Обсуждение автоматного программирования

(http://lambda-the-ultimate.org/node/2072)

Использование теории автоматов, на первый взгляд, конфликтует с самим понятием «программное обеспечение». Возможно, поэтому автоматная теория очень редко упоминается в информатике (computer science) и даже при обсуждении проблем разработки программного обеспечения (software engineering). Представьте же мое удивление, когда я наткнулся на две эти статьи в Википедии: Automata-Based Programming (в это время там находился текст, сейчас расположенный по адресу http://en.wikipedia.org/wiki/Automata-based_programming_%28programming_technology%29) и Switch Technology.

By Hank Thediek at 2007-02-18 18:32

Обсуждение

1. Использование теории автоматов

> Использование теории автоматов, на первый взгляд, конфликтует с самим понятием «программное обеспечение».

Я бы сказал по-другому. В контексте языков программирования автоматы часто используются при операционной спецификации в форме виртуальных машин (известный пример – SECD машина (http://www.ict.edu.ru/ft/005135//ch10.pdf)). В этом контексте автоматная теория не так уж и конфиликтует с чем-либо, а просто является другим уровнем представления. Однако, большинство ПО, кажется, более продуктивно разрабатывать на более "высоком" уровне при помощи языков, которые потом компилируются в автоматы.

Даже когда задача состоит в разработке автомата самого по себе, сложные системы, возможно, стоит разрабатывать на языках высокого уровня, а потом трансформировать (вручную или автоматически) в автоматную форму. Например, некоторые языки, используемые в разработке управляющих систем (например, языки синхронного программирования *Esterel* и *Lustre*), могут быть скомпилированы в конечные автоматы.

> Возможно, поэтому автоматная теория очень редко упоминается в информатике (computer science).

Может быть, это потому, что автоматы скорее воспринимаются, как фундаментальные основы информатики. Разве термин «Машина Тьюринга» сам по себе ни о чем не говорит? ;)

By Anton van Straaten at Sun, 2007-02-18 19:24

2. Семантика состояния

Изложенное в статье «Автоматное программирование» (http://en.wikipedia.org/wiki/Automata-based_programming_%28programming_technology%29) несколько отличается от теоретических «использований» автоматов в информатике. Автоматы и состояния играют малую роль в ПО. При императивном стиле программирования абсолютно темным кажется вопрос о конкретном состоянии, которое неизбежно присутствует.

«Явное» использование автоматов и состояний в ПО имеет много преимуществ, упомянутых в статьях по двум ссылкам выше. Существует разница между семантикой состояний и использованием состояний в качестве семантики. Грубо говоря, я бы охарактеризовал «семантику состояний» как материальное или телесное. Подавляющая часть семантик в информатике являются детонационными (denotational) и аксиоматичными (axiomatic). Это уже само по себе, конечно, не материально. Недостаток явной телесности в компьютерных языках (ПО) – корень многих проблем. Только представьте себе, что невозможно говорить о материальном мире на натуральном языке.

By Hank Thediek at Sun, 2007-02-18 21:42

3. Недостаток материальности

Я как раз думал об этом вчера, когда ковырялся с STM и не распараллелил ничего.

Одна из главных причин, почему атомарность слишком жесткое требование – это то, что некоторые величины нужно сохранять. Классический пример – сбережения в банке, которые должны мигрировать из одного места в другое. Если бы можно было проще выражать неизменные величины, то, возможно, появились бы более натуральные стили программирования.

By Gavin Mendel-Gleason at Mon, 2007-02-19 10:16

4. Хм, Теорема Нётер

Теорема Нётер для логики/информатики (http://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B5%D0%BC%D0%B0_%D0%9D%D1%91%D1%82%D0%B5%D1%80) ...

By Derek Elkins at Mon, 2007-02-19 19:38

5. А у меня другой опыт

Я первый семестр учусь в качестве студента по компьютерной специальности, и нам читают «ядро» курса «Фундаментальные основы информатики». Курс имеет дело исключительно с автоматной теорией и ее применениями в информатике на разный уровнях (программирование, парсинг, верфикация ...)

By Chris King at Sun, 2007-02-18 20:49 | login or register to post comments

6. Контрпример

Я считаю, что конечные автоматы часто используются в разработке ПО. Например, в нашем web XML-редакторе *Xopus*. Мы используем автоматы для (пре-)валидации XML и создания детей новых элементов, вводимых пользователем. Конечные автоматы делают все это быстро, даже будучи написанными на *JavaScript*.

By Sjoerd Visscher at Sun, 2007-02-18 20:58

7. Хорошая книга

Hopcroft J.E., Motwani R, Ullman J. D. Introduction to Automata Theory, Languages, and Computation одна из моих любимых книг. Несколько месяцев подряд после ее прочтения все есть конечный автомат :-)

By Luke Gorrie at Mon, 2007-02-19 09:32

8. Переизобретение колеса

Событийное исчисление разбавляет старую тему о состояниях новым интересным витком. В событийном исчислении мы начинаем с переменных (тут мы не используем это старое слово «состояние») и вносим в них суть, используя аксиомы постоянства, доменные аксиомы, и достаточно нежный процесс, называемый «circumscription». В качестве результата получается большое транзитное правило, которое для всего мира выглядит как конечный автомат (но они не используют это слово). Существует книга Muller E.T. Commonsense-reasoning и много другой литературы, которую можно найти, поискав по словосочетанию «event calculus». Переизобретение вещей не такая уж и плохая идея.

By Hank Thediek at Mon, 2007-02-19 13:20

9. ASML

Я не нахожу эти статьи таким уж большим сюрпризом, учитывая вещи типа программирования на Abstract State Machines, например, http://research.microsoft.com/fse/asml/.

By Andreas Bauer at Tue, 2007-02-20 09:32

10. Упоминавшейся, например, тут

ASML упоминается, например, тут **– http://lambda-the-ultimate.org/node/1833.**

By Ehud Lamm at Tue, 2007-02-20 10:53

11. VisualState семантика и реализация

Как счастливого пользователя *StateCharts* для real time-программирования встраиваемых систем, меня разочаровывает, что этой технологии так мало внимания уделяется в академических исследованиях.

Одно исключение... A. Wasowski and P. Sestoft: On the Formal Semantics of VisualSTATE Statecharts. Technical report TR-2002-19, IT University of Copenhagen, Denmark, September 2002. http://www.itu.dk/~sestoft/papers/ITU-TR-2002-19.pdf

Также смотри... A. Wasowski: Flattening statecharts without explosions. ACM SIGPLAN Notices, Volume 39, Issue 7, July 2004. http://delivery.acm.org/10.1145/1000000/997200/p257-wasowski.pdf?key1=997200&key2=9209361121&coll=GUIDE&dl=GUIDE&CFID=29294912&CFTOKEN=19125348

By e at Thu, 2007-02-22 17:13 | login or register to post comments

12. Еще одно имя для реактивного/событийного/сигнального программирования

Этот тип программирования лучше для моделирования ситуаций, встречающихся в реальном мире.

Кстати, какой существует хороший синтакс для этого стиля программирования? Я пробовал использовать exceptions-like-синтаксис, но он делает код практически нечитабельным, потому что реакция на события может быть очень далеко от источника.

By Achilleas Margaritis at Tue, 2007-02-20 11:26

13. Action Languages

Action languages (http://www.cs.utexas.edu/users/vl/papers/refart.ps) очень близки к событийному счислению, упомянутому выше, но они несколько более интуитивны и на несколько более высоком уровне. Например, язык *C+*. Я нашел несколько публикаций по запросу «C+ Action Language». Кроме того, глава 15 книги «Commonsense Reasoning», упомянутой выше, обсуждает связь между событийным счислением и несколькими другими подобными техниками, включая *Action Languages* и *C+*.

By Hank Thediek at Tue, 2007-02-20 15:15

14. W3C вступает в игру

State Chart XML (SCXML): State Machine Notation for Control Abstraction

By Sjoerd Visscher at Wed, 2007-02-21 18:53