Министерство образования и науки российской федерации

Севастопольский государственный университет

Институт информационных технологий и управления в технических системах

# РЕФЕРАТ

на тему: методология разработки программного обеспечения Cleanroom

по дисциплине: «Технологии создания программных продуктов»

Выполнила ст.гр. ИС/б-25

Изосимина Юлия

Проверил:

Строганов В.А.

Севастополь 2016г.

# ОГЛАВЛЕНИЕ

[ОГЛАВЛЕНИЕ 1](#_Toc452013683)

[ВВЕДЕНИЕ 2](#_Toc452013684)

[1.1 Разработка и демонстрация 4](#_Toc452013688)

[1.2 Проверка и практика 5](#_Toc452013689)

[1.3 Cleanroom и объектно-ориентированный подход 7](#_Toc452013690)

[ЗАКЛЮЧЕНИЕ 11](#_Toc452013691)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 12](#_Toc452013692)

# ВВЕДЕНИЕ

Программные системы в настоящее время присутствуют повсеместно: практически любые электронные устройства содержат программное обеспечение (ПО) того или иного вида. Без соответствующего программного обеспечения в современном мире невозможно представить индустриальное производство, школы и университеты, систему здравоохранения, финансовые и правительственные учреждения. Многие пользователи применяют ПО для самообразования, для развлечений и т.д. Создание спецификации требований, разработка, модификация и сопровождение таких систем ПО составляет суть технической дисциплины инженерия программного обеспечения(software engineering, SE).

Даже простые системы ПО обладают высокой степенью сложности, поэтому при их разработке приходится применять весь арсенал технических и инженерных методов. Проектирование ПО – процесс определения архитектуры, компонентов, интерфейсов, других характеристик системы и конечного результата.

Методологии представляют собой ядро теории управления разработкой программного обеспечения. К существующей классификации в зависимости от используемой в ней модели жизненного цикла (водопадные и итерационные методологии) добавилась более общая классификация на прогнозируемы и адаптивные методологии.

Адаптивныеметодологии нацелены на преодоление ожидаемой неполноты требований и их постоянного изменения. Когда меняются требования, команда разработчиков тоже меняется. Команда, участвующая в адаптивной разработке, с трудом может предсказать будущее проекта. Существует точный план лишь на ближайшее время. Более удаленные во времени планы существуют лишь как декларации о целях проекта, ожидаемых затратах и результатах.

Выделяют следующие основные модели процесса или методологии разработки ПО:

* Каскадная разработка или модель водопада (англ. waterfall model) — модель процесса разработки программного обеспечения, в которой процесс разработки выглядит как поток, последовательно проходящий фазы анализа требований, проектирования, реализации, тестирования, интеграции и поддержки.
* Итеративная разработка (англ. iteration — повторение) — выполнение работ параллельно с непрерывным анализом полученных результатов и корректировкой предыдущих этапов работы. Проект при этом подходе в каждой фазе развития проходит повторяющийся цикл: Планирование — Реализация — Проверка — Оценка (англ. plan-do-check-act cycle).

Существуют такие методологии как SCRUM (методология, предназначенная для небольших команд (до 10 человек)), KANBAN (гибкая методология разработки программного обеспечения, ориентированная на задачи), DYNAMIC SYSTEM DEVELOPMENT METHOD (появился в результате работы консорциум из 17 английских компаний), MICROSOFT SOLUTIONS FRAMEWORK (методология разработки программного обеспечения, предложенная корпорацией Microsoft), RATIONAL UNIFIED PROCESS (методология разработки программного обеспечения, созданная компанией Rational Software) и др.

# В данной работе я рассмотрела методологию Cleanroom Software Engineering

## 1.1 Разработка и демонстрация

Теоретические основы Cleanroom были установлены в конце 1970-х и в начале 1980-х Harlan Mills, опытным математом и IBM. Он связал фундаментальные идеи в математике, статистике и разработке в программное обеспечение. Под влиянием Edsger Dijkstra на структурном программировании, Nicholas Wirth на пошаговом усовершенствовании, и David Parnas на модульной конструкции, Mills определил научные основы для технического подхода к программному обеспечению. Два фундаментального понимания управляло работой Mills: во-первых, это программы являются правилами для математических функций и во-вторых, что потенциальные программы выполнение есть у бесконечного числа населения, требующее статистической выборки для качественной сертификации. Первое понимание открыло всю теорию функций к разработке программного обеспечения и вывод технологий спецификации структуры поля и схем, функционально-теоритическая проверка правильности и поэтапная разработка. Второе понимание открыло всю статистическую теорию к тестированию программного обеспечения и привело к технологии статистическое тестирование использования и качественная сертификации.

Идеи Mills были усовершенствованы и продемонстрированы в сотрудничестве с коллегами Alan Currit, Michael Dyer, Alan Hevner, Richard Linger, Bernard Witt, и другими в Подразделении Федеральных систем IBM.

Теория и Практика (Linger, Mills и Witt), опубликованный в 1979 Addison-Wesley, были представлены функционально-теоретические методы для требований к программному обеспечению, проектированию, проверки и повторной разработке.

Анализ и проектирование (Mills, Linger, and Hevner; Academic Press, Inc., 1986) представлял методы структуры поля для системной спецификации, проектирования и проверки, и была представлена поэтапная разработка для управления проектами. В 1987 эти идеи были интегрированы под титульные данные cleanroom, срок занял от полупроводниковой отрасли, чтобы отразить акцент на дефектное предотвращение вместо того, чтобы дезертировать удаление.

"Cleanroom Software Engineering" (by Mills, Dyer, and Linger), был опубликован в выпуске в мае 1987 программного обеспечения IEEE.

Первым проектом программного обеспечения Cleanroom управлял Richard Linger IBM в середине 1980-х. Проект COBOL Structuring Facility разработало коммерческое ПО повторно сооружающее продукт, что показанные замечательные уровни качество и надежность в использовании заказчика, и предусмотрел исходный контроль данных процесса «чистое помещение».

## 1.2 Проверка и практика

В 1990 Richard Linger установил IBM Cleanroom Software Technology Center, где дальнейшее совершенствование методов «чистого помещения», автоматизации, и передача технологии была достигнута. В начале 1990-х массовое хранение элементов управления адаптера, разработанное используя Cleanroom, было представлено IBM. Тысячи модулей были проданы, и после расширенной жизни продукт был удален в 1997 без единственного полевого отказа, сообщенный против микрокода Cleanroom. Разработка была во главе с Mike Brewer и включала Paul Fisher, Dave Fuhrer, Karl Nielson, и другич членов команды. Сертификационное испытание было во главе с Joe Ryan и Mike Houghtaling.

Сегодня, лаборатория тестирования в Разделении Систем хранения IBM возможно мировой лидер в практике статистического тестирования использования из программного обеспечения.

В конце 1980-х и в начале 1990-х высоко оцененная Разработка программного обеспечения Лаборатория (SEL), а тогда Аэронавтика и Космическое Администрирование (НАСА), Центр космических полетов имени Годдарда (GSFC) провел ряд экспериментов Cleanroom под руководством Vic Basili, Scott Green, Rose Pajerski, Jon Valett, и других. Серии SEL экспериментов Cleanroom рассматривают что-то, чтобы быть одной из самых больших исследовательских анализов, проводящих, для того чтобы датировать разработка программного обеспечения. Четыре системы программного обеспечения наземного управления увеличивающегося размера были разработаны, используя разработку «чистого помещения», с результатами, показывающими непротиворечивый улучшение по качеству и производительность по уже впечатляющему НАСА базовая линия GSFC.

В течение формирующего периода американского Министерства обороны (DoD) программа ARPA STARS (Разработка программного обеспечения для Адаптируемых, Надежных Систем) в середине 1980-х, Cleanroom было выбрано как ключевая технология для разработки и коммерциb лидерами STARS, включая Dave Ceely, Dick Drake, BillEtt, Joe Greene, John Foreman, Jim Moore и другие. Компания Dr.Mills, основанная с Arnie Beckhardt, чтобы усовершенствовать Разработку программного обеспечения «чистого помещения», был выбран как механизм STARS для коммерциализации технологии «чистого помещения». Значительные шаги вперед в методах «чистого помещения» и инструментах были сделаны множеством под поддержкой STARS. В начале 1990-х американский армейский fрсенал Picatinny продил проект «чистое помещение», во время которого 20:1 доход от инвестиций в технологии «чистого помещения» введение было понято. В 1996 Центр Данных и Анализа DoD Программного обеспечения сообщило о существенном преимуществе стоимости и качества для «чистого помещения» в сравнительном анализе методов программного обеспечения. Другие организации с историческим данными по производительности программного обеспечения и качеству провели использование крупных проектов «чистое помещение» и опубликовало результаты в открытой литературе. Методы «чистое помещение» произвели драматические улучшения итогов проекта программного обеспечения в IBM, Ericsson, НАСА, DoD и многих других организациях. Данные по Cleanroom находится, и они последовательно показывают, что существенное улучшение производительности команды программного обеспечения возможна под дисциплиной Cleanroom.

Институт программной инженерии (SEI) в Университете Карнеги-Меллон предусмотрело существенное национальное лидерство для усовершенствования в программном обеспечении практики. Модель технологической завершенности SEI для программного обеспечения (CMM) стала общепринятым и широко распространенным управлением моделируют для улучшения программного обеспечения. В 1996 SEI завершил проект, чтобы определить справочную модель «чистого помещения» и карту технических технологий «чистого помещения» в процессах управления CMM. Основное открытие этой работы состояло в том, что «чистое помещение» и CMM совместимые и взаимно поддерживают. Эта работа была распространена в двух технических отчетах SEI: Cleanroom Software Engineering Reference Model (by Linger and Trammell 1996), и Cleanroom Software Engineering Implementation of the Capability Maturity Model (CMM) для программного обеспечения (by Linger, Paulk, and Trammell 1996).

## 1.3 Cleanroom и объектно-ориентированный подход

Процесс проектирования в технологии Cleanroom сводится к формальному описанию функций, необходимых для реализации поведения "черного ящика", т. е. к созданию модуля-описателя. Данный процесс напоминает проектирование объектов в объектном программировании, когда данные и функции (методы) инкапсулируются в единой сущности. При написании программ должны использоваться только базовые конструкции из технологии структурного программирования (блоки, ветвления, циклы). Качество программного кода (соответствие работы программы заложенным спецификациям) проверяется в ходе верификации.

Идеологи Cleanroom считают, что сочетание принципов Cleanroom и ОО подхода с его концентрацией на повторном использовании программных компонентов позволяет создавать ПО, отвечающего как концепции повторного использования, так и предсказуемости и высокого качества. Например, ОО методы могут быть использованы при анализе предметной области, а методики Cleanroom – при проектировании и разработке.

Преимущества Cleanroom заключаются в корректности, надежности и понятности программного продукта. Они проявляются в виде сокращения числа ошибок, обнаруживаемых в период эксплуатации ПО, сокращению времени разработки, простоты поддержки ПО и удлинения сроков эксплуатации.

Данная технология минимизирует затраты компании-разработчика на стадии поддержки программного продукта за счет увеличения затрат на стадии разработки. Как правило, такой подход в целом снижает совокупные затраты разработчика на всем сроке жизненного цикла продукта.

В разработке ПО методом "чистая комната" можно выделить пять ключевых моментов:

1. Формальная спецификация*.*Длясоздаваемой системы разрабатывается формальная спецификация. Для записи спецификации используется модель состояний, в которой отображены отклики системы на стимулы.
2. Пошаговая разработка*.*Разработка ПО разбивается на несколько этапов, которые выполняются и проверяются методом "чистая комната" независимо друг от друга. Этапы определяются совместно с заказчиком на ранних стадиях процесса создания программного продукта.
3. Структурное программирование*.*Используется только ограниченное количество управляющих конструкций и абстракций данных. Процесс разработки программы – это процедура поэтапной детализации спецификации.
4. Статическая верификация*.*Разрабатываемое ПО проверяется статическим методом строгого инспектирования ПО. Для модулей или отдельных элементов тестирование кода не проводится.
5. Статистическое тестирование системы*.*На каждом шаге разработки проводится тестирование статистическими методами, позволяющими оценить надежность программной системы. Как показано на рис. 20.5, статистические тесты базируются на операционном профиле, который разрабатывается параллельно созданию спецификации системы.

Процесс разработки ПО методом "чистая комната" планируется таким образом, чтобы обеспечить строгое инспектирование программ. Спецификация системы представлена моделью состояний, которая через ряд последовательных моделей постепенно преобразуется в исполняемую программу. Инспектирование программ дополняются строгими математическими доказательствами согласованности и корректности преобразований.

Обычно разработкой больших систем методом "чистая комната" занимаются три группы разработчиков.

1. Группа спецификации*.*Отвечает за разработку и поддержку системной спецификации. Этой группой создаются спецификации пользовательских требований и формальные спецификации для верификации системы. В некоторых случаях, например после окончания разработки спецификации, эта группа может присоединиться к группе разработки.
2. Группа разработки*.*Занимается разработкой и проверкой ПО. При проверке используется структурированный формальный подход, основанный на инспектировании кода, подкрепленный доказательством правильности работы системы.
3. Группа сертификации*.*Занимается разработкой статистических тестов, применяемых после окончания разработки ПО. Все тесты основаны на использовании формальной спецификации. Контрольные тесты разрабатываются параллельно с созданием системы и используются для сертификации надежности ПО.

На рис. 1 представлена модель процесса разработки ПО методом "чистая комната".



Рисунок 1. Метод разработки ПО «Чистая комната»

# ЗАКЛЮЧЕНИЕ

В результате использования метода "чистая комната" готовый программный продукт содержит крайне мало ошибок и его стоимость меньше, чем у разработанного традиционными методами. В процессе разработки методом "чистая комната" оказывается рентабельной статическая проверка. Огромное количество дефектов обнаруживается еще до исполнения программы и исправляется в процессе разработки ПО. Во время тестирования проектов, которые разрабатывались с использованием метода "чистая комната", в среднем обнаруживается только 2,3 дефекта на тысячу строк исходного кода. В целом расходы на разработку не увеличиваются, так как сокращаются расходы на тестирование и исправление ошибок в разрабатываемой программной системе.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

# [COBOL/SF 1988a) COBOL Structuring Facility Re-Engineering Concepts. IBM Publication SC34-4079. 1988.

# (Mills 1987) Mills. H. D.: Dyer. M.: & Linger. R. C. "Cleanroom Software Engineering." IEEE Software. (Sept. 1987).

# Cleanroom Software Engineering: Technology and Process Stacy J. Prowell Carmen J. Trammell Robert C. Linger Jesse H. Poore. 1999.