УДК 004.415

П.Д. Дробинцев, И.В. Никифоров, В.П. Котляров

МЕТОДИКА ПРОЕКТИРОВАНИЯ ТЕСТОВ СЛОЖНЫХ ПРОГРАММНЫХ КОМПЛЕКСОВ НА ОСНОВЕ СТРУКТУРИРОВАННЫХ UCM МОДЕЛЕЙ

P.D. Drobintsev, I.V. Nikiforov, V.P. Kotliarov

FORMAL MODELS STRUCTURIZATION BASED TECHNIQUE OF COMPLEX SOFTWARE PROJECTS TESTING

Описана методика тестирования, основанная на применении структуризации к формальной модели разрабатываемого программного комплекса. В рамках методики описаны подходы и критерии структуризации, а также приведен пример автоматической генерации тестов на основе структурированной модели.

Предлагаемая методика позволяет существенно упростить процесс анализа формальной модели и обеспечивает возможность автоматической генерации тестов под контролем пользователя.

UCM. СТРУКТУРИЗАЦИЯ. ПОКРЫТИЕ. ТЕСТОВЫЙ НАБОР.

The paper describes a technique of software testing based on structurization of formal model. In the scope of the technique approaches and criterions of structurization are described. An example of automated tests generation is shown.

Suggested technique allows to resolve problems of formal model understanding and gives an opportunity of automated tests generation with usage of branch coverage criterion.

UCM. STRUCTURIZATION. COVERAGE. TEST SUI.

Растущая сложность разрабатываемых программных комплексов существенно осложняет обеспечение высокого уровня их качества. При этом коллективный подход к разработке программного продукта (ПП) как правило ограничивается компетентностью разработчиков и тестировщиков проектной команды, знанием исключительно запланированных ими конкретных модулей и интерфейсов внутримодульного взаимодействия и взаимодействия с окружением. В связи с этим при проектировании сложных систем развивается т. н. *инкрементальный подход*, позволяющий интегрировать проверку корректности программного обеспечения (ПО) по частям (по компонентам) с проверкой корректности системы в целом, что существенно снижает суммарную трудоемкость разработки.

Данная статья посвящена описанию применения инкрементального подхода к тестированию и верификации сложного программного продукта. В основе подхода лежит идея автоматизированного получения структурированной формальной модели разрабатываемой системы. Структурированная формальная модель наряду с упрощением понимания функциональности и поведенческих свойств системы, обеспечивает упрощение анализа, верификации и служит основой для автоматической генерации кодов и тестов разрабатываемого ПП.

Описание UCM нотации

Для описания сложных программных систем все чаще используются формальные языки проектирования, в частности, в данной работе использован язык UCM (Use Case Maps) [1].

Модель UCM проекта (рис. 1) представляет собой набор связанных и структурированных диаграмм, каждая из которых состоит из последовательности элементов нотации UCM. В совокупности набор диаграмм задает возможные поведения систе-

мы, описанные в требованиях. К основным элементам UCM нотации относятся: компоненты Team, в границах которых задается поведение объекта; элементы Responsibility (✕), отражающие точки выполнения каких-либо действий; StartPoint (●) и EndPoint (┫), задающие начальную и конечную точки поведенческого сценария; элементы OrFork (⅄) и OrJoin (⅂), описывающие альтернативы и недетерминизм; элементы AndFork (┻) и AndJoin (┳), описывающие параллельные сценарии; элемент FailurePoint (⍒), описывающий механизм генерации и обработки исключений; элемент Timer (◷), задающий временную задержку, в т. ч. и со сложным логическим поведением, структурный элемент Stub (◇), описывающий иерархию в поведении системы, что позволяет вести разработку системы покомпонентно от верхнего уровня абстракции до детального описания низкоуровневых диаграмм.

Следует отметить, что использование формальной модели открывает возможность различного рода преобразований, так, например, во входной язык верификатора [2] или во входной язык системы автоматической генерации тестов [6].

Структуризация формальной модели системы

Структуризация – это этап системного анализа, который состоит в том, что вся совокупность объектов и процессов, имеющих отношение к поставленной цели, сначала разделяется на собственно изучаемую систему и внешнюю среду, затем выделяются отдельные составные части – подсистемы и элементы изучаемой системы. С точки зрения UCM единицей структуризации может быть любой элемент или выделенная группа элементов поведения, лишь бы это позволяло упростить процесс анализа исходной модели. В рамках настоящей работы в роли единиц структуризации используется элемент Stab или явно выделенная совокупность элементов UCM диаграммы.

Определим UCM диаграмму как ориентированный граф с вершинами из множества элементов диаграммы и дугами, определяющими достижимость одного элемента диаграммы из других. То есть UCM диаграмма – это четверка $\mathrm{UCM} = (U, S, E, R)$, где U – конечное множество элементов; $S \subseteq U$ – множество стартовых точек; $E \subseteq U$ – множество конечных точек; $R \subseteq U \times U$ – отношение переходов, определяющее достижимость одного элемента диаграммы из других.

С точки зрения анализа поведения системы для процессов верификации и тестирования важным является понятие пути (π), которое определяется как $\pi = u_1 \to u_2 \to \ldots \to u_n$, где $u_i \in U$. При проведении тестирования также важно условие, что в тестовых сценариях каждый путь начинается в стартовой точке и заканчивается в конечной точке $u_1 \in S \wedge u_n \in E$.

Структурированная UCM диаграмма отличается от исходной только тем, что ис-

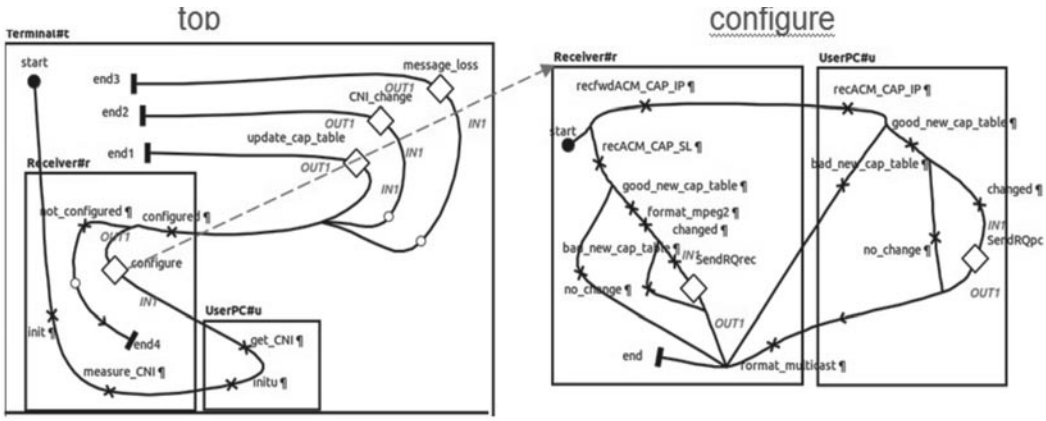

Рис. 1. UCM диаграмма для телекоммуникационного проекта Satellite

пользует элементы Stub для декомпозиции поведения на основании функциональных или структурных критериев. При этом элемент Stub является вложенной UCM диаграммой и определяется как шестерка $ST = (U_{st}, S_{st}, E_{st}, R_{st}, Sr, Er)$, в которой определение $U_{st}, S_{st}, E_{st}, R_{st}$ аналогично определению UCM диаграммы; $Sr \subseteq S_{st} \times U$ – отношение переходов, определяющее связь стартовых точек элемента Stub с элементами диаграммы верхнего уровня, и $Er \subseteq E_{st} \times U$ – отношение переходов, определяющее связь конечных точек элемента Stub с элементами диаграммы верхнего уровня.

Таким образом, процесс структуризации можно определить как преобразование: $\mathrm{UCM} \rightarrow \mathrm{UCM}'$, в котором $U' = U \cup ST_1 \cup ST_2 \cup ... \cup ST_k$. Очевидно, что существует множество способов реализации подобного преобразования, однако с точки зрения анализа сценариев поведения системы оно может проводиться на основании двух дополняющих критериев – функционального и структурного.

Функциональный критерий определяется создателем формальной спецификации и подразумевает выделение в элемент Stub тех элементов исходной UCM диаграммы, которые служат выполнению определенной функциональности, заданной в требованиях на ПО. Использование этого критерия требует понимания как исходной спецификации требований системы, так и того, какими поведенческими сценариями каждое из требований может быть проверено при тестировании системы. Ограничением подобного подхода является обязательное участие специалиста предметной области (представителя заказчика), участвующего в формализации и утверждающего предлагаемое преобразование $\mathrm{UCM} \rightarrow \mathrm{UCM}'$. Заметим, что декомпозиция формализованной спецификации практически не поддается автоматизации.

На рис. 1 представлен пример двух уровней структурированной UCM диаграммы, описывающей протокол взаимодействия между спутником и оператором, предоставляющим IP услуги на основе стандарта ETSI [3]. В данном примере в соответствии с требованиями поведение системы разбито на компоненты top и configure.

Структурный критерий в отличие от функционального использует структуру самой UCM диаграммы для построения элементов Stub и проведения декомпозиции. Характерным примером использования данного критерия является объединение в Stub элементов диаграммы, принадлежащих к различным путям, приводящим систему в одно и то же состояние. Следует отметить, что композиция поведенческих сценариев вложенных диаграмм разных уровней на основе структурного критерия может быть автоматизирована.

Вне зависимости от применяемого критерия структурирования любая UCM диаграмма может быть подвержена декомпозиции с целью упрощения дальнейшего анализа.

Автоматическая генерация тестового набора по структурной модели

Для автоматической генерации тестового набора по структурированной модели необходимо определить возможные критерии покрытия, применение которых будет гарантировать качество тестирования данной модели. В силу того факта, что UCM диаграмма содержит информацию о структуре разрабатываемого ПО, для обеспечения ее покрытия удобно использовать структурные критерии. Наиболее известными из них являются критерий покрытия по ветвям, определяющий необходимость покрытия каждой ветви поведения системы хотя бы одним из тестов, и критерий покрытия по путям, определяющий необходимость покрытия каждого пути поведения системы хотя бы одним тестом [4].

Рассмотрим критерий покрытия по ветвям. С точки зрения UCM диаграммы ветвь определяется как отрезок пути, удовлетворяющий следующим требованиям:

- каждая ветвь начинается в стартовой точке диаграммы или в точке альтернативного выбора OrFork, или в точке начала параллельного поведения AndFork;
- каждая ветвь заканчивается или в конечной точке диаграммы, или в точке окончания альтернативного поведения OrJoin,

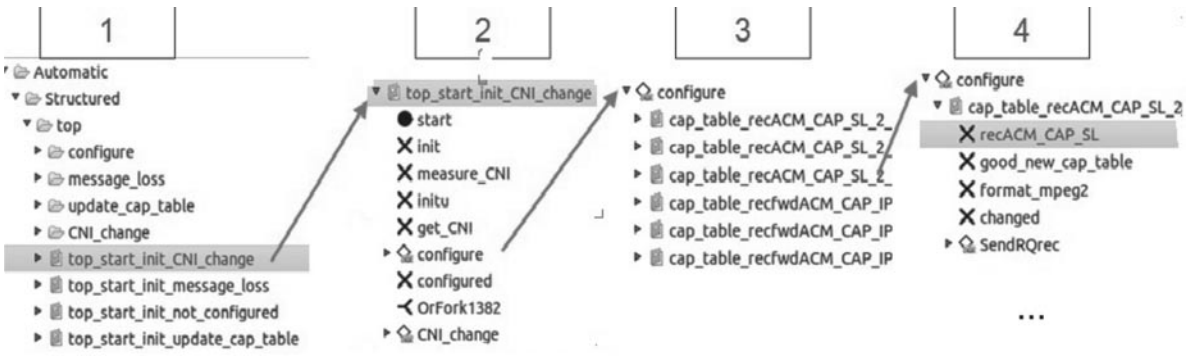

Рис. 2. Структурные гиды UCM проекта

или в точке синхронизации параллельных сценариев AndJoin.

С точки зрения структурированного представления UCM диаграммы для выполнения критерия покрытия по ветвям необходимо покрытие всех ветвей основной диаграммы (диаграммы верхнего уровня), а также всех ветвей диаграмм, находящихся в элементах Stub. При этом тесты для подобного покрытия могут быть построены независимо для каждого элемента Stub.

В рассматриваемом подходе генерация тестов основана на использовании гидов [5], которые могут быть получены путем обхода структурированной модели.

Технология генерации тестовых сценариев для промышленных проектов [7] предусматривает три этапа: 1) генерация т. н. гидов [8, 9], кортежей событий однозначно направляющих генерацию верифицированных тестовых сценариев; 2) генерация множества детальных символьных сценариев [8], обеспечивающих покрытие функциональности системы по заданному критерию; 3) генерация множества конкретных тестовых сценариев или трасс [6, 9], по которым генерируются тестовые наборы для автоматического тестирования.

В настоящей статье описывается первый этап. Например, по UCM проекту Satellite генерируются структурные гиды (рис. 2). Для диаграммы top генерируется соответствующая директория top, которая содержит в себе папки с именами элементов Stub и гиды, покрывающие возможные поведения системы на верхнем уровне абстракции (1). Каждый из гидов раскрывается как последовательность UCM элементов (2) и содержит в себе элементы Reference ( ), которые ссылаются на гиды детального описания соответствующего уровня (3). Элемент Reference может содержать множество гидов детального уровня. Каждый из гидов детального уровня также раскрывается как последовательность UCM элементов со ссылками на нижние уровни (4). Такая детализация может быть осуществлена вплоть до самого низкого уровня абстракции системы. В результате структурного подхода образуется дерево гидов проектируемой системы.

Полученное множество гидов может использоваться как для покрытия ветвей отдельного элемента Stub, так и для покрытия системы в целом, в этом случае гиды «склеиваются» таким образом, чтобы каждый полученный результирующий гид начинался в стартовой точке диаграммы и заканчивался после покрытия определенной ветви системы. Например, гид top_start_init_CNI_change может быть определен как гид, проходящий через ветку корректной конфигурации системы good_new_cap_table с последующим прохождением элемента Stub CNI_change.

Если «склейку» выполнить для всех комбинаций гидов из каждого Stub, то получится нежелательный «взрыв» тестовых сценариев. Поэтому для тестирования необходим избирательный подход к «склейке» гидов отдельных Stub. Для этого предусмотрен механизм селекции актуальных гидов в множестве гидов каждого Stub (рис. 3),

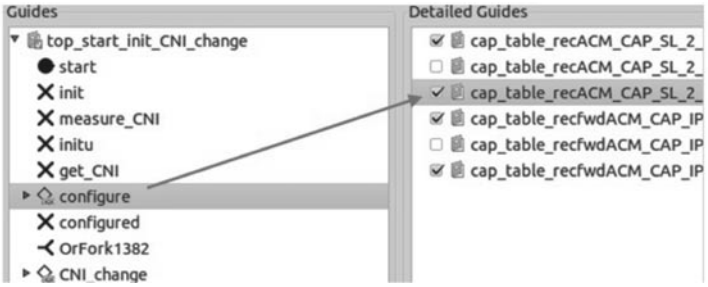

Рис. 3. Выбор гидов из множества гидов, сгенерированных для configure

реализуемый с помощью пометки chek-box в выбранном гиде.

Механизм селекции позволяет из всего множества поведений системы выбрать только интересующие: режимы функционирования и соответствующие им поведенческие сценарии, обеспечивающие полное покрытие исходных требований.

Как в случае покрытия ветвей, так и в случае покрытия селектированных пользователем режимов поведения, использование структурированного представления облегчает анализ модели и дает пользователю возможность контролировать процесс генерации гидов, используемых для получения тестового набора.

Использование структурированного представления UCM модели в процессе тестирования сложных программных комплексов позволяет существенно сократить время на анализ сгенерированного тестового набора, а также предоставляет пользователю возможности по управлению процессом генерации тестов с использованием гидов.

СПИСОК ЛИТЕРАТУРЫ

1. Z.151 : User requirements notation (URN) – Language definition [Электронный ресурс] / Режим доступа: http://www.itu.int/rec/T-REC-Z.151-200811-I/en

2. **Никифоров, И.В.** Генерация формальной модели системы по требованиям, заданным в нотации USE CASE MAPS [Текст] / И.В. Никифоров, А.В. Петров, Ю.В. Юсупов // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. –СПб.: Изд-во Политехнического ун-та, 2010. –№ 4 (103). –С. 191–195.

3. ETSI TS 102 441 V 1.1.1 [Электронный ресурс] / Режим доступа: http://www.etsi.org/deliver/etsi_ts/102400_102499/102441/01.01.01_60/ts_102441v010101p.pdf

4. **Котляров, В.П.** Основы тестирования программного обеспечения: Учеб. пособие [Текст] / В.П. Котляров, Т.В. Коликова. –М.: Интернет-Университет Информационных Технологий, 2006. – 256 с.

5. **Дробинцев, П.Д.** Автоматизация тестирования на основе покрытия пользовательских сценариев [Текст] / П.Д. Дробинцев, В.П. Котляров, И.Г. Чернорудский // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. –СПб.: Изд-во Политехнического ун-та, 2012. –№ 4 (152). –С. 123–126

6. Recommendation ITU-T Z.120. Message Sequence Chart (MSC), 11/2000 [Электронный ресурс].

7. **Baranov, S.** The technology of Automation Verification and Testing in Industrial Projects [Электронный ресурс] / S. Baranov, V. Kotlyarov, A. Letichevsky, P. Drobintsev // Proc. of St. Petersburg IEEE Chapter, International Conf. –May 18-21, 2005. – P. 81–86.

8. **Анureев, И.С.** Средства поддержки интегрированной технологии для анализа и верификации спецификаций телекоммуникационных приложений [Текст] / И.С. Анureев, С.Н. Баранов, Д.М. Белоглазов, Е.В. Бодин, П.Д. Дробинцев, А.В. Колчин, В.П. Котляров, А.А. Летичевский, А.А. Летичевский мл., В.А. Непомнящий, И.В. Никифоров, С.В. Потиенко, Л.В. Прийма, Б.В. Тютин // Труды СПИИРАН. –2013. –№ 1. –28 с.

9. **Колчин, А.** Метод генерации тестовых сценариев в среде инсерционного моделирования [Текст] / А. Колчин, В. Котляров, П. Дробинцев // Управляющие системы и машины. –Киев: Академпериодика, 2012. –Т. 6. –С. 43–48.

REFERENCES

1. Z.151: User requirements notation (URN) – Language definition. Availiable http://www.itu.int/rec/T-REC-Z.151-200811-I/en

2. **Nikiforov I.V., Petrov A.V., Yusupov Yu.V.** Generatsiia formal'noi modeli sistemy po trebovaniiam, zadannym v notatsii USE CASE MAPS [*Generation of formal model of a sysmem from requirements specified in use case map*] / Nauchno-tekhnicheskie vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie [*St. Petersburg State Polytechnical University Journal. Computer Science. Telecommunications and Control Systems*]. – St. Petersburg: Izd-vo Politekhnicheskogo un-ta, 2010. – № 4 (103). – S. 191–195. (rus)

3. ETSI TS 102 441 V 1.1.1. Availiable http://www.etsi.org/deliver/etsi_ts/102400_102499/102441/01.01.01_60/ts_102441v010101p.pdf

4. **Kotliarov V.P., Kolikova T.V.** Osnovy testirovaniia programmnogo obespecheniia: Ucheb. posobie. – Moscow: Internet-Universitet Informatsionnykh Tekhnologii, 2006. – S. 256. (rus)

5. **Drobintsev P.D., Kotliarov V.P., Chernorutskii I.G.** Avtomatizatsiia testirovaniia na osnove pokrytiia pol'zovatel'skikh stsenariev [*Approach for testing automation based on user scenarious*] / Nauchno-tekhnicheskie vedomosti SPbGPU. Informatika. Telekommunikatsii. Upravlenie [*St. Petersburg State Polytechnical University Journal. Computer Science. Telecommunications and Control Systems*]. – St. Petersburg: Izd-vo Politekhnicheskogo un-ta, 2012. – № 4 (152). – S. 123–126. (rus)

6. Recommendation ITU-T Z.120. Message Sequence Chart (MSC), 11/2000.

7. **Baranov S., Kotlyarov V., Letichevsky A., Drobintsev P.** The technology of Automation Verification and Testing in Industrial Projects / Proc. of St. Petersburg IEEE Chapter, International Conf.; May 18-21, 2005. –P. 81–86. (rus)

8. **Anureev I.S., Baranov S.N., Beloglazov D.M., Bodin E.V., Drobintsev P.D., Kolchin A.V., Kotliarov V.P., Letichevskii A.A., Letichevskii A.A. ml., Nepomniashchii V.A., Nikiforov I.V., Potienko S.V., Priima L.V., Tiutin B.V.** Sredstva podderzhki integrirovannoi tekhnologii dlia analiza i verifikatsii spetsifikatsii telekommunikatsionnykh prilozhenii / Trudy SPIIRAN. – 2013. – № 1. – 28 s. (rus)

9. **Kolchin A., Kotliarov V., Drobintsev P.** Metod generatsii testovykh stsenariev v srede insertsionnogo modelirovaniia / Upravliaiushchie sistemy i mashiny. – Kiev: Akademperiodika, 2012. – T. 6. – S. 43–48.

ДРОБИНЦЕВ Павел Дмитриевич – *доцент кафедры информационных и управляющих систем Санкт-Петербургского государственного политехнического университета, кандидат технических наук.*

195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.

DROBINTSEV, Pavel D. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.

НИКИФОРОВ Игорь Валерьевич – *аспирант кафедры информационных и управляющих систем Санкт-Петербургского государственного политехнического университета.*

195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.

NIKIFOROV, Igor V. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.

КОТЛЯРОВ Всеволод Павлович – *профессор кафедры информационных и управляющих систем Санкт-Петербургского государственного политехнического университета, кандидат технических наук.*

195251, Россия, Санкт-Петербург, ул. Политехническая, д. 29.
E-mail: vpk@spbstu.ru

KOTLYAROV, Vsevolod P. *St. Petersburg State Polytechnical University.*
195251, Politekhnicheskaya Str. 29, St. Petersburg, Russia.
E-mail: vpk@spbstu.ru