0 Неформализованное описание задачи

Необходимо реализовать программу, которая визуализирует успеваемость студента, группы, потока, факультета (баллы должны высчитываться с учетом степени важности предметов).

1 Название программы

Universal Rating Visualizer (универсальный визуализатор рейтинга), далее будет использоваться сокращение (URV).

2 Цель и назначение программной системы

Цель программной системы – улучшить зрительное восприятие рейтинга студента, группы, потока или факультета в целом, заменив табличную систему отображения рейтинга на визуальное отображение. Под визуальным отображением рейтинга понимается сопоставление среднему баллу студента, группы, потока или факультета некоему изображению, отражающему успеваемость обучающейся единицы в интересующий нас промежуток времени (за первую контрольную неделю, за вторую контрольную неделю или за весь семестр).

Программа предназначена для визуализации рейтинга студента, группы, потока или факультета на основе записей, введенных пользователем.

3 Область применения программной системы

Планируется использовать URV в качестве плагина к сайту, занимающемуся отображением рейтинга (напр. edu.vstu).

4 Анализ аналогов программной системы

Первый «Когнитивный визуализатора рейтинга». Данная программа представляет рейтинг студента в виде колоды карт. Градация идет как по виду карты, так и по масти.

Второй «Когнитивный рейтинг студента в виде "лица Чернова"». Программа представляет собой апплет на языке Java и предназначена для

отображения 15 параметров в виде «лица Чернова».

Между двумя данными программными продуктами есть ряд общих черт: обе системы высчитывают средний рейтинг студента, учитывая степени важности тех или иных предметов. Кроме того, обе системы выводят результаты по обеим контрольным неделям, экзамену и итоговые результаты. Обе системы, каждая своим образом, осуществили визуализацию рейтинга и его градацию: в первом случае через колоду карт, во втором - с помощью мимики лица Чернова.

Перейдем к положительным моментам в аналогах.

В первой ПС можно отметить возможность визуализации рейтинга сразу нескольких студентов, то есть, осуществлена возможность сравнения рейтинга. Также градация по номиналу карт предельно понятна - чем ниже карта по “статусу”, тем ниже студент по рейтингу.

Во второй ПС нужно отметить хороший подход к высчитыванию 15 критериев для получения лица Чернова: такой подход можно считать более качественным и полным по сравнению с первой системой. Также визуализация в виде человеческого лица понятна всем - чем хуже у студента с учебой, тем злее или печальнее будет выражение лица Чернова.

Перейдем к недостаткам.

Обе ПС не предназначены для визуализации рейтинга потоков и факультетов, максимум был достигнут в первой ПС, в ней можно было визуализировать рейтинги всех студентов в группе. Также обе ПС не поддерживают визуализацию каких-либо промежуточных срезов, например, контрольных работ.

Первый визуализатор, на наш взгляд, непонятно отображает рейтинг на отдельных срезах. Все 4 среза (2 контрольных недели, экзамен и итог) обозначаются разными мастями. Немного непонятно, показывают ли масти степень важности того или иного среза или же просто были выбраны для разнообразия отображения. Кроме того, вывод визуализации не оформлен: выводится ФИО студента, а под ним карта. Если студентов выводится много, то на визуализацию неприятно смотреть.

Второй визуализатор, на наш взгляд, более проработан, но есть пара непонятных моментов. Первое - не всегда совокупность всех 15 параметров для построения лица Чернова в совокупности дают естественное выражение лица. Второе - возможность пользователя самому назначать степень важности предмета, ведь тогда общая картина о рейтинге студнта может отобразиться неверно.

После анализа прототипов, можно сделать вывод, что в большей степени можно опираться на второй аналог, но внести свои исправления в способ отображения лица Чернова, а также добавить возможности сравнения потоков, групп и факультетов.

|  |  |  |  |
| --- | --- | --- | --- |
| **Свойство** | **Когнитивный визуализатор рейтинга.** | **Когнитивный рейтинг студента в виде "лица Чернова"** | **Универсальный визуализатор рейтинга** |
| Рейтинг одного студента | + | + | + |
| Рейтинг группы | + | - | + |
| Рейтинг кафедры | - | - | + |
| Рейтинг факультета | - | - | + |
| Рейтинг вуза | - | - | + |
| Многокритериальная зависимость | - | + | + |
| Гибкость системы к числу предметов | ? | - | + |
| Контрольные срезы | + | + | + |
| Промежуточные срезы | - | - | ? |
| Визуализация с градацией рейтинга | + | + | + |
| Наглядность визуализаци | + | + | ? |
| Удобность использования | - | + | ? |

5 Протокол извлечения знаний

На данный момент вопросы к заказчику не возникли.

6 Роли пользователей

1. Анонимный пользователь - незарегистрированный пользователь, который может только просматривать рейтинг.

2. Модератор - человек, вносящий данные об успеваемости студентов в систему (например, староста группы).

3. Администратор - человек, проверяющий работу модератора. Он может принять или отклонить введенные модератором данные. Также способен сам вносить нужные данные в систему (например, преподаватель).

7 Описание объекта проектирования

На основе данных, хранящихся в БД, система вычисляет средний балл студента, группы, потока или факультета с учетом сложности предметов. Затем программа выводит этот рейтинг в визуальной форме. Можно выводить визуальный рейтинг как для одной обучающейся единицы, так и для нескольких сразу, чтобы сравнить результаты учебы.

8 Описание функциональных требований к программе

1. Программа должна визуализировать данные о рейтинге студента, группы, потока, факультета в виде персонажа из мира «Властелин колец».

2. Программа должна сохранять в базу данных записи, вводимые пользователем.

3. Программа должна уметь высчитывать средний рейтинг студента, группы, потока и факультета, учитывая степень сложности предметов, за счет особенностей математической логики программы.

4. Программа должна обеспечивать возможность визуализировать рейтинги нескольких равнозначных обучаемых единиц (студент-студент, группа-группа, поток-поток, факультет-факультет).

5. Программа должна позволять администратору назначать сложности для предметов, а также внутри себя производить разделение предметов на три категории: профильные предметы, «полезные» и «дополнительные».

6. Программа должна реагировать на увеличение или уменьшение среднего балла обучающейся единицы сменой состояния персонажа мира.

7. Программа должна обеспечивать изменение визуального рейтинга в зависимости от среднего балла в следующих промежутках: 61 - 66, 67-72, 73-76, 77 - 81, 82 - 85, 86 - 89, 90 - 94, 95 - 100.

8. Если пользователь хочет сравнить две обучаемые единицы на разных факультетах, то программа должна уметь преобразовать один вид персонажа в другой для лучшего понимания разницы в рейтинге.

9. Программа должна уметь выводить визуализацию рейтинга для следующих срезов: первая контрольная неделя, вторая контрольная неделя, экзамен и итог.

10. Программа должна обеспечивать защищенность данных в БД путем ограничения возможностей обычных пользователей и модератора.

11. Программа должна позволять сравнивать не более четырех равнозначных обучаемых единиц между собой.

9 Описание нефункциональных требований

1. Программа должна корректно реагировать на ежегодное изменение количества студентов и предметов благодаря структуре БД.

10 Входные и выходные данные

Входные данные: основным входом является список баллов студента за каждый предмет, но также пользователь может добавлять в базу новых студентов, группы, факультеты, предметы.

Выходные данные: визуальное отображение рейтинга студента, группы, факультета в виде персонажа. На выходе может вывестись один персонаж или несколько, если необходимо произвести сравнение.

11 Ограничение границ проекта

Визуализатор должен корректно работать с текущим количеством студентов, обучающихся в ВолгГТУ, а также с полным набором предметов, имеющихся в учебном плане.

Визуализатор должен работать с тремя группами предметов (профильные, «полезные», «дополнительные») и не более того.

Программа должна уметь производить сравнение рейтингов следующих видов: «студент – студент», «группа – группа», «поток – поток» и «факультет – факультет» и никаких других.

Программа должна уметь сравнивать до 4 обучаемых единиц и не более того.

Программа должна работать с количеством предметов в диапазоне от 5 до 18.

12 Распределение ролей в команде

Клевцов В. - руководитель/тех. райтер/кодер

Никитский Н. - кодер/архитектор

Пестун М. - тестер

Поцелуйко А. - кодер/тестер/верстальщик/тех. райтер