МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**
Инженерная школа информационных технологий и робототехники

МОЛОДЕЖЬ И СОВРЕМЕННЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Сборник трудов
XVIII Международной научно-практической конференции
студентов, аспирантов и молодых ученых

22–26 марта 2021 г.

Томск 2021

ТЕСТИРОВАНИЕ DART-РЕАЛИЗАЦИИ КЛИЕНТСКОЙ ЧАСТИ ПРОТОКОЛА POP3

С.А. Прокопенко, к.т.н., доц.,
Н.В. Шабалдина, к.т.н., доц.
А.П. Сотников, аспирант,
Томский государственный университет
E-mail: sotnikhtc@gmail.com

Введение

Жизненный цикл любого программного продукта, в том числе мобильного приложения, является многоэтапным, начиная от формулировки формальных требований к продукту и заканчивая непосредственно эксплуатацией. Неотъемлемой частью данного цикла является тестирование разрабатываемого продукта.

Dart – молодой язык программирования, используемый для разработки мобильных, десктопных, серверных и веб приложений. В данной работе мы рассматриваем Dart-реализацию почтового клиента (клиентскую часть протокола) POP3 [1].

Тесты можно строить по-разному, однако, для того чтобы гарантировать полноту обнаружения определенного класса ошибок, необходимо иметь адекватную математическую модель формальных требований к работе приложения. Расширенный автомат является такой моделью [2], поскольку в данной модели учитываются параметры у входных воздействий и выходных реакций, а также имеются внутренние переменные. Кроме того, переходы между состояниями зависят от истинности предикатов, которые в свою очередь определяются входными параметрами и внутренними переменными. В данной работе мы рассматриваем тест, построенный на основе модели расширенного автомата, извлеченной из спецификации протокола POP3 [3], для обнаружения выходных неисправностей.

Описание формальной модели и теста

На рисунке 1 представлен расширенный автомат, который был построен нами на основе RFC-спецификации протокола POP3. Данный автомат имеет три состояния. Состояние 1 является начальным состоянием, в этом состоянии осуществляется установление соединения клиента с сервером, при успешном установлении соединения автомат переходит в состояние 2. В состоянии 2 происходит авторизация пользователя, при успешной авторизации автомат переходит в состояние 3. В состоянии 3 можно выполнять различные операции с почтовым ящиком авторизованного пользователя. Кроме того, в состояниях 2 и 3 можно закрыть соединение, при этом автомат перейдет в начальное состояние 1.

Каждый переход из состояния в состояние помечен парами «входное воздействие (входные параметры) / выходная реакция (выходные параметры)». У некоторых входных воздействий или выходных реакций параметры могут отсутствовать.

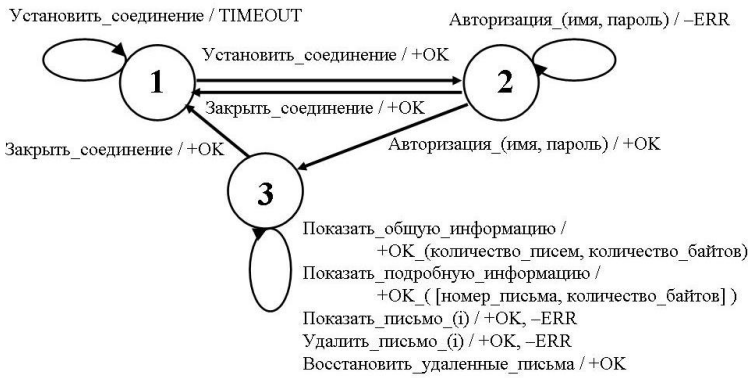

Рис. 1. Расширенный автомат, описывающий поведение POP3-клиента.

Под тестом понимается конечное множество конечных входных последовательностей. Иногда для удобства к тестовым последовательностям приписываются эталонные выходные реакции. По реакциям на тестовые последовательности можно распознать неисправную реализацию. Для обнаружения выходных неисправностей требуется достичь каждое состояние автомата и подать из него все входные воздействия. Для автомата на рис. 1 тест состоит из 11 последовательностей.

Для проведения эксперимента была написана так называемая «заглушка», т.е. простая реализация POP3-сервера. Данная заглушка отвечает «+OK» на авторизацию только одного пользователя с параметрами имя_верное, пароль_верный. Кроме того, мы считаем, что в почтовом ящике данного пользователя имеются пять писем, каждое из которых по 24 байта, суммарно 120 байтов. Соответственно, ответы сервера на входные воздействия «Показать_письмо_(*i*)» и «Удалить_письмо_(*i*)» для писем с $1 \le i \le 5$ будут «+OK» (если письмо с номером *i* еще не помечено как удаленное). В ответ на входные воздействия «Показать_общую_информацию» и «Показать_подробную_информацию» сервер присылает ответ «+OK» и соответственно общую или подробную информацию согласно спецификации. На воздействие «Восстановить_удаленные_письма» сервер всегда отвечает «+OK», на воздействие «Закрыть_соединение» отвечает «+OK» и закрывает соединение.

Результаты эксперимента

Построенный тест из 11 входных последовательностей был подан на Dart-реализацию POP3-клиента, которая в свою очередь взаимодействовала с «заглушкой». На тестовую последовательность «Установить_соединение Авторизация_(имя_верное, пароль_верный) Удалить_письмо_(*i*≤Number_of_Messages И письмо *i* не помечено как удаленное) / +OK +OK +OK» реализация произвела неверную выходную реакцию. Опишем найденную ошибку подробнее.

Тестом обнаружена ошибка в реализации команды DELE (рис. 2; вместо команды DELE указана команда LIST). Отправке команды DELE соответствует переход в автомате на рис. 1 из состояния 3 в состояние 3 под действием входного воздействия «Удалить_письмо_(*i*)». Данная команда имеет параметр – номер письма, которое пользователь хочет удалить. Ответом на эту команду согласно спецификации является «+OK» или «–ERR». В случае если сообщение с таким номером есть в почтовом ящике и оно еще не помечено как удаленное (т.е. команда DELE с данным параметром применена первый раз в эту сессию), должен быть ответ «+OK». Ответ «–ERR» должен быть в случае, если письмо уже помечено как удаленное либо номер письма больше чем число писем в ящике.

Однако Dart-реализация на входную последовательность «Установить_соединение Авторизация_(имя, пароль) Удалить_письмо_(*i*)» выдает реакцию «+OK +OK +OK_[номер_письма, количество_байтов]», т.е. реакция на «Удалить_письмо_(*i*)» такая же, как на «Показать_подробную_информацию» (на команду LIST, которая, как мы видим по рис. 2, указана в реализации вместо команды DELE).

```
3   class PopDeleteCommand extends PopCommand<void> {
4     PopDeleteCommand(int messageId) : super('LIST $messageId');
5   }
```

Рис. 2. Фрагмент программного кода, содержащий обнаруженную ошибку.

Заключение

В данной работе мы построили тест обходом графа переходов на основе модели расширенного автомата для POP3. Последовательности теста преобразованы в соответствующие вызовы методов тестируемой Dart-реализации POP3-клиента. В результате тестирования обнаружена ошибка, связанная с некорректной реализацией команды DELE. Данная ошибка существенно влияет на функционирование реализации, поскольку письма по запросу клиента вообще не удаляются. Найденная ошибка реализации была исправлена, создан pull request в основной репозиторий [4].

Список использованных источников

1. Dart-реализация протокола POP3 enough_mail. [Электронный ресурс]. – URL: https://github.com/Enough-Software/enough_mail (дата обращения 04.03.2021).
2. Petrenko A., Boroday S., Groz R. Confirming configurations in EFSM testing // IEEE Trans. Software Eng. 2004. № 30(1). P. 29– 42.
3. RFC 1939 – Post Office Protocol – Version 3. [Электронный ресурс]. – URL: https://tools.ietf.org/html/rfc1939#page-8 (дата обращения 04.03.2021).
4. Pull request исправления реализации POP3 enough_mail [Электронный ресурс]. – URL: https://github.com/Enough-Software/enough_mail/pull/128 (дата обращения 05.03.2021).