Модульное и интеграционное тестирование

Модульное тестирование - это тестирование программы на уровне отдельно взятых модулей, функций или классов.

Цель модульного тестирования – выявление локализованных в модуле ошибок в реализации алгоритмов, а также определение степени готовности системы к переходу на следующий уровень разработки и тестирования.

Особенности модульного тестирования:

-модульное тестирование проводится по принципу "белого ящика", то есть основывается на знании внутренней структуры программы, и включает те или иные методы анализа покрытия кода

-в ходе модульного тестирования для каждого модуля обычно создаются заглушки для соответствующих интерфейсов, причем некоторые из них могут использоваться для подачи входных значений, другие для анализа результатов, присутствие третьих может определяться требованиями, накладываемыми компилятором и сборщиком

Модульное тестирование

Модульное тестирование характеризуется степенью, в которой тесты выполняют или покрывают логику программы (исходный текст).

Модульные тесты, связанные со структурным тестированием,

формируются по следующим принципам:

-на основе потока управления, когда элементы, которые должны быть покрыты при прохождении тестов, определяются на основе структурных критериев тестирования C0 (команд), C1 (ветвей), C2 (путей) , к которым относятся вершины, дуги, пути управляющего графа программы, условия, комбинации условий и т. п.

-на основе потока данных, когда элементы, которые должны быть покрыты при прохождении тестов, определяются при помощи потока данных, т. е. информационного графа программы

Тестирование программного обеспечения

Структурные критерии основываются на основных элементах управляющего графа программы, операторах, ветвях, путях:

- критерий тестирования команд (критерий C0) - набор тестов в совокупности должен обеспечить прохождение каждой команды

не менее одного раза. Это слабый критерий и, как правило, используется в больших программных системах, где другие критерии применить невозможно.

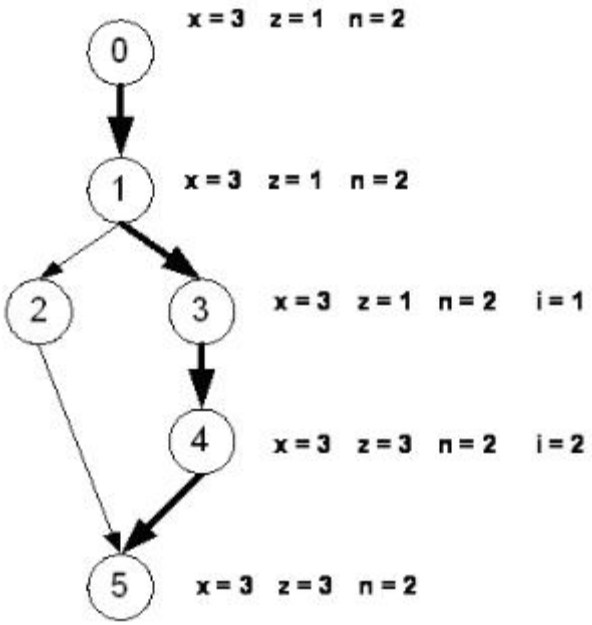
- критерий тестирования ветвей (критерий C1) - набор тестов в совокупности должен обеспечить прохождение каждой ветви

не менее одного раза. Этот достаточно сильный и, при этом, экономичный критерий, поскольку множество ветвей в тестируемом приложении конечно и не велико, используется в системах автоматизации тестирования.

- критерий тестирования путей (критерий C2) - набор тестов в совокупности должен обеспечить прохождение каждого пути

не менее одного раза. Если программа содержит цикл (особенно,

с неявно заданным числом итераций), то число итераций ограничивается константой (часто - двум или количеством классов выходных путей).



Тестирование программного

обеспечения

Управляющий граф программы – все возможные варианты путей выполнения кода программы.

Трасса - это "сохраненный путь" на управляющем графе программы, т.е. зафиксированные в журнале записи о состояниях переменных в заданных точках в ходе выполнения программы.

Трасса, проходящая через вершины 0-1-3-4-5

№ вершины - оператора	Значение переменной x	Значение переменной z	Значение переменной n	Значение переменной i
0	3	1	2	нет

1	3	1	2	нет
3	3	1	2	1
4	3	3	2	2
5	3	3	2	нет

Дамп – область памяти, состояние которой фиксируется в контрольной точке в виде единого массива или нескольких связанных

массивов.

–

Модульное тестирование

Тестирование на основе потока управления - к структурным критериям тестирования С0 (команд), С1 (ветвей), С2 (путей) следует добавить критерий покрытия условий, заключающийся в покрытии всех логических (булевских) условий в программе.

Критерии покрытия решений и условий не заменяют друг друга, поэтому на практике используется комбинированный критерий покрытия условий/решений, совмещающий требования по покрытию и решений и условий.

Также часть используется критерий покрытия вызовов функций программы, в

соответствии с которым каждый вызов каждой функции в программе должен быть осуществлен хотя бы один раз.

Тестирование на основе потока данных – направлено на выявление ссылок на неинициализированные переменные и избыточные присваивания, т.е. на аномалии потока данных. При этом выполняется тестирование всех

взаимосвязей, включающих в себя ссылку (использование) и определение переменной, на которую указывает ссылка. Недостаток этого вида тестирования в том, что он не включает критерий С1, и не гарантирует покрытия решений.

Модульное тестирование

Для достижения заданной степени тестированности в структурном тестировании целесообразно выделить три этапа процесса построения набора тестов:

-конструирование управляющего графа программы соответствует статическому анализу программы, задача которого состоит в получении управляющего графа программы и зависящего от него и от критерия тестирования множества элементов, которые необходимо покрыть тестами.

-выбор тестовых путей обеспечивает выбор и построение тестовых путей

иможет быть реализован какими-либо из следующих методов:

-статические методы,

-динамические методы,

-методы реализуемых путей.

-генерация тестов, соответствующих тестовым путям, осуществляется поиском подходящих тестов, реализующих прохождение этих известных путей.

Модульное тестирование

Статические методы выбора тестовых путей.

Самое простое и легко реализуемое решение - построение каждого пути посредством постепенного его удлинения за счет добавления дуг, пока не будет достигнута выходная вершина управляющего графа программы.

В адаптивных методах каждый раз добавляется только один тестовый путь (входной тест), используя предыдущие пути (тесты) как руководство для выбора последующих путей в соответствии с некоторой стратегией. Основной недостаток статических методов заключается в том, что не учитывается возможная нереализуемость построенных путей тестирования.

Достоинство статических методов выбора тестовых путей состоит в сравнительно небольшом количестве необходимых ресурсов, как при использовании, так и при разработке, однако их реализация может содержать непредсказуемый процент брака (нереализуемых путей). Кроме того, в этих системах переход от покрывающего множества путей к полной системе тестов осуществляется вручную, а эта работа достаточно трудоемкая.

Модульное тестирование

Динамические методы выбора тестовых путей.

Динамические методы выбора тестовых путей предполагают построение полной системы тестов, удовлетворяющих заданному критерию, путем одновременного решения задачи построения покрывающего множества путей и тестовых данных. При этом можно автоматически учитывать реализуемость или нереализуемость ранее рассмотренных путей или их частей. Основной идеей динамических методов является подсоединение к начальным реализуемым отрезкам путей дальнейших их частей так, чтобы:

-не терять при этом реализуемости вновь полученных путей;

-покрыть требуемые элементы структуры программы.

Динамические методы выбора тестовых путей требуют значительно больших ресурсов как при разработке, так и при эксплуатации, однако

увеличение затрат происходит, в основном, за счет разработки и эксплуатации способов определения реализуемости пути (символический интерпретатор, решатель неравенств). Достоинство этих методов заключается в том, что получаемый с их помощью результат имеет некоторый качественный уровень - реализуемость путей.

–

Интеграционное тестирование

Интеграционное тестирование - это тестирование части системы, состоящей из двух и более модулей. Основная задача интеграционного тестирования - поиск дефектов, связанных с ошибками в реализации и интерпретации интерфейсного взаимодействия между модулями.

С технологической точки зрения интеграционное тестирование является количественным развитием модульного, поскольку так же, как и модульное тестирование, работает с интерфейсами модулей и подсистем и требует создания тестового окружения, включая заглушки на месте отсутствующих модулей.

Основное различие между модульным и интеграционным тестированием состоит в целях, то есть в видах обнаруживаемых дефектов, которые определяют стратегию выбора входных данных и методов анализа. При интеграционном тестировании применяются методы, связанные с покрытием интерфейсов, например, вызовов функций или методов, или анализ использования интерфейсных объектов, таких как глобальные ресурсы, средства коммуникаций, предоставляемых операционной системой.