

МЕХАНИЗМЫ ВЫБОРА УРОВНЯ МОНИТОРИНГА ПРИ ОБЕСПЕЧЕНИИ НАДЕЖНОСТИ ПРОГРАММНЫХ СРЕДСТВ.

Н.В. Успенская, В.В. Бахтизин

Кафедра программного обеспечения информационных технологий,
Белорусский государственный университет информатики и радиоэлектроники

Минск, Республика Беларусь

E-mail: nvuspenskaya@gmail.com

Использование проектирования по контракту является мощным механизмом для обеспечения надежности программных средств. В связи с чем существует необходимость определения оптимального уровня и механизмов мониторинга для обеспечения надежности и высокой производительности программных средств..

Для обеспечения надежности программных средств (ПС) предлагается использовать проектирование по контракту (Design by Contract). Данный способ устанавливает отношение между модулями программного средства, однозначно устанавливающий права и обязанности сторон. Такой подход обеспечивает точное определение требований и ответственности для каждого модуля, что повышает тестируемость и диагностируемость программных средств, что в свою очередь обеспечивает их надежность [1].

Основная идея контрактного программирования — это модель взаимодействия элементов ПС. Взаимодействие между модулями ПС происходит по схеме «клиент-поставщик» в соответствии с определенным контрактом. Контракт некоторого объекта включает в себя обязательства клиентского модуля (предусловия), обязательства, присутствующие после выполнения метода (постусловия), обязательства по выполнению конкретных свойств — инвариантов, которые должны выполняться при получении поставщиком сообщения, а также при выходе из метода [2].

Любая открытая функция класса — это не просто абстрактный фрагмент кода, предназначенный для выполнения неопределенной задачи. Каждая функция обладает строгими семантическими свойствами, которые отражают, что делает эта функция, независимо от того, как она это делает. Для четкого выражения задачи, выполняемой конкретной функцией, с ней связывают два утверждения — предусловие и постусловие. Предусловие определяет условия, которые должны выполняться всякий раз перед вызовом функции, а постусловие определяет свойства, гарантируемые после завершения ее выполнения. Предусловие и постусловие определяют контракт функции со всеми ее клиентами [2].

Нарушением контракта является отклонение конкретной реализации от заданной спецификации. При включенном мониторинге нарушение утверждения в период выполнения приводит к генерации исключения.

Не только выбор уровня мониторинга является обязанностью разработчика, но и реализация способа контроля мониторинга. Мною были проведены исследования на реальных проектах для иностранного заказчика с целью определения оптимальной реализации мониторинга. Был разработан модульный контроль мониторинга, управление которым велось при помощи опционального файла, хранящегося на системе пользователя; при помощи опционального файла, хранящегося на удаленном сервере, доступ на чтение к которому был у всех систем пользователя, а доступ на запись был лишь у привилегированного пользователя; при помощи настроек ключа в реестре.

По результатам проведенного исследования было установлено, что наиболее неудачным вариантом мониторинга является вариант с возможностью влияния пользователя на настройки опционального файла. При удалении данного файла система тратила дополнительные ресурсы на дополнительное чтение, на определения причин неудачного чтения и по факту использовала настройки по умолчанию, заложенные в самой системе. Более удачными вариантами управлением мониторингом следует признать управление уровнем мониторинга через опциональный файл, находящийся на удаленном сервере и через ключ в реестре. Однако следует упомянуть о сложностях, вызванных данными подходами. При управлении из реестра, следует поставить пользователю программу для изменения данного ключа или как минимум скрипт. Поставляемое приложение должно менять значение ключа в реестре на необходимое без информирования пользователя об имени ключа.

При управлении уровнем мониторинга при помощи удаленной системы следует информировать заказчика, что данная система должна быть доступна для всех систем пользователя, использующих продукт с функцией мониторинга.

При вышеуказанном методе управления мониторингом следует учесть, что опциональный файл должен быть защищен элементами синхронизации на случай наличия нескольких привиле-

гированных пользователей для избегания непредумышленного повреждения информации, вложенной параллельно. Данная проблемная ситуация может быть разрешена при помощи того же удаленного сервера и каталога с файлами, размещенного на данном сервере. Каждый файл из каталога должен иметь имя системы и содержать информацию по уровню мониторинга. Таким образом системные администраторы будут иметь возможность закрыть доступ на запись и исполнение для других пользователей, так как не будет необходимости разделять ресурсы между несколькими системными администраторами.

Выбор уровня мониторинга зависит от разработчика. Предлагается использовать максимальный уровень мониторинга как при отладке ПС, так и в модулях, критичных к высокой надежности выполнения операций. Для модулей, в которых критично время выполнения операций, предлагается рассчитывать уровень мониторинга в зависимости от производительности приложения.

В случае значительного падения показателей производительности ПС, предлагается не отключать мониторинг полностью и обязательно проверять предусловия. Опыт показывает, что это значительно менее критично к ресурсам, нежели проверка постусловий и обеспечивает достаточно высокий уровень надежности по сравнению с полным отсутствием мониторинга.

Так же следует учесть периодичность использования модуля с функцией мониторинга. Установлено, что для модулей, запущенных в фоновом режиме и выполняющим некоторую работу по ожиданию некоторого события нет необходимости устанавливать мониторинг в какой либо уровень за исключением инициализации компонентов.

Для модулей, выполняющих критическую работу по наступлению события, максимальный уровень мониторинга является необходимым в 20% функций. Остальная часть функционала должна быть защищена мониторингом предусловий. Опыт показывает, что большее покрытие максимальным мониторингом лишь ухудшает показатели производительности.

Следует упомянуть, что данные показатели могут различаться для различных программных средств и их модулей. Однако для графиче-

ческих модулей, модулей фильтрации сообщений, модулей ожидания события, модулей ведения статистического учета некоторых данных, не влияющих на поведение ПС, а необходимых для контроля его работы, данные значения являются полученными опытным путем. Вышеуказанные данные были апробированы на программном средстве для системы резервного копирования виртуальных машин, на плагине для переноса данных с операционной системы Windows на операционную систему Mac в системе иерархического хранения, на программном средстве иерархического копирования при работе с файловой системой ReFS.

На данный момент разработанные программные средства находятся у пользователей и ведется учет критических проблем, если таковые возникнут, не выявленных посредством установки описанного выше мониторинга. Так же пользователи обеспечивают данными по падению либо росту производительности в зависимости от выставленных значений по уровню мониторинга. Данные сведения будут учтены для дальнейшей работы по поиску оптимального подхода для выставления значений уровня мониторинга.

Как показывает практика проектирования ПС, контрактное программирование повышает уровень повторного использования кода и позволяет однозначно определить причину проблемы в повторно используемом коде.

Для обеспечения целостности данных рекомендуется выполнять проверку контракта методами, реализованными в классе-источнике. Опыт показывает, что стратегия, при которой каждый модуль ПС несет ответственность только за свои действия, приносит результаты при эксплуатации, сопровождении и доработке модифицированных ПС.

Как показывает практика проектирования ПС, контрактное программирование повышает уровень повторного использования кода и позволяет однозначно определить причину проблемы в повторно используемом коде.

1. Мейер, Б. Объектно-ориентированное конструирование программных систем / Б. Мейер // М.: Русская редакция. – 2005. – С. 125–127.
2. Meyer, B. Touch of Class. Learning to Program Well with Objects and Contracts. /B. Meuer // London. – 2009. С. 87-92.