ной разработкой компании NVIDIA. Однако он не подходят в качестве универсального средства создания параллельных программ для гибридных вычислительных систем из-за своей низкоуровневости.

В работе описаны способы программирования задачи слежения за «космическим мусором» с помощью средств языка OpenCL. Приведены результаты разработки для реализации задачи на платформе CUDA. Описывается технология подготовки программ с помощью инструментальной ЭВМ на базе персонального компьютера.

Литература

1. Aaftab Munshi, Benedict R. Gaster, Timothy G. Mattson, James Fung, Dan Ginsburg. OpenCL Programming Guide. - New Jersey: Addison-Wesley, 2008-2011.
2. Тарас Шаповалов. Дизайн OpenCL. - http://opencl.ru/design, 2014.
3. OpenCL. Что это такое и зачем он нужен. - http://habrahabr.ru/post/72247, 2009.
4. OpenCL. Подробности технологии. - http://habrahabr.ru/post/72650, 2009.
5. Apple Inc. OpenCL Programming Guide for Mac // Mac Developer Library. - https://developer.apple.com/library/mac/documentation/Performance/Conceptual/OpenCL_MacProgGuide, 2013.
6. Advanced Micro Devices, Inc. Heterogeneous Computing with OpenCL // Université du Québec à Chicoutimi. - Elsevier Inc., 2013.
7. Ofer Rosenberg. OpenCL Overview lecture // The Haifa Linux Club. - haifux.org/lectures/267/OpenCL_Overview.pdf, 2009.
8. Neil Trevett. OpenCL Introduction. - http://www.khronos.org/opencl, 2013.

УДК 650.1

СРАВНЕНИЕ ЭТАПОВ ЖИЗНЕНОГО ЦИКЛА ПРОГРАММ И МОДЕЛЕЙ

В.В. Литвинов, А.А. Задорожний
Черниговский национальный технологический университет, Украина

Некоторые этапы жизненного цикла программного обеспечения схожи с этапами жизненного цикла моделей. Это означает, что инструментальные средства, которые используются на этапах жизненного цикла разработки программного обеспечения, можно использовать на схожих этапах жизненного цикла моделей.

Рассмотрим, какие этапы жизненного цикла программ и моделей схожи между собой (рис. 1).

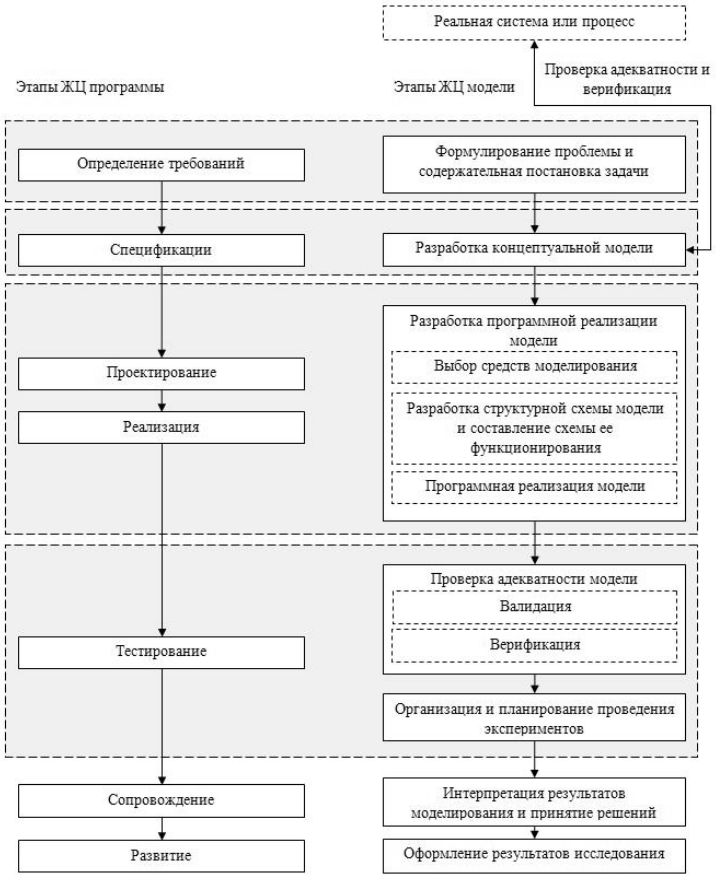

Рис. 1. Сравнение этапов жизненного цикла программ и моделей

Если для моделирования сложной системы создается имитационная модель, то некоторые этапы жизненного цикла моделей очень схожи с этапами жизненного цикла программ.

Например, если говорить об этапе формулирования проблемы и содержательной постановке задачи, то одной из важных задач, которую необходимо решить на этом этапе при создании имитационной модели системы является определение требования к модели и к моделируемой системе. Таким образом, при создании модели системы можно выделить два вида требований: требования к моделируемой системе, и требования к модели. Требования к системе можно получить из готовой системы или из разрабатываемой системы. Например, если мы создаем модель программной системы, то, скорее всего, требования к

ней уже были выдвинуты на этапе анализа требований к системе. После того как требования к системе и к модели системы были выдвинуты, их необходимо проверить, чтобы они были непротиворечивы. И для этого можно использовать те же инструменты, которые используются в жизненном цикле программы на этапе анализа требований к системе. Также на этапе формулирования проблемы решаются вопросы, связанные с определением целей создания программы либо модели, определением временных рамок разработки и ресурсов, которые можно затратить на разработку. Производится активное взаимодействие с заказчиком для более четкого формирования требований.

Этап спецификации жизненного цикла разработки программы соответствует этапу разработки концептуальной модели в жизненном цикле создания модели. Спецификацию программы можно считать ее концептуальной моделью, особенно если спецификация представлена в виде сущности описанной на некотором специализированном языке спецификаций. Задачей этапа спецификации является сбор всех требований и устранение противоречивости этих требований. Этап разработки концептуальной модели также в некоторой мере предназначен для устранения противоречивостей, таких как создание максимально точной модели при как можно большем ее упрощении по сравнению с реальным объектом, что позволит сэкономить ресурсы, затрачиваемых при создании модели, а также ресурсы, затрачиваемые на прогоны модели.

Наиболее схожими являются этапы проектирования и реализации в ЖЦ создания программ и этап разработки программной реализации модели. Схожесть проявляется особенно четко в том случаи, если для создания был выбран прикладной язык программирования. Этап разработки программной реализации у модели тесно связан с разработкой концептуальной модели. Хотя разработка имитационной модели очень сильно похожа на разработку обычной программы, но разработка модели происходит на основании зафиксированной архитектуры, которая выбирается на основании разработки концептуальной модели.

Этап разработки концептуальной модели при создании имитационной модели системы состоит из трех шагов:

1. Выбор степени детализации описания объекта моделирования, на котором определяется, насколько детально должен быть описан объект моделирования. На этом шаге необходимо выбрать между стоимостью модели и погрешностями моделирования. Чем выше будет степень детализации объекта моделирования, тем дольше будет происходить процесс ее разработки и калибровки и тем меньше будут отклонения от реальной системы полученных результатов.

2. Описание переменных модели, на котором определяются входные, выходные, внутренние параметры и их распределения.
3. Формализованное изображение концептуальной модели, который является самым важным шагом на этапе разработки концептуальной модели, и который тесно связывает этот этап с этапом разработки программной реализации модели.

Необходимо рассмотреть, каким образом формализованное изображение концептуальной модели влияет на этап программной реализации модели, а также определить, что такое программа с фиксированной архитектурой. Фиксированная архитектура имитационной программы появляется в тот момент, когда происходит формальное изображение концептуальной модели. После того как был выбран формализм для описания имитационной модели, у этого формализма, как правило есть набор подходов, которые применяют, чтобы их основании создать программную реализацию модели.

Этап тестирования ЖЦ программы соответствует этапам проверки адекватности модели и организации и планирования проведения экспериментов. Также можно говорить о том, что экспериментирование над программной реализацией модели можно проводить с использованием каких же инструментальных средств, как и при ее тестировании. Можно говорить о том, что процесс экспериментирования над моделью схож с процессом ее тестировании, но без сравнения результатов эксперимента с заведомо известным результатом. Вместо того чтобы сравнивать результат прогона модели с заведомо известным результатом, производиться их постобработка и вывод окончательного результата на экран.

Выводы. Сравнение жизненных циклов программ и моделей показало, что этапы жизненного цикла программы подобны этапам жизненного цикла модели. Для создания моделей наиболее походящим является итерационная модель жизненного цикла, которая также используется для создания программ. Особенностью применения итерационной модели ЖЦ для создания моделей является применение процессов проверки модели на адекватность и валидации модели в качестве условий остановки итерации.

Поскольку этапы жизненного цикла создания моделей сходи с этапами жизненного цикла создания программ, то инструментальные средства, которые используются на этапах жизненного цикла программ можно использовать на схожих этапах жизненного цикла моделей.

Литература

1. Томашевський В.М. Моделювання систем. – К.: Видавнича група BHV, 2007. – 352 с.

2. Литвинов В.В., Марьянович Т.П. Методы построения имитационных систем. – К.: Наук. Думка, 1991. – 120 с.
3. Макаров, В.Л. Социальное моделирование – новый компьютерный прорыв (агент-ориентированные модели) / В.Л. Макаров, А.Р. Бахтизин. – М.: Экономика, 2013. – 295 с.
4. Клаус Н.Г., Свечкарев В.П. Многоагентное моделирование конфликтных ситуаций: Учебное пособие. – Ростов-на-Дону: Изд-во СКНЦ ВШ ЮФУ, 2012. – 124 с.
5. Collier N, Howe T, North MJ: Onward and upward: The transition to Repast 2.0. In Proceedings of the first annual North American Association for Computational Social and Organizational Science conference. Edited by Carley K. Carnegie Mellon University, Pittsburgh; 2003.
6. Balci O. Verification, Validation And Accreditation Of Simulation Models // Proceedings of the 29th conference on Winter simulation. – N.Y./^ ACM Press, 1997. – P. 135-141.

УДК 681.3

МОДЕЛИРОВАНИЕ В ОБУЧАЮЩИХ СИСТЕМАХ

В.В. Литвинов, И.С. Посадская
Черниговский национальный технологический університет, Украина

Для выхода на современные рубежи в культурном, социальном, экономическом, научном и техническом развитии Украине необходимы разносторонне развитые, профессионально подготовленные специалисты. Поэтому все большую актуальность приобретает проблема разработки и внедрения в практику наиболее эффективных информационных технологий обучения, отвечающих общей концепции образования. Суть этой концепции состоит в подготовке личности, которая обладает, кроме достаточного для требований современного уровня производства, науки, культуры и государства объемом базовых знаний, умений и навыков, способностью к активной творческой профессиональной и общественной деятельности. Важная роль в этом отводится новым программно-техническим средствам индивидуализации процесса обучения.

Среди разнообразия обучающего программного обеспечения гибкость обучения обеспечивает только один класс систем - интеллектуальные обучающие системы (ИОС). В них, как правило, используют модели знаний: знания о предметной области учебного курса, рассматриваемые как эталон; знания о предметной области учебного курса, сформированные в представлениях обучаемого; знания о процессе обучения.

Имеется достаточно широкий спектр способов представления знаний: логические методы; семантические сети; фреймы; продукционные системы.