Лекция 9

Современные направления проверки правильности программ

- Современные направления проверки правильности программ - это формальные спецификации, методы доказательства, верификация, валидация и тестирование
- **Формальные спецификации** появились в программировании в 70-х годах прошлого столетия. Они предоставляют средства, облегчающие описание рассуждения о свойствах и особенностях программ в математической нотации.
- На формальных спецификациях базируются методы доведения программ, которые были начаты работами по теории алгоритмов А.А. Маркова [1], А.А. Ляпунова [2], схемами Ю.И. Янова [3] и формальными нотациями описания взаимодействующих процессов К.А. Хоара [4] и др..

- Для проверки формальной спецификации программы применяют математический аппарат для описания правильного решения некоторой задачи, для которой она разработана.
- Вместе со спецификацией разрабатываются дополнительные аксиомы, утверждение о описание операторов и условий, так называемые предварительные условия или предпосылки, и постусловием, определяющие заключительные правили получения правильного результата.

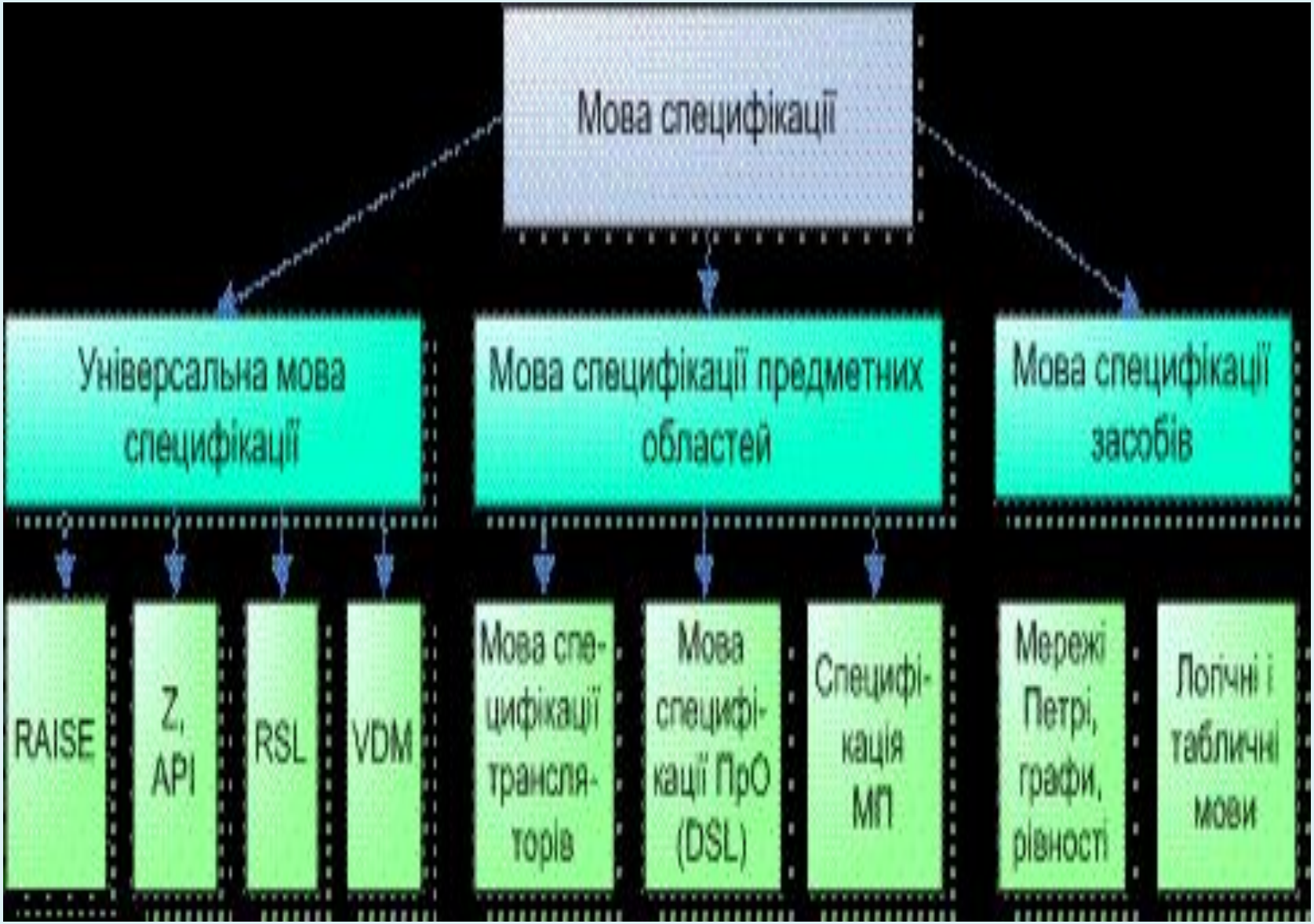
- **Доказательство программ** производится с помощью утверждений, состоящих в формальном языке и служат для проверки правильности программы в заданных ее точках. Если утверждение соответствует конечному оператору программы, то делается окончательный вывод о частичной или полной правильность работы программ.
- **Верификация и валидация** - это методы обеспечения проверки и анализа правильности выполнения заданных функций программы в соответствии с заданными требованиями заказчика к ним и системы в целом.
- **Тестирование** - это метод выявления ошибок в ПС путем выполнения исходного кода на тестовых данных, сбора рабочих характеристик в динамике выполнения в конкретном операционной среде, выявления различных ошибок, дефектов, отказов и сбоев, вызванных нерегулярными, аномальными ситуациями или аварийным прекращением работы

- Теоретические средства реализуются как процессы программирования и проверки правильности программного продукта.
- В настоящее время процессы верификации, валидации и тестирования ПС регламентированы стандартом ISO/IEC-12207 из жизненного цикла ПС.
- В некоторой зарубежной литературе процессы верификации и тестирования на практике отождествляются, они ориентированы на достижение правильности программы.

Языки спецификации программ и их классификация

- Языки формальной спецификации, которые используются для формального описания свойств выполнения программ путем задания утверждений, являются языками высокого уровня.
- В общем случае формальная спецификация программы - это однозначный специфицированный описание программы с помощью математических понятий, терминов, правил синтаксиса и семантики формального языка.

Категории формальных языков спецификации



Универсальные языки спецификации имеют общую математическую основу с такими средствами:

- 2) арифметические операции;
- 3) множества и операции над множествами;
- 4) описания последовательностей и операции над ними;
- 5) описания функций и операций над ними;
- 6) описания древовидных структур;
- 7) средства построения моделей областей;
- 8) процедурные средства языков программирования (операторы присваивания, цикла, выбора, выхода);
- 9) операции композиции, аргументами и результатами которых могут быть функции, выражения, операторы;
- 10) механизм конструирования новых структур данных.

Языки спецификации предметных областей (доменов) в программировании:

- 1) спецификации доменов;
-
- 2) описания взаимодействий и параллельного выполнения;
-
- 3) спецификации языков программирования и трансляторов;
-
- 4) спецификации баз данных и знаний;
-
- 5) спецификации пакетов прикладных программ.
-

- Языки спецификации специфики доменов DSL (Domain Specific Language) предназначены для формализованного описания задач в терминах предметной области, подлежащей моделированию. Эти языки можно подразделить на внешние и внутренние.
- **Внешние языки** (типа UML) По уровню выше языков программирования, например, предметно-ориентированный язык DSL, который используется для представления абстрактных понятий и задач ПрО. Их описание трансформируется к понятиям некоторой внутренней речи или языка программирования специальными генераторами или текстовыми редакторами.
- **Внутренние языки** - языки описания

- Языки программирования предметной области, дополнены средствами и механизмами технологий. Метапрограммирование является эффективным средством автоматизации спецификаций разработанных программ и в настоящее время находят широкое применение в области информационных технологий.

классификация спецификаций по способу выполнения

- Кроме приведенной классификации языков спецификаций, существуют другие. Например, возможна классификация спецификаций по способу выполнения:
-
- - Выполняемая (executable),
-
- - Алгебраическая (algebraic),
-
- - Сценарная (use case or scenarios),
-
- - В ограничениях (constraints).

классификация спецификаций по способу выполнения

- Выполняемые спецификации предполагают разработку прототипов систем для достижения установленной цели
- Алгебраические спецификации, включают в себя механизмы описания аксиом и утверждений, предназначенные для доведения специфицированных программ.
- Сценарные спецификации (UML) позволяют описывать различные способы возможного применения системы

Верификация и валидация программ

- **Верификация ПО** - процесс обеспечения правильной реализации ПО в соответствии со спецификациями, выполняется в течение всего жизненного цикла. Верификация дает ответ на вопрос, правильно создается система.
  Валидация - процесс проверки соответствия ПО функциональным и нефункциональным требованиям и ожидаемым потребностям заказчика.
  Верификация и валидация - это методы анализа, проверки спецификаций и правильности выполнения программ в соответствии с заданными требованиями и формального описания программы

- **Метод верификации** помогает сделать вывод о корректности созданной программной системы при ее проектировании и после завершения ее разработки.
- **Валидация** позволяет установить выполнимость заданных требований путем их просмотра, инспекции и оценки результатов проектирования на процессах ЖЦ для подтверждения того, что осуществляется корректная реализация требований, соблюдение заданных условий и ограничений к системе.
- **Верификация и валидация** обеспечивают проверку полноты, непротиворечивости и однозначности спецификации и правильности выполнения функций системы.

- Верификация и валидация, как методы, обеспечивают соответственно проверку и анализ правильности выполнения заданных функций и соответствия ПО требованиям заказчика, а также заданным спецификациям. Они представлены в стандартах] как самостоятельные процессы ЖЦ и используются, начиная от этапа анализа требований и кончая проверкой правильности функционирования программного кода на заключительном этапе, а именно, тестировании.
- Для этих процессов определены цели, задачи и действия по проверке правильности создаваемого продукта (рабочие, промежуточные продукты) на этапах ЖЦ. Рассмотрим их трактовку в стандартном представлении.

- **Верификации и валидации** подвергаются:

- - Компоненты системы, их интерфейсы (программные, технические и информационные) и взаимодействие объектов (протоколы, сообщения) в распределенных средах;

- - Описания доступа к базам данных, средства защиты от несанкционированного доступа к данным различных пользователей;

- - Документация к системе;

- - Тесты, тестовые процедуры и входные наборы данных.

Процесс верификации.

- **Цель процесса** - убедиться, что каждый программный продукт проекта отражает согласованные требования к их реализации.
- Этот процесс основывается:

- - На стратегии и критериям верификации всех рабочих программных продуктов на ЖЦ;

- - На выполнении действий по верификации в соответствии со стандартом;

- - На устранении недостатков, выявленных в программных (рабочих, промежуточных и конечных) продуктах;

- - На согласовании результатов верификации с заказчиком.

Процесс верификации

- Процесс верификации может проводиться исполнителем программы или другим сотрудником той же организации или сотрудником другой организации, например представителем заказчика. Этот процесс включает в себя действия по его внедрению и исполнению.

Внедрение процесса заключается в определении критических элементов (процессов и программных продуктов), которые должны подвергаться верификации, в выборе исполнителя верификации, инструментальных средств поддержки процесса верификации, в составлении плана верификации и его утверждения. В процессе верификации выполняются задачи проверки условий: контракта, процесса, требований, интеграции, кода и документации.

Соответствии с планом и требованиями заказчика проверяется правильность выполнения функций системы, интерфейсов и взаимосвязей компонентов, а также доступ к данным и к средствам защиты.

Процесс валидации.

- Цель процесса - убедиться, что специфические требования для программного выполнено, и осуществляется это с помощью:
-
- - Разработанной стратегии и критериев проверки всех рабочих продуктов;
-
- - Оговоренных действий по проведению валидации;
-
- - Демонстрации соответствия разработанных программных продуктов требованиям заказчика и правилам их использования;
-
- - Согласование с заказчиком полученных результатов валидации продукта.

- Процесс валидации может проводиться самим исполнителем или другим лицом, например, заказчиком, осуществляет действия по внедрению и проведению этого процесса по плану, в котором отражены элементы и задачи проверки. При этом используются методы, инструментальные средства и процедуры выполнения задач процесса для установления соответствия тестовых требований и особенностей использования программных продуктов проекта на правильность реализации требований.
- На других процессах ЖЦ выполняются дополнительные действия:
- - Проверка и контроль проектных решений с помощью методик и процедур обзора хода разработки;
- - Обращение к CASE-систем, которые включают в себя процедуры проверки требований к продукту;
- - Просмотры и инспекции промежуточных результатов на соответствие требованиям для подтверждения того, что они удовлетворяет условиям выполнения системы.

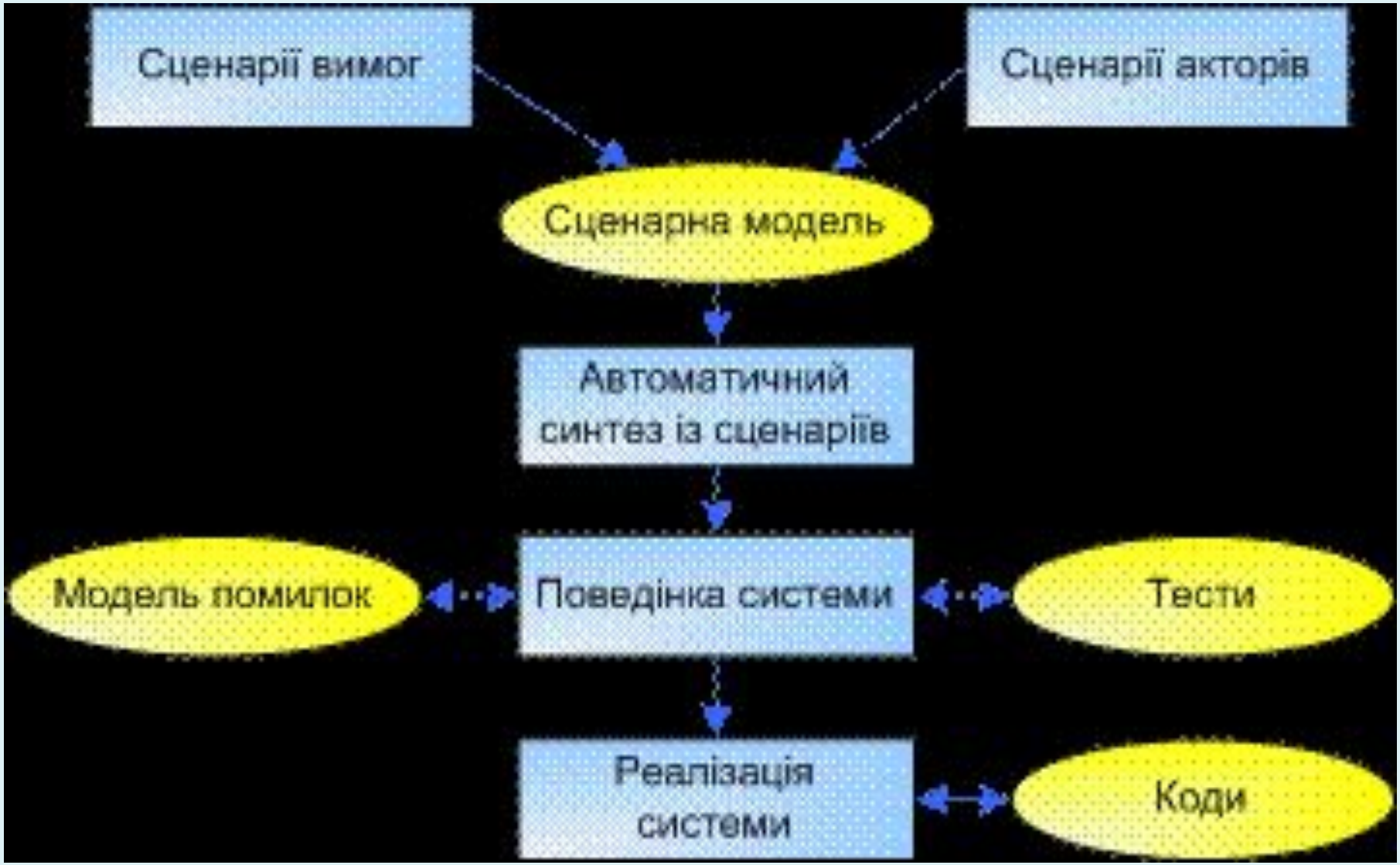
- Таким образом, основные задачи процессов верификации и валидации состоит в том, чтобы проверить и подтвердить, что конечный программный продукт соответствует назначению и удовлетворяет требованиям заказчика. Эти процессы взаимосвязаны и определяются, как правило, одним общим термином «верификация и валидация» или «Verification and Validation» (V & V)
- «верификация и валидация» основаны на планировании их как процессов, так и проверки для наиболее критичных элементов проекта: компонентов, интерфейсов (программных, технических и информационных), взаимодействий объектов (протоколов и сообщений), передачи данных между компонентами и их защиты, а также создания тестов и тестовых процедур.
- После проверки отдельных компонентов системы проводятся их интеграция, повторная верификация и валидация интегрированной системы, создается комплект документации, отражающей правильность выполнения требований по результатам инспекций и тестирования.

Подход к валидации сценария требований

- К процессу создания программ принадлежит описание требований на языке UML с помощью сценариев и действующих исполнителей - актеров как внешних сущностей по системе.
- Требования нужно проверять до их перестройки в программные элементы. Сценарий после трансформации - это последовательность взаимодействий между одним или несколькими актерами и системой, в которой актер исполняет цель сценария при взаимодействии с ней.
- В модели требований сценарий задает несколько альтернативных событий, заданных на языке диаграмм UML. Они разделяются на функциональные (системные) и внутренние, как определяющее поведение системы.
- На основе описания сценария требования проверяются путем валидации для выявления ошибок в представлении сценарных требований

- Эта проверка происходит итерационно и состоит из следующих шагов:
- 1. Формализованное описание требований в виде сценариев;
- 2. Создание модели требований;
- 3. Создание специальных сценариев для валидации требований;
- 4. Применение валидационных сценариев в модели требований;
- 5. Оценка результатов поведения модели требований;
- 6. Проверка условий завершения процесса валидации и при обнаружении каких-либо неточностей повторение шагов, начиная с п. 2.
- При выполнении сценариев могут возникнуть ошибочные ситуации, при которых поведения системы становится не детерминированным. В этих целей проводится контроль покрытия сценариев в модели требований валидационных сценариям с целью выявления ошибок или рисков

Валидация сценариев требований к системе



- Составная часть валидации требований по сценариям - определение классов эквивалентности входных и выходных данных для валидации и синтеза сценариев. Входная информация для синтеза сценариев - сценарная модель, задается на языке взаимодействия.

- Информация используется при генерации дополнительных сценариев с целью улучшения процесса валидации, автоматического синтеза сценариев модели и получения модели поведения системы под управлением актера.

- Модель проверяется с помощью тестов и модели ошибок, что в целом позволяет найти неполноту исходных требований или противоречия в требованиях.

Автоматический синтез программы основан на следующих процедурах:

- - Валидация требований путем выполнения валидационных сценариев
- - Добавление проверенных сценариев к набору валидационных сценариев и их использование в качестве входных данных для синтеза;
- - Поиск ошибок в сценариях и проверка различных композиций сценариев.

- Синтез спецификаций сценариев требований, трансформированных в диаграммам взаимодействия, может проводиться в среде системы Rational Rose.

Верификация объектных моделей

- Верификация объектной модели (ОМ) основывается на спецификации:
- - Базовых (простых) объектов ОМ, атрибутами которых являются данные и операции объекта - функции над этими данными;

- - Объектов, которые считаются проверенными, если их операции используются как теоремы, применяемые над подобъектов и не выводят их из множества состояний этих объектов.

Доказательство правильности построения ОМ предусматривает:

- - Введение дополнительных или удаления лишних атрибутов объекта и его интерфейсов в ОМ, доведение правильности объекта ОМ на основе спецификации интерфейсов и взаимодействия с другими объектами;

- - Доведение правильности задания типов для атрибутов объекта, т.е. правильности того, что выбранный тип реализует операцию, а множество его значений определяется множеством состояний объекта.

- Это доказательство является завершающим при проверке правильности ОМ.

Подход к верификации композиции компонентов

- Метод верификации композиции компонентов базируется на спецификации функций и временных свойств готовых проверенных компонентов (типа reuse) и выполняется с помощью абстракций модели проверки Model Checking
- Общая компонентная модель - это совокупности проверенных спецификаций компонентов, временных свойств и условий функционирования для асинхронной передачи сообщений (АПП).

Модель проверки обеспечивает верификацию программ на надежность путем решения следующих задач:

- - Спецификация компонентов на языке UML [с описанием временных свойств;
-
- - Описание спецификации интерфейса и временных свойств;
-
- - Проверка свойств сложных компонентов композиционным аппаратом.

- Компоненты модели могут быть примитивными и сложными.
-
- Свойства примитива проверяются с помощью модели проверки, а свойство сложного компонента - на абстракции компонентов, собранных из примитивов и проверенных их свойств в интегрированной среде.
-
- Данный подход может использоваться в распределенных программных системах, функционирующих на платформах типа CORBA, DCOM и EJB.

Общие перспективы верификации программ

- Методы формальной верификации использовались для проверки правильности моделей ПрО, функций в языке API, безопасности и целостности БД - в проекте SDV фирмы Microsoft и в международном проекте по формальной верификации ПС.
-
- Идея создания этого проекта принадлежит Т.Хоару и обсуждалась на симпозиуме по верифицированного ВС в феврале 2005г. в Калифорнии. Затем в октябре того же года на конференции IFIP в Цюрихе был принят международный проект сроком на 15 лет по разработке целостного автоматизированного набора инструментов для проверки корректности ПС [14, 15].

- В проекте сформулированы следующие основные задачи:
- - Разработка единой теории создания и анализа программ;
- - Построение всеобъемлющего интегрированного набора инструментов верификации для всех процессов, включая разработку спецификаций и их проверку, генерацию тестовых примеров, уточнение, анализ и верификацию программ;
- - Создание репозитария формальных спецификаций и верифицированных программных объектов различных видов и типов.
- Репозитарий - это хранилище правильных программ, спецификаций и инструментов.

Функции репозитория:

- - Накопление верифицированных спецификаций, методов доказывания, программных объектов и реализаций кодов для различных программных приложений;
-
- - Накопление всевозможных методов верификации, их оформление в виде, пригодном для поиска и отбора реализованной теоретической концепции для дальнейшего применения;
-
- - Разработка стандартных форм для задания и обмена формальными спецификациями различных объектов, инструментов и готовых систем;
-
- - Разработка механизмов взаимодействия для переноса готовых верифицированных продуктов с репозитория в новые распределенные и сетевые среды для их использования в новых ПС.

- Данный проект предполагается развивать в течение 50 лет. Известно, что более ранние проекты задавали подобные цели: улучшение качества ПС, формализации сервисных моделей, снижение сложности за счет использования КПВ, создания отладочного инструментария для визуальной диагностики ошибок и их устранение т.д.. Однако коренного изменения в программировании пока не произошло. Привлечение техники формальной спецификации программ еще не означает, что в программе будут отсутствовать ошибки, поскольку ошибки в программных проектах, в интерпретации спецификаций МП, в документации пока нельзя распознать. Реализация международного проекта по верификации ПС поможет решить многие из этих вопросов.