

# Этапы технологии проектирования встроенных бортовых систем с поддержкой верификации

И.В. Шошмина<sup>1</sup>

<sup>1</sup>Санкт-Петербургский государственный политехнический университет,

ishoshmina@dcn.ftk.spbstu.ru

**Аннотация** — В работе рассматривается проблема интеграции метода формальной верификации – проверки на модели – и процесса проектирования сложных распределенных программных систем для повышения качества ПО. Предлагается при проектировании опираться на построение платформенно-независимой модели программы. Подход позволяет структурировать процесс верификации: выделить и независимо верифицировать ядро системы, состоящее из независимых подсистем и отвечающее за логическое управление всей системой как единым целым. Предлагаемая методика опробована на реальной системе управления энергоснабжением судна.

**Ключевые слова** — проверка модели, проектирование на основе модели, BoUML, SPIN, BuDDy, линейная темпоральная логика LTL.

## I. Введение

В связи с расширением сфер применения и возрастающей сложностью программного обеспечения (ПО) методы повышения качества ПО становятся важнейшей областью информатики.

Целью исследования является разработка этапов методики проектирования распределенных бортовых программных систем с поддержкой формальной верификации.

В работе используется метод формальной верификации, называемый проверка модели. Этот метод позволяет автоматически проверить, что заданная формула (формула темпоральной логики) истинна на конкретной структуре, представляющей собой формальную модель проектируемой системы.

Использование в промышленной разработке ПО метода проверки модели очень перспективно, т.к. при этом проводится исчерпывающий анализ всех возможных поведений системы, и поэтому в верифицируе-

мых системах можно гарантировать выполнение проверенных свойств.

## II. Проектирование на основе моделей

Включение методов формальной верификации в процесс проектирования программных средств сталкивается с необходимостью построить формальную модель системы и описать требования к ее поведению.

Сложная бортовая система состоит из большого числа подсистем. В методике рассматриваются бортовые системы, имеющие распределенный характер, т.е. содержащие такие подсистемы, которые работают независимо по своим алгоритмам, но взаимодействуют друг с другом для выполнения общих целей и задач.

Не все множество подсистем распределенной бортовой системы влияет на верифицируемые свойства друг друга, некоторые подсистемы можно верифицировать независимо. Даже в тех подсистемах, которые участвуют во взаимодействиях, часть внутренних алгоритмов не имеет непосредственного отношения к процессу взаимодействия. Поэтому предлагается структурировать процесс верификации: выделить и независимо верифицировать ядро системы, состоящее из независимых подсистем и отвечающее за логическое управление всей системой как единым целым.

Ядро управления бортовых систем относится к классу реагирующих систем – это системы, дающие отклик на внешнее событие в зависимости от своего состояния.

Для описания модели ядра подсистемы используется подход, предложенный консорциумом OMG: проектирование на основе моделей (Model-Driven Engineering – MDE). В этом подходе описание проектируемой системы производится независимо от реализации и может быть формально транслировано на большое число платформ (в т.ч., Java, XML, SOAP). В соответствии с MDE формальная модель ядра управления строится на платформенно-независимом языке моделирования UML.

Поведение каждой независимой подсистемы ядра управления моделируется конечным автоматом, выполняемым в отдельном процессе. Конечные автоматы компактно описываются визуальным формализмом, картами состояний. Последние представлены в языке UML диаграммами конечных автоматов (UML state machine diagrams). Статические связи между объектами описываются диаграммами классов (UML class diagrams).

Верификация методом проверки модели проводится для замкнутых систем: кроме ядра управления необходимо моделировать поведение среды, порождающей события, например, поведение оператора за пультом. Поведение среды можно ограничить, задав сценарий, т.е. конечную фиксированную последовательность событий (с помощью UML диаграмм взаимодействия). В этом случае модель среды соответствует поведению среды, согласующемуся с каким-либо вариантом ее использования. Однако такой подход не предусматривает появления случайного, не рассматриваемого в стандартных сценариях события, например, внештатного нажатия кнопки включения противопожарной системы на подводной лодке, а потому не позволяет выявить тех поведений ядра управления, которые никогда не должны происходить. Описание поведения среды недетерминированной картой состояний покрывает все возможные сценарии поведения.

Графическая модель системы строится в среде VoUML. Программное обеспечение

VoUML является свободно распространяемым. VoUML поддерживает представление UML модели в формате обмена метамоделями XMI.

UML модель системы в формате XMI 2.1 транслируется в язык Promela пакета SPIN [1] с помощью транслятора, разработанного в рамках представляемого проекта. Свободно распространяемый пакет SPIN, созданный в лаборатории Bell Labs, позволяет строить модели параллельных программ и широкого класса дискретных систем с конечным числом состояний, выразить требуемые свойства их поведения на языке линейной темпоральной логики LTL, автоматически проверить выполнение темпоральных свойств на моделях.

Для перевода формул линейной темпоральной логики в автоматы Бюхи, используемые при верификации, разработан символьный алгоритм на основе бинарных решающих диаграмм (БРД) [2]. Использование БРД в реализации алгоритма обеспечивается библиотекой свободно распространяемого пакета BuDDY [3].

Синхронная композиция модели ядра системы и автомата Бюхи верифицируется SPIN. Если в результате верификации будет обнаружено нарушение требования к поведению системы, то строится контрпример, который демонстрирует последовательность шагов каждой подсистемы, приводящую к опровержению свойства. Контрпример, заданный, например, в виде диаграммы последовательностей – UML sequence diagram, является источником информации для понимания причины ошибки и последующей модификации модели системы. После модификации этапы верификации повторяются вновь. Если ошибок не обнаружено, то модель ядра управления корректна по отношению к данному свойству и может быть использована для дальнейшего проектирования.

### III. Верификация системы энергоснабжения судна

Разрабатываемая методика была апробирована на конкретной системе управления энергоснабжением судна (СУЭ). Система энергоснабжения состоит из двух электродвигателей, двух генераторных автоматов, двух электрогенераторов, двух пультов операторов, многочисленных датчиков и т.п. Управление каждым физическим устройством осуществляется независимой подсистемой в соответствии с показаниями датчиков, сообщениями других подсистем и текущим состоянием системы управления. В задачи СУЭ входит автоматическое подключение резервных мощностей в случае перегрузки, обнаружение и корректная обработка аварийных ситуаций, связанных с состоянием физических устройств и другие подобные функции. Алгоритмы системы были получены от независимых разработчиков в виде кода на C++.

При верификации построенной в BoUML модели системы был найден ряд некорректностей. Перечислим некоторые из них. Одной из часто встречающихся некорректностей распределенных систем является взаимная блокировка процессов. В построенной модели блокировки процессов возникают из-за переполнения каналов обмена. Выявление данной ошибки позволило обнаружить большое количество избыточных с логической и алгоритмической точки зрения взаимодействий между процессами, имевшихся в исходной реализации. Данную проблему удалось обойти, удалив лишние пересылки и снизив размер системы.

В технической спецификации СУЭ утверждается, что выход из режима защиты генераторного автомата (ГА) возможен только

при нажатии кнопки "Сброс защиты". Для проверки этого требования было сформулировано следующее свойство на языке LTL:  $Fq \Rightarrow (qUp)$ , где  $p$  — утверждение о том, что кнопка "Сброс защиты" нажата, а  $q$  — утверждение о том, что система в режиме защиты. Проверяемое свойство гарантирует, что никогда не будет так, что система вышла из режима защиты без нажатия кнопки сброса. При проверке этого свойства обнаружена трасса, на которой свойство не выполняется. Анализ показал, что некорректность объясняется отсутствием переходного режима; ошибка была исправлена его добавлением.

Найденные ошибки подтверждаются тестированием исходной реализации на C++. Исправление этих ошибок в реализации приводит иногда к значительной модификации кода.

### IV. Заключение

В работе представлены результаты разработки этапов методики проектирования встроенных бортовых систем с поддержкой верификации. При верификации реальной системы энергоснабжения судна выявлен и исправлен ряд тонких ошибок.

Использование свободного программного обеспечения – BoUML, SPIN, BuDDY – позволило сосредоточиться на решении научных задач.

### Литература

- [1] Holzmann G. Spin Model Checker. The Primer and Reference Manual.— Addison Wesley, 2003.— 608 с.
- [2] Карпов Ю.Г. Model checking. Верификация параллельных и распределенных программ – СПб.:БХВ-Петербург, 2010. – 560 с.
- [3] Andersen H.R., Henrik H. Boolean expression diagrams// In proceedings 12<sup>th</sup> Annual IEEE Symposium on Logic in Computer Science. – 1997. – P. 88-98.