УДК 004.415.53

Верификация промышленных алгоритмов управления методом Model checking в сочетании с концепцией виртуальных объектов управления

Лях Т.В. (Институт автоматики и электрометрии СО РАН, Новосибирский государственный университет),
Зюбин В.Е. (Институт автоматики и электрометрии СО РАН, Новосибирский государственный университет)

На сегодняшний день текущая практика промышленной автоматизации такова, что тестирование управляющих алгоритмов в подавляющем большинстве случаев начинается только при запуске ПО на реальном объекте. В результате проверка алгоритма откладывается до этапа пуско-наладочных работ на объекте автоматизации. В статье предложен подход к тестированию алгоритмов управления на основе концепции виртуальных объектов управления. Для гарантии, что алгоритм управления удовлетворяет полностью накладываемым на него требованиями, используется метод верификации Model checking.

Ключевые слова: *Алгоритмы управления, промышленная автоматизация, процесс-ориентированное программирование, язык Reflex, верификация, Model checking..*

1. Введение

На сегодняшний день текущая практика промышленной автоматизации предполагает, что автоматизированные системы управления создаются исключительно на базе цифровой техники в виде программно-аппаратных комплексов. При этом на современном этапе наблюдается четкая тенденция к усложнению программной составляющей таких систем, повышению ее функциональности и общей трудоемкости ее реализации. Рост значимости программного обеспечения в области промышленной автоматизации, высокая стоимость логических ошибок в программах давно уже находятся в противоречии с текущей практикой разработки управляющих программ, которая ведется в рамках водопадной модели. Тестирование управляющих алгоритмов в подавляющем большинстве случаев начинается только при запуске ПО на реальном объекте. В результате проверка алгоритма

откладывается до этапа пуско-наладочных работ на объекте автоматизации. Такая практика чревата высокими рисками, нештатными ситуациями или даже авариями на объекте.

Для решения проблемы тестирования управляющих алгоритмов в Институте автоматики и электрометрии СО РАН была предложена концепция виртуальных объектов управления (ВОУ) – программных имитаторов автоматизируемого технического процесса, со свойствами, схожими со свойствами моделируемого объекта [1]. Код ВОУ (Рис. 1) исполняется независимо от алгоритма управления (АУ), создаваемого разработчиком. Унифицированный обмен данными между ВОУ и алгоритмом управления обеспечивает сохранение связей при изменении алгоритма. Такой подход позволил использовать итерационную модель разработки и отлаживать код алгоритма управления до этапа пуска-наладки.

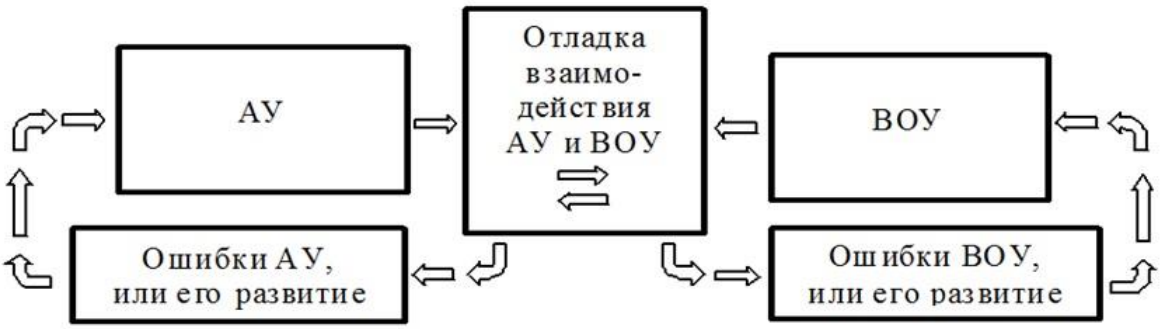

Рис. 1. Итерационная модель разработки алгоритма управления (АУ) с использованием виртуального объекта управления (ВОУ)

Было создано ПО на базе среды LabVIEW, которое позволило запускать одновременно и тестировать ВОУ и АУ, имитируя обмен данными и внешние события [2], и с помощью него был отлажен алгоритм управления Большим солнечным вакуумным телескопом.

Однако в процессе работы был выявлен ряд недостатков такого подхода. Эти недостатки связаны с тем, что тестирование не обеспечивает полноту покрытия тестами всей функциональности алгоритма. Также при таком подходе множество операций приходилось выполнять вручную оператору.

Это растягивало процесс тестирование и увеличивало возможность пропущенных ошибок и неисследованных поведений алгоритма управления. Поэтому для автоматизации проверки алгоритма управления было предложено воспользоваться подходом Model chescking. Этот подход хорошо себя зарекомендовал при верификаии реаигирующих систем, т.е. систем, взаимодействующих с окружением. На данный момент подход Model checking активно развивается.

В статье рассматриваются особенности разработки управляющих алгоритмов промышленного уровня, исследуются проблемы тестирования алгоритмов управления в приборостроении, описывается подход к тестированию алгоритмов управления на основе концепции виртуальных объектов управления. Предложен способ автоматизации тестирования АУ с использованием подхода Model checking [3].

2. Специфика разработки алгоритмов управления и преимущества языка Reflex

Задачи автоматизации имеют ряд характерных особенностей, и потому на ПО и языки программирования, которые используются в разработке АУ, накладывается ряд особых требований. Поскольку система управления воздействует на объект управления через органы управления и реагирует события на объекте так, как это определено в алгоритме управления, от управляющего алгоритма требуется цикличность: он считывает входные сигналы, обрабатывает и формирует выходные сигналы. От алгоритма также требуется адекватность реакции по времени событиям на объекте, т.е. синхронизация исполнения алгоритма с физическими процессами во внешней среде. Поскольку на объекте управления зачастую множество процессов возникают и протекают одновременно, требуется, чтобы средства разработки предоставляли возможность обеспечить логический параллелизм алгоритма.

Благодаря этим особенностям стратегии создания управляющих алгоритмов в промышленной автоматизации и используемые языки программирования отличаются от практики, применяемой при создании ПО, не взаимодействующего с реальными объектами.

Для создания промышленных алгоритмов управления используется множество подходов: языки стандарта МЭК 61131-3, языки общего назначения (такие, как C, C++ или Delphi), языково-ориентированное программирование с использованием предметно-ориентированных языков и прочее [4]. Каждый из таких подходов имеет определенные преимущества и недостатки, и, не вдаваясь глубоко в детали, следует упомянуть, что в конечном счете выбор делается в пользу того решения, которое наилучшим образом соответствует особенностям автоматизируемого объекта. Однако в последнее время в связи с недостатками стандарта МЭК [5] наблюдаемая тенденция такова, что при разработке промышленных алгоритмов управления все чаще отказываются от языков МЭК в пользу либо языков общего назначения, либо новых, специализированных для узкого применения, формализмов.

Процесс-ориентированный язык Reflex был создан для описания алгоритмов управления при решении задач промышленной автоматизации [6]. Язык Reflex отличается рядом достоинств:

1) Адекватность задачам промышленной автоматизации;
2) Легкость в изучении;
3) Язык Reflex – высокоуровневый язык программирования, разработчику не требуется работать в терминах низкоуровневых операций с оборудованием;
4) Алгоритмы, созданные на языке Reflex, не зависят от среды исполнения;
5) Язык Reflex допускает вызовы функций, написанных на других языках программирования.

3. Концепция виртуальных объектов управления на базе LabVIEW с использованием языка Reflex

Концепция ВОУ для итерационной разработки АУ была реализована с использованием механизма DLL, пакета LabVIEW [7] и транслятора языка Reflex. Интерфейс был создан средствами пакета прикладных программ технических вычислений LabVIEW, который широко используется для имитационного моделирования. Алгоритм управления и описание ВОУ создается на языке Reflex.

При итерационной разработке алгоритма управления на основе концепции ВОУ работа происходит по схеме, изображенной на рис. 2:

1) На языке Рефлекс создается описание логически обособленной части АУ (например, часть алгоритма, отвечающая за определенную функциональность);
2) На языке Рефлекс создается описание элемента ВОУ, соответствующего функционированию этого АУ;
3) ВОУ и АУ транслируются в DLL, которые встраиваются в отладочное ПО. Дополнительно транслятор создает конфигурационные файлы, которые автоматически интегрируются в отладочное ПО;
4) Оператор за отладочным интерфейсом проводит тестирование блока АУ: запускает одновременно и тестирует пошагово АУ и ВОУ, имитирует передачу данных от оператора интерфейса управления для АУ, имитирует передачу данных с датчиков объекта управления. Это позволяет имитировать нештатные ситуации на объекте управления: аварии, поломки оборудования и отсутствие связи с оборудованием – и оценивать реакцию АУ на них;

5) Если было выявлено несоответствие поведения АУ требованиям спецификации, или же найдены ошибки в описании АУ или ВОУ на языке Reflex, вносятся изменения в код АУ или ВОУ, и трансляция и тестирование происходят заново;
6) Если тестирование прошло успешно, не было выявлено ошибок в описаниях ВОУ и АУ, и было установлено, что АУ удовлетворяет накладываемым на него признакам, описывается следующий логический модуль АУ и создается соответствующий блок ВОУ.

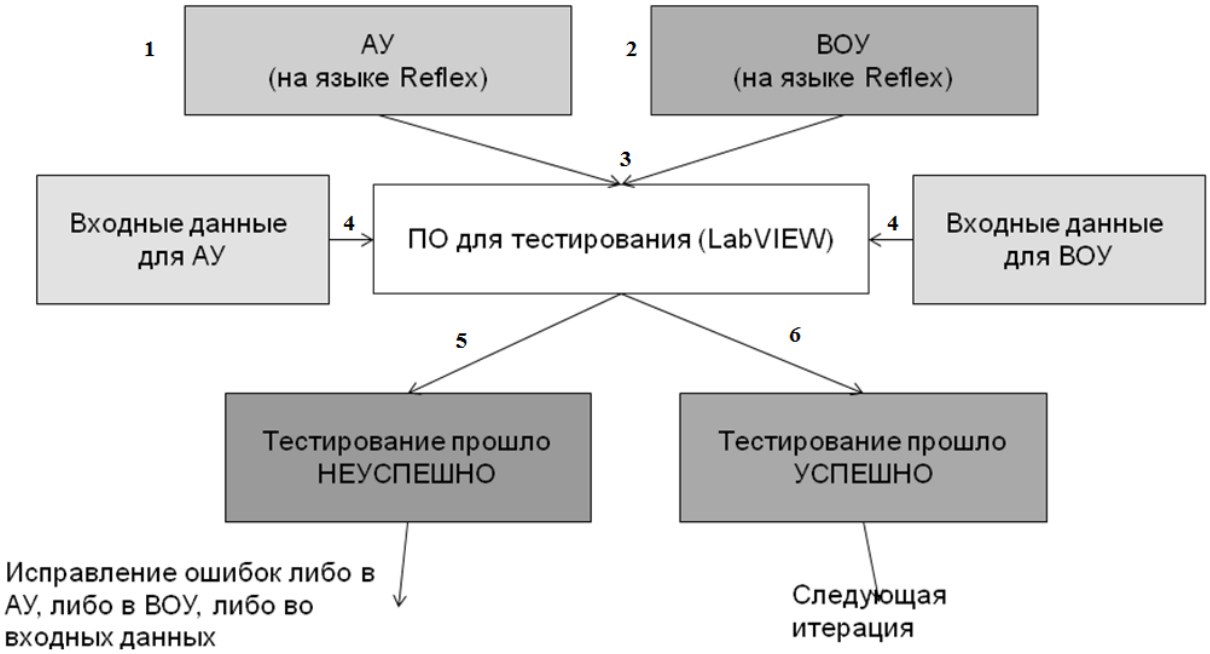

Рис. 2. Итерационная схема разработки алгоритмов управления на основе концепции ВОУ

Рассмотренная выше схема разработки была опробована при разработке алгоритма управления системы вакуумирования Большим солнечным вакуумный телескопом (БСВТ, г Иркутск, поселок Листвянка).

3.1. Недостатки разработанного метода проверки безопасности АУ

Однако при работе над системой вакуумирования БСВТ было обращено внимание на ряд неудобств данного подхода к разработке АУ:

- Большое количество работы «вручную». Оператору приходится вручную запускать ВОУ и АУ, тестировать их пошагово, имитировать передачу данных от оператора интерфейса управления для АУ и передачу данных с датчиков объекта управления;
- Полнота покрытия алгоритма тестами неизвестна. Тестирование не способно дать точный ответ, насколько полно тесты покрывают функциональность АУ, и насколько

точно выполняются требования, накладываемые на АУ. К тому же, тестирование чаще всего выявляет частые ошибки, в то время как редкие, но критические ошибки, могут ускользнуть от внимания. С помощью тестирование методом «черного ящика» невозможно доказать отсутствие ошибок в программе.

Для точного доказательства, что АУ удовлетворяет накладываемым на него требованиям, необходим строгий и непротиворечивый математический аппарат.

4. Использование метода Model checking для тестирования АУ

Для проверки корректности АУ в идеале необходимо перебрать все возможные пути его вычисления, однако для систем, взаимодействующих с окружающей средой, эта задача невыполнима, так как таких путей может оказаться бесконечное множество. В настоящий момент для того, чтобы показать, что алгоритм соответствует накладываемым на него требованиям, используются методы верификации.

Среди существующих методов верификации для автоматизации тестирования АУ был выбран метод Model Checking. При верификации алгоритма подходом Model Checking проверяется, что некоторое свойство поведения алгоритма управления, выраженное формулой темпоральной логики, выполняется для модели системы с конечным числом состояний [3].

Так как язык Reflex базируется на модели конечного гипераавтомата, предоставляет логический параллелизм, средства взаимодействия между процессами и дает возможность контролировать время нахождения процесса в текущем состоянии, это делает Reflex удобным для описания верифицируемой модели системы, которая при верификации методом Model Checking представляется в виде модифицированного конечного автомата.

Итерационная схема разработки АУ была изменена (рис.3)

1) На языке Reflex создается описание части АУ, а также на формальном языке темпоральной логики описываются требования к АУ. Требования, описываемые на этом не этапе, не учитывают взаимодействие АУ с ВОУ – это те требования, которые рассматривают внутреннюю логику функционирования АУ, не зависящую от обмена данными с внешней средой.
2) Автоматическая верификация АУ (выполняется верифицирующим ПО). Если верфикатор выносит решение, что требования не выполняются, происходит поиск ошибок (или в описании АУ, или в формализации требований), после чего вносятся необходимые исправления и верификация повторяется

3) На языке Reflex создается описание части ВОУ, а также на формальном языке темпоральной логики описываются требования к ВОУ. Требования к ВОУ, описываемые на этом этапе, проверяют лишь то, что поведение ВОУ соответствует поведению реального объекта управления.
4) Автоматическая верификация ВОУ (выполняется верифицирующим ПО). Если верификатор выносит решение, что требования не выполняются, происходит поиск ошибок (или в описании АУ, или в формализации требований), после чего вносятся необходимые исправления и верификация повторяется
5) Формулировка требований на языке темпоральной логики, истинность которых зависит от взаимодействия АУ и ВОУ
6) Автоматическое построение общей модели АУ и ВОУ (выполняется верифицирующим ПО)
7) Автоматическая верификация общей модели АУ и ВОУ(выполняется верифицирующим ПО)
8) Если по результатам верификации было вынесено решение, что накладываемые требования не выполняются, происходит поиск ошибок (или в описании ВОУ, или в описании ВОУ, или в описании требований), после чего повторяется верификация общей модели. Если верификация прошла успешно, то АУ удовлетворяет накладываем требованиям. Происходит возврат на п. 1. и описание следующего логического блока алгоритма.

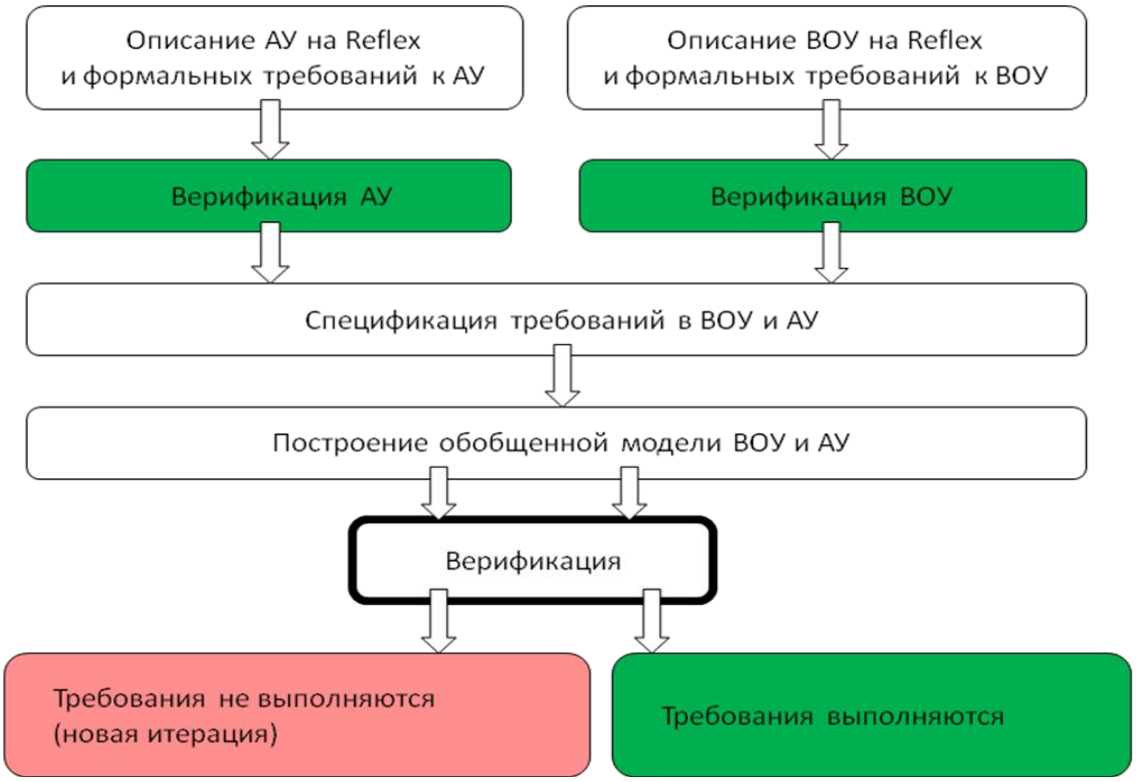

Рис. 3. Схема тестирования АУ и ВОУметодом Model checking

4.1. Схема верификации кода, созданного на языке Reflex, методом Model Checking с помощью верификатора SPIN

Для реализации описанной выше схемы требовалось создание верификатора кода на языке Reflex. Однако создание верификатора для метода Model Checking с нуля – задача крайне объемная, и требующая серьезных затрат. Поэтому было решено воспользоваться уже существующим верификатором Spin.

Верификация кода на языке Reflex проходит по следующей схеме:

1) Создается описание алгоритмического блока на языке Reflex
2) Транслятор автоматически преобразует код на языке Reflex в код на языке Promela – стандартном языке верификатора SPIN
3) Описание формальных требований методами верификатора SPIN
4) Трансляция требований в язык Promela с помощью верификатора SPIN
5) Автоматическая верификация верификатором SPIN
6) Требования выполняются – задача выполнена. Требования не выполняются – поиск ошибок или в описании алгоритма на языке Reflex, или в описании требований.

Таким образом, в предложенной схеме автоматически проходит верификация алгоритма, и для реализации необходимо было только реализовать транслятор из языка Reflex в язык Promela.

Такой подход позволяет избежать одного из самых значимых недостатков подхода Model checking: необходимости дальнейшего тестирования результирующего кода. Это связано с тем, что при верификации методом Model checking верифицируется не результирующий код, а построенная модель алгоритма. Однако верифицируемая модель, созданная на языке Reflex, транслируется в исполняемый код алгоритма на языке Си, который уже не требует дополнительного тестирования.

5. Заключение

Таким образом, в работе был предложен вариант реализации концепции итерационной разработки управляющих алгоритмов на основе виртуального объекта управления (ВОУ). Была разработана схема автоматической верификации алгоритма управления с помощью метода Model Checking. Так как описанный на языке Reflex алгоритм транслируется в исполняемый код алгоритма, это значит, что использование предложенной схемы позволит уйти от самого значительного недостатка верификации методом Model Checking: необходимости повторного тестирования, так как в классическом подходе верифицируется не сама конечная система, а только ее модель. Используя подход Model Checking можно проверять как корректное функционирование АУ, так и поведение ВОУ.

Концепция итерационной разработки управляющих алгоритмов на основе ВОУ эффективна для задач снижения рисков при вводе систем управления в эксплуатацию. Использование метода в реальных проектах по автоматизации позволяет:

1. тестировать создаваемые алгоритмы, начиная с самых ранних стадий разработки, внедрить итерационную модель разработки для случая промышленной автоматизации;
2. обеспечить контроль процесса создания управляющих алгоритмов и снизить психологическую нагрузку на коллектив разработчиков;
3. сократить время выполнения проекта и имеющиеся риски этапа пуско-наладки;
4. гибко расширять круг лиц, участвующих в процессе разработки, в частности, чтобы своевременно выявлять и устранять ошибки в техническом задании.

Разработанный подход был использован для отладки алгоритма управления вакуумной системой БСВТ.

Список литературы

1. Зюбин В. Е. Итерационная разработка управляющих алгоритмов на основе имитационного моделирования объекта управления // Автоматизация в промышленности. 2010. № 11. С. 43-48
2. Лях Т. В., Зюбин В. Е. Применение концепции виртуальных объектов управления для решения задач промышленной автоматизации // Материалы Девятой международной Ершовской конференции PSI-2014 (г. Санкт-Петербург, Россия, июнь 2014г). С. 57-64.
3. C. Baier, J.P. Katoen, Principles of Model Checking. The MIT Press. Massachusetts Institute of Technology, 2007.
4. Горячкин А. А., Зюбин В. Е., Лубков А. А. Разработка графического формализма для описания алгоритмов в процесс-ориентированном стиле // Вестн. Новосиб. гос. ун-та. Серия: Информационные технологии. 2013. Т. 11, вып. 2. С. 44–54.
5. Зюбин В. Е. К пятилетию стандарта IEC 1131-3. Итоги и прогнозы // Приборы и системы. Управление, контроль, диагностика. 1999. № 1.
6. В. Е. Зюбин. «Си с процессами» - язык программирования логических контроллеров // Мехатроника, автоматизация, управление. 2006. № 12 С. 31-35
7. Дж. Тревис. LabVIEW для всех. М.: ДМК Пресс, 2011. 912 с.