УДК 681.326/.069: 001.13 (045/046)

Б. Т. УКУЕВ, К. МУКУДИН УУЛУ
B. T. UKUEV, K. MUKUDINUULU

E.mail. ksucta@elcat.kg

МЕТОДЫ И ТЕХНОЛОГИИ ПРИМЕНЕНИЯ UML – МОДЕЛЕЙ НА ОСНОВЕ КОМПОЗИЦИИ И ДЕКОМПОЗИЦИИ ДЛЯ РЕШЕНИЯ ИНЖЕНЕРНЫХ ЗАДАЧ

METHODS AND APPLICATION TECHNOLOGIES UML - MODEL BASED ON THE COMPOSITION AND DECOMPOSITION TO SOLVE ENGINEERING PROBLEMS

Бул макалада UML – моделдеринин өзгөчө бөлүктөрүнө карата композиция жана декомпозиция методдорунун колдонулушуна негизделген жана аткарылган UML – моделдерин трансформациялоо ыкмалары каралган. Бул ыкмалар модель түшүнүктөрүн жеңилдетет жана аны менен иштөөнү жөнөкөйлөтөт. Мындан тышкары трансформациянын жаңы түрлөрү сунушталып UML – моделдеринин ар кандай элементтерин өзгөртүп түзүүдөгү колдонулуулар каралган.

Ачкыч сөздөр: *трансформация, модель, композиция, декомпозиция, моделдөөнүн унифицирленген тили, рефакторинг.*

В данной статье рассматриваются способы трансформации исполняемых UML – моделей, которые основаны на композиции или декомпозиции отдельных частей модели и позволяют облегчить понимание модели и упростить работу с ней; предлагаются и обсуждаются новые виды таких трансформаций, иллюстрируется применение преобразований к различным элементам UML – моделей.

Ключевые слова:*трансформация, модель, композиция, декомпозиция, унифицированный язык моделирования, рефакторинг.*

This article describes the methods of transformation executable UML – models? Which are based on the composition or decomposition of the individual parts of the model and allow to facilitate understanding of the model and to simplify the work with it; proposed and discussed new types of such transformations, illustrated by the use of changes to various elements of UML – models.

Keywords: *transformation, model, composition, decomposition, Unified Modeling Language, refactoring.*

Выразительных средств языка UML в совокупности с мощными механизмами расширения достаточно для того, чтобы описать любую программную систему со всех точек зрения, актуальных на различных этапах жизненного цикла. Более того, все шире становится область применения концепций модельно – ориентированной разработки (MDD – ModelDriveDevelopment), отводящих моделям главную роль в процессе создания и поддержки системы.

Объектно-ориентированный подход использует объектную декомпозицию. При этом статическая структура системы описывается в терминах объектов и связей между ними, а поведение системы описывается в терминах обмена сообщениями между объектами. Каждый объект системы обладает своим собственным поведением, моделирующим поведение объекта реального мира. Концептуальной основой объектно-ориентированного подхода является объектная модель. Основными элементами объектно-ориентированного подхода являются:

- абстрагирование (abstraction);
- инкапсуляция (encapsulation);

• модульность (modularity);
• иерархия (hierarchy).

Кроме основных имеются еще три дополнительных элемента, не являющихся в отличие от основных строго обязательными:

• типизация (typing)',
• параллелизм (concurrency)',
• устойчивость (persistence).

Абстрагирование - это выделение существенных характеристик некоторого объекта, которые отличают его от всех других видов объектов и, таким образом, четко определяют его концептуальные границы относительно дальнейшего рассмотрения и анализа. Абстрагирование концентрирует внимание на внешних особенностях объекта и позволяет отделить самые существенные особенности его поведения от деталей их реализации. Выбор правильного набора абстракций для заданной предметной области представляет собой главную задачу объектно-ориентированного проектирования.

Инкапсуляция - это процесс отделения друг от друга отдельных элементов объекта, определяющих его устройство и поведение. Инкапсуляция служит для того, чтобы изолировать интерфейс объекта, отражающий его внешнее поведение, от внутренней реализации объекта. Объектный подход предполагает, что собственные ресурсы, которыми могут манипулировать только методы самого класса, скрыты от внешней среды. Абстрагирование и инкапсуляция являются взаимодополняющими операциями: абстрагирование фокусирует внимание на внешних особенностях объекта, а инкапсуляция (или, иначе, ограничение доступа) не позволяет объектам-пользователям различать внутреннее устройство объекта.

Модульность - это свойство системы, связанное с возможностью ее декомпозиции на ряд внутренне связных, но слабо связанных между собой модулей. Инкапсуляция и модульность создают барьеры между абстракциями.

Иерархия - это ранжированная или упорядоченная система абстракций, расположение их по уровням. Основными видами иерархических структур применительно к сложным системам являются структура классов (иерархия по номенклатуре) и структура объектов (иерархия по составу). Примерами иерархии классов являются простое и множественное наследование (один класс использует структурную или функциональную часть соответственно одного или нескольких других классов), а иерархии объектов - агрегация.

Типизация - это ограничение, накладываемое на класс объектов и препятствующее взаимозаменяемости различных классов (или сильно сужающее ее возможность). Типизация позволяет защититься от использования объектов одного класса вместо другого или по крайней мере управлять таким использованием.

Параллелизм - свойство объектов находиться в активном или пассивном состоянии и различать активные и пассивные объекты между собой.

Устойчивость - свойство объекта существовать во времени (вне зависимости от процесса, породившего данный объект) и/или в пространстве (при перемещении объекта из адресного пространства, в котором он был создан).

В рамках модельно – ориентированного подхода собственно разработка сводится к последовательному уточнению модели системы, начинающемуся с абстрактной модели и заканчивающемуся готовой программной системой. Таким образом, при использовании MDD сложность создаваемых моделей будет напрямую отражать сложность системы, а работа с моделью может создавать такие же трудности, как те, которые возникают при чтении и понимании большого количества исходного кода на традиционных языках программирования.

Данное свойство сближает такие методы трансформации моделей с известными приемами рефакторинга объектно–ориентированного программного обеспечения. Однако графическая природа языка UML накладывает свой отпечаток на возникающие проблемы и методы их решения. Кроме того, языкUML продолжает развиваться, включая в себя множество новых элементов, многие из которых могут быть использованы для упрощения восприятия моделей.

Суть предлагаемого подхода состоит в том, что исходная модель сложной системы посредством описываемых преобразований трансформируется и дополняется элементами, образуя иерархическое представление, удобное как в случаях, когда нужно получить представление об общих принципах работы системы, так и в случаях, когда необходим детальный анализ конкретных аспектов. Такие преобразования могут быть классифицированы как методы композиции и декомпозиции автоматных моделей.

Под композицией в данной работе понимается такое преобразование модели, результатом которого является создание единой новой целостной сущности из нескольких имеющихся сущностей. Например, объединение в одной диаграмме элементов всех диаграмм, описывающих конкретный автомат, есть примитивное преобразование – композиция. Следует отметить, что такая трансформация является примером графической композиции, так как затрагивает только элементы, описывающие графические сущности модели. Как правило, применение композиции способствует получению целостного представления о каком – либо элементе системы; при этом должны быть приняты меры, предотвращающие чрезмерное усложнение создаваемого элемента. На рисунке [Рис.1.] показано пример композиции одной системы. Элементы А и В объединившись в один конечный автомат создают одну систему.

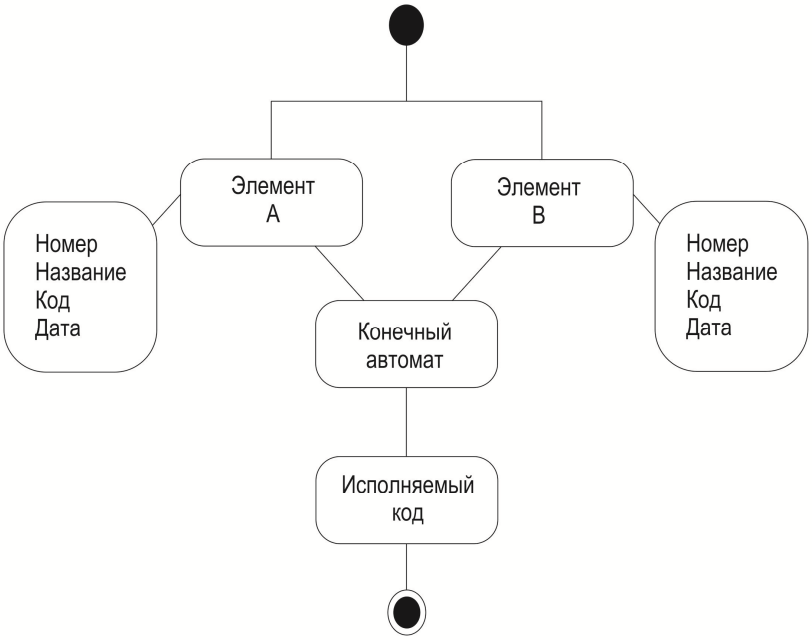

Рис. 1. Пример метода композиции

Под декомпозицией следует понимать преобразование, обратное композиции, то есть такое преобразование, результатом которого является создание нескольких новых сущностей на основе единственной исходной сущности. Для приведенного примера графической композиции обратным будет преобразование, в результате которого элементы одной диаграммы распределяются между несколькими другими диаграммами. Метод декомпозиции системы, то есть разбиение системы на отдельные подсистемы раскрывается на рисунке 2.

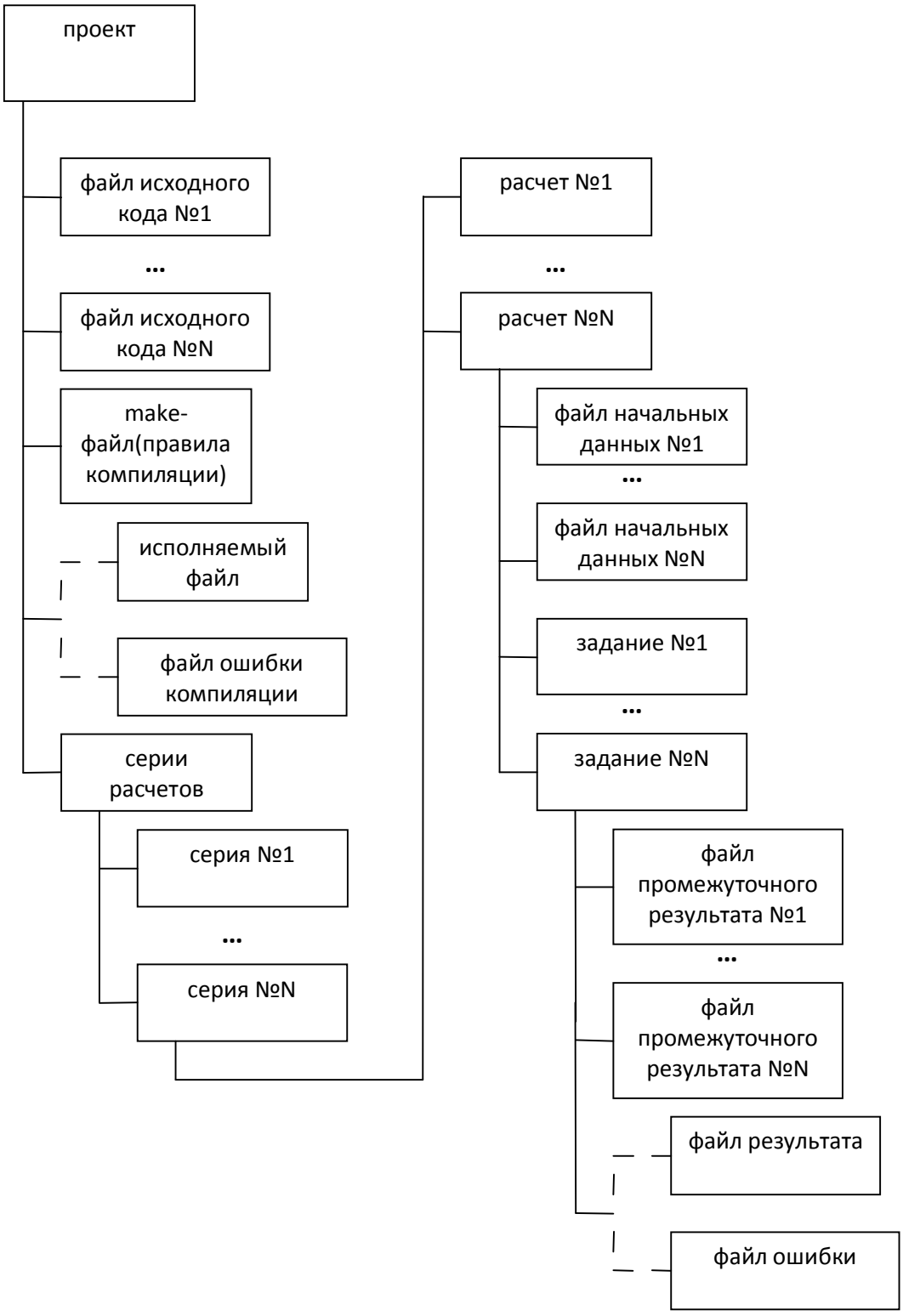

Рис. 2. Структура декомпозиции пользовательской системы

Применяя композицию и декомпозицию, можно построить иерархическое описание сложных элементов системы, облегчающее ее восприятие. При этом понимание работы системы будет происходить сверху вниз по иерархии описания. С помощью иерархической структуры системы можно организовать доступ к необходимым материалам с помощью кластерной системы.

В настоящее время вычислительные системы кластерного типа получают все большее распространение. Данная тенденция обусловлена различными факторами, главным из которых является насущная потребность в решении актуальных задач фундаментальной и прикладной науки, для анализа исследования которых

производительности существующих средств вычислительной техники оказывается недостаточно.

С аппаратной точки зрения кластер представляет собой набор серийно выпускаемых вычислительных узлов (электронно-вычилительных машин), объединенных сетью передачи данных. С точки зрения программного обеспечения кластер является единой виртуальной машиной, имеющей несколько процессоров и распределенную память. Производительность кластера определяется производительностью его узлов и характеристиками среды передачи данных. Сначала пользователь знакомится с основными компонентами описываемой части модели и принципами их взаимодействия, и далее по необходимости углубляется, переходя к более детальному описанию интересующего его компонента. На практике даже для достаточно сложных конечных автоматов редко требуется больше 4 – 5 уровней детализации.

Заключение

В статье были предложены новые методы трансформации исполняемых моделей на языке UML, основанных на композиции и декомпозиции. Была продемонстрирована практическая применимость предлагаемых методов как средства упрощения восприятия и улучшения структуры сложных автоматов.

Каждый из предложенных методов допускает как автоматизацию выполнения преобразования, так и автоматический поиск частей модели, к которым это преобразование применимо и вероятно, его применение целесообразно. Приоритетными направлениями дальнейшей работы являются реализация модуля автоматического поиска и применения трансформаций в среде моделирования, наряду с продолжением работ по разработке и исследованию новых методов трансформации.

Список литературы

1. Укуев Б.Т. Теория и методы моделирования управленческих и инженерных задач на базе новых информационных технологий [Текст] / Б.Т.Укуев. - Бишкек: Б.и., 2014.– 220с.
2. Рефакторинг: улучшение существующего кода[Текст] / М. Фаулер, К. Бек,Д. Брант и др. –СПб.: Символ – Плюс, 2002. -432с.
3. Укуев Б.Т. Разработка моделей автоматизации инженерных задач на основе объектно-ориентированного подхода[Текст] / Б.Т.Укуев // Материалы IX-гоМеждунар. симп.: «Фундаментальные и прикладные проблемы науки».– М., 2014.–Т. 7.– С.41-48.
4. Волкова Е.Д., Страбыкин А.Д. Анализ и трансформации исполняемых UML – моделей[Текст] / Е.Д. Волкова,А.Д. Страбыкин // Труды Института системного программирования. – М.: ИСП РАН, 2006. -С. 171 – 192.