УДК 004.415

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

ОГРАНИЧЕНИЯ НА МНОГОПОТОЧНЫЕ КОНСТРУКЦИИ И ВРЕМЕННЫЕ ЗАДЕРЖКИ ЯЗЫКА UCM

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

RESTRICTIONS ON CONCURRENT CONSTRUCTION AND TIME DELAYS OF UCM LANGUAGE

Описано уточнение семантики языка UCM для моделирования систем реального времени с использованием временных задержек и параллельных потоков. Изучен подход преобразования многопоточных конструкций, временных задержек языка UCM в язык базовых протоколов.

Уточнение семантики предложено исходя из того, что существующая версия стандарта UCM позволяет создавать семантически некорректные модели. Описанные расширения и ограничения на язык позволяют решить проблемы некорректности модели в индустриальных проектах.

UCM. ТАЙМЕР. ЗАДЕРЖКА. СИНХРОНИЗАЦИЯ. ПОТОКИ.

The paper describes an approach to adjustment of semantics for UCM real time constructions in implementation of translator into Basic Protocols notation. The following constructions and their adjustment are described: multithreading and delays.

The main problem of such constructions is that initial version of UCM standard allows to create semantically incorrect models. Proposed extensions and restrictions of UCM semantics allowed solving of these problems for different types of projects

UCM. TIMERS. DELAYS. SYNCHRONIZATION. THREAD.

Разработка программной системы начинается с создания требований. Документы, описывающие спецификацию промышленной системы, как правило, задаются на естественном языке и состоят из сотен или тысяч пунктов. Нередко исходные спецификации содержат ошибки, связанные с противоречивостью, неполнотой и недетерменизмом в поведении системы. Обнаружение и исправление ошибок, связанных с требованиями, эффективно осуществлять на ранних этапах разработки [1].

Проанализировать спецификации промышленных систем на наличие ошибок вручную, без поддерживающего инструментария, практически невозможно. Существующие системы верификации и тестирования не могут работать с неформализованными спецификациями. Поэтому актуальна задача формализации исходных текстовых описаний требований с использованием входных языков средств верификации и тестирования.

Одной из перспективных интегрированных технологий автоматизации тестирования и верификации на основе формальных моделей является технология VRS/TAT [2], в которой для высокоуровневого описания поведенческих моделей используется нотация Use Case Maps (UCM) [3, 5], а инструменты автоматизации проверки и генерации работают с моделью на языке базовых протоколов [4].

Язык спецификаций UCM стандартизован, но тем не менее содержит ряд неточностей, не позволяющих однозначным и корректным образом отобразить семантику моделируемых систем.

В статье предложены ограничения, накладываемые на разработку многопоточных моделей систем, а также уточнения семантики конструкций языка UCM, моделирующих временные задержки и прерывания.

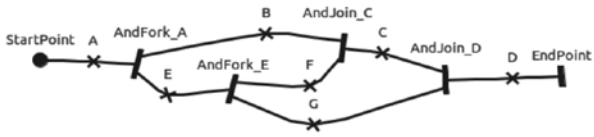

Рис. 1. Нарушение структуры параллельных потоков

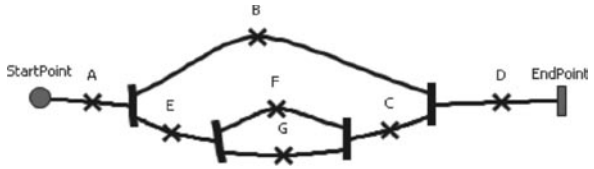

Рис. 2. Граф с правильной структурой потоков

Ограничения на разработку многопоточных систем

Несмотря на все преимущества использования языка UCM и семантики его элементов, существующий стандарт Z.151 [3] содержит ряд неточностей, затрудняющих моделирование многопоточных систем.

Баланс скобок в спецификации параллельных потоков. Рассмотрим ситуацию, когда синтаксически правильные элементы порождения и синхронизации потоков на графе могут приводить к нарушению структуры параллельных потоков, а следовательно, к некорректному поведению системы.

На рис. 1 после элементов AndFork_A и AndFork_E происходит порождение потоков B, E и F, G соответственно, а на элементах AndJoin_C и AndJoin_D выполняется синхронизация потоков соответственно B, F и C, G.

Легко заметить, что синхронизация потоков, порожденных разными элементами, значительно затрудняет механизм поиска и отладки ошибок в системе, а также затрудняет возможность отслеживания связи «родитель/ребенок» в иерархии потоков. Такие связи полезны при анализе ситуаций, когда поток «ребенок» продолжает свое выполнение после завершения потока «родителя».

Граф поведения системы, который содержит корректную структуру порождения и синхронизации потоков, изображен на рис. 2.

Анализ структуры потоков можно сравнить с анализом скобочной формы математических выражений. Если скобочная форма выражения нарушена, то оно считается синтаксически некорректным. Также и в моделировании параллельных потоков: если нарушена структура потоков, то и вся система считается синтаксически неверной.

Изучение потоков на наличие ошибок в структуре и их исправление позволяет создавать синтаксически корректные модели систем.

Неограниченное порождение потоков. Рассмотрим ситуацию, изображенную на рис. 3. После элемента D происходит порождение потоков B и E. Поток B завершает свое исполнение на элементе EndPoint. Поток E возвращается по циклу, не содержащему условия ограничения итераций, через элемент D и производит порождение новых потоков B’ и E’. Сценарий поведения повторяется для потока E’.

Неограниченные циклы приводят к порождению неограниченного числа незавершенных потоков, что приводит к дефициту памяти или других ресурсов. Поэтому необходимо вводить количественные ограничения на использование таких конструкций в разрабатываемых моделях.

Гонка данных при доступе параллельных процессов к общим ресурсам. Рассмотрим ситуацию, когда на параллельных ветвях используются разделяемые ресурсы без синхронизации. На рис. 4 представлена ситуация, в которой два параллельных потока используют общий разделяемый ресурс var без синхронизации. Подобная формализация приводит к гонке при доступе к данным [6].

В модели наиболее интересны два сце-

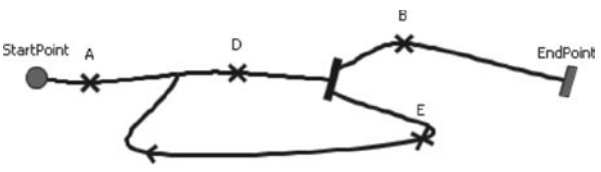

Рис. 3. Рекурсивное порождение потоков

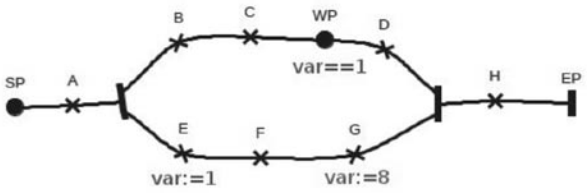

Рис. 4. Разделяемые ресурсы без синхронизации

нария исполнения:

если исполняется сценарий «E»-«F»-«G»-«WP», то элемент «D» никогда не будет применен, этот сценарий ведет в deadlock;

если исполняется сценарий «E»-«F»-«WP»-«G», то элемент «D» можно применить, и данный сценарий достигает конечной точки EP.

Устранить deadlock можно, добавив синхронизацию и тем самым исключив параллельный доступ к общему ресурсу системы (рис. 5).

Предлагаемые ограничения на разработку многопоточных систем в UCM. Запрещается использовать:

параллельные конструкции с нарушением структуры потоков;

неограниченное рекурсивное порождение потоков;

разделяемые ресурсы без синхронизации на параллельных участках исполнения.

Особенности моделирования временных задержек

В индустриальных системах часто встречаются требования создавать временные задержки. Следует отметить, что в рассматриваемом случае используется событийное моделирование с относительным временем – длительностью между событиями. События – это изменение значений атрибутов системы.

По стандарту [3] с элементом Timer (рис. 6) связывают два исходящих пути: нормальный путь исполнения (regular path – RP) и путь по истечению (timeout path – TOP). Для выбора каждого из путей существуют условия CRP и CTOP соответственно. Также существует триггер путь (trigger path или trigger counter), влияющий на поведение таймера и позволяющий отметить задержку.

Семантика элементов, моделирующих временные задержки, по стандарту Z.151 [3] содержит описание вариантов возможных поведений модели в зависимости от наступивших событий, но при этом не описывает, какие типы событий связаны с таймерами, и не определены типы ряда событий, характерных для спецификации телекоммуникационных приложений.

Расширим описание семантики таймера следующими событиями.

- Установка таймера: TIMER_SET <имя таймера>. Событие происходит при достижении элемента Timer.
- Истечение таймера: TIMER_EXPIRE <имя таймера>. Происходит после выполнения условия CTOP.
- Остановка таймера: TIMER_RESET <имя таймера>. Событие происходит после начала исполнения одного из путей RP или TOP, или наступления триггер события.

Используя семантику элемента Timer и связанных с ним событий, можно выделить три типа временных задержек:

1) простая задержка, особенность моделирования которой – строго зафиксированные условия исходящих путей (false) и отсутствие триггер события;

2) задержка с прерыванием, особенность моделирования которой – наличие триггер события;

3) задержка с прерыванием исполнения, особенность моделирования которой – наличие прерывания FailurePoint на таймаут пути.

Рис. 5. Модель системы с синхронизацией

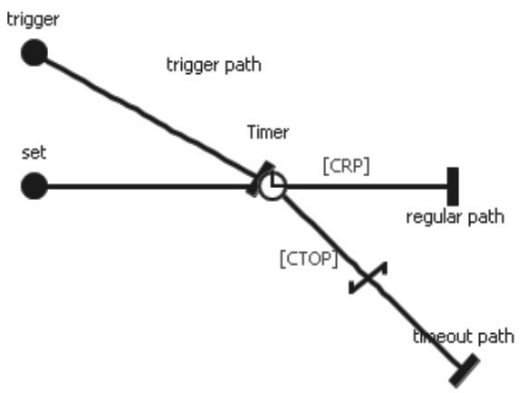

Рис. 6. UCM диаграмма с элементом Timer

Предложенное расширение семантики таймера позволило решить проблему описания задержек для телекоммуникационных проектов.

Преобразование временных задержек в базовые протоколы

Для инструментов верификации и тестирования VRS/TAT разработан инструмент преобразования моделей на языке UCM в модели на языке базовых протоколов [8–10]. В трансляторе UCM→BP реализована концепция преобразования временных задержек и прерываний, а также реализованы проверки сформулированных ограничений на разработку многопоточных систем.

Для элемента Timer существует некоторый атрибут timer_var, отвечающий за состояние таймера и принимающий два значения: true – если таймер установлен, false – если таймер остановлен. По умолчанию значение атрибута – false

В базовом протоколе для элемента Timer, отвечающем за установку таймера, в постусловии генерируется выражение timer_var:=true, в поле процесса базового протокола генерируется выражение TIMER_SET.

Для каждого исходящего пути (RP и TOP) из элемента Timer генерируется по одному базовому протоколу.

Предусловие протокола для пути RP (выражение для выбора регулярного пути) генерируется в соответствии с логической формулой, выведенной на основе [3]:

$$(timer\_var=true)\&(CRP)\ V\ (timer\_var=true)\&(trigger)\&(CTOP),$$

где trigger – логическое выражение для триггер события.

В поле процесса БП генерируется действие TIMER_RESET.

В постусловии этого протокола генерируется выражение, моделирующее остановку таймера: timer_var:=false.

Рассмотрим протокол для таймаут пути (TOP). В общем случае в предусловии генерируется выражение:

$$(timer\_var=true)\&(\sim CRP)\&[(CTOP)\ V(\sim trigger)\&(\sim CTOP)]$$

В поле процесса БП для пути TOP генерируются операции TIMER_EXPIRE, TIMER_RESET.

Таким образом, преобразование временных задержек сводится к генерации трех базовых протоколов с различными логическими выражениями в предусловиях.

Для каждого из рассмотренных случаев моделирования таймера можно произвести оптимизацию логических выражений, т. к. значения используемых конъюнктов и дизъюнктов заранее известно.

Рассмотренные в статье методы уточнения семантики элементов стандарта UCM, моделирующих временные задержки и прерывания, а также накладываемые ограничения на разработку многопоточных систем позволяют производить моделирование комплексных телекоммуникационных систем, при этом уменьшая возможность создания семантически неверных поведений в модели.

Методы реализованы в трансляторе UCM→BP, что позволяет сделать применение технологической цепочки VRS/TAT более удобным и эффективным на проектах средней и большой сложности.

Предложенный транслятор совместно с поддерживающим инструментарием технологии VRS/TAT был применен при разработке модулей телекоммуникационных приложений и показал существенное сокращение затрат при производстве качественного индустриального программного продукта.

СПИСОК ЛИТЕРАТУРЫ

1. **Booch, Gr.** Object-Oriented Analysis and Design with Applications [Text] / Gr. Booch, R. Maksimchuk, M. Engel [et al.]. —3rd ed. —Addison-Wesley Professional, 2007. —720 p.

2. **Веселов, А.О.** Автоматизация тестирования в области телекоммуникаций [Текст] / А.О. Веселов, В.П. Котляров // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. —СПб.: Изд-во Политехнического ун-та, 2010. —№ 4 (103). —С. 180—185.

3. Recommendation ITU-T Z.151 [Электронный ресурс] User requirements notation (URN), 11/2008.

4. **Летичевский, А.А.** Спецификация систем с помощью базовых протоколов [Текст] / А.А. Летичевский, Ю.В. Капитонова, А.А. Летичевский (мл.) [и др.] // Кибернетика и системный анализ. —2005. —№ 4. —С. 3—21.

5. **Buhr, R.J.A.** Use Case Maps for Object-Oriented Systems [Text] / R.J.A. Buhr, R.S. Casselman. —Prentice Hall, 1995.

6. **Гергель, В.П.** Высокопроизводительные вычисления для многопроцессорных многоядерных систем [Текст] / В.П. Гергель. —Нижний Новгород: Изд-во Нижегородского ун-та, 2010. — 544 с.

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

8. **Никифоров, И.В.** Генерация формальной модели системы по требованиям, заданным в нотации USE CASE MAP [Текст] / И.В. Никифоров, А.В. Петров, Ю.В. Юсупов // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. —2010. —№ 4 (103). —С. 191—195.

9. **Ануреев, И.С.** Средства поддержки интегрированной технологии для анализа и верификации спецификаций телекоммуникационных приложений [Текст] / И.С. Ануреев, 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 [и др.] // Тр. СПИИРАН. —2013. —№ 1. —28 с.

10. **Никифоров, И.В.** Статический метод отладки тестовых сценариев, сгенерированных с помощью эвристик [Текст] / И.В. Никифоров, А.В. Петров, В.П. Котляров // Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. —СПб.: Изд-во Политехнического ун-та, 2012. —№ 4 (152). —С. 114—119.

REFERENCES

1. **Booch Gr., Maksimchuk R., Engel M., Young B., Conallen J., Houston K.** Object-Oriented Analysis and Design with Applications; 3rd ed. — Addison-Wesley Professional, 2007. — 720 p.

2. **Veselov A.O., Kotliarov V.P.** Avtomatizatsiia testirovaniia v oblasti telekommunikatsii [*Testing automation of projects in telecommunication domain*] / 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. 180—185. (rus)

3. Recommendation ITU-T Z.151. User requirements notation (URN), 11/2008.

4. **Letichevskii A.A., Kapitonova Iu.V., Letichevskii A.A. (ml.) i dr.** Spetsifikatsiia sistem s pomoshch'iu bazovykh protokolov / Kibernetika i sistemnyi analiz. — 2005. — № 4. — S. 3—21. (rus)

5. **Buhr R.J.A., Casselman R.S.** Use Case Maps for Object-Oriented Systems. — Prentice Hall, 1995.

6. **Gergel' V.P.** Vysokoproizvoditel'nye vychisleniia dlia mnogoprotsessornykh mnogoiadernykh sistem. — Nizhnii Novgorod: Izd-vo Nizhegorodskogo universiteta, 2010. — 544 s. (rus)

7. **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)

8. **Nikiforov I.V., Petrov A.V., Iusupov Iu.V.** Generatsiia formal'noi modeli sistemy po trebovaniiam, zadannym v notatsii USE CASE MAP [*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)

9. **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)

10. **Nikiforov I.V., Petrov A.V., Kotliarov V.P.** Staticheskii metod otladki testovykh stsenariev, sgenerirovannykh s pomoshch'iu evristik [*Static approach for debugging of test scenarious generated with usage of heuristics*] / 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. 114–119. (rus)

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

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.