Технические науки

ТЕХНОЛОГИЧЕСКИЕ СРЕДСТВА И МЕТОДЫ ПОВЫШЕНИЯ НАДЕЖНОСТИ ПРОГРАММ

Аббазова Р.А., Белова С.В.

Волжский политехнический институт, филиал Волгоградского государственного технического университета, Волжский, e-mail: rimma2705@mail.ru

На показатели качества проектирования и созданного программного продукта влияют многие факторы – это структурная упорядоченность комплекса программ и данных, степень комплексной автоматизации технологии проектирования программ, документированность создаваемых программ и всего комплекса, квалификация специалистов и др.

Жизненный цикл программного продукта – это непрерывный процесс, который начинается с момента принятия решения о необходимости его создания и заканчивается в момент его полного изъятия из эксплуатации, состоит из нескольких этапов: требования/спецификации, проектирование, реализация проекта, отладка, сопровождение проекта.

Определение полного комплекса требований к программному обеспечению системы является первоначальной задачей. На этом этапе следует убедиться в том, что к программному обеспечению предъявлены четкие и корректные требования, которые действительно могут быть выполнены.

На этапе проектирования создается структура будущей программой системы, на этом этапе закладывается качество и надежность будущего программного продукта. Необходимо проверить, насколько точно отражены в проекте все требования к программному обеспечению, то есть должно быть установлено соответствие проекта основным требованиям. В результате деятельности на этапах требований и проектирования должен быть получен проект системы, содержащий достаточно информации для реализации системы.

Реализация подразумевает выбор языка программирования и составление текста программы. В реализации обычно выделяют два этапа – реализацию компонент программного обеспечения и интеграцию компонент в готовый продукт.

На этапе отладки затрачиваются наибольшие усилия по повышению надежности программного обеспечения. В процессе отладки производится проверка, включающая тестирование и другие методы. Тестирование – это процедура обнаружения ошибок в программе, является составляющей частью отладки. Успех отладки в значительной степени предопределяет рациональная организация тестирования.

Сопровождение охватывает все действия по повышению надежности после завершения отладки. Этот этап включает следующие виды работ: анализ несоответствий в программной системе, вызывающих сбои в ее работе; коррекцию программных ошибок; разработку усовершенствованных версий программного обеспечения; функциональное расширение или улучшение производительности.

Влияние на надежность различных этапов жизненного цикла разное. Существуют технологические средства и методы, которые являются необходимыми для разработки надежного программного обеспечения на разных фазах жизненного цикла программы:

Тестирование – выполнение программы при заданных условиях с целью получения реальных результатов ее работы. По этим данным делается вывод о степени соответствия программы предъявляемым требованиям.

Анализ – логическая или математическая обработка аналитических или эмпирических данных. Анализ может включать оценку выполняемых логических функций, числовых или статистических характеристик алгоритмов и формул, затрат памяти и времени, использование внешней памяти, системы приоритетов и т. д.

Демонстрация – выполнение функциональных задач перед квалифицированными программистами.

Инспекция – проверка программы на соответствие требованиям, указанным в документации.

Для создания комплексов программ высокого качества, в том числе по показателям надежности функционирования, необходимо разрабатывать не только методы, способствующие повышению надежности, но и методы, позволяющие рассчитывать надежность в зависимости от затрат на различные средства ее повышения.

Список литературы

1. Хашемиан Х.М., Датчики технологических процессов: характеристики и методы повышения надежности, Издательство: Бином 2008 – 336 с.

2. Гласе Р. Руководство по надежному программированию / Пер. с англ. Ю.П. Кондранина, В.М. Рабиновича; Под ред. В.М. Рабиновича. Предисл. В.В. Липаева. – М.: Финансы и статистика, 1982 – 256 с.

ПОШАГОВАЯ ДЕТАЛИЗАЦИЯ КАК МЕТОД ПРОЕКТИРОВАНИЯ АЛГОРИТМОВ

Абросимова А.С., Белова С.В.

Волжский политехнический институт, филиал Волгоградского государственного технического университета, Волжский, e-mail: smile-1909@mail.ru

Проектирование является одной из основных фаз жизненного цикла программного обеспечения. Задачей этапа проектирования является исследование структуры системы и логических взаимосвязей ее элементов. На этапе проектирования создается структура будущей программы.

Современный подход к проектированию программ основан на декомпозиции задачи, которая в свою очередь основана на использовании абстракций. Целью при декомпозиции является создание модулей, которые представляют собой небольшие, относительно самостоятельные программы, взаимодействующие друг с другом. Если эта цель достигнута, то разработка отдельных модулей может осуществляться различными людьми независимо друг от друга, при этом объединенная программа будет функционировать правильно.

Сначала производится проектирование архитектуры программной системы. Это предполагает первичную стадию проектирования структуры системы.

Следующим шагом является детальное проектирование. На этом этапе происходит процедурное описание программы, выбор и оценка алгоритма для реализации каждого модуля.

Для проектирования модульных программ применяются два основных метода: нисходящего и восходящего проектирования.

В соответствие с методом нисходящего проектирования сначала кодируются, тестируются и отлаживаются модули самого высокого уровня.

Применение метода нисходящего проектирования основано на пошаговой детализации решения задачи. Начиная с верхних, самых общих шагов, на каждом следующем происходит все большее уточнение функций, выполняемых программой, до полной их реализации.

Метод нисходящего проектирования позволяет обнаружить и исправить ошибки взаимосвязи

блоков и логические ошибки на более ранних этапах программирования, когда внесение изменений еще не приводит к коренной перестройке всей программы.

Основная идея метода нисходящего проектирования – не пытаться программировать сразу. Пошаговая детализация автоматически заставляет программиста формировать понятную ему же структуру программы. Аккуратное проектирование приводит к тому, что программист хорошо представляет себе работу каждой конкретной подзадачи, ее входные и выходные данные, и потому в состоянии протестировать именно ее. Также упрощается и последующая отладка – при получении неверного результата программа может быть протрассирована, и проверка результата на очередном шаге сведется к пониманию, верно или неверно отработала очередная подзадача.

Достоинства метода пошаговой детализации:

• сохраняется целостность программы: от сложного к простому;

• проектирование программы, кодирование, проверку и документирование можно делать параллельно;

• в каждый момент времени, даже в начале разработки имеется работающий вариант программы.

По методу восходящего проектирования в первую очередь разрабатываются модули самого нижнего уровня. Эти модули, работоспособность которых уже проверена, включаются в разрабатываемые модули более высокого уровня.

Методу восходящего проектирования присущ ряд недостатков:

1) выявление ошибок алгоритма и сопряжений блоков осуществляется в конце разработки, что усложняет процесс внесения изменений;

2) при переходе на новый уровень требуются новые тестовые данные, что увеличивает трудоемкость разработки;

3) затрудняется процесс отладки, т.к. на каждом новом уровне тестированию подвергается все большее число блоков и связей.

Оба метода обладают как достоинствами, так и недостатками. При нисходящем проектировании до его завершения остаются неизвестными размер программы и ее эксплуатационные характеристики, так как они определяются в основном модулями нижнего уровня. А в методе восходящего проектирования принципиальные ошибки в проекте модулей нижнего уровня будут выявлены лишь на заключительной стадии работы.

На практике наилучшие результаты дает сочетание обоих методов. В таких случаях первый шаг разработки программы заключается в создании общей логической структуры, а затем общих модулей при котором сначала создаются модули верхних уровней и самые критичные модули нижнего уровня, которые наиболее часто используются, после чего применяется метод проектирования «сверху-вниз».

Список литературы

1. Окулов С. Программирование в алгоритмах, Из-во: Бином, 2007. – 384 с.

Секция «Проблемы моделирования, проектирования и разработки программных средств», научный руководитель – Рыбанов А.А., канд. техн. наук, доцент

ИССЛЕДОВАНИЕ МЕТОДОВ ЗАЩИТЫ ДЕЦЕНТРАЛИЗОВАННЫХ СИСТЕМ ОТ DOS И СПАМ АТАК (НА ПРИМЕРЕ ПРОГРАММЫ FROST В СЕТИ FREENET)

Шумихин И.И., Моженков В.В.[1]

Волжский политехнический институт, филиал ВолгГТУ, Волжский, e-mail: igor13409@mail.ru

В настоящее время многие государства вводят цензуру коммуникаций, решают какую информацию разрешить, какую запретить. Тем самым ограничивают такие права людей, как право на свободу слова и право на получение информации. Вследствие этого появляется проблема защиты информации от цензуры.

Данную проблему помогают решать системы, построенные на основе децентрализованных сетей, позволяющие анонимно обмениваться информацией, а также устранить возможность для любой группы лиц навязывать свои убеждения и ценности другим.

С появлением анонимных сетей также появилась проблема защиты данных сетей от различных видов атак, целью которых чаще всего является уменьшение уровня доступности информации.

Целью данной работы является модификация программы Frost, предоставляющей эквивалент форумов в сети Freenet. Целью модификации является защита существующей системы форумов от DoS и спам атак.

Одним из существующих методов защиты от такого вида атак является Сеть Доверия или Web of Trust. Принципом действия сети доверия является присваивание пользователями друг другу оценки качества сообщений. Вследствие этого появляется возможность фильтровать сообщения пользователей с оценкой выше указанной планки. Также, для значительного усложнения манипуляций с рейтингом со стороны злоумышленника, происходит автоматическое обновление личного списка оценок на основании списка оценок других пользователей.

Данный метод стабильно работает в сети небольшим количеством пользователей, но при значительном увеличении пользователей, также падает производительность.

В данной работе используется модификация первого метода, также называющаяся сетью агентов. Этот метод похож на предыдущий, но имеет несколько значительных отличий в алгоритме. Он заключается в том, что некоторые из пользователей регистрируются как агенты и подтверждают каким-либо образом, что они живые люди. Далее агенты дают оценку значимости каждому прочитанному сообщению, добавляя в личный список ключ сообщения и его оценку. Улучшение производительности заключается в том, что пользователи оценивают не каждый каждого, а только агентов, и значительно ускоряется процесс обновления оценок. Также у каждого агента есть общедоступный ключ списка сообщений, к которому обращаются пользователи для определения значимости сообщения. Список оценок каждого агента периодически дополняется списками других агентов.

Для последующего решения ставятся следующие задачи:

1. Разработка и программная реализация алгоритма используемого метода.

2. Анализ и тестирование модифицированного приложения.

Список литературы

1. Моженков В.В. Anonymous XML. – Kingston University, 2009.

2. Ian Clarke. FMS: Spam-proof anonymous message boards on Freenet, 2008-05-11, http://blog.locut.us/2008/05/11/fms-spam-proof-anonymous-message-boards-on-freenet.

[1] Научный руководитель – Моженков В.В.