|  |  |
| --- | --- |
| **#** | **Речь** |
| 1 | Здравствуйте, уважаемые члены аттестационной комиссии! Вашему вниманию представляется дипломный проект, посвящённый реализации Системы проектирования и тестирования каркасов программных продуктов |
| 2 | Существует масса подходов к построению и разработке ПО.  Ошибки могут появиться на любом из этапов, но, как правило, большинство из них возникает вследствие неправильного решения на этапе проектирования. Цена ошибки растёт к концу процесса разработки |
| 3 | В данной работе предлагается система проектирования и тестирования каркасов программных продуктов, основанная на автоматной модели. На слайде представлена автоматная модель прототипа архитектуры, в которой выделены состояния, спецификации и сценарии использования. Такая модель была введена для обобщения описания архитектур систем. В настоящее время ведутся работы по созданию практик, методологии и инструментов, позволяющих тестировать модели, заданные подобным образом, однако идеальное решение пока не получено.  [ Модель прототипа является результатом отображения исходной архитектуры разрабатываемой системы в программную среду и представляет собой сущность, характеризующуюся: конечным множеством состояний, начальным состоянием, множеством конечных состояний, набором спецификаций требований и множеством сценариев использования. Состояние можно определить как момент в работе прототипа архитектуры, который объединяет в неявной форме все входные воздействия прошлого, а также влияет на реакцию в текущий момент времени. Каждое состояние имеет вполне определённый смысл и качественно отличается от всех других состоянии, и кроме того однозначно определяет действия, которые могут совершаться в этом состоянии] {Сценарий прототипа представляет собой множество правил перехода между состояниями, каждое из которых отражает возможность выполнения каждого требования спецификации. Главная задача сценария состоит в том, чтобы подтвердить или опровергнуть выполнение какого-либо свойства из набора спецификации.} |
| 4 | Формальное описание модели программы позволяет провести тестирование или верификацию, то есть появляется возможность проверить соответствие между прототипом и требованиями к нему.  Таким образом : **Пересечение Возможного поведения программы с неправильным должно быть пусто**  {Верификация программ в общем случае алгоритмически неразрешима (см. Теорема Райса)} |
| 5 | На основе формальной модели мной была спроектирована структура системы.  Для понимания общей структуры системы можно рассмотреть типичный вариант её использования. Пользователь, в терминах предметной области, согласно интерфейсу программирования, разрабатывает требуемый прототип архитектуры на специализированном языке описания моделей. Система транслирует полученный прототип архитектуры на выбранный целевой язык автоматного программирования. Полученная модель на целевом языке является пригодной для верификации. При помощи верификатора можно получить анализатор для модели прототипа и получить результат верификации или ошибку в виде контрпримера. Контрпример – трасса, содержащая ошибку. Процесс проектирования циклически повторяется после исправления модели на основе контрпримера. |
| 6 | В качестве базового верификатора был выбран сторонний верификатор SPIN.  Данный верификатор представляет собой средство моделирования и верификации протоколов, параллельных программ и широкого класса дискретных систем. Он используется для систем, предъявляющих высокие требования к надёжности, например, верификация аэрокосмических систем в NASA.  В основе верификации лежит метод Model Checking, который представляет собой Метод автоматической формальной верификации систем с конечным числом состояний. Для того, чтобы воспользоваться верификатором необходимо   1. Задать как система устроена и как она должна быть устроена 2. Таким образом получаем две нотации: описание поведения то есть устройство системы и описание требований или свойства правильности 3. Программа-верификатор проверяет, что устройство системы удовлетворяет свойствам правильности   {Проверка свойства на конечной модели программы/Исчерпывающий поиск по пространству состояний} |
| 7 | Для того чтобы провести тестирование на основе предложенной модели прототипа необходимо было выбрать технологию конвертирования модели на язык Promela, с которым работает SPIN.  Promela позволяет описывать поведение системы на основе автомата Крипке и поведение системы в терминах темпоральной логики.  Поэтому для описания предложенной модели прототипа в процессе дипломного проектирования был разработан специальный язык для описания модели прототипа.  Таким образом, процесс верификации проходит 2 стадии: на первой стадии вырабатываются состояния и переходы |
| 8 | На второй стадии для описания требований, предъявляемых к модели прототипа, используются формулы темпоральной логики. Особенность этого типа логики заключается в возможности описания поведения системы во времени.  {ПРОБЛЕМА: Полнота спецификации – охватывают ли спецификации всё поведение системы} |
| 9 | Мной была спроектирована архитектура системы, состоящая из трёх основных компонентов – подсистемы лексического и синтаксического анализа, подсистемы построения графа потока исполнения и подсистемы кодогенерации.  Для того, чтобы описать прототип архитектуры в процессе дипломного проектирования был разработан специализированный язык описания моделей прототипов. Назовём этот язык PROTO.  Таким образом, в качестве входных данных система принимает описание прототипа архитектуры с использованием языка PROTO. В качестве выходных данных система отдаёт представление исходного прототипа на целевом языке автоматного программирования |
| 10 | Основополагающей компонентой разработанной системы является – подсистема лексического и синтаксического анализа.  Данная подсистема отвечает за разбор исходного файла прототипа архитектуры на языке PROTO и построение абстрактного синтаксического дерева, соответствующего исходному прототипу архитектуры.  Система преобразует модель в абстрактное синтаксическое дерево, структура которого была выбрана с учётом дальнейших алгоритмов его обработки. Процесс построения лексического и синтаксического анализа был автоматизирован с помощью ANTLR. Для автоматизации сборки системы был использован Apache Maven. |
| 11 | Компонентой внутреннего слоя системы является подсистема построения графа потока исполнения. Данная подсистема ответственна за перевод информации, содержащейся в абстрактном синтаксическом дереве. Грамматика языка обеспечивает описание узлов графа и направленных дуг, соответствующих условиям перехода. Эти свойства отражаются в абстрактном синтаксическом дереве, которое затем преобразуется в более удобную с точки зрения автоматных языков форму – граф потока исполнения.  При построении такого графа каждому узлу ставится в соответствие состояние в исходной модели, а каждой инструкции перехода между состояниями соответствует ребро между состояниями. |
| 12 | Основной задачей подсистемы кодогенерации является генерация кода на целевом языке для исходной модели прототипа архитектуры  Требования спецификации к модели, описанные с помощью разработанного языка описания моделей, приводятся к специализированному виду, пригодному для использования верификатором SPIN.  [ При вопросах ссылаться на приложение ] |
| 13 | В качестве иллюстрации использования языка рассмотрим описание программы, представляющей из себя имитационную модель некоторого объекта. Поскольку для любой программной системы может быть построен соответствующий ей конечный автомат, то поведение любой программы может быть описано в терминах перехода конечного автомата из состояния в состояние, а, следовательно, эта программа может быть описана на предлагаемом языке Proto.  В качестве моделируемой программы возьмём прототип, реализующий имитационную модель автоматической коробки передач. Выбор этого примера объясняется как простотой реализации данной имитационной модели, так и наличием достаточно большого количества состоянии, что позволяет продемонстрировать основные возможности предлагаемого языка.  На данном слайде вы можете видеть взаимно однозначное соответствие между моделью и её изображением в виде конечного автомата. |
| 14 | Во время дипломного проектирования были разработаны визуальные средства, помогающие разработчику создавать описания проектируемых систем.  Был реализован плагин для среды разработки Eclipse, осуществляющий полноценную подсветку синтаксиса, автоматическое завершение ввода, а так же проверку ошибок. |
| 15 | В результате работы над дипломным проектом, мною была разработана и формализована модель системы проектирования и тестирования каркасов программных продуктов. На базе модели спроектирована архитектура системы. На основе формализованной модели был спроектирован и реализован каркас системы проектирования и тестирования прототипов архитектур с применением современных подходов и технологий программирования.  Также был реализован редактор для работы с языком PROTO в виде плагина для среды разработки Eclipse.  {Модель системы, представляющая из себя транслятор из разработанного языка Proto в язык автоматного программирования Promela} |
| 16 | Можно выделить следующие пути развития проекта. Во-первых, для расширения возможностей системы требуется добавить поддержку верификации распределённых и параллельных систем; Во-вторых, генерация других целевых языков позволит расширить список используемых верификаторов. Наконец, совершенствование компонентов и оптимизация алгоритмов базовой платформы системы позволит повысить эффективность её использования и сократить накладные расходы в работе полезной части. |
| 17 | На этом, у меня все. Спасибо за Ваше внимание. |