Добрый день, уважаемая комиссия.
Я Клеванец Игорь, студент группы РК9-121.
Тема моего дипломного проекта: Подсистема хранения состояния модели РДО на основе реляционной базы данных.

[лист 1]
На момент постановки задачи среда имитационного моделирования РДО представляла результаты своей работы в ограниченном количестве представлений. Она могла представить пользователю системы некоторые показатели, которые запросил пользователь.
Также она выводила в текстовый файл трассировку. Проблему представляет собой именно текстовый формат данных. Для того, чтобы пользоваться ими в автоматизированном режиме пользователю нужно разработать систему, которая разберет этот текст и только после этого проанализирует их - это нетривиальная задача.
РДО-студия позволяла построить графики. Не очень удобные для работы, но принципиально возможность имелась.
Расширить возможности работы с результатами моделирования можно с помощью повышения открытости системы, т.е. выгрузки части данных в стороннюю систему в открытом формате.
Я остановился на реляционной базе данных. Это могли быть XML-формат или другой вид форматированного текста. Варианты использования открытых данных ограничиваются только фантазией и потребностями пользователя. Для проверки работы разработанной системы я выбрал наглядный пример - построение графиков. На роль СУБД выбран PostrgeSQL. Это мощный инструмент с наиболее либеральным лицензионным соглашением - BSD.
Одной из причин выбора баз данных стала идея проверки гипотезы. Один из алгоритмов поиска в РДО работает нерационально. Это объективный факт. Гипотеза заключалась в том, что алгоритм поиска СУБД работает гораздо более эффективно чем алгоритм РДО. Проверка этой гипотезы является одной из целей этого дипломного проекта.

[лист 2]
На диаграмме целей она отмечена в этом !* блоке. Состояние модели представлено текущими данными на каждый момент модельного времени - состояние ресурсов !*, а также историей изменения этих данных - трассировкой !*. Соответсвенно для каждого из блоков данных нужно разработать свою систему.
Здесь !* представлена структура базы данных для хранения ресурсов. Прошу обратить внимание на то, что связь этих блоков !*, хранящих целочисленные, вещественные и прочие значения, связаны с отношением rdo_value логически с помощью тригеров.

[лист 3]
Здесь представлена стурктура для хранения трассировки. Она более сложная из-за того, что во-первых, хранение истории - задача сама по себе более сложная чем хранение только текущего значения; во-вторых, в РДО история собирается не только с ресурсов, но и с других сущностей: системные события, поиск по графу и прочие.

[лист 4]
На этом листе с помощью диаграммы последовательнсотей представлен пример записи ресурса, у которого есть один параметр, хранящий массив из одного числа. Это тестовый пример, который демонстрирует схему взаимодействия компонент проектируемой системы.

[лист 5, лист 6]
На листах 5 и 6 представлены диаграммы классов, участвующих в работе разработанной мной подсистемы. Штрих-пунктиром выделены вновь разработанные классы. Остальные уже существовали в исходном коде РДО и подверглись модификации. Например, в классе RDOSimulatorBase происходит вызов подключения к СУБД и создание структуры баз данных.
А класс RDOValue обеспечивает запись универсальной сущности РДО, которая хранит значение любого поддерживаемого типа.

На листе 5 представлены классы, которые обеспечивают работу с ресурсами. На листе 6 классы для работы с трассировкой.

[лист 7]
На этом листе представлены диаграммы компонентов. Первая представляет собой перечень тех библиотек и файлов, которые подверглись изменению в моей работе. Вторая представляет перечень всех файлов, которые нужны, чтобы запустить среду РДО. Эта !* и эта !* часть появилась в результате моей работы.

[лист 8.1]
На диаграмме состояний укрупненно представлены те состояния, через которые проходит РДО-студия во время свой работы. Подробнее расммотрим этот !* блок в рамках работы с ресурсами.

[лист 8.2]
Мы видим диаграмму актвиностей, которая описывает алгоритм работы процесса записи конкретного ресурса. Он представляет собой запись иденфитифкатора ресурса и всех его параметров. Далее декомпозую этот !* блок.

[лист 9.1]
В этом алгоритме проверяется тип величины и происходит запись этой величины соответствующим методом.

[лист 9.2]
Далее представлен алгоритм записи массива. Прошу обратить внимание на то, что между этими двумя алгоритмами - рекурсиная связь.

[лист 9.3]
На следующем листе представлен алгоритм обновления значения массивов. Обновление значений для остальных типов непримечательно - это одна строка. Здесь же снова рекурсия.

[лист 10]
В рамках исследовательской части была исследовано быстродействие разработанной системы вцелом и влияние каждого модуля в отдельности. Для этого были собраны отдельные версии РДО, в которых работает модуль для ресурсов, трассировки, оба модуля вместе и версия, в которой модули присутствуют, но отключены.
Рассмотрено влияние на два этапа работы РДО: инициализации модели и сам процесс моделирования. А также проверена гипотеза о работе алгоритма поиска.
Разработы тестовые модели различной нагруженности: на 1000, 5000 и 10000 ресурсов. Для измерения каждой величины взята выборка из пяти замеров. Итого проведено 210 экспериментов.
Версия, в которой разработанные модули выключены, работает так же быстро как и текущая версия РДО.
Иницилизация работы модуля для ресурсов в разы медленнее работы модуля для трассировки. Однако разницы при объединении работы модулей не обнаружено.
Сам процесс моделирования медленнее при работе модуля трассировки.
Использование СУБД в алгоритме поиска не оправдало ожиданий: потеря производительности самого алгоритма - около 20%. Он неприемлим. Наработка исключена из кода РДО.

[лист 11, лист 12]
На диатограммах представлены все результаты, полученные в ходе экспериментов.
В результате проделанной работы были достигнуты все поставленные цели: РДО выводит свои внутренние данные в стороннюю систему. Адекватность работы проверена через систему Navicat, которая способна строить отчеты по содержимому баз данных. Схема

работы с этой системой представлена на последнем листе.
К отрицательным результатам работы следует отнести большие потери быстродействия, а также невозможность использования СУБД в алгоритме поиска РДО.