0 Неформализованное описание задачи

Необходимо реализовать программу, которая визуализирует успеваемость студента, группы, 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 или факультета в целом, заменив табличную систему отображения рейтинга на визуальное отображение. Под визуальным отображением рейтинга понимается сопоставление среднему баллу студента, группы или факультета некому изображению, отражающему успеваемость обучающейся единицы в интересующий нас промежуток времени (за первую контрольную неделю, за вторую контрольную неделю, экзамен/зачет или за весь семестр). Под обучающейся единицей понимается студент, группа или факультет.

Программа предназначена для визуализации рейтинга студента, группы или факультета на основе таблицы баллов, заполняемой по ходу учебного года.

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 студента 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.

Таблица 1. Сравнительный анализ аналогов.

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | + | + | + |

5 Протокол извлечения знаний

1. Нужна ли Вам возможность визуализировать рейтинг преподавателей? Если да, то нужна более подробная информация о том, как он строится.

6 Роли пользователей

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

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

3. Администратор-преподаватель - человек, проверяющий работу модератора. Он может принять или отклонить введенные модератором данные. Также он может внести некоторые коррективы в данные, внесенные модератором. Преподаватель назначает модераторов. У каждого преподавателя по своему предмету(-ам) есть таблица норм баллов за срезы, которые он может заполнять и редактировать.

4. Администратор факультета – человек, который назначает преподавателей на каждый предмет.

5. Администратор вуза – человек, который занимается добавлением и редактированием факультетов, данных о преподавателях, связывает преподавателей с факультетами. Также он назначает администраторов факультетов.

6. Администратор БД – человек, который занимается редактированием и заполнением базы данных. Он вносит новые предметы, студентов и преподавателей в базу каждый семестр.

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, факультета в виде персонажа в мире «Властелин колец».

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

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

4. Программа должна позволять администратору БД назначать сложности для предметов.

5. Программа должна реагировать на увеличение или уменьшение среднего балла (с учетом сложности предметов) обучающейся единицы сменой состояния персонажа мира.

6. Программа должна обеспечивать изменение визуального рейтинга в зависимости от среднего балла с учетом сложности предметов в промежутках, указанных в приложении 1.

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

8. Программа должна обеспечивать защищенность данных в БД путем предоставления прав на изменение БД только ограниченному числу пользователей.

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

10. Программа должна давать возможность преподавателю вносить собственную норму баллов за каждый срез по каждому предмету и редактировать ее.

11. Программа должна давать возможность преподавателю назначать модераторов по каждому предмету; администраторам факультетов назначать преподавателей на все предметы, а администратору вуза назначать администраторов факультетов.

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

10 Входные и выходные данные

Входные данные для визуализации: список баллов студента(ов) по предметам в выбранный срез, а также нормы баллов по предметам в выбранном срезе.

Выходные данные для визуализации: визуальное отображение рейтинга студента, группы, факультета в виде персонажа. На выходе может вывестись один персонаж или несколько, если необходимо произвести сравнение. Все персонажи отображаются в едином мире, то есть, на одной картинке.

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.

Программа должна уметь визуализировать рейтинги следующих видов: «студент – студент», «группа – группа» и «факультет – факультет» и никаких других.

Программа допускает одновременную визуализацию рейтингов до четырех обучаемых единиц.

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

13 Технологии реализации

Так как мы разрабатываем веб-приложение, выбор языка и технологии разработки ограничен следующим списком:

Java;

Html, Css, Php, JS;

C#;

Flash;

Html, Css, Python, JS

node.js.

Так же нам необходимо использовать базу данных, используем mysql. Для поддержки различных когнитивных миров, а так же возможности использования различных типов баз данных программа будет иметь структуру, поддерживающую плагины. Для создания диаграмм используется astah и glify. Для написания документации используется Libreoffice. В качестве системы контроля версий используется Git.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Кроссплатформенность/  Распостраненность | Знание | Простота разработки | Стоимость платформы |
| Java | - | + | + | бесплатно |
| Html /  Php /  JS | + | частично + | + | бесплатно |
| С# | - | частично + | + | платно |
| Flash | - | - | - | бесплатно |
| Html /  Python /  JS | - | частично + | - | бесплатно |
| node.js | + | частично + | + | бесплатно |

Исходя из таблицы нами выбрано два варианта: node.js/jade/mysql/jscoverage/Mocha и Html/Php/JS.

Первый вариант более оптимистичный, нацеленный на платформы будущего. Данный вариант включает в себя следующие компоненты:

node.js -- организует главный цикл программы,

jade -- генерирует страницы для представления,

mysql -- для хранения необходимых данных,

jscoverage -- проверка покрытия кода тестами,

Mocha -- средство тестирования кода,

nide -- интегрированная среда разработки.

Второй вариант более реалистичный, выбор основан на знаниях команды. Данный вариант включает в себя следующие компоненты:

html -- генерирует страницы для представления,

php -- генерирует страницы для представления, организует главный цикл программы,

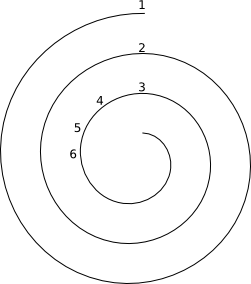
mysql -- для хранения необходимых данных,

phpunit -- средство тестирования кода.

javascript -- оброботка событий(нажатия клавиш).

eclipse -- интегрированная среда разработки.

14 Жизненный цикл



1. 2-ой релиз 10 апреля 2014.
2. 1-ой релиз 10 марта 2014.
3. Сдача проектной документации 24 декабря 2014.
4. Вторая сдача проектного решения 10 декабря 2014.
5. Первая сдача проектного решения 26 ноября 2014.
6. Сдача спецификации 12 ноября 2014.

Приложение 1

Оценочная шкала рейтинга в процентах

0 - 60 - неудовлетворительно (Неуспевающий студент) - Ужасно  
61 - 66 - слабо-удовлетворительно (Слабо-успевающий студент) – Плохо-  
67 - 71 - твердое удовлетворительно (Студент-троечник) - Плохо  
72 - 75 - удовлетворительно ближе к хорошо (Студент-троечник ближе к слабому хорошисту) - Плохо+  
76 - 81 - слабо-хорошо (Студент-слабый хорошист) - Хорошо-  
82 - 85 - твердое хорошо (Студент-хорошист) - Хорошо  
86 - 89 - хорошо ближе к отлично (Студент-хорошист ближе к слабому отличнику) - Хорошо +  
90 - 94 - слабое отлично (Студент-слабый отличник) - Отлично-  
95 - 100 - твердое отлично (Студент-отличник) - Отлично