среди равных партнеров. Таким образом, опыт, приобретаемый в группах, оказывает противодействие отчуждению, помогая решению проблем, возникающих при межличностном взаимодействии. Именно групповая работа с использованием компьютера или при обучении на компьютере, при которой дети формулируют совместно общую задачу и сами выбирают себе роль в группе, дает максимальный образовательный эффект.

Более того, само использование компьютеров в аудитории увеличивает возможности интерактивного обучения. Это особенно верно при активном внутригрупповом взаимодействии учащихся. Стимулируя взаимообучение, компьютеры существенно повышают мотивацию, учащиеся научаются обращаться за помощью друг к другу, что улучшает результаты их обучения. Ряд экспериментальных педагогических исследований показал преимущества различных моделей группового компьютеризованного обучения.

З.Х. Маматова

## СОЗДАНИЕ ПРОГРАММ НА ОСНОВЕ ФОРМАЛЬНЫХ ПРЕОБРАЗОВАНИЙ

(Ферганский государственный университет )

Технологии**,** модели и процессы создания ПО**.** Процесс создания программного обеспечения (ПО) – это множество взаимосвязанных процессов и результатов их выполнения, которые ведут к созданию программного продукта. Процесс создания ПО может начинаться с разработки программной системы "с нуля", но чаще новое ПО разрабатывается на основе существующих программных систем путем их модификации. Несмотря на то, что наблюдается огромное многообразие подходов, методов и технологий создания ПО, существуют фундаментальные базовые процессы, без реализации которых не может обойтись ни одна технология разработки программных продуктов. Фундаментальные процессы, присущие любому проекту создания ПО:

-Разработка спецификации требований на ПО (определяют функциональные характеристики системы и обязательны для выполнения).

-Создание программного обеспечения (создание ПО согласно спецификации).

-Аттестация ПО (Созданное ПО должно пройти аттестацию для подтверждения соответствию требованиям заказчика).

-Модернизация ПО (совершенствование ПО согласно измененным требованиям потребителя).

Модель процесса создания ПО – последовательность этапов, необходимых для разработки создаваемого ПО. Модели процесса разработки ПО:

1. Каскадная модель. 2. Эволюционная модель. 3. Формальное преобразование. 4. Сборка программных продуктов из ранее созданных компонентов (модель сборки). 5. Итерационная (спиральная) модель.

Методы создания ПО представляют собой структурный подход к созданию ПО, который способствует производству ПО эффективным, с экономической точки зрения, способом. Все методы основаны на использовании моделей системы в качестве спецификации ее структуры:

1. Функционально-ориентированные (структурный анализ, JSD) основаны на определении основных функциональных компонент системы.

2. Объектно-ориентированные используют подходы, основанные на использовании унифицированного языка моделирования UML.

Базовые процессы создания ПО: 1.Разработка спецификации. 2.Проектирование и реализация. 3. Аттестация. 4. Эволюция.

Формальные преобразования заключаются в автоматическом построении программы на основе ее формального описания, что гарантирует получение кода, который будет абсолютно точно соответствовать исходной спецификации. Формальные преобразования - это набор методов, позволяющих математически корректно трансформировать исходные спецификации в код целевой программной системы. Т.к. переход от требований к коду происходит математически корректно, то проблема тестирования и доказательства корректности конечной программы по отношению к спецификации исчезает. Но верификация спецификаций по отношению к требованиям к системе остается. Очевидно, что этот метод требует наличия формальных спецификаций, составление и доказательство корректности которых является нетривиальной задачей. Именно поэтому метод формальных преобразований редко используется для всего программного комплекса. В большинстве случаев он применяется для той части сложной системы и реализации тех алгоритмов, где исходные требования хорошо формализуемы.

Подходы на основе формальных преобразований**.**

-прозрачного (белого) ящика, представляющего реализацию в виде совокупности функций при пошаговом уточнении.

-Метод ящиков удобен, когда нам надо работать с функциями либо с закрытой, либо с открытой реализацией. Использование ящиков определяют следующие три принципа:

-все определенные при проектировании данные скрыты (инкапсулированы) в ящиках;

-все процессы определены как использующие ящики последовательно или параллельно;

-каждый ящик занимает определенное место в системной иерархии.

Черный ящик представляет собой точную спецификацию внешнего, видимого с пользовательской точки зрения поведения. Ящик с состояниями получаем из черного ящика выделением элементов истории стимулов, которые сохраняют состояние (инварианты состояний) в процессе ящике. Прозрачный ящик получаем из ящика с состояниями, определяя процедуру, выполняющую требуемое преобразование.

Формальные генетические подходы**.** Сложились методы программирования, обладающие свойством доказательности и не теряющие это точное, накопленное знание. Три таких метода соответствуют уже исследованным генетическим подходам, но с учетом формальных, математических спецификаций.

-Формальное синтезирующее программирование использует математическую спецификацию – совокупность логических формул. Существуют две разновидности синтезирующего программирования:

-логическое, в котором программа извлекается как конструктивное доказательство из спецификации, понимаемой как теорема;

-трансформационное, в котором спецификация рассматривается как уравнение относительно программы и символическими преобразованиями превращается в программу.

-Формальное сборочное программирование использует спецификацию как композицию уже известных фрагментов.

-Формальное конкретизирующее программирование использует такие подходы, как смешанные вычисления и конкретизацию по аннотациям.

Трансформационный подход к программированию**.** Программирование как вид деятельности почти исключительно состоит из следующих видов обработки программ: 1. Выполнение программ. *2.* Трансляция программ. *3.*Построение программ. Его можно рассматривать как систематический процесс преобразования теоремы существования решения задачи ∀x ∃y =S(x,y) в программу p(X), доставляющую в качестве p(x) требуемый y: ∀x ∃y S(x, y) –> p(X): ∀x S(x, p(x)). 4.Оптимизация программ. *5.* Перевод программ. Мы его отличаем от трансляции тем, что при переводе модели вычислений входного и объектного языков существенно различны (рекурсивные – итеративные, аппликативные – с памятью, последовательные – параллельные). 6. Комплексирование программ. Широкий круг манипуляций, включающий ассемблирование, макрообработку, редактирование и работу с библиотеками и в целом обеспечивающий сборку программы из частей, имеющих самостоятельное содержание. 7. Генерация программных комплексов. Также разнообразный набор.

В.А. Осипов, В.В. Графкин

## ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДОСТУПНОСТИ ИНФОРМАЦИИ ДЛЯ ПОЛЬЗОВАТЕЛЕЙ С НАРУШЕНИЯМИ ЗРЕНИЯ

(Самарский университет)

В данной статье рассматривается проблема доступности веб-контента для пользователей с нарушениями зрения. Определены основные принципы, которым должна соответствовать веб-страница с учетом разных правил, утвержденных документом Web Content Accessibility Guidelines (WCAG 2.0), целью которого является формирование единых стандартов доступности веб-контента, от-