**Тестирование разрабатываемой системы**

При создании программного проекта около 50% общего времени и более 50% общей стоимости расходуется на проверку (тестирование) разрабатываемой программы или системы. Кроме того, доля стоимости тестирования в общей стоимости программ имеет тенденцию возрастать при увеличении сложности комплексов программ и повышения требований к их качеству.

Учитывая это, при отработке технологии тестирования программ следует четко выделять определенное (по возможности не очень большое) число правил отладки, обеспечивающих высокое качество программного продукта и снижающих затраты на его создание.

Тестирование – очень важный и трудоемкий этап процесса разработки программного обеспечения, так как правильное тестирование позволяет выявить подавляющее большинство ошибок, допущенных при составлении программ. Процесс разработки программного обеспечения предполагает три стадии тестирования: автономное, комплексное и системное, каждая из которых соответствует завершению соответствующей части системы. Различают два подхода к формированию тестов: структурный и функциональный. Каждый из указанных подходов имеет свои особенности и области применения.

Недостаточно выполнить проектирование и кодирование программного обеспечения, необходимо также обеспечить его соответствие требованиям и спецификациям. Многократно проводимые исследования показали, что чем раньше обнаруживаются те или иные несоответствия или ошибки, тем больше вероятность их правильного исправления и ниже его стоимость.

Тестирование – это процесс исполнения программы с целью обнаружения ошибок. Никакое тестирование не может доказать отсутствие ошибок в хоть сколько-нибудь сложном программном обеспечении. Для такого программного обеспечения выполнение полного тестирования, т. е. задания всех возможных комбинаций исходных данных, становится невозможным, а, следовательно, всегда имеется вероятность того, что в программном обеспечении остались не выявленные ошибки. Однако соблюдение основных правил тестирования и научно обоснованный подбор тестов может уменьшить их количество.

Процесс разработки программного обеспечения, в том виде, как он определяется в современной модели жизненного цикла программного обеспечения, предполагает три стадии тестирования:

* автономное тестирование компонентов программного обеспечения;
* комплексное тестирование разрабатываемого программного обеспечения;
* системное или оценочное тестирование на соответствие основным критериям качества.

В качестве методов автономного тестирования компонентов системы используются различные подходы:

* Метод ручного контроля;
* Тестирование по принципу «белого ящика»;
* Тестирование по принципу «черного ящика».

Согласно [Иванова], существуют следующие подходы для комплексного тестирования компонентов ПО:

* Восходящее тестирование;
* Нисходящее тестирование.

Восходящее тестирование предполагает, что каждый компонент тестируют отдельно на соответствие имеющимся спецификациям на него, затем собирают оттестированные компонент в компоненты более высокой степени интеграции и тестируют их. При этом проверяют интерфейсы между компонентами, используемые для подключения компонентов более низкого уровня иерархии. И так далее, пока не будет собран весь программный продукт.

Нисходящее тестирование заключается в том, что, когда проектирование какого-либо компонента заканчивается, его кодируют и передают на тестирование. При его тестировании все вызываемые им компоненты заменяют компонентами, которые в той или иной степени имитируют поведение вызываемых компонентов. Такие компоненты принято называть «заглушками». В отличие от тестирующих программ заглушки очень просты, например, они могут просто фиксировать, что им передано управление. Часто заглушки просто возвращают какие-либо фиксированные данные. Как только тестирование основного компонента завершено, к нему подключают компоненты, непосредственно им вызываемые, и необходимые заглушки, а затем проводят их совместное тестирование. Далее последовательно подключают следующие компоненты, пока не будет собрана вся система.

При разработке репозитория инсталляционных пакетов первым делом разрабатывались классы моделей сущностей, которые были выделены в результате объектной декомпозиции, затем производилась разработка контроллеров, которые описывают бизнес-процессы системы. Параллельно с разработкой моделей и контроллеров велась разработка пользовательского интерфейса. При этом, контроллеры, которые должны были вызываться элементами пользовательского интерфейса были заменены на «заглушки». Таким образом, тестирование репозитория инсталляционных пакетов происходило с использованием как восходящего подхода, так и нисходящего подхода.

**Тестирование методом ручного контроля**