СБОРНИК **НАУЧНЫХ ТРУДОВ НГТУ. – 2016. – № 2(84).** – 107–115

УДК 004.434:004.94

ПРИМЕНЕНИЕ МЕТОДОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ «АРХИТЕКТУРА, УПРАВЛЯЕМАЯ МОДЕЛЬЮ»*

И.Р. НАЗАРОВ[1], А.В. АНИКИН[2]

[1] *630073, РФ, г. Новосибирск, пр. Карла Маркса, 20, Новосибирский государственный технический университет, магистрант кафедры автоматики. E-mail: igorfresh@mail.ru*

[2] *630073, РФ, г. Новосибирск, пр. Карла Маркса, 20, Новосибирский государственный технический университет, магистрант кафедры автоматики. E-mail: anikantonsd@yandex.ru*

В настоящее время устройства и системы содержат большое количество элементов и модулей, управляемых программами. Сложность систем растет, а значит, появляется потребность в качественном программном обеспечении, разработка которого традиционно сопровождается различными проблемами. Идет поиск наиболее оптимальных методов разработки, способных повысить качество ПО и уменьшить время его проектирования. В данной статье описывается применение методологии разработки ПО, созданной с целью решить имеющиеся проблемы проектирования систем.

Ключевые слова: разработка программного обеспечения, моделирование, модель, UML, MDA, объектно-ориентированное программирование, CASE, платформа

DOI: 10.17212/2307-6879-2016-2-107-115

ВВЕДЕНИЕ

Многие десятилетия стоит задача поиска методологии разработки программного обеспечения, с помощью которой можно было бы избежать ошибок разработки ПО, добиться повышения качества, надежности и скорости его создания.

* Статья получена 26 январь 2016 г.

1. ПРОБЛЕМЫ РАЗРАБОТКИ ПО

Можно выделить следующие проблемы разработки ПО [1–3].

1. Отсутствие прозрачности – невозможность отследить степень готовности проекта.

2. Ручное написание кода, сопровождающееся ошибками.

3. Низкое качество кода, определяющееся его читаемостью и неудовлетворительным использованием вычислительных ресурсов.

4. Недостаточная надежность. Устранение ошибок требует большого опыта и времени.

5. Формулирование требований. Заказчик не всегда может объяснить свои потребности, а разработчик не всегда может их понять, кроме того, нередко возможно изменение требований в процессе разработки.

6. Мультиплатформенность. Создание ПО под несколько платформ или перенос с одной на другую является трудозатратным процессом [4].

7. Документация. Ее наличие и актуальность необходимы на всех этапах разработки.

Оптимальный подбор методологии должен помочь в решении перечисленных проблемы разработки. Основным направлением в поиске является моделирование и увеличение уровня абстракции, переход от машинного кода к ассемблеру, далее – к языкам высокого уровня [5–6].

2. MDA

В настоящее время для создания ПО, служащего для управления различными устройствами, все более широкое применение получила архитектура, управляемая моделью (Model-Driven Architecture, MDA), – методология, созданная с целью решения проблем разработки [7–8].

При применении данной методологии этап проектирования логического аспекта системы обособляется от его реализации. Модель системы проектируется без оглядки на технические особенности реализации системы: язык программирования и особенности платформы. В основе методологии лежит объектно-ориентированный язык моделирования UML. С помощью построения средствами языка визуальных моделей (диаграмм) происходит полное описание проектируемой системы, ее логика, структура и поведение [9–10].

При применении MDA в первую очередь создается платформо-независимая модель системы (Platform independent model, PIM). В ее диаграммах отражаются все требования к системе, ее элементы, алгоритмы работы и интерфейсы без конкретных примеров физической реализации и языков программирования [11].

Далее производится трансформация PIM в платформо-зависимую модель (Platform-specific model, PSM), уже с конкретными деталями реализации, языком и платформой. Далее с помощью имеющихся в настоящее время программных средств происходит автоматическая генерация программного кода. При этом модель и код всегда непосредственно связаны друг с другом, и изменение модели влечет моментальное изменение кода.

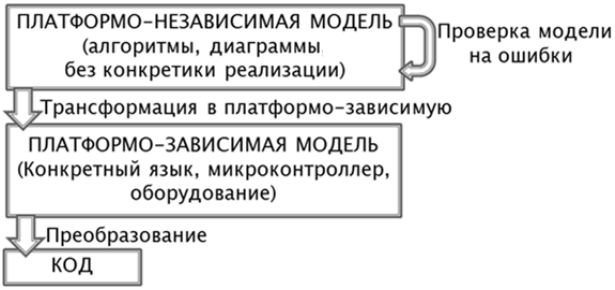

Рис. 1. Этапы разработки с применением MDA

3. ПРЕИМУЩЕСТВА MDA

Рассмотрим преимущества методологии для создания ПО системы автоматизированного управления аквариумом. Для аквариума заданного размера с конкретным видом рыб определенным микроконтроллером для реализации программного управления требуется проработать логику системы, написать и отладить программу управления на определенном языке программирования. Если возникнет потребность в реализации управления аквариумом с другими параметрами, с другими исполнительными устройствами, то вся разработка ПО начинается заново. А из единожды созданной с помощью языка UML платформо-независимой модели можно автоматически генерировать платформо-зависимый код приложения на необходимом языке и для необходимого контроллера. И так как код и модель связаны друг с другом, можно быстро изменять части системы, например, добавляя и удаляя необходимое аквариумное оборудование, а также проводить тестирование непосредственно во время разработки.

Достоинства подхода MDA

1. Хорошее понимание предметной области за счет полного погружения в разработку логического аспекта системы, а не кода.

2. Доступность и наглядность UML диаграмм, создание которых не требует профессиональных навыков программирования. При этом диаграммы являются

понятными для заказчика, что упрощает его взаимодействие с проектировщиком, позволяющее избежать большого числа ошибок проектирования.

3. Возможность повторного использования разработанного ПО на разных платформах реализации, гарантия соответствия систем на разных платформах одной строгой логической модели.

4. Устранение ручного программирования на этапе PIM, за счет этого – повышение производительности и большего внимания к проработке логики.

5. Возможность обеспечения межплатформенной совместимости за счет отношений между разными PSM, преобразованных из одной PIM, благодаря возможности MDA генерировать мосты между ними.

6. Наличие актуальной документации.

4. ПРИМЕНЕНИЕ

Представленный подход был применен для разработки ПО для автоматизированного управления аквариумом. С использованием среды IBM Rational Rhapsody, реализующей подход MDA [12], были созданы модели в диаграммах UML, из которых автоматически генерировался программный код. Далее будут описаны три основных вида диаграмм, используемых в работе.

На рис. 2 приведен фрагмент диаграммы вариантов использования (Use case diagram). С ее помощью отражаются функции, которые должны выполняться элементами системы, учитываемые при формировании требований [13]. Так, например, устройства (Devices) должны создавать управляющие воздействия для слива и налива воды, подачи корма и т. д.

Рис. 2. Диаграмма вариантов использования

На рис. 3 приведен фрагмент диаграммы классов. Она демонстрирует классы системы, понятие которых используется в объектно-ориентированном программировании, их атрибуты и связи [14]. Так, для класса «Кормушка»

(Feeder), родительским классом которого является класс «Устройства» (Devices), имеет атрибуты FeederID для обозначения его номера в системе и FeederPower для обозначения статуса включения. Возможные события: включен, выключен и «выдать порцию корма сейчас».

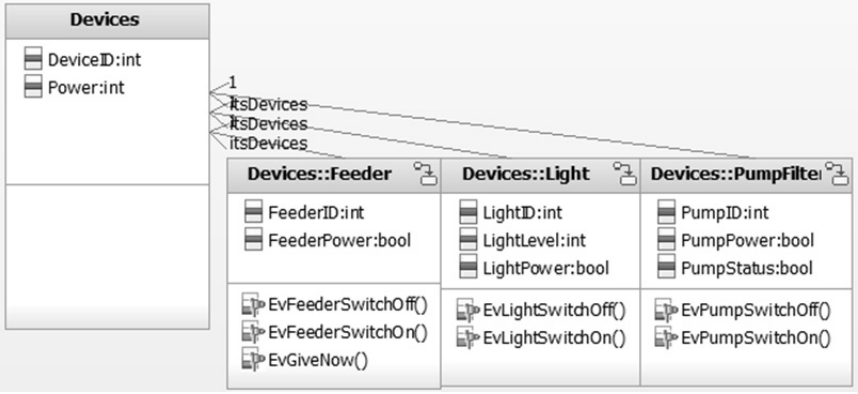

Рис. 3. Диаграмма классов

Поведение каждого класса описывается соответствующей диаграммой состояний (Statechart diagram) [15]. В ней отражаются все возможные состояния системы при выполнении какой-либо операции или при каком-либо условии. В качестве примера на рис. 4 приведена диаграмма состояний для класса «Датчик температуры». Начальным состоянием датчика является простой (Idle), каждые 200 миллисекунд датчик переходит в состояние измерения и в зависимости от результатов инициирует соответствующее событие.

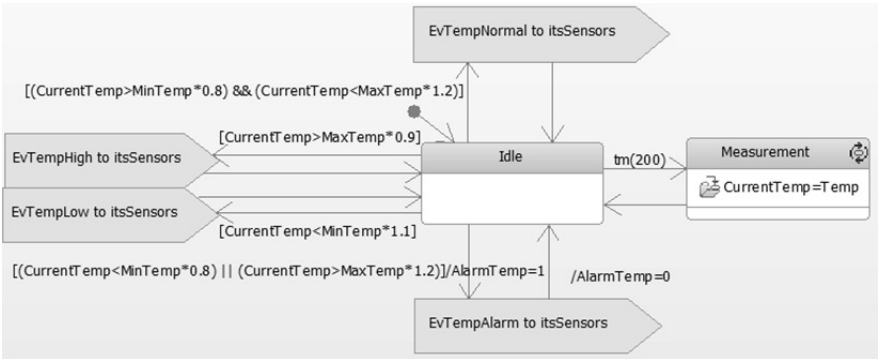

Рис. 4. Диаграмма состояний для датчика температуры

Как уже было сказано, при подходе MDA код генерируется из диаграмм автоматически. В проекте управления аквариумом программный код высокого качества генерировался на языке C++.

Модель, созданная с применением MDA, является наглядной и легко модернизируемой. Избавление от этапа ручного написания кода позволило углубиться в проработку логического аспекта системы. Платформо-независимую модель можно применять для создания систем автоматизированного управления аквариумами различных конфигураций оборудования, размеров и назначений.

ЗАКЛЮЧЕНИЕ

В данной статье описана методология создания программного обеспечения, позволяющая решить основные проблемы, приводящие к ошибкам и увеличивающие сроки проектирования.

Легко отметить, что понятие разработчика программного обеспечения может сильно видоизмениться с внедрением MDA. Со смещением акцента на создание модели разработкой ПО смогут заниматься не только программисты, но и специалисты предметных областей.

СПИСОК ЛИТЕРАТУРЫ

1. *Безуглый Д.* Технология разработки программного обеспечения [Электронный ресурс] // Корпоративные системы. – 2002. – № 1. – URL: http://www.interface.ru/rational/teh.htm (дата обращения: 23.06.2016).

2. *Лапыгин Д., Новичков А.* Управление конфигурацией и изменениями: RUP или ITIL? [Электронный ресурс] // Открытые системы. СУБД. – 2005. – № 2. – URL: http://www.osp.ru/os/2005/02/185318/ (дата обращения: 23.06.2016).

3. *Элгебели А.Р.* Проблемы качества программного обеспечения и практические рекомендации [Электронный ресурс] // IBM DeveloperWorks: web-сайт. – 2013. – URL: https://www.ibm.com/developerworks/ru/library/r-software-quality-challenges-practice-recommendations/ (дата обращения: 23.06.2016).

4. *Шикуть А.В.* К вопросу о переносимости кода и некоторых возможностях использования кроссплатформенного программного обеспечения // Инженерный журнал: наука и инновации. – 2013. – № 6.

5. *Вендров А.М.* Современные технологии создания программного обеспечения // Jet Info. – 2004. – № 4. – С. 32.

6. *Себеста Р.У.* Основные концепции языков программирования. – 5-е изд. – М.: Вильямс, 2001. – 672 с. – ISBN 5-8459-0192-8.

7. *Кузнецов М.* MDA – новая концепция интеграции приложений место в ИТ [Электронный ресурс] // Открытые системы. СУБД. – 2003. – № 9. – URL: http://www.osp.ru/os/2003/09/183391/ (дата обращения: 23.06.2016).

8. *Грибачев К.Г.* Delphi и Model Driven Architecture. Разработка приложений баз данных. – СПб.: Питер, 2004. – 348 с.

9. *Колин Ю.* Применение Rational Software Architect в разработке, управляемой моделями: Часть 1. Обзор парадигмы разработок, управляемых моделями с шаблонами [Электронный ресурс] // IBM DeveloperWorks: web-сайт. – 2007. – URL: http://www.ibm.com/developerworks/ru/library/1121_yu/ (дата обращения: 23.06.2016).

10. *Рамбо Дж., Блаха М.* UML 2.0. Объектно-ориентированное моделирование и разработка. – СПб.: Питер, 2007. – 544 с.

11. MDD. Разработка, управляемая моделями, и ее место в ИТ [Электронный ресурс] // IBM developerWorks: web-сайт. – 2007. – URL: http://www.ibm.com/developerworks/ru/library/mdd/ch6/ch6.html (дата обращения: 23.06.2016).

12. *Boldt R.* Model-driven architecture, embedded developers and IBM Rational Rhapsody [Электронный ресурс]. – 2009. – URL: ftp://public.dhe.ibm.com/software/emea/de/rational/neu/Model_driven_architecture_EN_2009.pdf (дата обращения: 23.06.2016).

13. *Леоненков А.В.* Нотация и семантика языка UML. – М.: Интуит, 2016. – 205 с. – ISBN 5-94774-408-2.

14. *Буч Г., Рамбо Дж., Якобс А.* Язык UML. Руководство пользователя. – М.: ДМК Пресс, 2007. – 496 с. – ISBN 5-94074-334-X. – ISBN 0-321-26797-4.

15. *Леоненков А*. Самоучитель UML. – СПб.: БХВ-Петербург, 2004. – 418 с. – ISBN 5-94157-342-1.

Назаров Игорь Романович, магистрант кафедры автоматики Новосибирского государственного технического университета по направлению «Управление в технических системах». E-mail: igorfresh@mail.ru

Аникин Антон Викторович, магистрант кафедры автоматики Новосибирского государственного технического университета по направлению «Управление в технических системах». E-mail: anikantonsd@yandex.ru

Application of software development technology «Model driven architecture»*

I.R. Nazarov[1], A.V. Anikin[2]

[1] *630073, Russia, Novosibirsk, PR. Karla Marksa, 20, Novosibirsk state technical University, undergraduate of department of automation. E-mail: igorfresh@mail.ru*
[2] *630073, Russia, Novosibirsk, PR. Karla Marksa, 20, Novosibirsk state technical University, undergraduate of department of automation. E-mail: anikantonsd@yandex.ru @mail.ru*

At the present time devices and systems contain a large number of software driven components and modules. The complexity of the system is increasing, creating a need for quality software, which development traditionally accompanied with various problems. Developers are in search of the more optimal methods of development that can improve software quality and reduce the time of its design. This article describes a software development methodology, created to solve some of existing problems of system design.

Keywords: software development, modeling, model, UML, MDA, object-oriented programming, CASE, platform

DOI: 10.17212/2307-6879-2016-2-107-115

REFERENCES

1. Bezuglyi D. Tekhnologiya razrabotki programmnogo obespecheniya [Software engineering technology]. *Korporativnye sistemy*, 2002, no. 1. Available at: http://www.interface.ru/rational/teh.htm (accessed 23.06.2016)

2. Lapygin D., Novichkov A. Upravlenie konfiguratsiei i izmeneniyami: RUP ili ITIL? [Configuration and change management: RUP or ITIL?]. *Otkrytye sistemy. SUBD – Open Systems. DBMS*, 2005, no. 2. Available at: http://www.osp.ru/os/2005/02/185318/ (accessed 23.06.2016)

3. Elgebeely A.R. Problemy kachestva programmnogo obespecheniya i prakticheskie rekomendatsii [Software quality challenges and practice recommendations]. *IBM DeveloperWorks*: website. 2013. (In Russian) Available at: https://www.ibm.com/developerworks/ru/library/r-software-quality-challenges-practice-recommendations/ (accessed 23.06.2016)

4. Shikut' A.V. K voprosu o perenosimosti koda i nekotorykh vozmozhnostyakh ispol'zovaniya krossplatformennogo programmnogo obespecheniya [On issue of code portability and some possibilities of using cross-platform software]. *Inzhenernyi zhurnal: nauka i innovatsii – Engineering Journal: Sciense and Innovation*, 2013, no. 6.

* *Received 26 January 2016.*

5. Vendrov A.M. Sovremennye tekhnologii sozdaniya programmnogo obespecheniya [Modern technologies of software engineering]. *Jet Info*, 2004, no. 4, p. 32.

6. Sebesta R.W. *Concepts of programming languages*. 5th ed. Boston, Addison Wesley, 2001. 720 p. (Russ. ed.: Sebesta R.U. *Osnovnye kontseptsii yazykov programmirovaniya*. 5th ed. Moscow, Williams Publ., 2001. 672 p. ISBN 5-8459-0192-8).

7. Kuznetsov M. MDA – novaya kontseptsiya integratsii prilozhenii [A new concept of application integration]. *Otkrytye sistemy. SUBD – Open Systems. DBMS*, 2003, no. 9. Available at: http://www.osp.ru/os/2003/09/183391/ (accessed 23.06.2016)

8. Gribachev K.G. *Delphi i Model Driven Architecture. Razrabotka prilozhenii baz dannykh* [Delphi and Model Driven Architecture. Development of database applications]. St. Petersburg, Piter Publ., 2004. 348 p.

9. Kolin Yu. Primenenie Rational Software Architect v razrabotke, upravlyaemoi modelyami: Chast' 1. Obzor paradigmy razrabotok, upravlyaemykh modelyami s shablonami [Using of Rational Software Architect with MDA. Part 1. Overview of MDA with templates]. *IBM DeveloperWorks*: website. 2007. Available at: http://www.ibm.com/developerworks/ru/library/1121_yu/ (accessed 23.06.2016)

10. Rumbaugh J., Blaha M. *Object-oriented modeling and design with UML*. London, UK, Pearson, 2004. 496 p. (Russ. ed.: Rambo Dzh., Blaha M. *UML 2.0. Ob"ektno-orientirovannoe modelirovanie i razrabotka.* St. Petersburg, Piter Publ., 2007. 544 p.).

11. MDD. Razrabotka, upravlyaemaya modelyami, i ee mesto v IT [Model-driven development and its place in IT]. *IBM DeveloperWorks*: website. 2007. Available at: http://www.ibm.com/developerworks/ru/library/mdd/ch6/ch6.html (accessed 23.06.2016)

12. *Boldt R.* Model-driven architecture, embedded developers and IBM Rational Rhapsody. 2009. Available at: ftp://public.dhe.ibm.com/software/emea/de/rational/neu/Model_driven_architecture_EN_2009.pdf (accessed 23.06.2016)

13. Leonenkov A.V. *Notatsiya i semantika yazyka UML* [The notation and semantics of the UML]. Moscow, Intuit Publ., 2016. 205 p. ISBN 5-94774-408-2

14. Booch G., Jacobson I., Rumbaugh J. *The Unified Modeling Language user guide.* Boston, USA, Addison Wesley, 1998. 512 p. (Russ. ed.: Buch G., Rambo D. Dzhekobson A. *Yazyk UML. Rukovodstvo pol'zovatelya.* Moscow, DMK Press Publ., 2007. 496 p. ISBN 5-94074-334-X. ISBN 0-321-26797-4).

15. Leonenkov A. *Samouchitel' UML* [UML self-teacher]. St. Petersburg, BHV-Petersburg Publ., 2004. 418 p. ISBN 5-94157-342-1