|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Министерство науки и высшего образования РФ | | | | | | | | |
|  | | |  | | |  | | |
| Пермский государственный национальный  исследовательский университет | | | | | | | | |
|  | | |  | | |  | | |
|  | | ОТЧЁТ  по лабораторной работе №1  по дисциплине «Инструментальные средства построения баз знаний»  «Разработка оболочки экспертной системы продукционного типа» | | | | |  | |
|  | | |  | | |  | | |
|  | Работу выполнил  студент гр. ПМИ-1,2  Валеев Р. Р. \_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_» \_\_\_\_\_\_\_\_ 2023 | | |  | Проверил  старший преподаватель кафедры МОВС  Леонтьева Т.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_» \_\_\_\_\_\_\_ 2023 | | |  |
|  |  | | |  |  | | |  |
| Пермь 2023 | | | | | | | | |

# Постановка задачи

Разработать и реализовать оболочку экспертных систем. Использовать базу данных или объектную модель. Тестирование проводить на разработанной ранее экспертной системе.

Требования к оболочке экспертных систем:

* использовать созданную на курсе "Базы знаний" экспертную систему;
* обратный вывод;
* минимальная настройка;
* стратегия выборки правил – FIFO => внимание к порядку правил;
* минимальная тщательность;
* единый интерфейс для любого типа пользователей;
* стратегия тестирования посылки: eager;
* расширение – проприетарное;
* не набирать, а выбирать;
* домены описываются ЯВНО;
* при описании переменных задаётся не только домен, но и вид;
* реализовать взаимосвязанные комбобоксы;
* контекстное пополнение доменов и переменных;
* реализация рабочей памяти;
* в компоненте объяснения предоставить содержимое рабочей памяти;
* порядок правил значим => Drag & Drop;
* посылка – все переменные, заключение – только выводимые;
* возможно, установить запрет на редактирование переменной при её использовании;
* унификация интерфейса;
* исключить вычисления;
* при добавлении правил вставлять их после текущего выбранного;
* Drag & Drop значений домена;
* вопрос формируется как название переменной со знаком вопроса;
* компонента объяснения в виде treeview.

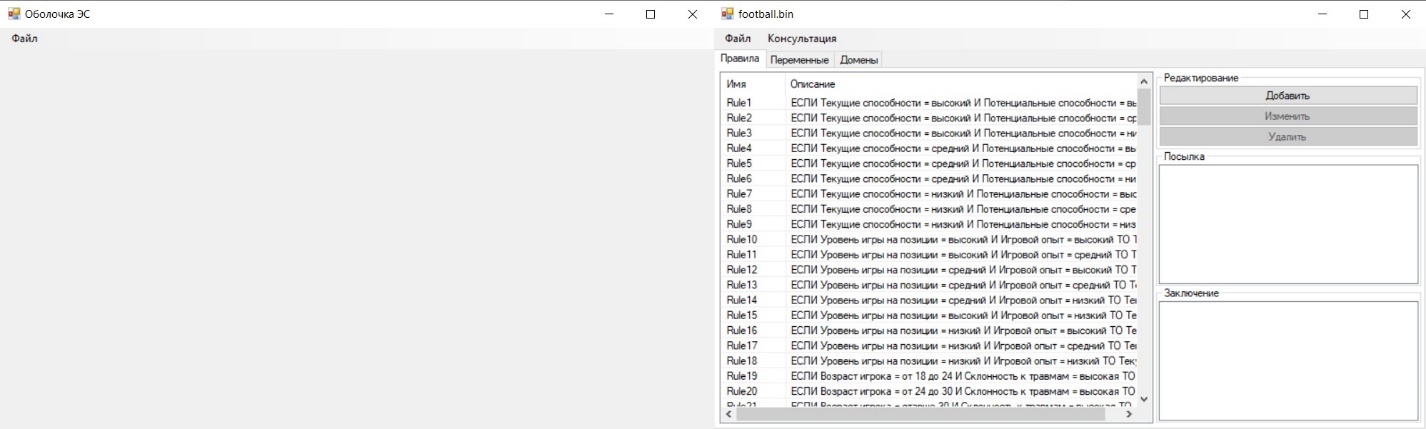
# Описание и реализация

Разработаем оболочку экспертных систем на языке программирования C#. Для создания GUI-приложения будем использовать Windows Forms.

## Разработка графического интерфейса

Реализация оболочки экспертных систем началась с создания графического интерфейса, а именно с создания необходимых форм и размещения на них элементов управления. В результате было реализовано семь форм, каждая из которых будет кратко описана далее.

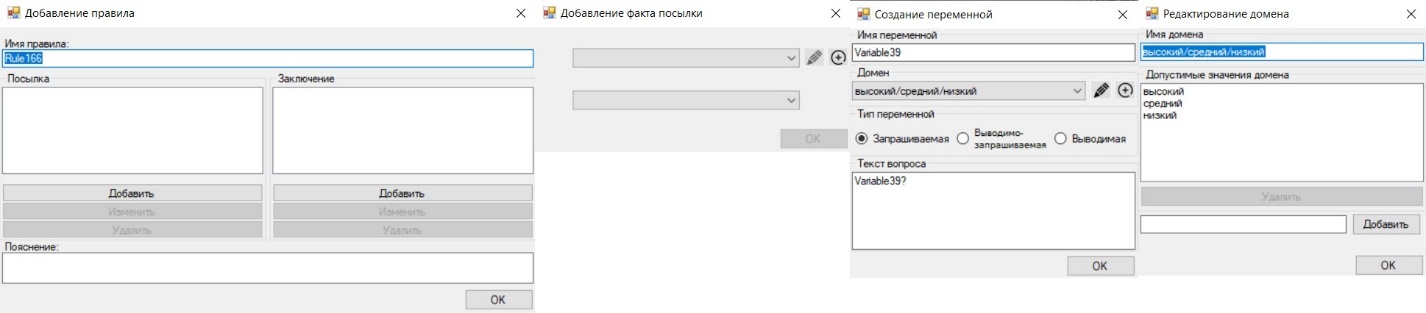
ExpertSystemForm – главная форма, содержащая меню приложения, а также три вкладки для добавления, редактирования и удаления правил/переменных/доменов. До момента открытия имеющейся базы знаний или создания новой, взаимодействие с приложением ограничено. Для вкладки «Правила» доступна возможность изменения порядка правил путем перетаскивания элементов списка. На рисунке 1 представлен внешний вид формы ExpertSystemForm.



Рисунок

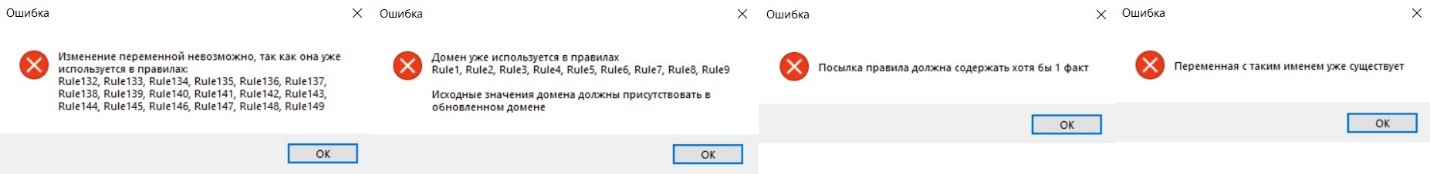
1 – Стартовое окно (слева) и окно после загрузки базы знаний (справа)

EditRuleForm, EditFactForm, EditVariableForm, EditDomainForm – формы добавления и изменения правил, фактов, переменных и доменов соответственно (рис. 2). Для удобства пользователей было реализовано контекстное пополнение переменных и доменов, т.е. при необходимости перед созданием правила можно добавить новую переменную или домен напрямую через форму EditRuleForm. Аналогично, через форму EditVariableForm можно создать новый домен или изменить уже имеющийся. Также форма добавления/изменения домена содержит элемент управления ListBox с поддержкой Drag & Drop для изменения порядка значений домена. В случае ввода некорректных данных и попытки их добавления/изменения пользователю будет выдано сообщение с текстом ошибки (рис. 3).



Рисунок

2 – Формы добавления/изменения правил, фактов, переменных, доменов



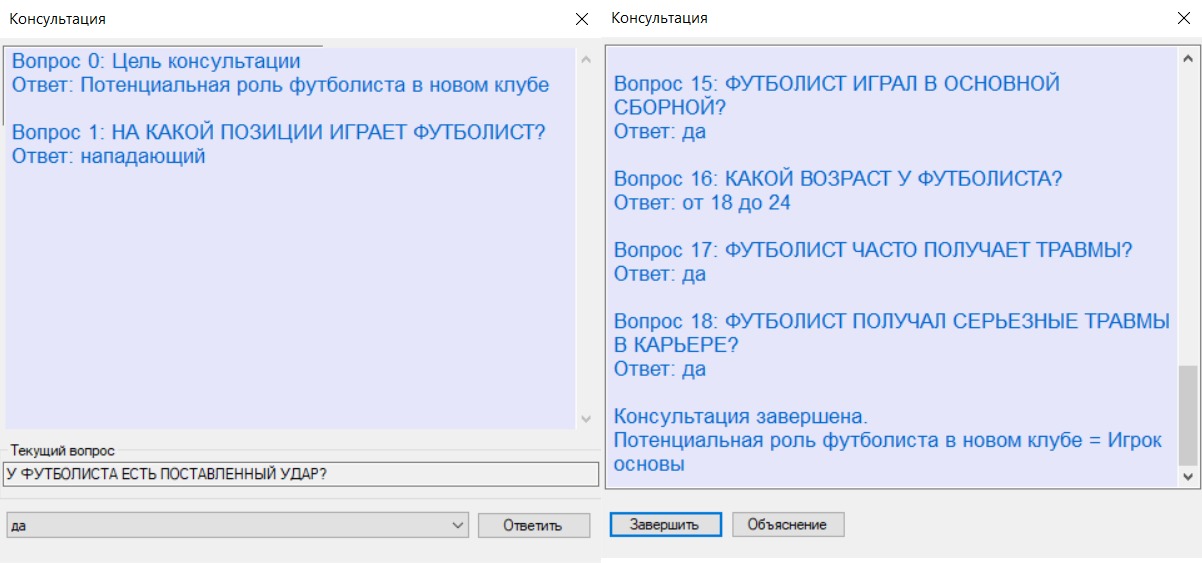
Рисунок

3 – Попытка добавления/изменения данных с ошибками

ConsultationForm – форма для консультации, состоящая из трех основных элементов:

1. ComboBox для выбора ответа на вопрос;
2. Label, где отображается текущий вопрос;
3. TextBox с историей консультации.

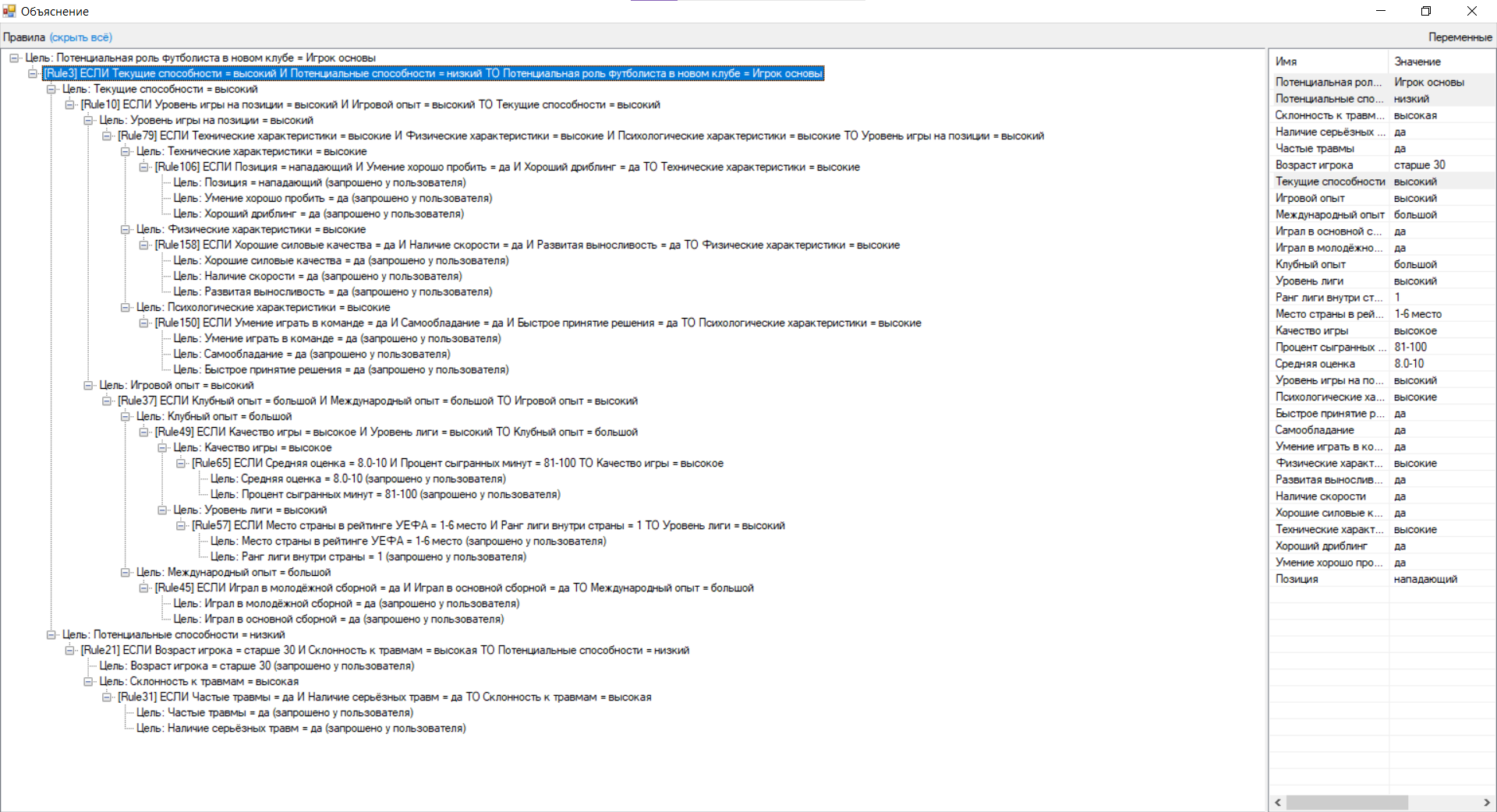
После завершения консультации, пользователю будет сообщено о результате, а также предложено посмотреть объяснение вывода. Внешний вид окна консультации представлен на рисунке ниже.



Рисунок

4 – Форма консультации

ExplanationForm – компонент объяснения, в котором можно посмотреть содержимое рабочей памяти, а также дерево вывода. Для реализации дерева использовался элемент управления TreeView. При выборе узла дерева вывода на панели переменных выделяются значения использованных переменных. На рисунке 5 представлен интерфейс компоненты объяснений.



Рисунок

5 – Дерево вывода и содержимое рабочей памяти

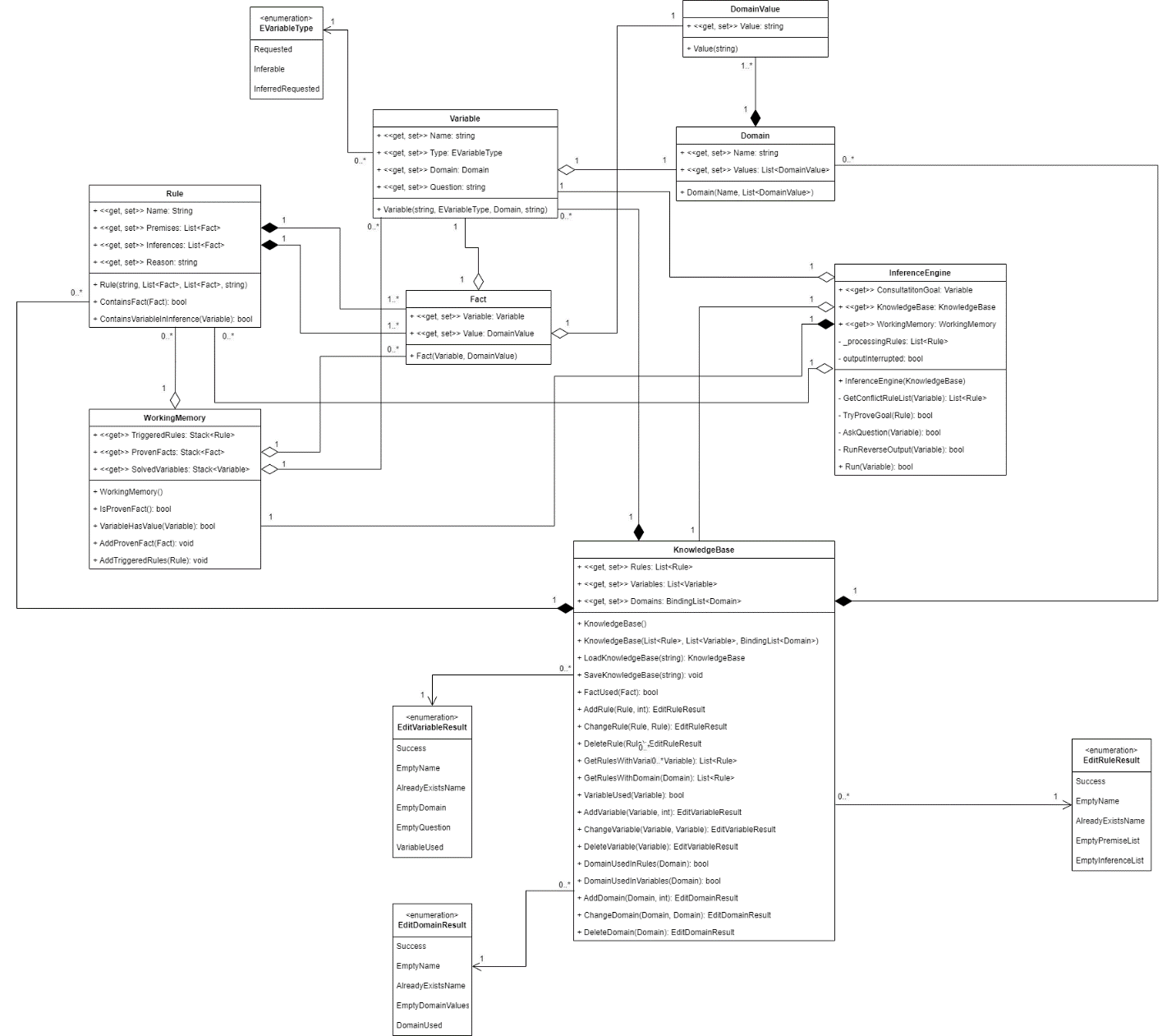
## Архитектура разрабатываемой системы

В соответствии со схемой оболочки ЭС (рис. 6) была спроектирована архитектура системы. На рисунке 7 представлена диаграмма классов разрабатываемой оболочки экспертных систем. Компоненты пополнения знаний и объяснений представлены в виде классов-форм и не отображены на данной диаграмме.



Рисунок

6 – Стандартная схема оболочки ЭС



Рисунок

7 – Диаграмма классов

Для представления знаний используются классы DomainValue, Domain, Variable, Fact, Rule. Класс KnowledgeBase представляет базу знаний. В нём хранятся списки правил, переменных и доменов. Основными методами класса являются AddRule, ChangeRule, DeleteRule, AddVariable, ChangeVariable, DeleteVariable, AddDomain, ChangeDomain, DeleteDomain, при помощи которых происходит приобретение знаний. Для загрузки и сохранения базы знаний используются методы LoadKnowledgeBase и SaveKnowledgeBase соответственно. Базы знаний для разрабатываемой системы, хранятся в бинарном формате. При попытке загрузить некорректный файл будет выдано сообщение об ошибке.

Класс WorkingMemory реализует рабочую память. В рабочей памяти будут хранится сработавшие в ходе логического вывода правила, доказанные факты, а также переменные, получившие значения.

Класс InferenceEngine ответственен за механизм логического вывода. Для каждой новой консультации создается экземпляр класса WorkingMemory и запоминается цель консультации (свойство ConsultationGoal). Сам вывод происходит для переданной базы знаний. Кроме этого, в классе описаны два атрибута: \_processingRules (набор правил, которые рассматриваются на данном этапе вывода) и outputInterrupted (переменная, определяющая был ли прерван вывод целевой переменной пользователем). Рассмотрим четыре основных метода, реализующие логический вывод:

1. RunReverseOutput – запускает логической вывод для переданной переменной.
2. GetConflictRuleList – формирует конфликтный набор правил, т.е. набор правил, которые могут быть использованы для вывода текущей цели.
3. TryProveGoal – рассматривает переданное правило, пытаясь доказать текущую цель.
4. AskQuestion – получает значение запрашиваемой переменной, путем задания вопроса пользователю.

## Описание механизма обратного логического вывода

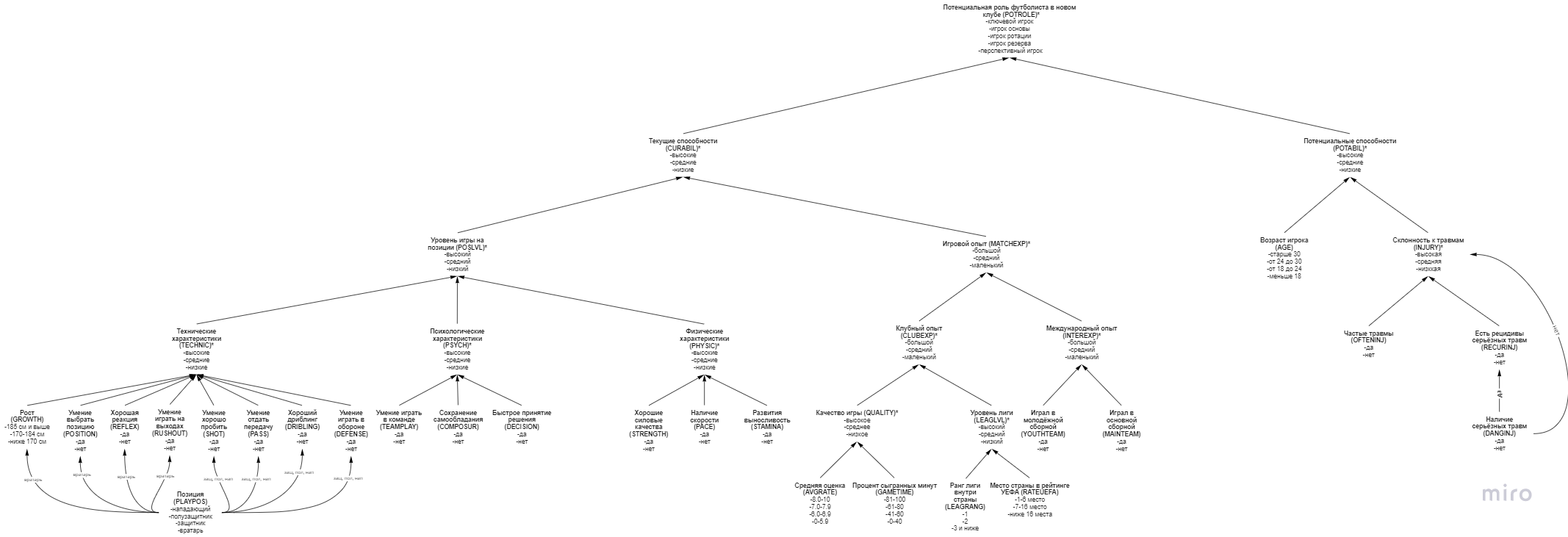
Ищем правило в заключении которого может быть определено значение целевой переменной. Если таких правил несколько – составляется конфликтный набор правил. Согласно требованиям к заданию, приоритет будет отдаваться правилам, идущим в начале списка. Далее, выбираем правило и проверяем истинность посылки. Если она истинна, то добавляем правило в список сработавших и также сохраняем доказанный факт. Если посылка ложна, то переходим к рассмотрению следующего правила. В случае, когда значение переменной в посылке не определено, данная переменная становится временной целью. В зависимости от типа переменной либо значение запрашивается у пользователя (запрашиваемая), либо запускается её обратный вывод (выводимая). В случае выводимо-запрашиваемой переменной сначала происходит попытка её вывода, но если вывести её не удалось, то задаётся соответствующий вопрос пользователю.

# Тестирование

Тестирование проводилось для ЭС, разработанной в ходе дисциплины «Базы знаний и оболочки экспертных систем». Для подтверждения правильности результатов разработанной оболочки экспертных систем были также приведены результаты, полученные из GURU. Для тестирования логического вывода с учетом выводимо-запрашиваемой переменной были внесены корректировки в исходную ЭС.

## Описание ЭС «Оценка потенциальной роли футболиста»

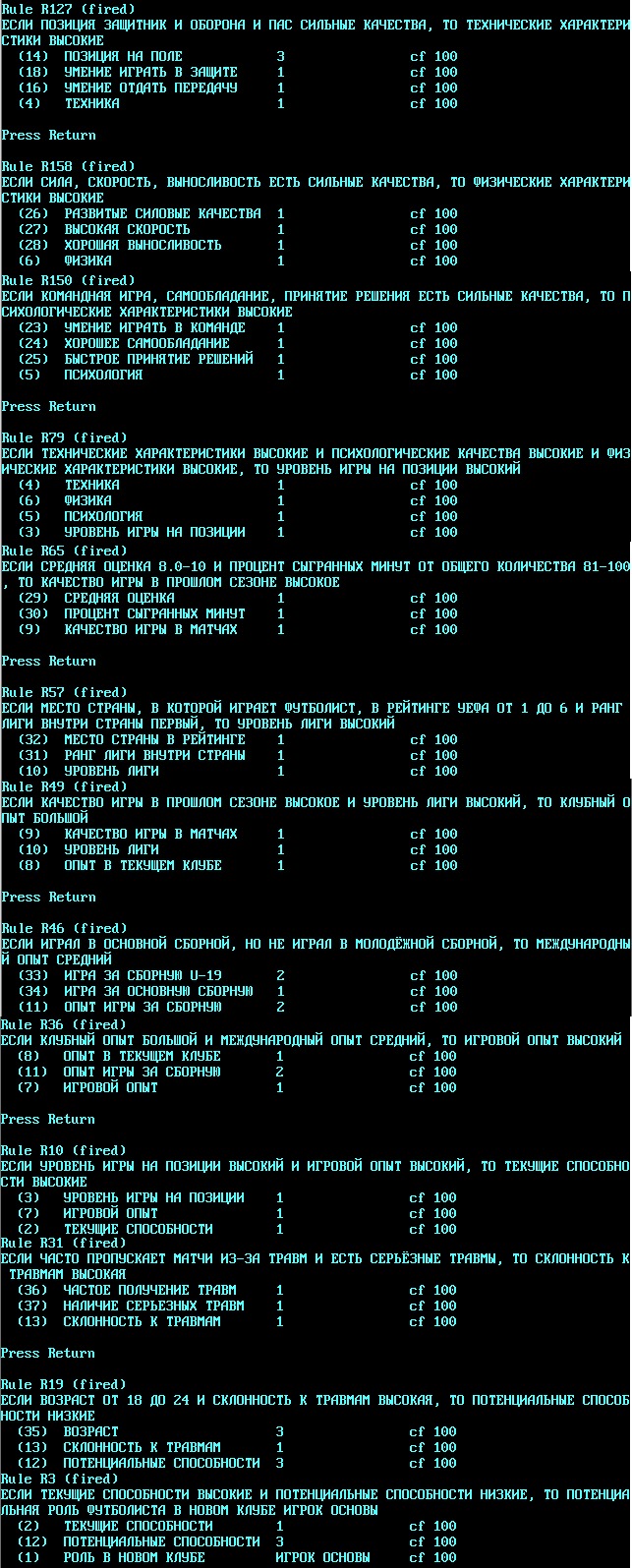
Предметная область – оценка потенциальной роли футболиста в новой команде. Для этого анализируется текущие характеристики игрока и его потенциал. Результат консультации с экспертной системы содержит оценку потенциальной роли игрока в новой команде: ключевой игрок, игрок основы, игрок ротации, игрок резерва. На рисунке 8 представлена концептуальная модель ПрО (дерево вывода) ЭС «Оценка потенциальной роли футболиста в новой команде».

Рисунок

8 – Концептуальная модель ПрО ЭС «Оценка потенциальной роли футболиста в новой команде»

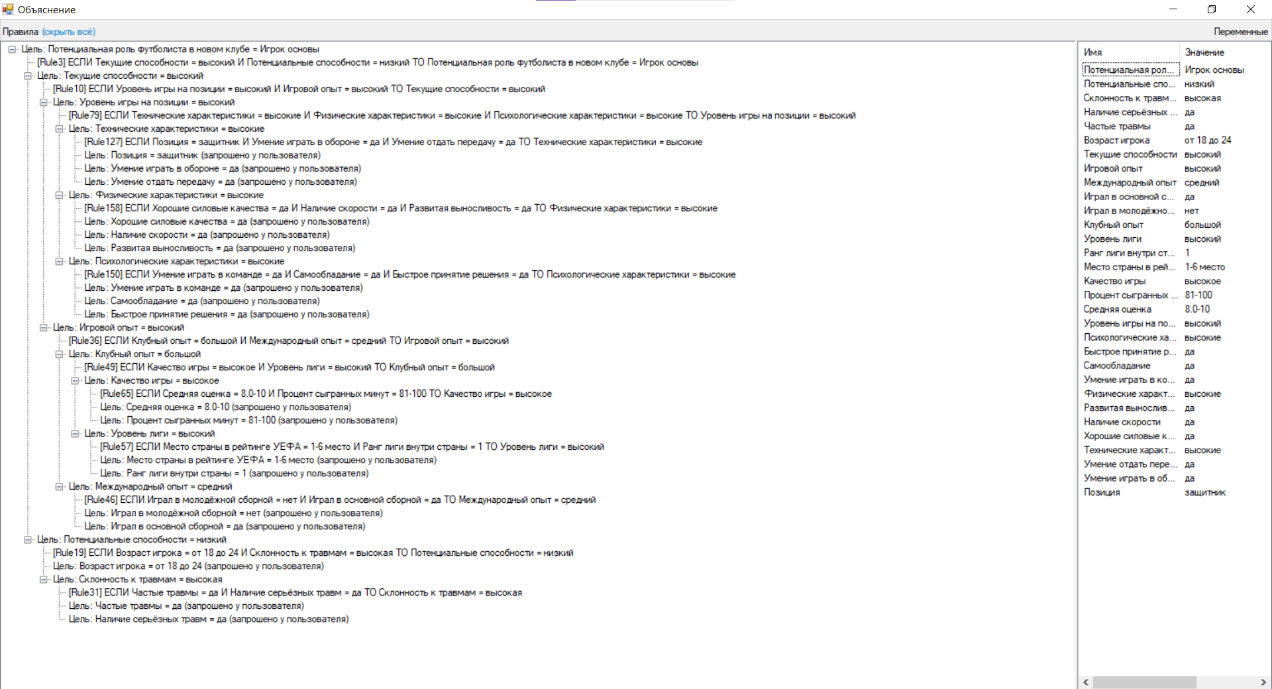
## Тест №1 для ЭС «Оценка потенциальной роли футболиста»

Проверить самую короткую ветвь в дереве. Результат тестирования представлен на рисунках 9-10.



Рисунок

9 – Тест №1 в GURU

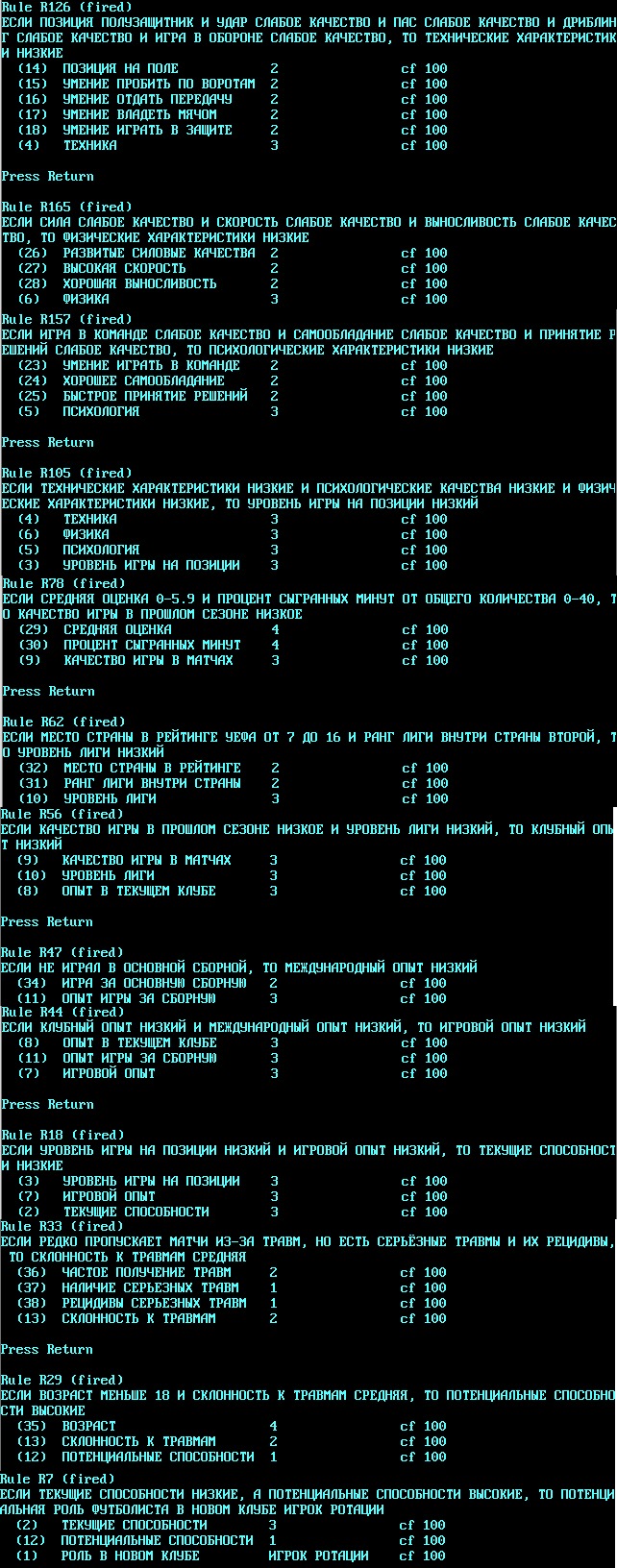


Рисунок

10 – Тест №1 в разработанной оболочке

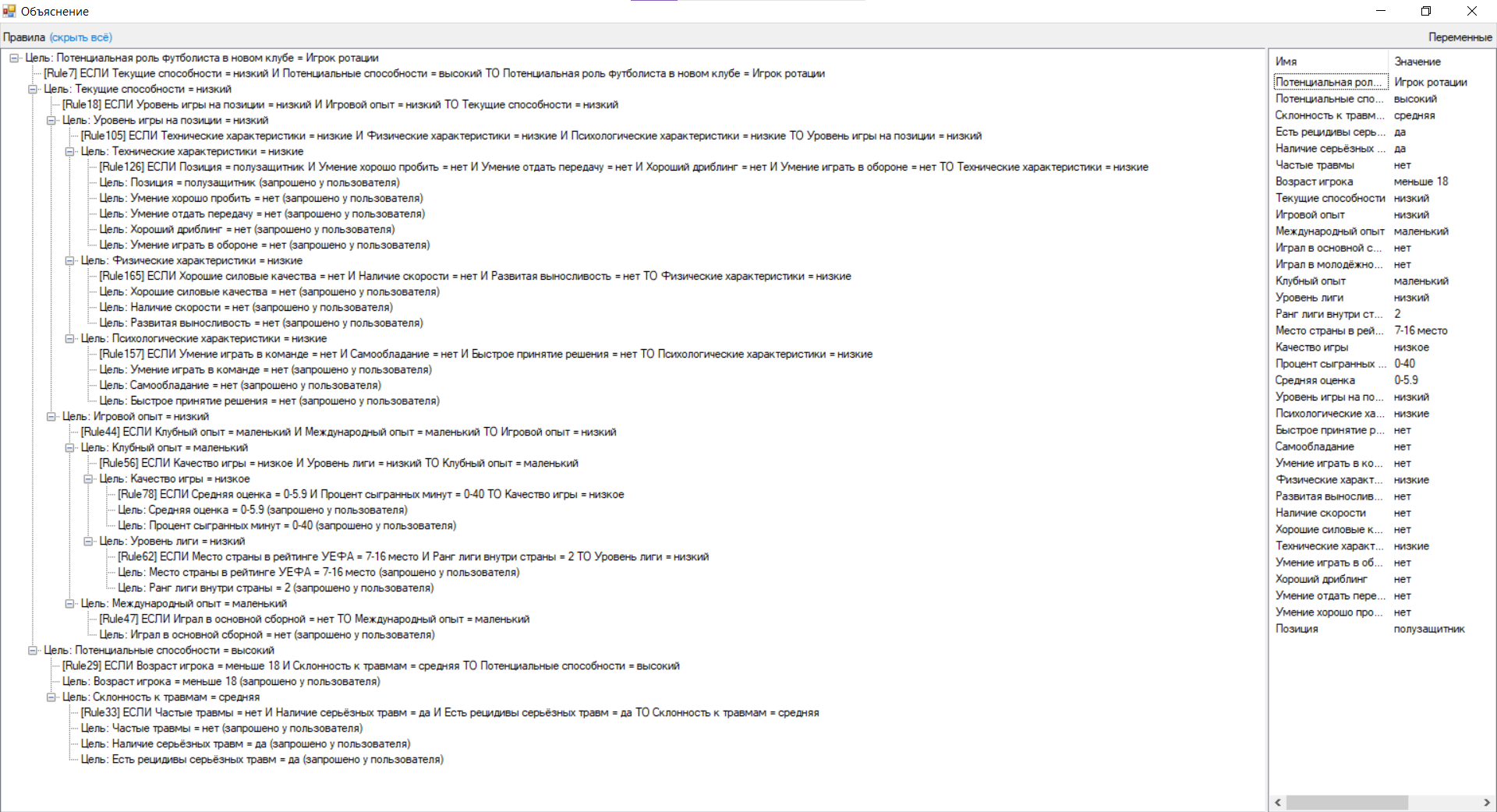
## Тест №2 для ЭС «Оценка потенциальной роли футболиста»

Проверить самую длинную ветвь в дереве. Результат тестирования представлен на рисунках 11-12.



Рисунок

11 – Тест №2 в GURU

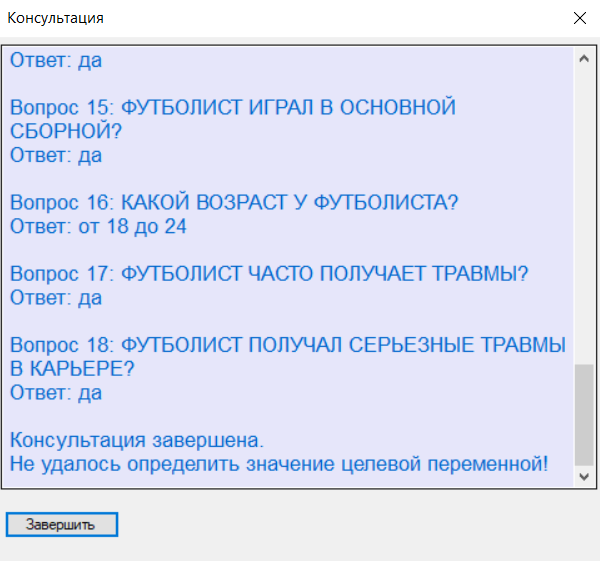


Рисунок

12 – Тест №2 в разработанной оболочке

## Тест №3 для ЭС «Оценка потенциальной роли футболиста»

Проверить случай, когда ЭС не может помочь пользователю. Такая ситуация в ЭС «Оценка потенциальной роли футболиста» невозможна, т.к. среди вопросов нет таких, ответ на которые может противоречить ответам на другие вопросы. Для того, чтобы убедиться, что такая ситуация корректно обрабатывается была внесена искусственная ошибка в имеющийся набор правил. Результат тестирования представлен на рисунке 13.

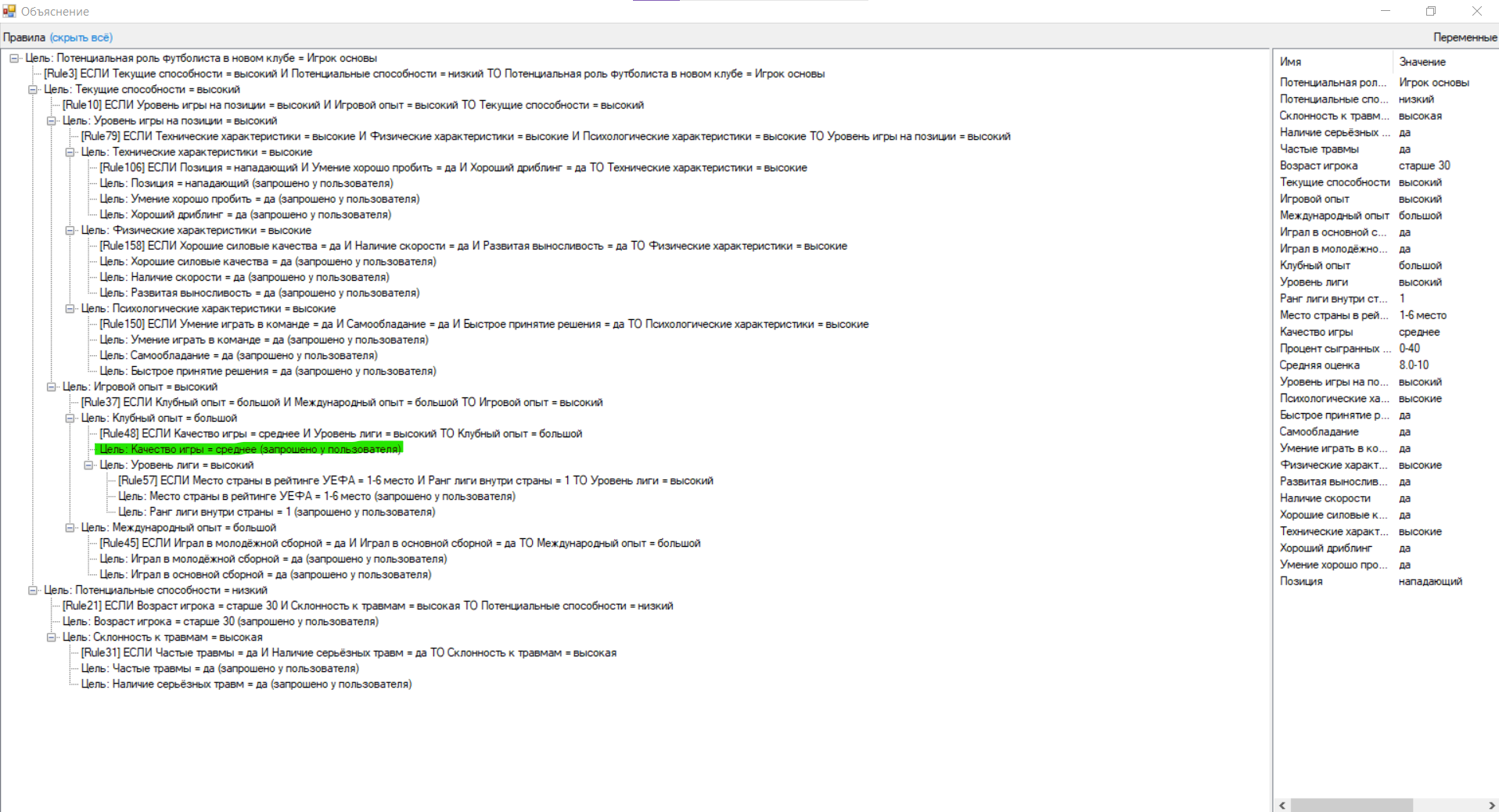


Рисунок

13 – Тест №3 в разработанной оболочке

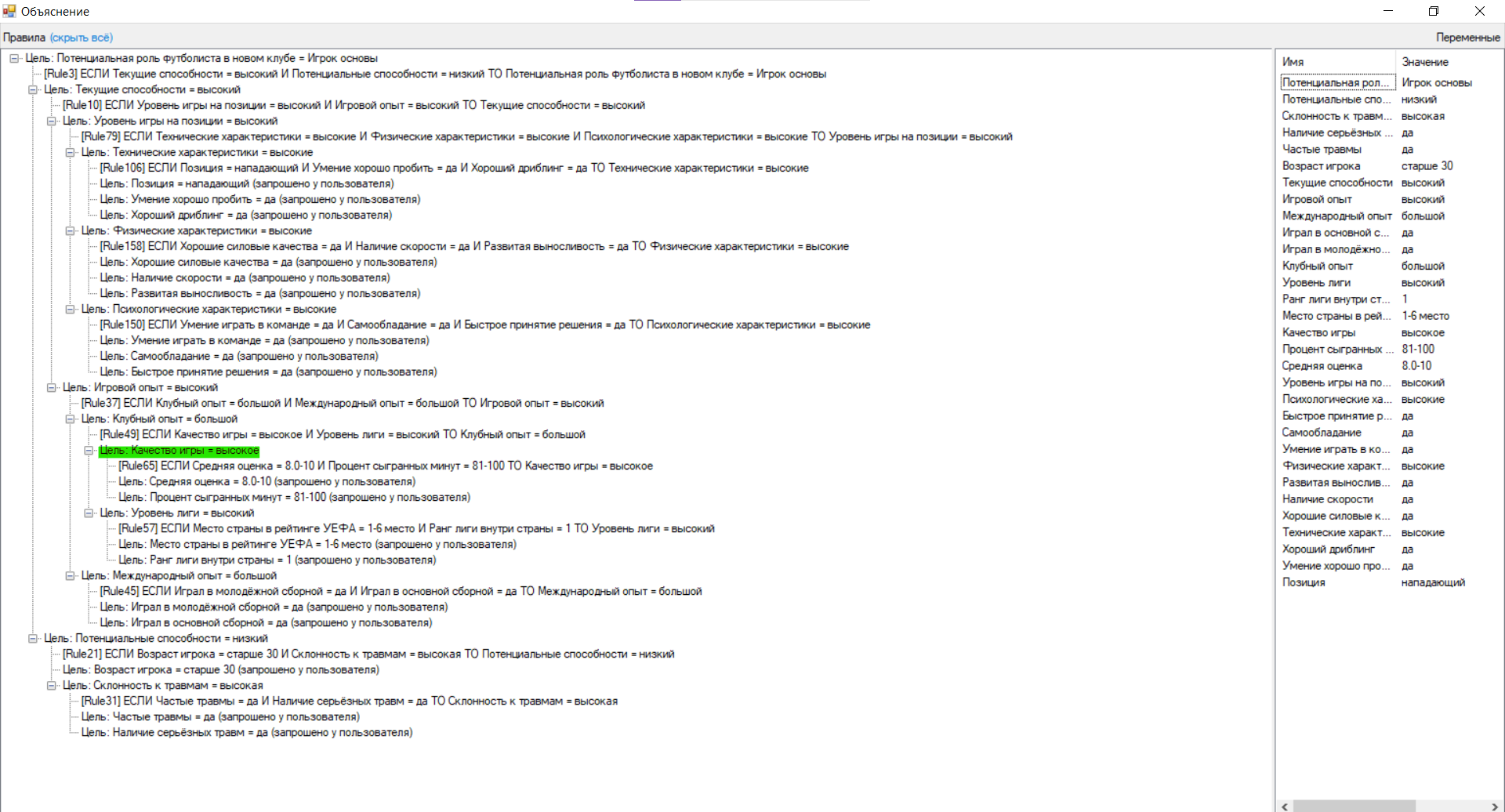
## Тест №4 для ЭС «Оценка потенциальной роли футболиста»

Проверить случай использования выводимо-запрашиваемой переменной. Для проведения теста переменная «Качество игры» стало выводимо-запрашиваемой, а некоторые правила, где данная переменная означивалась были удалены. Результат тестирования представлен на рисунках 14-15.



Рисунок

14 – Тест №4: выводимо-запрашиваемая переменная запрошена у пользователя



Рисунок

15 – Тест №4: выводимо-запрашиваемая переменная означена в результате срабатывания правила