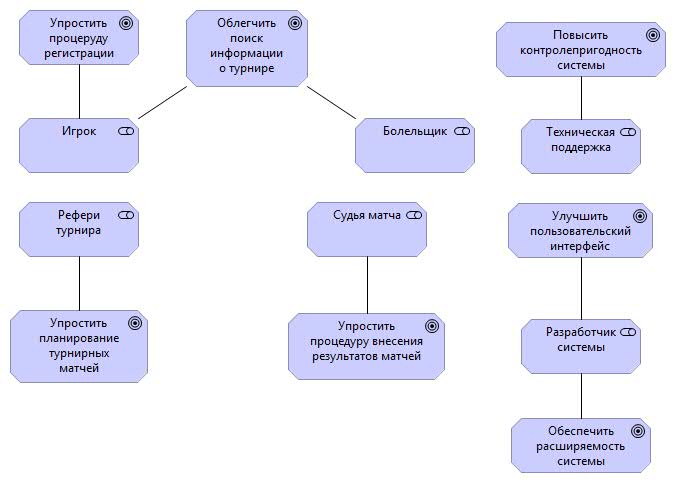
На данной диаграмме систематизированы лица, имеющие непосредственное отношение к системе. В первом круге – круге «Система», изображены стороны, наиболее тесно связанные с системой.

На втором круге изображены стороны, каким-либо образом поддерживающие работоспособность и создание системы.

На третьем круге изображены факторы окружающей среды, которые могут повлиять на систему, каким-либо позитивным или негативным образом

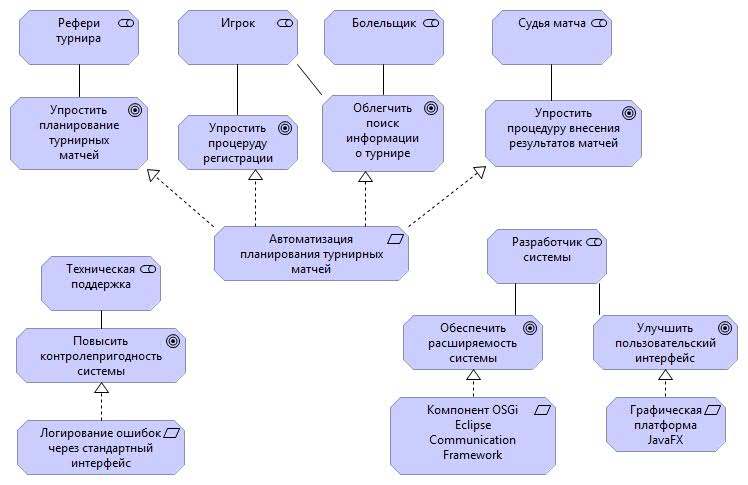
1. – Диаграмма заинтересованных сторон

У каждой из заинтересованных сторон системы есть свои собственные интересы, которые они хотят достичь, используя разрабатываемую систему. Все цели и их связи с заинтересованными лицами системы изображены на рисунке 4.



1. – Заинтересованные стороны и их цели

После построения диаграммы заинтересованных лиц и их целей необходимо выявить сценарии системы и ее ограничения. Ограничения системы показаны при помощи компонента «Ограничения» (Constraint). Все выявленные сценарии и ограничения системы изображены на рисунке 5.



1. – Сценарии и ограничения

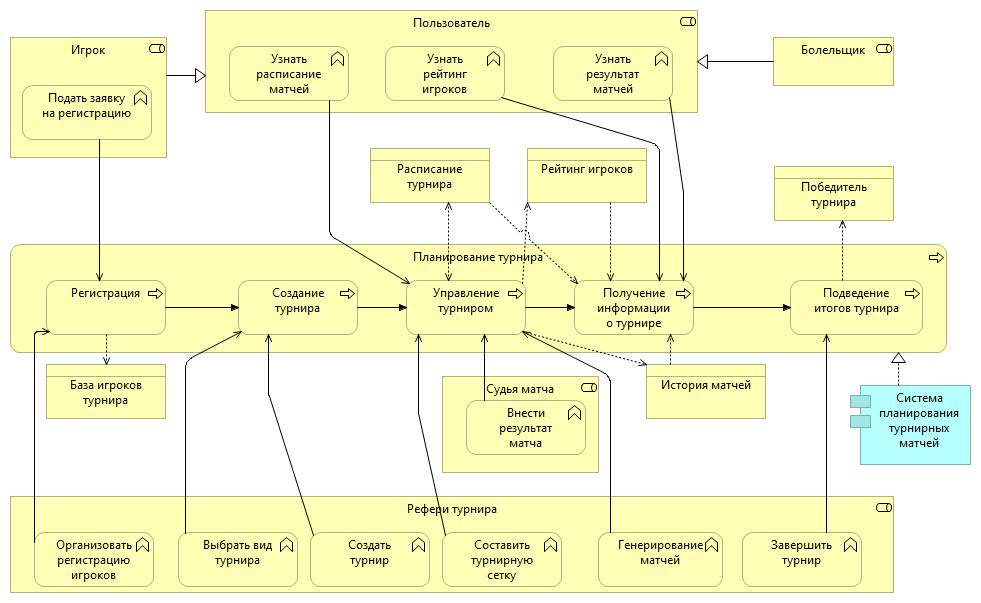
## Выявление функций системы

После того как был выполнен анализ заинтересованных сторон и их цели, необходимо выявить основные функциональные требования к системе.

Основными функциями системы являются:

* Создание универсальной модели турнира;
* Выбор настроек создаваемого турнира;
* Возможность регистрации игроков;
* Автоматическое генерирование турнирных матчей разных раундов;
* Графическое представление сетки матчей управляемого турнира;
* Выбор победителя для каждого матча;
* Сохранение истории сыгранных матчей;
* Формирование рейтинга игроков по результатам турнира для выявления победителя;
* Многопользовательский доступ к системе планирования несколькими пользователями;
* Создание понятного графического пользовательского интерфейса для планирования турнирных матчей.

На основе выявленных функций системы и диаграммы, представленной на рисунке 5, можно показать взаимосвязь выявленных бизнес-процессов и функций системы, а также бизнес-процесс планирования турнира (рисунок 6) [10].



1. – Бизнес-процесс планирования турнира

## Показатели качества системы

Помимо определения функциональных характеристик в процессе разработки требований к системе было необходимым выявить требования к показателям качества. [1] Для этого были созданы сценарии атрибутов качества разрабатываемой системы планирования турнирных матчей, которые представлены в таблицах 2-7.

1. – Показатели готовности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Внутренний | Неисправность сервера | Нормальные условия | Сервис планирования | Выдача сообщения о невозможности запуска системы в консольные сообщения из-за проблем сервера и перезапуск системы. | Необходимость восстановление системы за 1 минуту. |
| Внешний | Длительное бездействие | Нормальные условия | Сервис планирования | Система продолжает работу | Готовность к использованию. |
| Внешний | Аварийная ситуация | Нормальные условия | Сервис планирования | Выдача сообщения об ошибке через консольные сообщения и попытка устранения путем перезапуска системы или ее части. | Обеспечение работоспособности за 2-3 минуты. |

1. – Показатели модифицируемости системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Разработчик системы | Добавление алгоритмов автоматического генерирования матчей | В момент разработки системы | Исходный код программы | Модифицирование системы не привело к ошибкам | В течение трех часов |
| Разработчик системы | Изменение универсальной модели турнира | В момент разработки системы | Исходный код программы | Модифицирование системы не привело к ошибкам | В течении трех часов |
| Разработчик системы | Внесение изменений в пользовательский интерфейс | В момент разработки системы | Исходный код программы | Модифицирование системы не привело к ошибкам | В течение трех часов |

1. – Показатели производительности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Пользователи | Отправление запросов по управлению турниром | Нормальные условия | Система планирования | Обработка вызываемых методов | Обработка запросов не более 2 секунд. |
| Пользователи | Единовременное взаимодействие пользователей с сервером | Перегруженный режим | Система планирования | Система должна продолжать функционировать | Подключение до 100 устройств в минуту |

1. – Показатели безопасности

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Пользователь | Попытка внести изменения в данные при отсутствии прав доступа | В период использования | Данные в системе | Система не позволяет внести данные из-за ограниченности прав пользователя | Данные защищены от несанкционированного доступа |

1. – Показатели контролепригодности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Тестировщик | Тестирование турнира при разном количестве игроков | В период тестирования системы | Модель турнира | Система случайным образом назначает победителя каждого этапа и выводит победителя турнира | Наибольший массив тестируемых игроков 2000 |
| Тестировщик | Тестирование функции обмена данными с пользователями | В период тестирования системы | Сервис планирования | Система принимает запросы от пользователей и отсылает ответы | До 50 устройств одновременно |

1. – Показатели практичности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Пользователь | Изучить возможности системы по управлению турниром | В момент использования системы | Система планирования | Система возвращает победителя турнира | Цель пользователя по управлению турниром достигнута |
| Пользователь | Облегчить использования системы | В момент использования системы | Система планирования | Система имеет простой и понятный интерфейс | Повышение образовательного уровня пользователя |

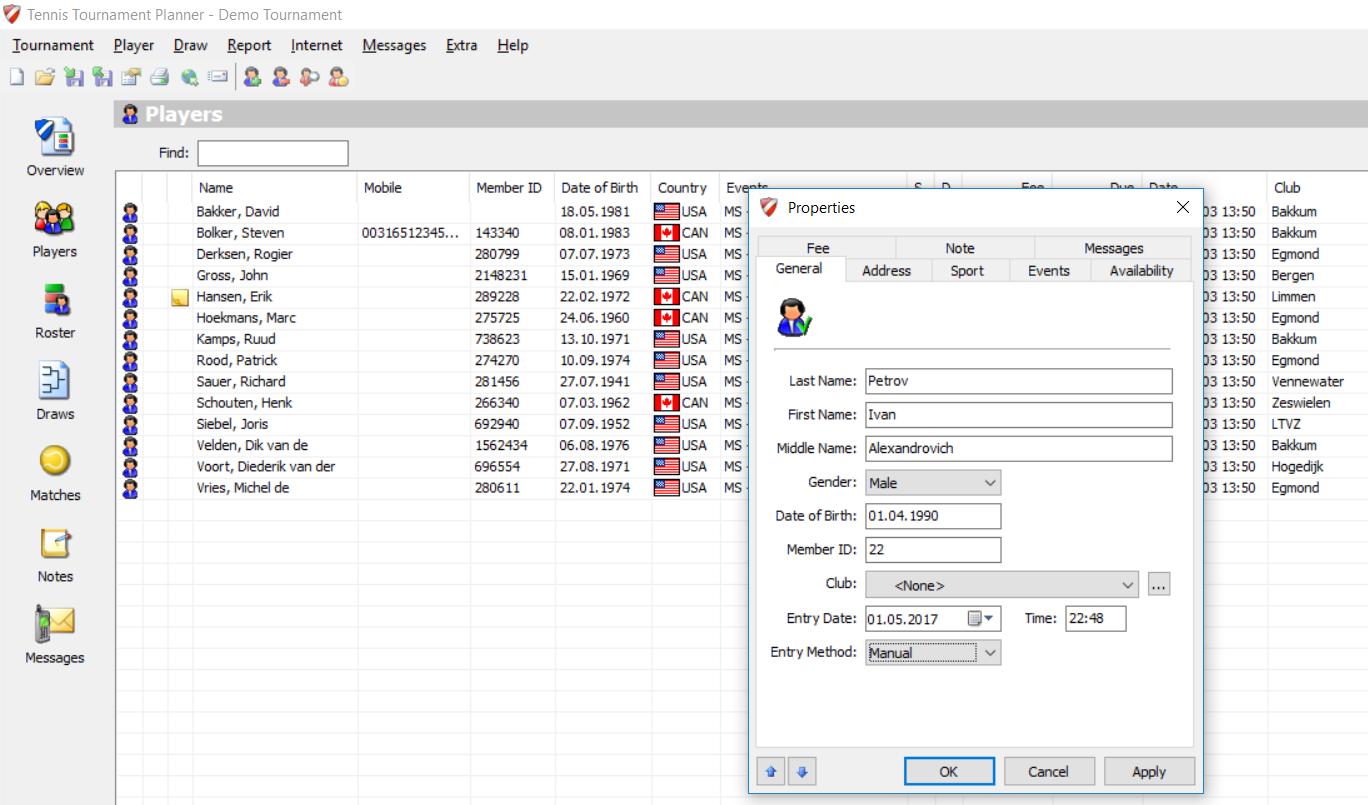
## Анализ существующих систем планирования

В процессе поиска аналогичных систем планирования турнирных матчей были найдены основные конкуренты разрабатываемой системы. Главным аналогом является приложение Tennis Tournament Manager, которое является официальным международным программным обеспечением. Также были проанализированы системы планирования Table Tennis Manager и Tournament Mango.

### Tennis Tournament Manager

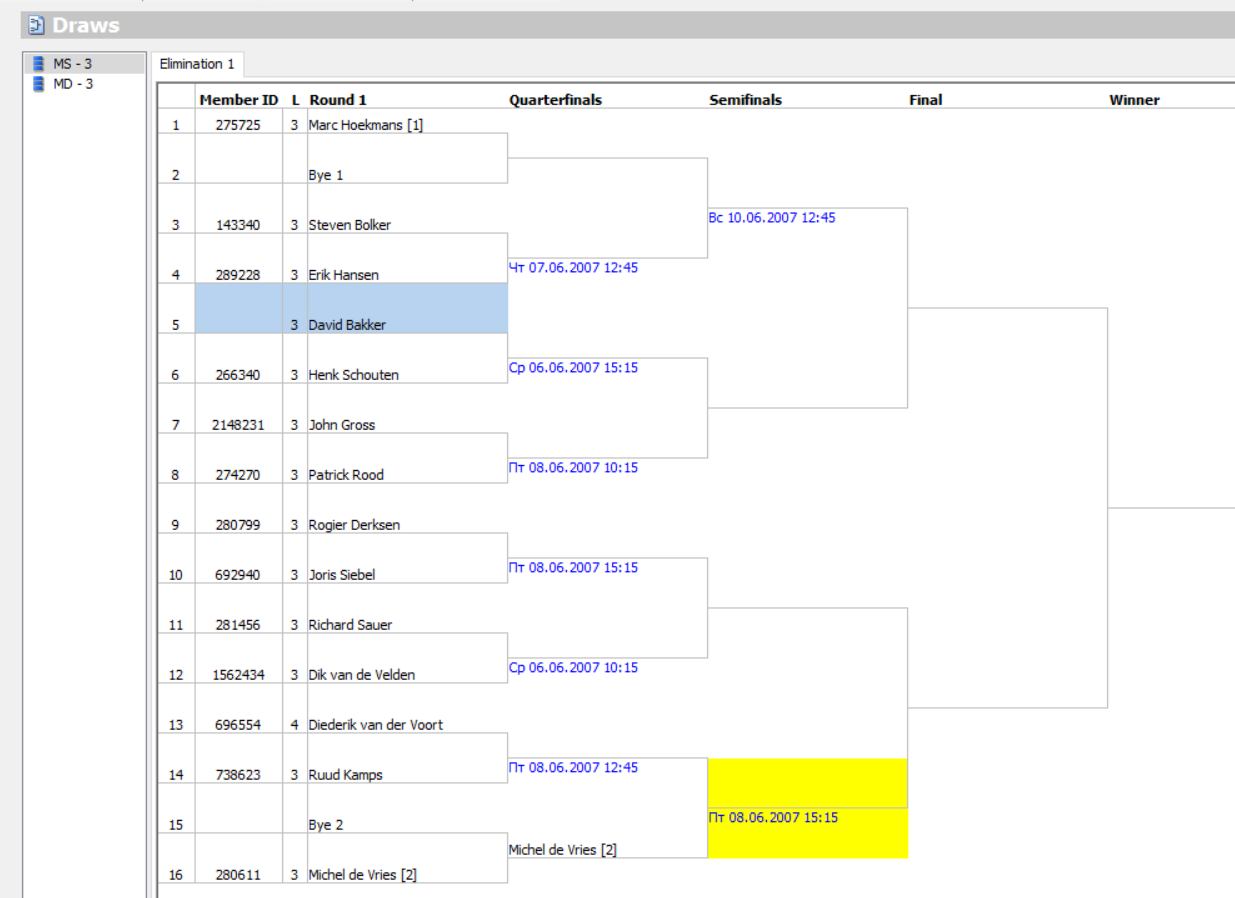
Для проведения крупных международных турниров по большому теннису, таких как Fed Cup, Davis Cup, Tennis Europe Junior Tour и ITF Beach Tennis World Championship, используется программа Tennis Tournament Planner. Демонстрационную версию программы можно скачать на сайте разработчика. Стоимость использования полной версии программы для планирования турнира составляет 150 €. [2]

Для создания и управления собственным турниром необходимо оплатить программу, разработанную Tournament Software. Финансирование международных турниров позволяет оплатить данное программное обеспечение на время проведения турнира. Менее значимые турниры зачастую не финансируются таким образом, чтобы использовать платную систему планирования. В связи с этим, было решено сделать аналог известной платной программы Tennis Tournament Planner, позволяющей реализовывать все необходимые функции для создания и планирования турнирами. Рассмотрим подробнее принцип работы платного аналога на примере демоверсии турнира. Функция добавления игрока в базу данных представлена на рисунке 7.



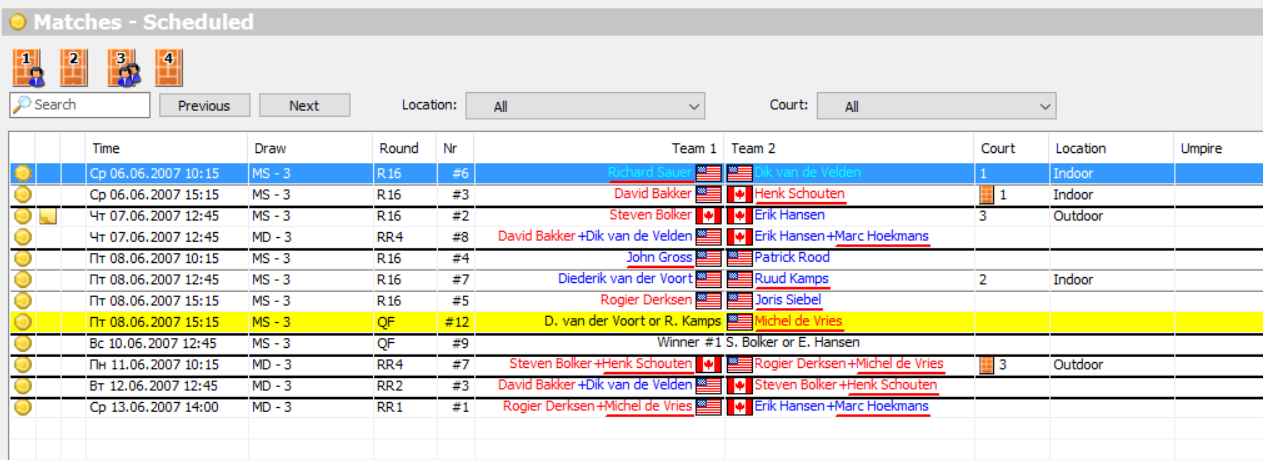
1. – Регистрация игрока на турнир

После регистрации игроков формируется сетка матчей, которая представлена на рисунке 8. Турнир в демоверсии создан по олимпийской системе. Этапы и их результаты наглядно представлены на сетке турнира, что позволяет проследить, на каком этапе каждый игрок покинет соревнование и с кем соперничал победитель турнира.



1. – Сетка матчей турнира

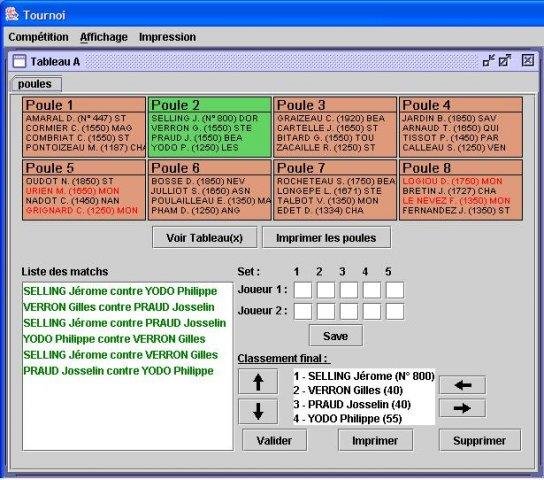
Все матчи турнира сохраняются в проекте турнира, что позволяет посмотреть статус любого матча и узнать счет встречи. Обратиться к истории матчей можно на панели инструментов по кнопке Matches (рисунок 7). Список матчей с датой проведения, игроками, раундом и месте проведения представлен на рисунке 9.



1. – Сетка матчей турнира

### Table Tennis Manager

Table Tennis Manager является системой планирования турниров по настольному теннису. Функциональность данной системы схожа с предыдущим аналогом. Расписание турнира, составленной в Table Tennis Manager представлено на рисунке 10.



1. – Расписание турнира

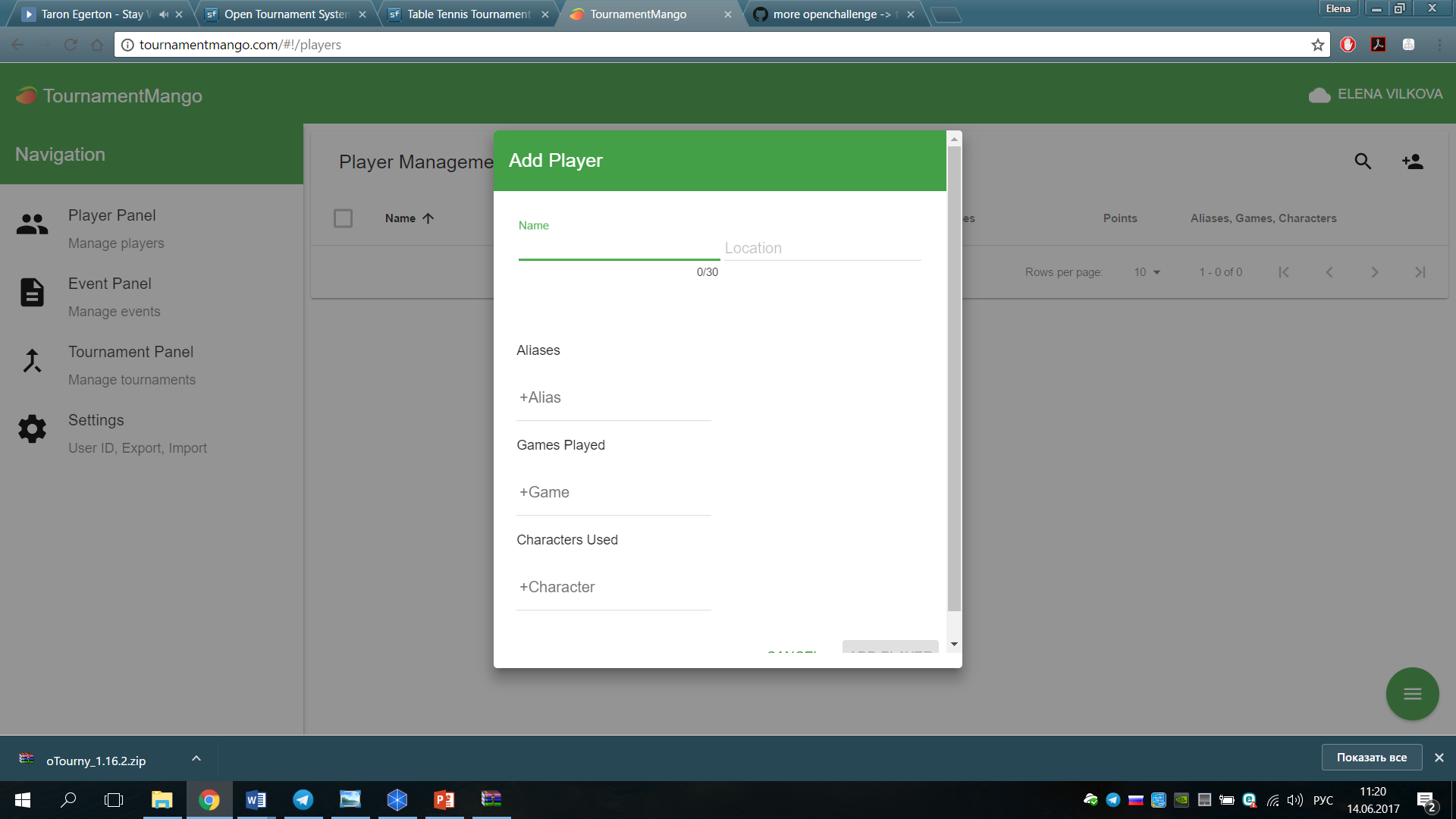
Данная программа позволяет планировать и управлять турниром, а также составлять сетку турнира, которая изображена на рисунке 11.



1. – Сетка турнира

### Tournament Mango

Система планирования и управления турнирами Tournament Mango является приложением, позволяющим хранить в базе данные об игроках, создавать турнир и управлять им как в момент подключения к Интернету, так и без доступа к сети с возможностью последующей загрузки данных после возобновления настроек сети. Функциональность Tournament Mango включает в себя добавление игроков (рисунок 12), создание турниров с различными настройками их проведения и управление ими.



1. – Добавление игрока в базу игроков

### Сравнение существующих аналогов

После того, как были сформулированы требования к разрабатываемой системе планирования турнирных матчей и рассмотрены ее аналог были составлены сравнительные характеристики систем по функциональности и атрибутам качества систем, представленные в таблицах 8 и 9 соответственно.

1. – Сравнение систем по функциональности

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Tennis Tournament Planner | Table Tennis Manager | Tournament Mango | Cистема планирования |
| Характеристики |
| Регистрация игроков | + | + | + | + |
| Создание турнира | + | + | + | + |
| Выбор модели турнира | + | + | + | + |
| Автоматическое генерирование матчей | + | + | + | + |
| Подведение итогов матчей | + | + | + | + |
| История матчей | + | + | − | + |
| Рейтинг игроков | + | − | + | + |
| Наличие внутреннего мессенджера | + | − | − | − |

1. – Сравнение систем по атрибутам качества

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Tennis Tournament Planner | Table Tennis Manager | Tournament Mango | Система планирования |
| Характеристики |
| Возможность добавлять новую модель турнира | − | − | + | + |
| Модифицируемость пользовательского интерфейса | − | − | + | + |
| Многопользовательский доступ к системе | − | − | − | + |
| Необходимость подключения к сети Интернет | − | − | − | + |
| Кроссплатформенность Windows/Linux/MacOS | + | + | + | + |
| Стоимость системы | 150 € | 0 | 0 | 0 |
| Доступ к исходному коды программы | − | − | + | + |
| Работа системы для разных конфигураций | − | − | − | + |
| Лицензия системы | closed | closed | LGPLv3.0 | EPL |

Данный анализ позволил на стадии проектировании системы выявить возможные преимущества и недоставки разрабатываемой системы планирования турнирных матчей. Таблица 8 показала, что набор функций рассматриваемых систем практически совпадает, делая эти программы аналогами, но качественные атрибуты систем, представленные в таблице 9, сильно отличаются для этих систем.

Таким образом, каждый из продуктов, представленных в таблице 9, имеет свои плюсы и минусы. По результатам сравнения выявлены явные преимущества разрабатываемой системы планирования турнирных матчей перед существующими аналогами. Среди них возможность изменения модели турнира и добавления новых видов, модифицируемость пользовательского интерфейса, многопользовательский доступ с возможностью реализации системы для разных конфигураций, а также наличие исходного кода, который любой разработчик может изменить для планирования турнира в другом виде спорта или с особенными параметрами проведения соревнования.

## Выводы к главе

В первой главе данной квалификационной работы была проведена работа по изучению предметной области, выявлению заинтересованных сторон системы, выявлению ее функций и атрибутов качества.

На основе заинтересованных сторон и анализа предметной области была проведена работа по определению целей заинтересованных сторон и определению сценариев и ограничений системы.

Далее были выявлены основные функции системы, представленные в пункте 1.3 и отображенные на диаграмме (рисунок 6).

На основе функций системы и выявленных сценариев и ограничений системы была проведена работа по выявлению атрибутов качества системы, которые представленные в пункте 1.4.

Результатом всей описанной выше работы является техническое задание, представленное в приложении А.

# Конструкторская часть

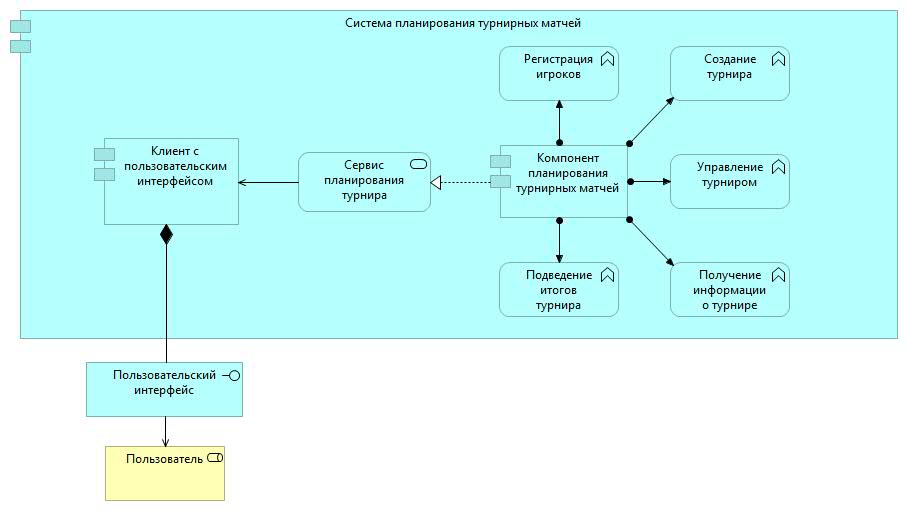
В начале данной части квалификационной работы представлена высокоуровневая концептуальная архитектура системы планирования турнирных матчей. Далее описан принцип работы используемой технологии модульности Open Services Gateway Initiative (OSGi) и компонента Eclipse Communication Framework (ECF), которые используются для реализации многопользовательского доступа к системе согласно техническому заданию.

Для перехода от концептуальной архитектуры к компонентной декомпозиции представлены два варианта схемы работоспособности системы, которые описывают принцип работы системы для одного и двух устройств. Проектирование компонентной архитектуры позволяет перейти к двум вариантам развертывания системы на оборудовании, которые описывают все устройства, функционирующие в системе планирования.

Заключительным этапом конструкторской части является детальное проектирование, которое более подробно описывает реализацию системы. В основе детального проектирования лежат UML-диаграммы, поэтому реализация системы представлена с помощью диаграмм классов, состояний и последовательностей.

## Концептуальная архитектура

Разрабатываемая система обладает высокомодульной структурой, что подразумевает работу системы по шаблону «клиент-сервер», а именно наличие сервера с управляемым турниром и клиента, который имеет доступ к нужному турниру через сервер. Таким образом, высокоуровневое представление система планирования турнирных матчей изображено на рисунке 13.



1. – Концептуальная архитектура системы планирования

Поскольку взаимодействие в системе происходит по шаблону «клиент-сервер», конструкция клиентов и серверов предполагает наличие в них совместимых интерфейсов, находящихся в сервисе планирования турнира (рисунок 13). [1] Этому содействует практика применения простых протоколов передачи сообщений. Результат соответствует положениям ряда тактик реализации модифицируемости: «поддержания стабильности интерфейсов», «замены компонентов» и «применения предписанных протоколов».

Для того, чтобы все компоненты системы работали вместе без ошибок, необходимо что бы они все были написаны на одном и том же языке. По требованиям технического задания, основным языком программирования системы явялется язык программирования Java, который позволит использовать систему на большинстве современных операционных систем. [3]

Для взаимодействия модулей системы необходимо было использовать технологию по организации взаимодействия бандлов OSGi-сервисы. Рассмотрим подробнее данную спецификацию и ее применение для разрабатываемой многомодульной системы.

## Спецификация OSGi

OSGi (Open Services Gateway Initiative) – спецификация динамической плагинной (модульной) шины для создания Java-приложений, разрабатываемая консорциумом OSGi Alliance. [4]

Основная идея фреймворка OSGi – в системе есть плагины (бандлы (bundles)). Основной способ взаимодействия между бандлами – сервисы: объекты, зарегистрированные в ядре системы с заявленными реализованными интерфейсами. Бандлы регистрируют сервисы для предоставления определенной функциональности другим бандлам. Помимо этого, OSGi предоставляет механизм создания и обработки событий, управление импортом/экспортом java-пакетов и библиотек, набор класслоадеров, методы адресации ресурсов. [5]

Понятие "динамическая шина" обозначает, что можно на лету, т.е. не перезапуская приложение, устанавливать, подключать, отключать и обновлять модули системы.

Бандл платформы OSGi (OSGi bundle) содержит java-классы и другие ресурсы, которые вместе могут реализовывать некие функции, а также предоставлять сервисы и пакеты другим плагинам. Конструктивно бандл может быть либо каталогом, либо jar-архивом. [13]

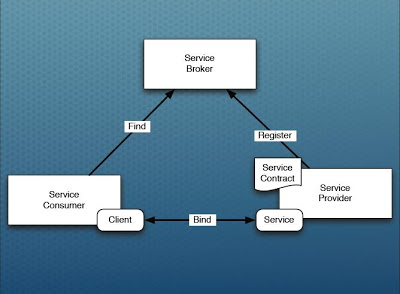
Взаимодействие бандлов можно осуществить с посредством указания их зависимостей. Однако в OSGi есть более гибкий способ взаимодействия бандлов, который называется сервисы.

### OSGi-сервисы

Спецификация OSGi задумывалась как среда выполнения и взаимодействия сервисов. Принцип работы заключается в следующем: один бандл выставляет сервисы, а другие их используют. Данный способ требует активации бандла, потому что выставление и подключение к сервисам, как правило, осуществляется в активаторе бандла. Активация бандла требует расходов на управление жизненным циклом бандлов, но в тоже время обеспечивает динамичность всей системы.

На рисунке 14 представлена схема взаимодействия сервисов. Один Java-объект, который называется Service, представляет некий интерфейс доступа к себе - контракт сервиса. Другой Java-объект, который называется Client, может взаимодействовать с сервисом, через предоставляемый тем контракт.

Чтобы данное взаимодействие было возможно Client должен найти нужный ему Service и получить к нему доступ. Для этого используется Service Broker, в роли которого предстает OSGi-реестр сервисов. Бандл, предоставляющий сервис, регистрирует его в реестре, а бандл-клиент извлекает сервис оттуда и использует. [13]



1. – Схема взаимодействия сервисов OSGi

### Eclipse Communication Framework

Eclipse — это прежде всего качественная платформа для построения самых разных приложений. Основным компонентом платформы является Equinox — реализация спецификации OSGi R4. На базе Equinox строятся другие компоненты, такие, как, например, Eclipse Communication Framework, о котором мы сегодня и поговорим.

Большинство создаваемых сегодня приложений работают в сетевой среде. Для создания программ, основанных на OSGi (в частности - Eclipse RCP - приложений), был разработан ECF.

ECF – это реализация концепции распределенных контейнеров, осуществляющих передачу данных по различным прикладным протоколам. Контейнеры могут обеспечивать связь с сохранением состояния (т.е. с сессией) и без сохранения. Поддерживается взаимодействие типа "точка-точка" (например, клиент - сервер) и типа "публикатор - подписчики". Само взаимодействие может быть, как синхронным, так и асинхронным.

ECF предоставляет единый API к контейнерам, обеспечивающим различные протоколы связи, причем этот API расширяемый и допускает использование возможностей, характерных для того или иного протокола. Обеспечивается такая возможность за счет того, что API многоуровневый. ECF в OSGi-среде содержит реализацию Distributed OSGi (RFC 119) для Equinox. [6]

Для реализации многомодульной системы и возможности обращения нескольких пользователей к одному сервису необходима специальная технология.

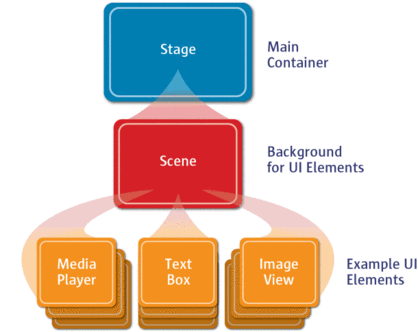
Некоторые контейнеры Eclipse Communication Framework поддерживают несколько разнородных API. В частности, R-OSGi и Generiс контейнеры, а также появившийся REST контейнер, поддерживают API вызова удаленных сервисов, так называемый. Remote Services API.

ECF Remote Services API это способ обеспечения работы бандлов в распределенной среде. Точно так же одни бандлы могут выставлять сервисы, доступные другим бандлам на другой JVM (и, естественно, даже на другой машине). Данная система, во-первых, не стандартизирована (т.е. это не OSGi подсистема, а часть непосредственно ECF), а, во-вторых, - более гибка, т.к. можно явно указывать, где находятся бандлы, выставляющие сервисы, (т.е. отпадает необходимость в процедуре поиска сервисов) и использовать для обеспечения взаимодействия различные протоколы и реализации.

## Платформа JavaFX

JavaFX — платформа на основе Java для создания приложений с насыщенным графическим интерфейсом. Может использоваться как для создания настольных приложений, запускаемых непосредственно из-под операционных систем, так и для интернет-приложений (RIA), работающих в браузерах, и для приложений на мобильных устройствах.

Основной управляющий класс расширяет класс Application и содержит два метода. Это базовая структура, которая необходима для запуска приложения JavaFX. Метод start(Stage primaryStage) автоматически вызывается при вызове метода launch(...) из метода main. Метод start(...) в качестве параметра принимает экземпляр класса Stage. На рисунке 15 представлена структура любого приложения JavaFX.



1. – Структура приложения JavaFX [7]

Объект Stage является основным контейнером, который, как правило, представляет собой обрамлённое окно со стандартными кнопками: закрыть, свернуть, развернуть. Внутрь Stage добавляется сцена (Scene), которая может быть заменена другой Scene. Внутрь Scene добавляются стандартные компоненты типа AnchorPane, TextBox и другие.

### FXDiagram

FXDiagram – это фреймворк для создания инструментов диаграмм на основе JavaFX. Доустановив этот фреймворк в свой проект, разработчик все еще может использовать весь функционал JavaFX. FXDiagram предоставляет дополнительные классы библиотеки, чтобы выборочно добавлять функциональные элементы более высокого уровня с целью улучшения пользовательского интерфейса.

Исходный код данного фреймворка доступен для пользователей с возможностью изучения примеров конструирования объектов и их дальнейшего использования в JavaFX проектах. FXDiagram почти полностью написан на языке Xtend, что обеспечивает краткий и читаемый код, будучи полностью совместимым с Java. FXDiagram работает с обычными Java-приложениями, но также хорошо интегрируется с Eclipse. [8]

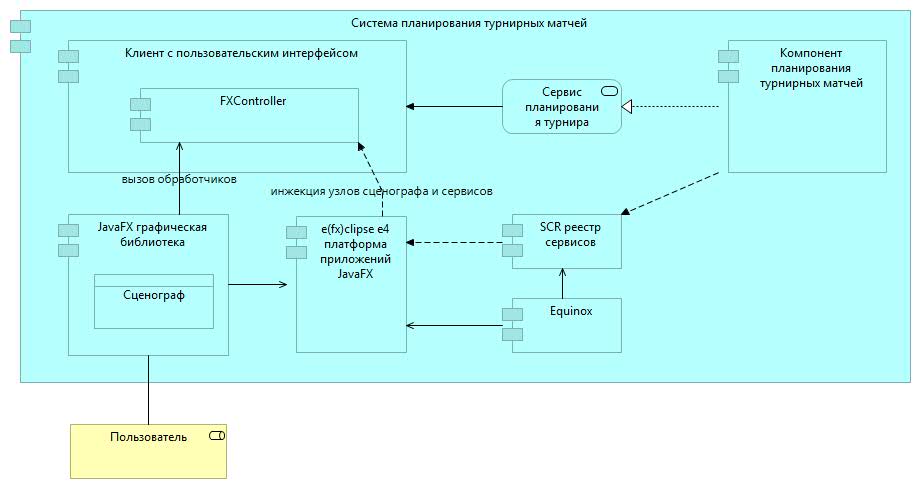
## Компонентная декомпозиция архитектуры

После того, как были выбраны все необходимые технологии и представлена высокоуровневая концептуальная архитектура системы, можно перейти к составлению диаграмм компонентной декомпозиции

Разрабатываемая система должна реализовывать многопользовательский доступ с возможностью работы системы на разных конфигурациях. Таким образом, представим компонентную декомпозицию для системы, где клиент и компонент планирования турнирных матчей взаимодействуют на одном устройстве и на разных.

### Компонентная декомпозиция архитектуры для одного устройства

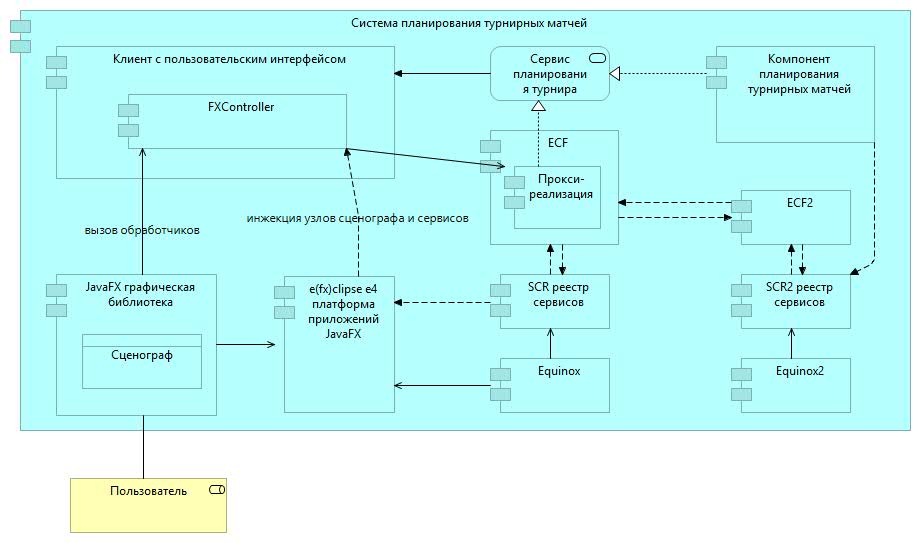
При работе системы на одном устройстве компонент планирования сообщает данные о регистрируемых сервисах в реестр сервисов. Графическая платформа JavaFX содержит сценограф пользовательского интерфейса, обработчики элементов которого вызываются в контроллерах. Платформа приложений e(fx)clipse позволяет инжектировать узлы сценографа и сервисов из реестра в клиент. Компонентная декомпозиция системы для одного устройства представлена на рисунке 16.



1. – Компонентная декомпозиция системы для одного устройства

### Компонентная декомпозиция системы для двух устройств

При работе системы на двух устройствах компонент планирования сообщает данные регистрируемых сервисов в свой реестр, который с помощью взаимодействия компонентов ECF создает прокси-сервис, передающий данный в реестр сервисов клиента, а далее взаимодействие программы происходит аналогично предыдущему случаю. Все данные клиенту поступают от прокси-реализации сервиса. Компонентная декомпозиция системы для двух устройств представлена на рисунке 17.



1. – Компонентная декомпозиция системы для двух устройств

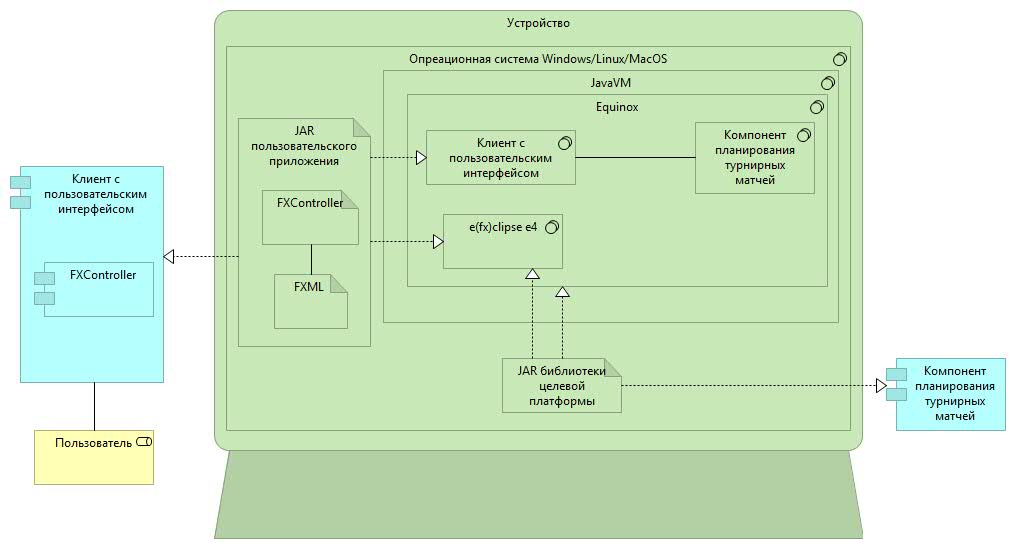
## Развертывание системы на оборудовании

Разрабатываемая система является высокомодульным приложением, поэтому для двух компонентных декомпозиций архитектуры представлены соответствующие варианты развертывания системы планирования турнирных матчей на оборудовании. [14]

### Развертывание системы на одном устройстве

На диаграмме, изображенной на рисунке 18, показано развертывание системы на оборудовании, где:

* Устройство, на котором запускается система;
* Операционная система в которой запускается система – Microsoft Windows/Linux/MacOS;
* Клиент и компонент планирования турнирных матчей работают на одной виртуальной машине JavaVM.

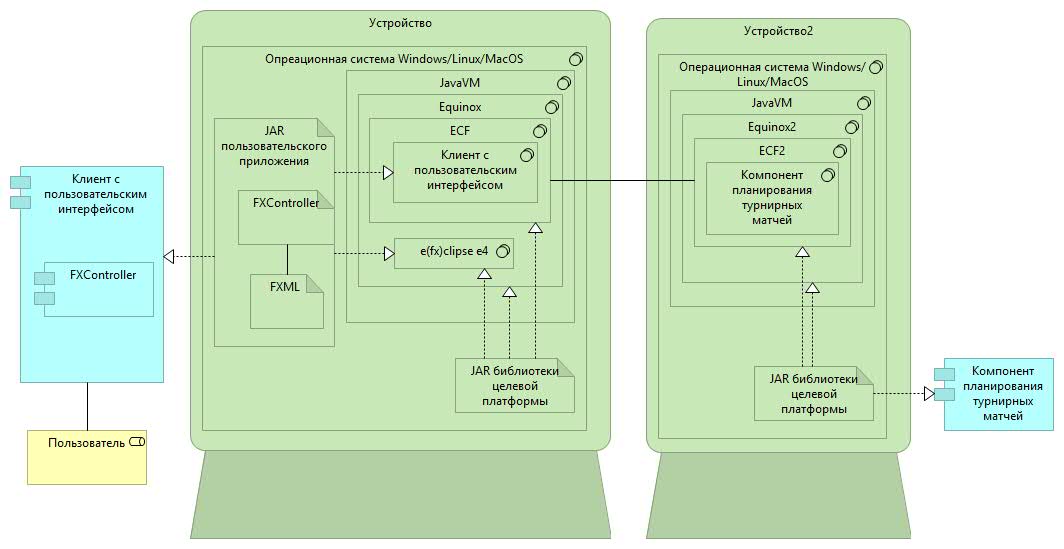


1. – Развертывание системы планирования турнирных матчей на одном устройстве

### Развертывание системы на двух устройствах

На диаграмме, изображенной на рисунке 19, показано развертывание системы на оборудовании, где:

* Устройства, на которых запускается система;
* Операционная система на устройствах, в которой запускается система – Microsoft Windows 10/Linux/MacOS;
* Клиент и компонент планирования турнирных матчей работают на разных виртуальных машинах JVM и взаимодействуют с помощью двух компонентов Eclipse Communication Framework (ECF и ECF2), реализованных на соответствующих реализациях Equinox.



1. – Развертывание системы планирования турнирных матчей на двух устройствах

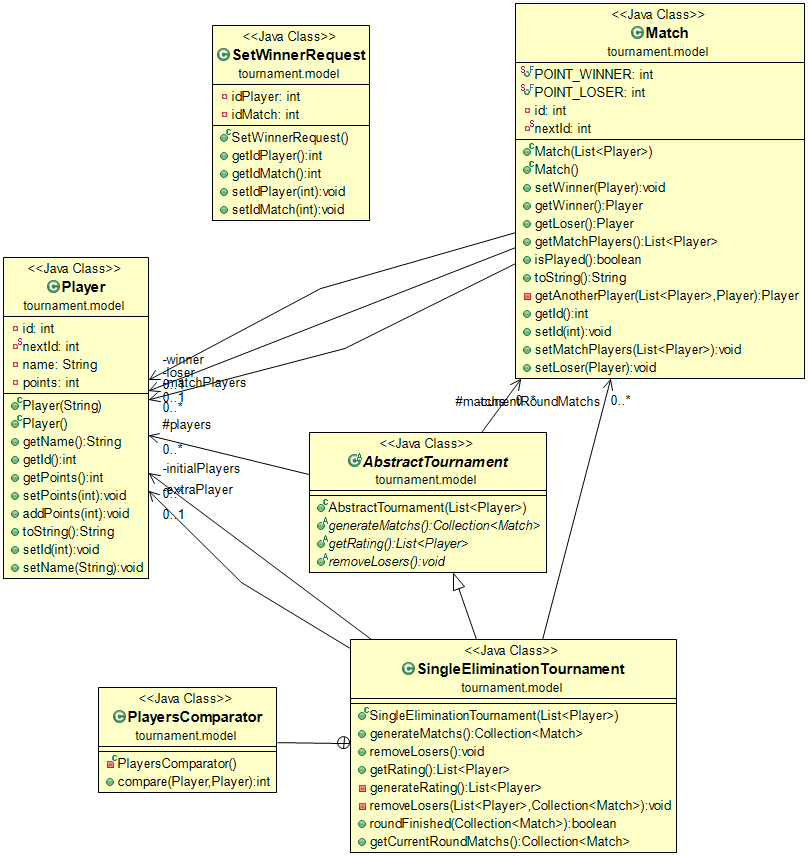
## Детальное проектирование

После того, как была представлена варианты компонентной декомпозиции системы и варианты ее развертывания на оборудовании, необходимо более детально представить реализацию программы с использованием вышеописанных выбранных технологий. Разрабатываемую систему планирования турнирных матчей было решено разделить на три составных блока и представить их взаимодействие с помощью диаграмм классов. [12]

### Диаграммы классов

#### Модель турнира

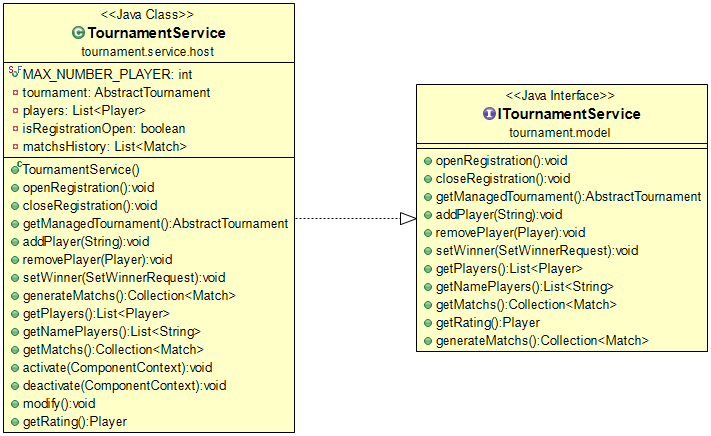
Основной единицей планирования и управления в системе является турнир. Для реализации модели турнира были выбраны такие главные элементы, как игрок, матч и вид турнира. Диаграмма классов модели турнира представлена на рисунке 20.



1. – Диаграмма классов модели турнира

#### Сервис планирования турнирных матчей

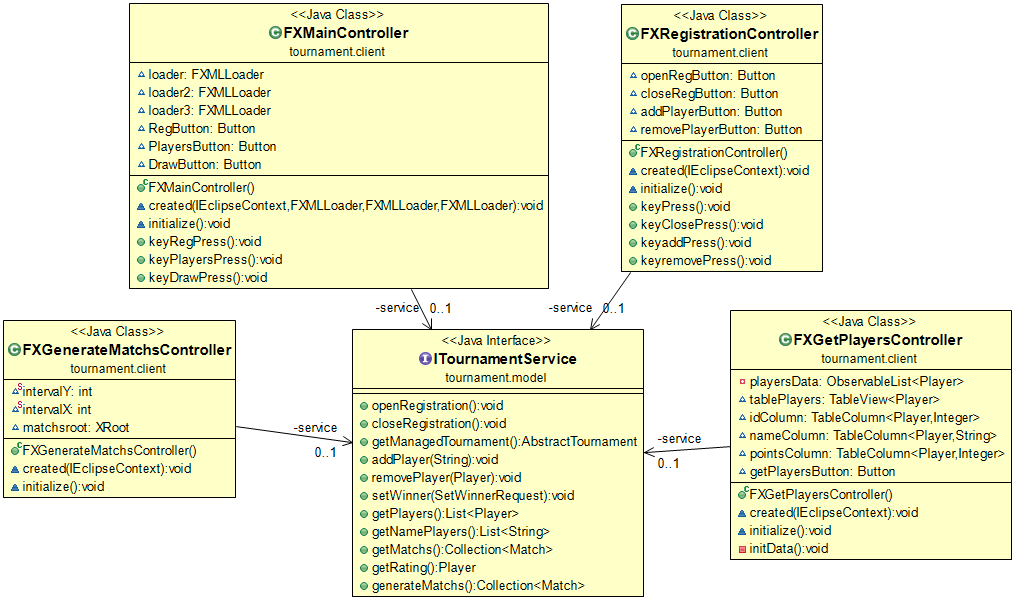
Интерфейс сервиса планирования задает методы по управлению турниром. Класс TournamentService реализует методы интерфейса ITournamentService и активирует компонент с целью создания сервера на выбранном порту. Все методы сервиса планирования турнирных матчей представлены на рисунке 21.



1. – Диаграмма классов сервиса планирования

#### Взаимодействие клиента и сервера

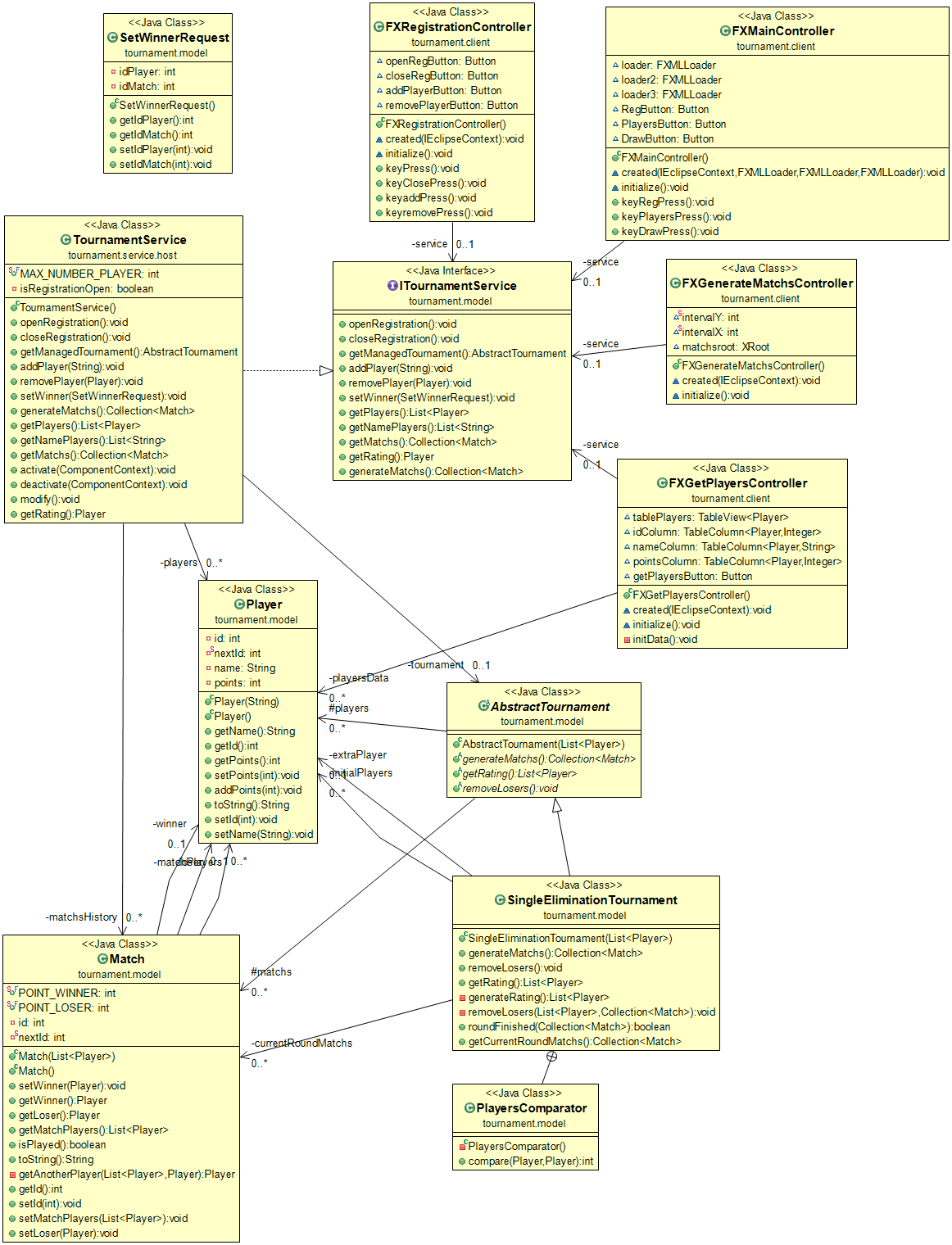
Для доступа пользователей к сервису планирования, им необходимо иметь доступ к интерфейсу ITournamentService, таким образом все контроллеры графического интерфейса должны знать об используемом сервере и использовать данные одного сервиса планирования. Диаграмма классов клиента представлена на рисунке 22.



1. – Диаграмма классов взаимодействия клиента и сервера

#### Диаграмма классов системы планирования турнирных матчей

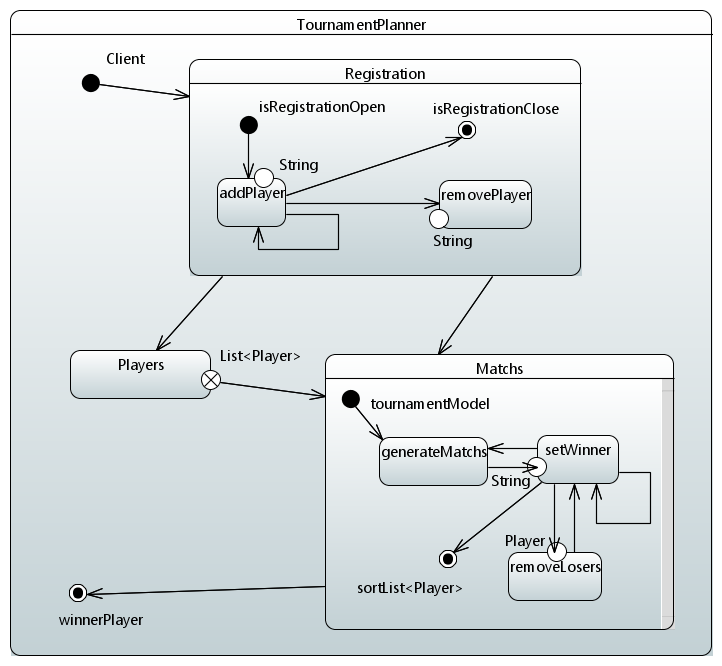
Проанализировав диаграммы классов каждого блока системы, представим на рисунке 23 диаграмму классов всей разрабатываемой системы планирования турнирных матчей.



1. – Диаграмма классов системы

### Диаграмма состояний конечного автомата

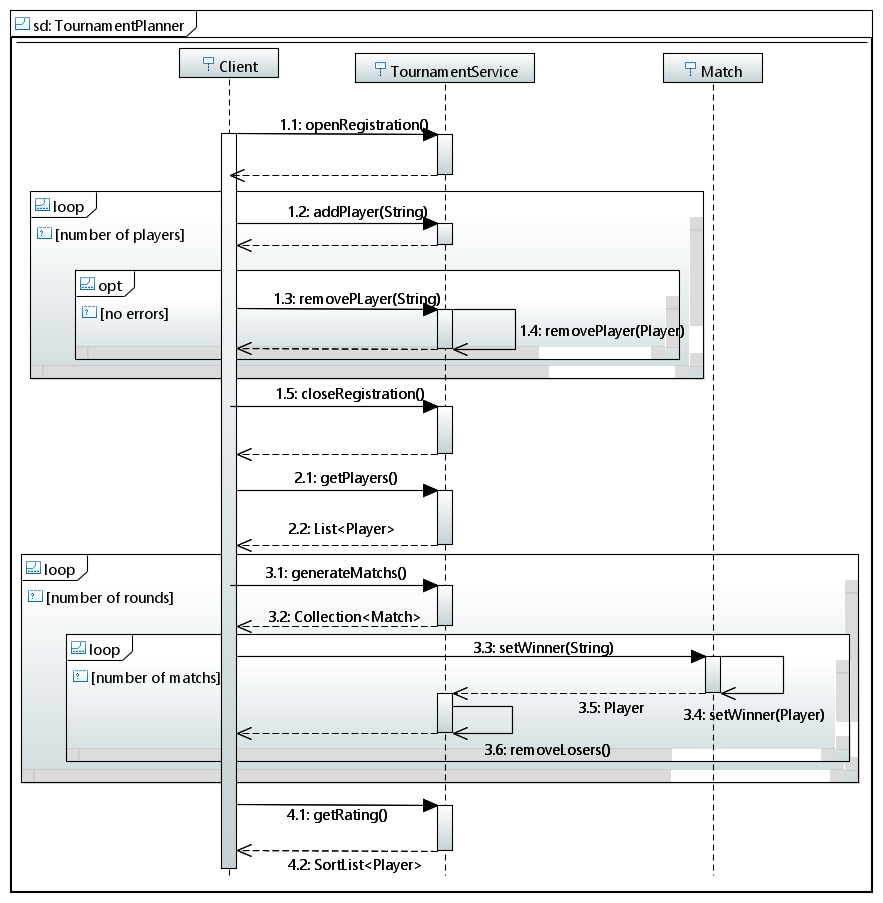
Клиент взаимодействует с компонентом планирования турнирных матчей по определенному плану, представленному в виде диаграммы состояний конечного автомата (рисунок 24). Данный процесс можно разделить на несколько блоков: регистрация, создание турнира, генерирование матчей и окончание турнира.



1. – Диаграмма состояний конечного автомата

### Диаграмма последовательностей

Диаграмма состояний позволяет проследить за состоянием системы в разные моменты работы программы. А хронология взаимодействия клиента и компонента планирования TournamentService с вызываемыми методами представлена на диаграмме последовательностей (рисунок 25).



1. – Диаграмма последовательностей

## Вывод к главе

В главе 2 этой квалификационной работы был описан процесс работы всех основных технологий, которые использовались в реализации системы планирования турнирных матчей.

В начале была построена концептуальная модель системы и в качестве основной технологии использовались OSGi-сервисы. Главным компонентом реализации спецификации Equinox является компонент Eclipse Communication Framework, реализующий взаимодействие клиента с пользовательским интерфейсом и компонента планирования турнирных матчей при их взаимодействии на разных устройствах.

После изучения принципа работы ECF была встроена в систему графическая платформа JavaFX с библиотекой FXDiagram для реализации пользовательского интерфейса и платформа приложений e(fx)clipse e4, позволяющая инжектировать узлы сценографа графического интерфейса и сервисы системы планирования.

На основе вышеописанных технологий были спроектированы два варианты компонентной декомпозиции системы, а также представлены варианты развертывания системы на оборудовании для одного и нескольких устройств.

Заключительным этапом конструкторской части является построение UML-диаграмм, а именно диаграмм классов подсистем и всей системы в целом, диаграммы состояний и последовательностей управления турниром, представляющих детальное проектирование системы планирования турнирных матчей.

.

# Технологическая часть

В технологической части данной работы описаны используемые средства, показан способ настройки IDE и получения бинарной сборки системы, а также определены методы и уровни тестирования и составлен план тестирования. На основе вышеописанных действий было проведено тестирование и приведены его результаты для полученной сборки системы.

## Используемые средства

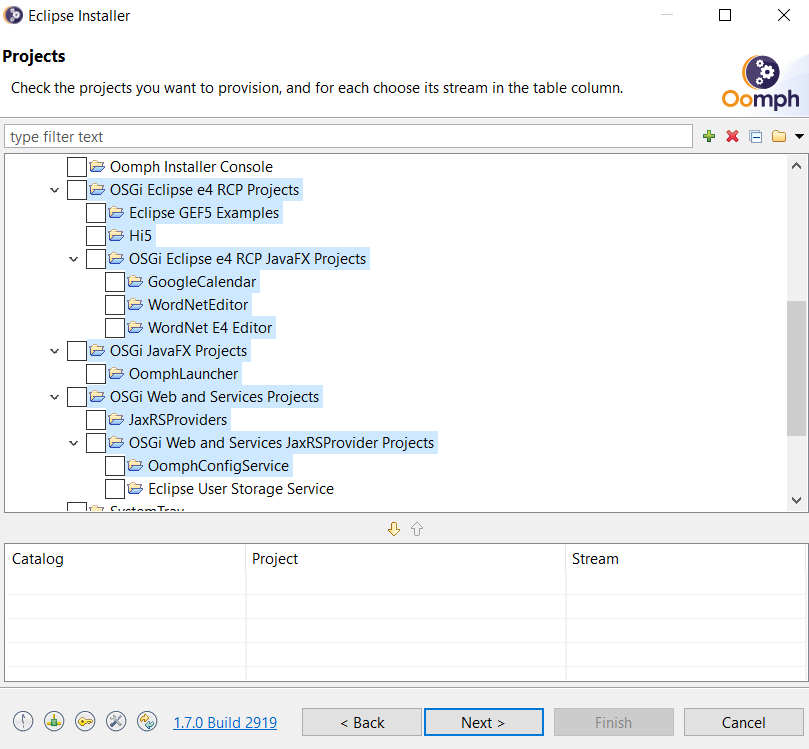
В данной работе использовались следующие средства:

* среда разработки Eclipse. Eclipse—свободная интегрированная среда разработки модульных кроссплатформенных приложений, развивается и поддерживается Eclipse Foundation;
* система контроля версий Git [15];
* язык программирования Java;
* операционная система Microsoft Windows 10;
* система установки IDE Eclipse Oomph.

## Настройка Eclipse IDE

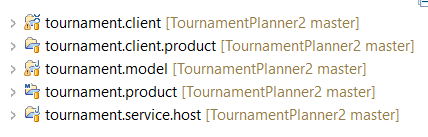
Для того, чтобы настроить среду разработки Eclipse IDE в рамках использования разработанной системы планирования турнирных матчей, необходимо:

* выполнить настройки в соответствии с используемой операционной системой Microsoft Windows 10, используя помощник: [https://github.com/agentlab/ru.agentlab.calendar/wiki/Настройки-Eclipse-IDE](https://github.com/agentlab/ru.agentlab.calendar/wiki/%D0%9D%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0-Eclipse-IDE);
* переключить каталог продуктов GitHub на AgentLab IDE Product;
* в окне Projects необходимо добавить файлы конфигурации, при помощи функции добавления проектов (зеленый знак «плюс»). Необходимо добавить OSGi Projects;
* после успешного добавления всех необходимых файлов конфигурации, для дальнейшей работы с системой необходимо выбрать проекты, выделенные на рисунке 26 и файл конфигурации Tournament.



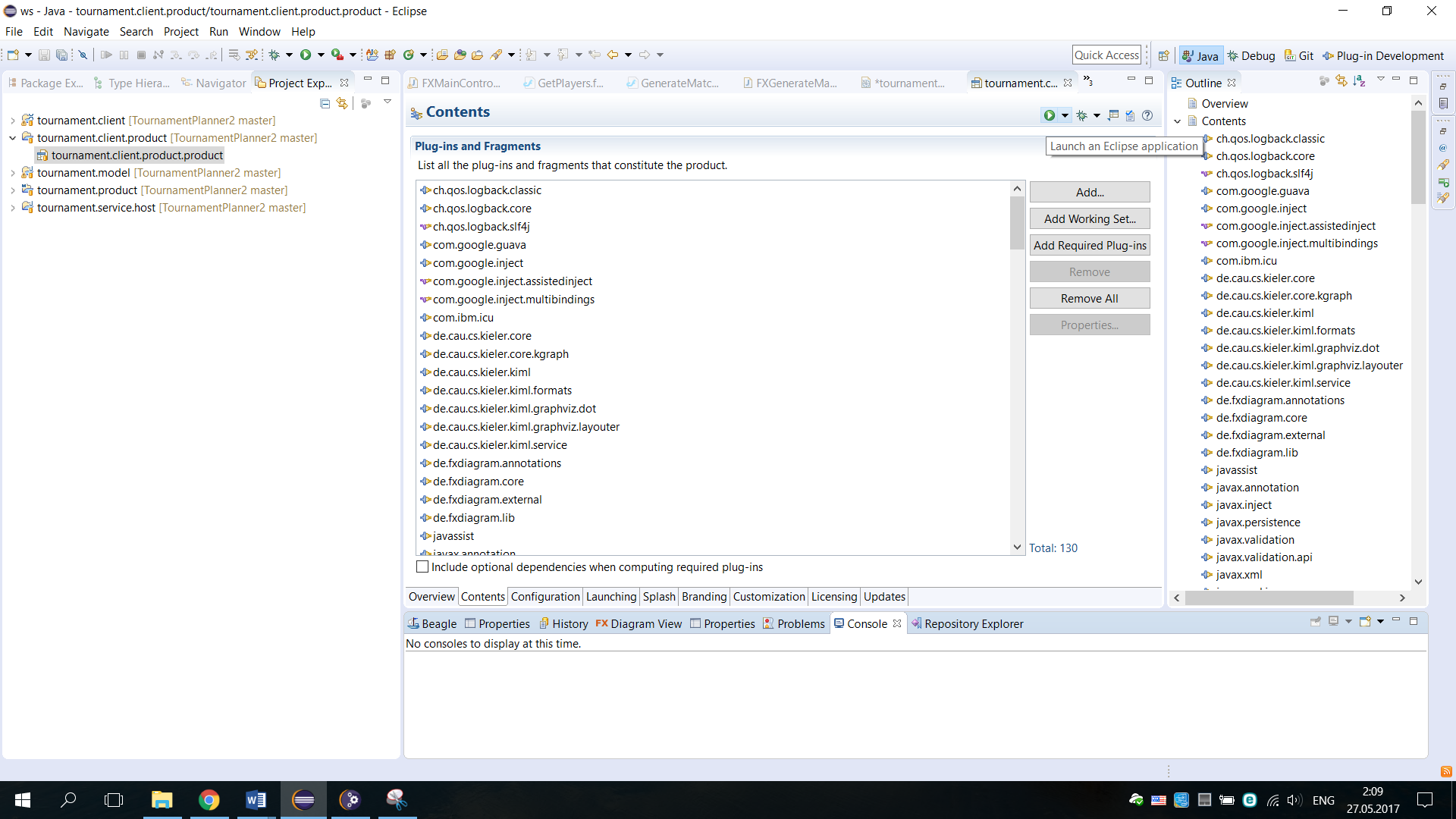
1. – Окно Projects

* после нажатия кнопки Finish на последнем окне Eclipse Installer начнется процесс скачивания выбранной IDE и выбранных проектов. После того как процесс скачивания завершится, откроется IDE Eclipse, где панель Package Explorer должна выглядеть также, как на рисунке 27.



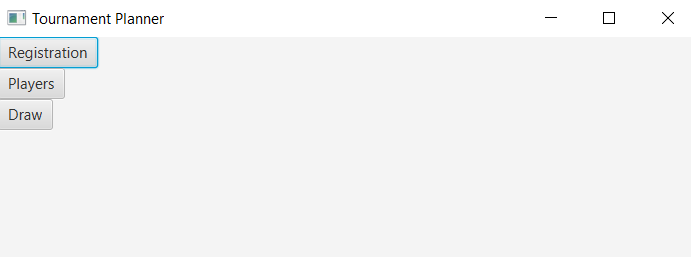
1. – Окно Package Explorer

* для запуска системы необходимо развернуть папку tournament.client.product, выбрать файл tournament.client.product.product и нажать на зеленую стрелочку запуска. Это действие показано на рисунке 28.



1. – Запуск системы

* после нажатия на кнопку запуска должно открыться окно приложения, с запущенным в нем сервисе планирования турнирного матча. Данное окно представлено на рисунке 29.



1. – Запущенная система планирования

Исходный код программы можно скачать по ссылке <https://github.com/agentlab/TournamentPlanner2>.

## Тестирование

### Планирование процесса тестирования

Для тестирования разрабатываемой системы планирования турнирных матчей была выбрана широко используемая методика тестирования программного обеспечения, в которую входят модульное, интеграционное и системное тестирования, которые проводятся в определенном порядке.

На начальном этапе разработки системы и создания классов проводится модульное тестирование. Как подсказывает название, это метод испытания на объектном уровне. Отдельные программные компоненты тестируются на наличие ошибок. Для разных модулей создаются юнит-тесты, которые проверяют, работают ли функции так, как задумывалось.

Следующим этапом тестирования является интеграционный тест. Отдельные модули, которые уже были подвергнуты модульному тестированию, интегрируются друг с другом, и проверяются на наличие неисправностей. Такой тип тестирования в первую очередь выявляет ошибки интерфейса и взаимодействия между компонентами, подсистемами или системами.

После проверки функциональности отдельных модулей и подсистем, переходи к анализу и тестированию кода всей системы в целом. В этом тестировании, вся система проверяется на наличие ошибок и багов. Этот тест осуществляется путем сопряжения аппаратных и программных компонентов всей системы, и затем выполняется ее проверка. [11]

Перед началом тестирования системы планирования турнирных матчей был разработан план тестирования, представленный в таблице 10.

После этого было проведено тестирование и полученные результаты были представлены в следующем пункте.

1. – План тестирования

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Объект тестирования** | **Уровень тестирования** | **Тестируемая ошибка** | **Виды деятельности** | **Частота тестирования** | **Ответственный исполнитель** |
| Класс SingleElimination.java | Модульный | Неработоспособность алгоритма | JUnit тест по планированию демо-версии турнира | До 2000 игроков | Тестировщик |
| Компонент  TournamentService | Интеграционный | Проблема создания сервера | Активация сервера TournamentService | До момента сдачи продукта заказчику | Тестировщик |
| Интерфейс ITournamentService | Интеграционный | Клиент и компонент планирования не взаимодействуют | Взаимодействие между клиентом и компонентом планирования через интерфейс сервиса | До момента сдачи продукта заказчику | Тестировщик |
| Пользовательский интерфейс | Системный | Проблемы при планировании турнира | Создание и управление турниром | До момента сдачи продукта заказчику | Тестировщик |

### Разработка тестовых случаев

Согласно плану тестирования, представленному в предыдущем пункте, на первом этапе проводится модульное тестирование, а именно был создан JUnit тест, представленный на рисунке 30. Данный тест позволяет проверить методы и алгоритмы планирования матчей в искусственно созданном турнире с различным количеством игроков и определить победителя турнира. [9]



1. – JUnit тест по определению победителя матча

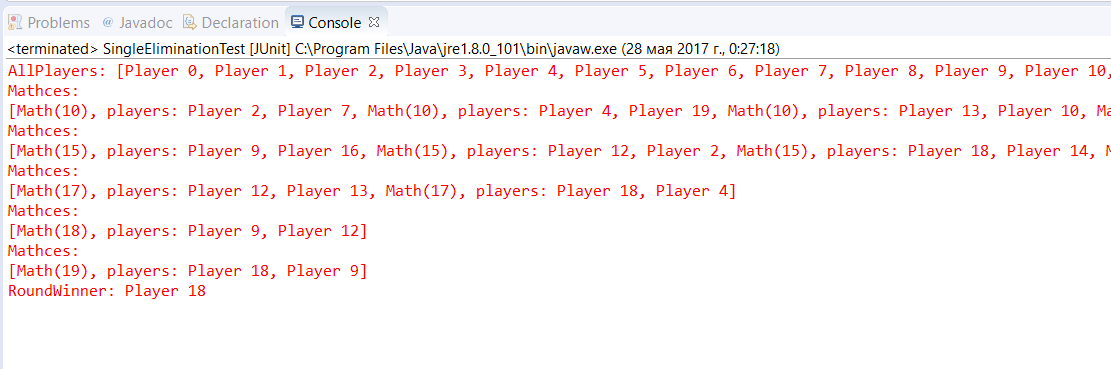
На этапе интеграционного тестирования компонента TournamentService необходимо получить информацию об активации сервера на локальном хосте 8080.

Следующим этапом интеграционного тестирования является тестирование связи клиента и сервера. Данный тест проводится путем отслеживания взаимодействия через интерфейс ITournamentService.

Завершающим этапом тестирования является тестирование на системном уровне. Результатом разработки системы планирования турнирных матчей является пользовательский интерфейс управления, который по требованиям ТЗ должен быть удобным и понятным в эксплуатации. В рамках тестирования были рассмотрены основные действия приложения, которые являются обязательными функциями для пользователей.

### Результаты тестирования

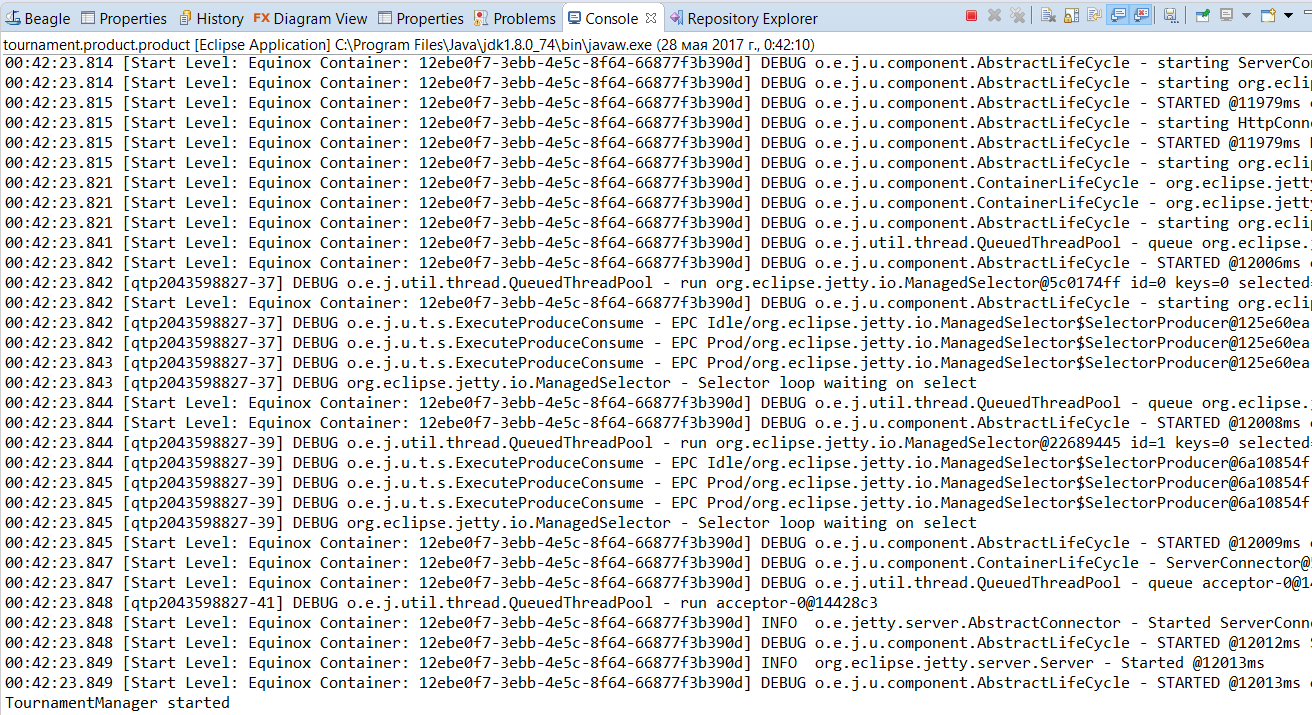
Результат модельного тестирования представлен на рисунке 31. Из данного рисунка видно, что в рамках созданного турнира определяются победители каждого матча в каждом раунде и итогом теста выводится победитель всего турнира.



1. – Результаты JUnit теста по определению победителя матча

Данное тестирование проводилось и для большего количества игроков, до 2000 игроков. Алгоритм автоматического генерирования был успешно протестирован с помощью JUnit тестов.

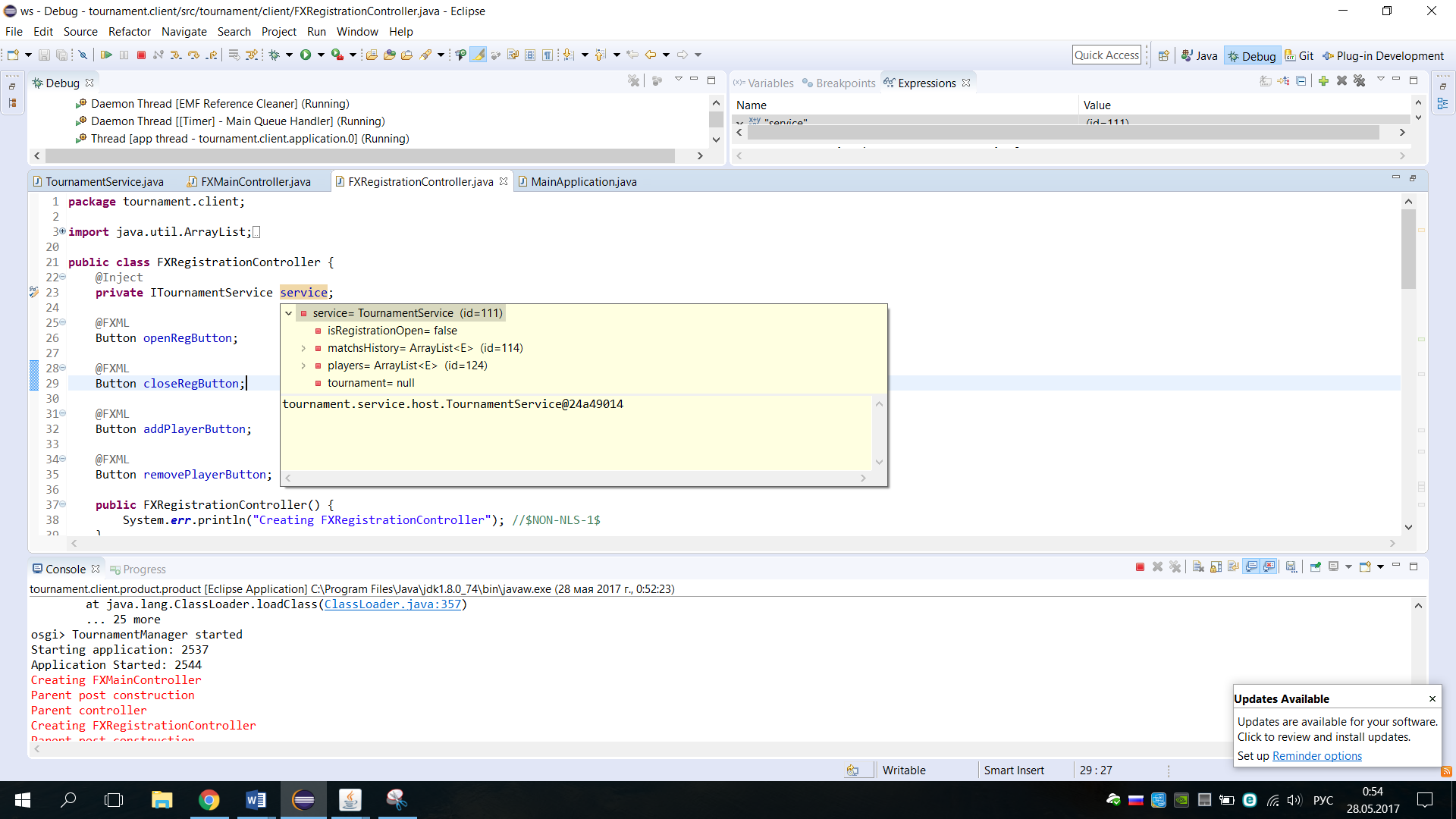
На рисунке 32 представлен результат одного из теста интеграционного уровня.



1. – Результаты теста активации сервера

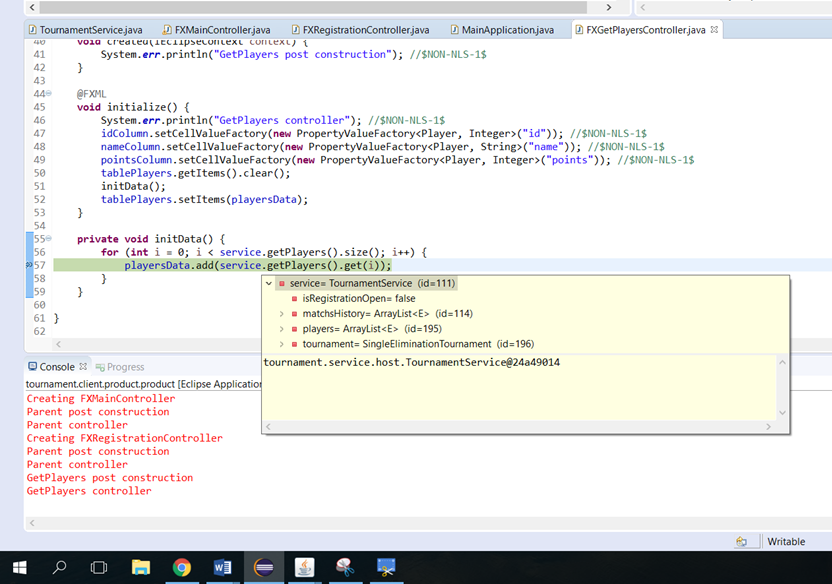
Результатом запуска продукта является вывод в консоли строки TournamentManager started, которая сообщает об успешной активации компонента, создающего сервер.

Результат следующего этапа интеграционного тестирования, где клиент обнаруживает необходимый сервер через ITournamentService, представлен на рисунке 33.



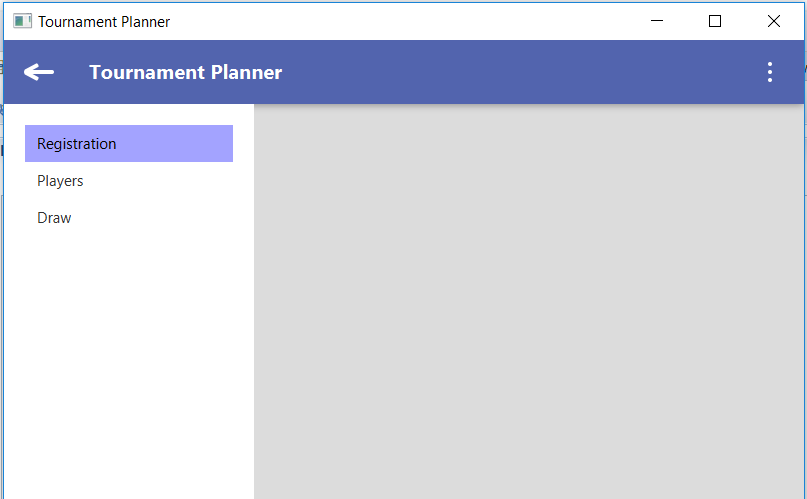
1. – Обнаружение клиентом сервера

При дальнейшем обращении к методам клиент работает с необходимым компонентом, который был выше инжектирован. На примере обращения к методу getPlayers(), данное тестирование было приведено в режиме отслеживания переменной service в Debug. На рисунках 33 и 34 значения id = 111 двух переменных service из двух разных классов совпадают, что означает успешное взаимодействие клиента и сервера в рамках работы системы планирования турнирных матчей.



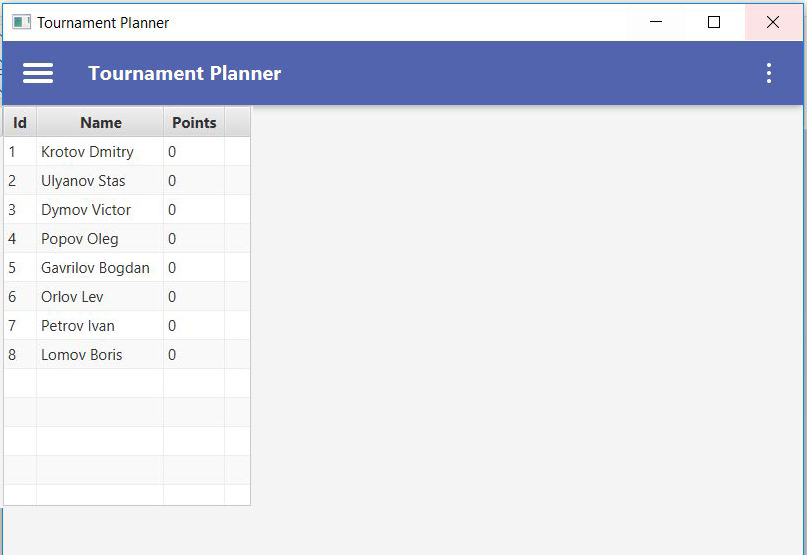
1. – Инспектирование переменной service на клиенте

Завершающим этапом тестирования является тестирование на системном уровне. На рисунке 35 представлено главное окно пользовательского интерфейса системы планирования турнирных матчей.



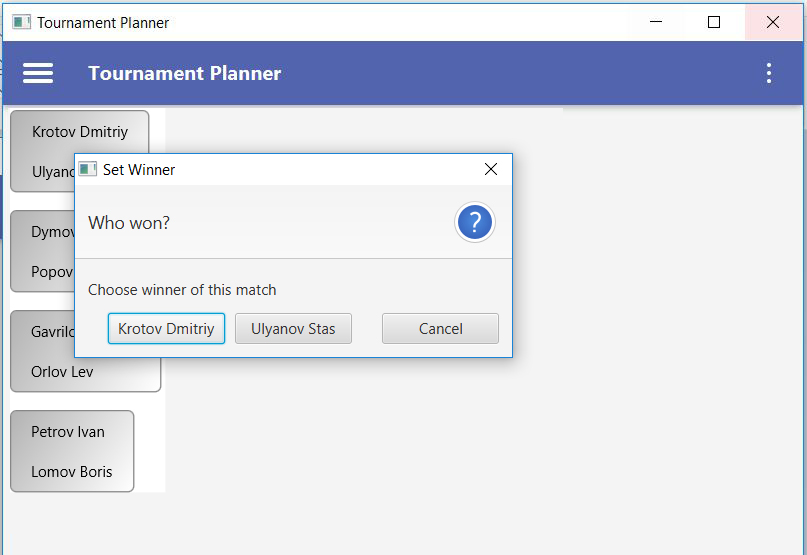
1. – Главное окно пользовательского интерфейса

После открытия регистрации рефери матча может перейти к добавлению игроков в турнир. После добавления или удаления необходимых игроков и закрытия регистрации создается турнир со всеми игроками, информацию о которых можно увидеть при переходе на кнопку Players. Результат представлен на рисунке 36.



1. – Окно Players

После того, как все игроки зарегистрированы, остается сгенерировать матчи и представить их в наглядном виде, представленном на рисунке 37. Победителя каждого матча можно выбрать при нажатии на нужный матч.



1. – Окно Draw

## Вывод к главе

В главе 3 данного квалификационной работы описаны основные используемые средства, которые применялись при разработке системы, а также описаны настройки среды разработки Eclipse IDE.

В данном разделе был разработан план тестирования. После разработки тестовых случаев было проведено тестирование. Результаты тестирования подтверждают работоспособность разработанной системы планирования турнирных матчей.

# ЗАКЛЮЧЕНИЕ

В данной квалификационной работе был проведен анализ заинтересованных сторон системы планирования турнирных матчей, были выявлены функции системы и ее показатели качества.

Далее, были изучены предложенные по ограничениям системы технологии спецификации OSGi, а именно принцип работы компонента реализации OSGi Eclipse Communication Framework. В рамках изучения были выявлены причины для использования данной технологии при реализации системы.

Также были изучены предложенные технологии графической реализации – платформа JavaFx для создания графических интерфейсов и фреймворк FXDiagram для создания инструментов графической платформы.

После того, как были изучены все технологии, использующиеся при разработке системы, спроектирована архитектура система и представлена реализация системы планирования турнирных матчей на основе предложенных технологий, была проделана работа по настройке Eclipse IDE и созданию бинарной сборки системы, разработке плана тестирования системы и тестированию системы.

В результате была решена поставленная цель ВКРБ – создание системы планирования турнирных матчей с возможностью проведения регистрации игроков, автоматическим генерированием матчей, подведением их итогов и выявлением победителя турнира, а также возможностью изменения, расширяемости системы и многопользовательский доступ с помощью разных конфигураций.

В дальнейшем планируется создать улучшенный пользовательский интерфейс, и расширить функциональность системы путем добавления алгоритмов проведения турнирных матчей.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Басс Л., Клементс П., Кацман Р., Архитектура программного обеспечения на практике. – СПб.: Питер, 2006. – 576 с. – (Классика Computer Science) – ISBN 5-469-00494-5.

[2] «Tournament Software»: [Электронный ресурс]. URL: https://www.tournamentsoftware.com (дата обращения: 15.10.2016).

[3] Герберт Шилдт, Java 8. Полное руководство. 9-е издание. – Вильямс, 2015. – ISBN 9780071808552.

[4] Jeff McAffer, Paul VanderLei, Simon Archer, OSGi and Equinox. – Addison Wesley. – Boston – 2010.

[5] Getting Started with Equinox & OSGi. [Электронный журнал]. URL: https://www.dzone.com (дата обращения: 10.03.2017).

[6] «ECF commits to OSGi Remote Services»: [Электронный ресурс]. URL: <http://www.eclipse.org/community/eclipse_newsletter/2013/july/article3.php> (дата обращения: 10.03.2017).

[7] Oracle Releases JavaFX2.0: [Электронный ресурс]. URL: <http://www.oracle.com/us/corporate/press/512728> (дата обращения: 22.04.2017).

[8] FXDiagram: [Электронный ресурс]. URL: <http://jankoehnlein.github.io/FXDiagram/index.html> (дата обращения: 26.04.2017).

[9] Tomek Kaczanowski, Practical Unit Testing with Junit and Mockito. –2013. – ISBN 978-83-934893-7-4.

[10] Archi4.0.1. Specificatoin, hints.

[11] Джон Макгрегор, Девид Сайкс, «Тестирование объектно-ориентированного программного обеспечения». – К.: ООО «ТИД «ДС», 2002. – 432 с.

[12] Дж. Рамбо: UML 2.0. Объектно-ориентированное моделирование и разработка. Питер, 2007. – ISBN 5-469-00814-2.

[13] Jayasoma S., Pradeep F., Demstifying OSGi, Colombo Java MeetUp, 2013.

[14] Marc Lankhorst. Enterprise Architecture at Work., Springer, N. York, 2000.

[15] Scott Chacon, Pro Git, 2009.

# ПРИЛОЖЕНИЕ А

**Техническое задание**

* 1. Общие сведения

В данной работе разрабатывается система планирования турнирных матчей.

Дополнительное ограничения:

* Использовать язык программирования Java.
* Использовать технологию модульности OSGi ECF.
* Использовать графическую платформу JavaFX.

Срок выполнения работы – 12 недель, с 9 февраля 2017 года.

* 1. Назначение и цели создания системы

Система предназначена для автоматизации планирования и удобства управления турнирами.

В результате создания системы должна быть достигнута цель создания системы с графическим интерфейсом, позволяющей проводить регистрацию игроков на турнир, автоматическое генерирование матчей с возможностью подведения их результатов на каждом этапе проведения турнира и выявления победителя турнира.

* 1. Характеристика объектов автоматизации

Данная система может применяться к турнирам, проводимым ФТР.

* 1. Требования к системе

В данной системе требуется проводить планирование и управление турнирными матчами.

Основные показатели качества, которые предъявляются к системе представлены в таблицах 11 – 16.

1. – Показатели готовности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Внутренний | Неисправность сервера | Нормальные условия | Сервис планирования | Выдача сообщения о невозможности запуска системы в консольные сообщения из-за проблем сервера и перезапуск системы. | Необходимость восстановление системы за 1 минуту. |
| Внешний | Длительное бездействие | Нормальные условия | Сервис планирования | Система продолжает работу | Готовность к использованию. |
| Внешний | Аварийная ситуация | Нормальные условия | Сервис планирования | Выдача сообщения об ошибке через консольные сообщения и попытка устранения путем перезапуска системы или ее части. | Обеспечение работоспособности за 2-3 минуты. |

1. – Показатели модифицируемости системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Разработчик системы | Добавление алгоритмов автоматического генерирования матчей | В момент разработки системы | Исходный код программы | Модифицирование системы не привело к ошибкам | В течение трех часов |
| Разработчик системы | Изменение универсальной модели турнира | В момент разработки системы | Исходный код программы | Модифицирование системы не привело к ошибкам | В течении трех часов |
| Разработчик системы | Внесение изменений в пользовательский интерфейс | В момент разработки системы | Исходный код программы | Модифицирование системы не привело к ошибкам | В течение трех часов |

1. – Показатели производительности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Пользователи | Отправление запросов по управлению турниром | Нормальные условия | Система планирования | Обработка вызываемых методов | Обработка запросов не более 2 секунд. |
| Пользователи | Единовременное взаимодействие пользователей с сервером | Перегруженный режим | Система планирования | Система должна продолжать функционировать | Подключение до 100 устройств в минуту |

1. – Показатели безопасности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Пользователь | Попытка внести изменения в данные при отсутствии прав доступа | В период использования | Данные в системе | Система не позволяет внести данные из-за ограниченности прав пользователя | Данные защищены от несанкционированного доступа |

1. – Показатели контролепригодности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Тестировщик | Тестирование турнира при разном количестве игроков | В период тестирования системы | Модель турнира | Система случайным образом назначает победителя каждого этапа и выводит победителя турнира | Наибольший массив тестируемых игроков 2000 |
| Тестировщик | Тестирование функции обмена данными с пользователями | В период тестирования системы | Сервис планирования | Система принимает запросы от пользователей и отсылает ответы | До 50 устройств одновременно |

1. – Показатели практичности системы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Источник стимула** | **Стимул** | **Условия** | **Артефакт** | **Реакция** | **Количественная мера** |
| Пользователь | Изучить возможности системы по управлению турниром | В момент использования системы | Система планирования | Система возвращает победителя турнира | Цель пользователя по управлению турниром достигнута |
| Пользователь | Облегчить использования системы | В момент использования системы | Система планирования | Система имеет простой и понятный интерфейс | Повышение образовательного уровня пользователя |

* 1. Состав и содержание работ по созданию системы

При создании системы необходимо провести следующие работы:

* выявить заинтересованные стороны системы;
* выявить функции системы
* выявить показатели качества системы;
* провести анализ спецификации OSGi;
* провести анализ компонента спецификации OSGi Eclipse Communication Framework;
* провести анализ платформы JavaFX;
* провести анализ фреймворка FXDiagram;
* реализовать систему;
* провести отладку системы;
* провести тестирование системы.

Система должна обладать следующей функциональностью:

* создание универсальной модели турнира;
* возможность регистрации игроков;
* автоматическое генерирование турнирных матчей разных раундов;
* графическое представление сетки матчей управляемого турнира;
* выбор победителя для каждого матча;
* сохранение истории сыгранных матчей;
* формирование рейтинга игроков по результатам турнира для выявления победителя;
* поддержка одновременного доступа к системе планирования несколькими пользователями;
* создание понятного графического пользовательского интерфейса для управления турниром.
  1. Порядок контроля и приемки системы

Для контроля работоспособности системы должен быть проведен экспериментальный запуск системы, подтверждающий работу системы.

* 1. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Не предъявляются.

* 1. Требования к документированию

Расчетно-пояснительная записка.

* 1. Источники разработки

Задание на ВКРБ.

# ПРИЛОЖЕНИЕ Б

**Графическая часть выпускной квалификационной работы**

В графическую часть входят:

1. Бизнес-процесс планирования турнира.
2. Цели заинтересованных сторон и требования к системе.
3. Анализ аналогичных систем планирования турнирных матчей.
4. Компонентная декомпозиция системы планирования для одного устройства.
5. Компонентная декомпозиция системы планирования для двух устройств.
6. Развертывание системы на оборудовании для одного устройства.
7. Развертывание системы на оборудовании для двух устройств.
8. UML-диаграммы. Диаграммы состояний и последовательностей.
9. Тестирование пользовательского интерфейса.