МОДЕЛИРОВАНИЕ КЛАССИФИКАЦИЙ НА ВИЗУАЛЬНОМ ЯЗЫКЕ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ UML SP

В.И. Гурьянов (Чебоксары)

В работе [1] представлена спецификация профиля UML, предназначенного для объектно-ориентированного имитационного моделирования (далее UML SP). Основная идея этого языка заключается в том, чтобы стереотипам профиля назначать двойственную семантику – предметную и вычислительную. Тем самым каждому концепту предметной области однозначно соответствует некоторая программная сущность. Метамодель профиля [1] составлена на основе обобщения опыта разработки имитационных моделей. При таком подходе метамодель неизбежно будет фрагментарной и неполной, поэтому актуальным остается вопрос обоснования выбора множества стереотипов UML SP. Концепты стереотипов можно организовать в онтологию (обзор методологий и методов построения онтологий [2]), что позволяет выделить четыре группы (или категории) стереотипов: структура, классификация, активность, коммуникация. Наибольшую сложность при построении модели анализа вызывает определение классов категории «классификация». В данной работе рассматривается один из подходов к теоретическому обоснованию подбора стереотипов категории «классификация» и предлагается метод выделения классов, помечаемых стереотипами этой категории. Примеры приведены на языке C++.

Стереотипы категории «классификация». С точки зрения вычислительной семантики, необходимость стереотипа «*Ontology Entity*» (пакет, группирующий сущности категории «классификация») не вызывает сомнений. В имитационных исследованиях, как правило, приходится работать не с одной имитационной моделью, а с несколькими вариантами одной и той же модели. Пакет «*Ontology Entity*» содержит повторно используемые компоненты программных систем, образующих линейку программного обеспечения имитационной модели.

Сложнее обстоит дело с обоснованием предметной семантики. Для дальнейшего уточнения онтологии воспользуемся методом «микротеорий» и обратимся к теории классификации, основанной на принципе двойственности таксономии и мерономии. Этот подход развит в работах С.В. Мейена и Ю.А.Шрейдера и в наиболее законченном виде изложен в книге [3]. Математическую основу теории составляет теория категорий. Множество таксонов является категорией C, при этом объекты категории – таксоны T, а морфизмы – отображения вложения таксонов. Архитипы также образуют категорию R, при которой в качестве морфизмов выступают гомоморфизмы. Классификация – это функтор $\Im: C \rightarrow R$ из категории таксонов в категорию архитипов.

Таким образом, данная теория классификации предполагает пять концептов верхнего уровня: классификация, таксономия, мерономия, таксон, мерон. Для определения стереотипов необходимо подобрать подобные элементы из метамодели UML (вер. 1.5). Основным ограничением будет требование соответствия стереотипов UML SP стереотипам UP, поскольку в противном случае мы получим противоречивые ситуации при разработке программной системы [1]. Мы предлагаем следующий набор элементов.

Ontology Entity «metaClass» Package (из Model Management). *Предметная семантика*: теория объекта исследования. *Вычислительная семантика*: повторно используемые классы. *Помеченные значения*: Concept = Наименование теории. *Ограничения*: используется пакетом «*World*».

Taxonomy «metaClass» Package (из Model Management). *Предметная семантика*: таксономия. *Вычислительная семантика*: группировочная сущность для абстрактных

классов, объявляющих интерфейс. *Помеченные значения*: Concept = Название таксономии. *Ограничения*: содержит классы «*Ontology Category*».

Ontology Category «metaClass» Class (из Core). *Предметная семантика*: таксон. Классы, помеченные стереотипом *Ontology Category*, определяются как абстрактные и, значит, они не могут иметь экземпляры; этот стереотип моделирует множество видов. *Вычислительная семантика*: абстрактные классы. *Помеченные значения*: Concept = Наименование таксона. *Ограничения*: входят только в пакет «*Taxonomy*» .

Meronomy «metaClass» Package (из Model Management). *Предметная семантика*: мерономия. *Вычислительная семантика*: классы, определяющие типы пользователя. *Помеченные значения*: Concept = Наименование мерономии. *Ограничения*: содержит только классы «*Ontology Space*».

Ontology Space «metaClass» Class (из Core). *Предметная семантика*: мерон, ячейка конфигурационного пространства системы. *Вычислительная семантика*: пользовательский тип. *Помеченные значения*: Concept = Наименование мерона. *Ограничения*: используется для определения композиции классов пакета «*Taxonomy*»; включаются только в пакет «*Meronomy*».

Categorization «metaClass» Dependence (из Core). *Предметная семантика*: функтор; поставляет таксону соответствующий архитип. *Вычислительная семантика*: зависимость, отражающая использование классов пакета «*Meronomy*» в классах пакета «*Taxonomy*». *Помеченные значения*: Concept = Наименование классификации; наименование функтора; принцип сопоставления. *Ограничения*: применяется только к пакетам «*Meronomy*» и «*Taxonomy*»; пакет «*Taxonomy*» зависит от пакета «*Meronomy*».

Спецификация [1] представляет, так сказать, «облегченную» версию UML SP и содержит определение только стереотипов *Ontology Entity*, *Substance* и *Ontology Space*.

Применение стереотипов. При таком выборе стереотипов отношение обобщения между классами «*Ontology Category*» можно рассматривать как метафору отношения включения между таксонами, а множественное наследование – как отношение пересечения. Таксономические структуры и архитипы не входят в метамодель UML SP, они создаются в процессе разработки модели. Пользовательские типы (классы «*Ontology Space*») определяют поля классов «*Ontology Category*»; либо непосредственно, либо в составе структур данных.

Теория классификации Мейена–Шрейдера позволяет не только определить стереотипы категории «классификация», но и предоставляет некоторые методы для построения «*Research Analysis Model*». Для выделения классов категории «классификация» предлагается использовать паттерн *Bridge* [4]. Действительно, интерфейс определяет внешние признаки классифицируемых объектов, а реализация – их строение. По определению, паттерн *Bridge* отделяет абстракцию (интерфейс) от реализации и минимизирует связность пакетов «*Taxonomy*» и «*Meronomy*». В ассоциации классов прежде всего необходимо найти класс, который может играть роль *Abstraction*. Этот класс должен быть абстрактным и объявлять интерфейс. Кроме того, класс должен иметь поле, которое содержит указатель на абстрактный класс, играющий роль *Implementor*. После чего классы раскладываются по пакетам «*Taxonomy*» и «*Meronomy*». Природа, конечно, не обязана быть устроена так, как того требует паттерн *Bridge*. Поэтому данный метод следует рассматривать как один из возможных подходов к выделению классов.

Пример 1. Наиболее распространенная система классификации – иерархическая. Пусть необходимо построить имитационную модель автобусного рейса с автовокзала некоторого города. Расписание движения автобусов составляет классификационную систему для данной задачи. Мы допустим, что с данного автовокзала выполняются рейсы по одному междугороднему маршруту и двум пригородным маршрутам. Диаграмма классов для имитационной модели приведена на рис. 1.

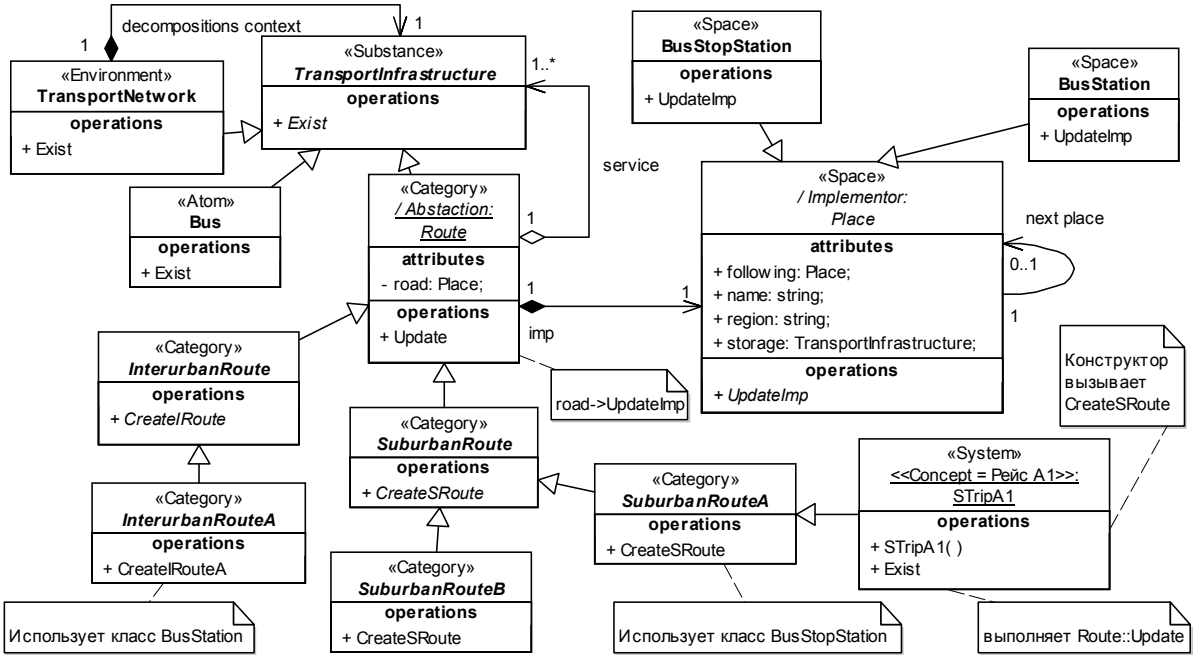

Рис. 1. Имитационная модель «Автобусный рейс»

Конкретный рейс моделируется экземплярами класса STripA1, который выполняется по маршруту SuburbanRouteA. Метод Exist задает единицу дискретно-событийного времени; моделирует посадку/высадку пассажиров в данном населенном пункте (операция Update) и перемещение объекта класса Bus в следующий населенный пункт. Маршрут моделируется динамическим списком из экземпляров подклассов класса Place. Классы BusStopStation и BusStation различаются операцией UpdateImp – в первом случае обновляется только часть пассажиров, во втором – все.

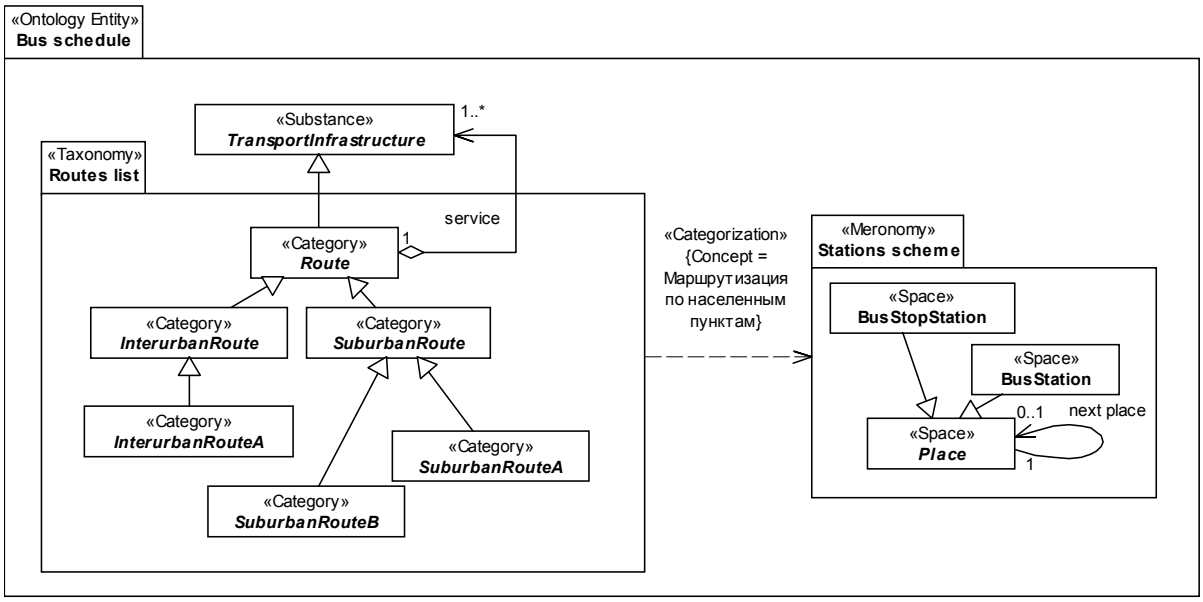

Рис. 2. Пример модели иерархической системы классификации

Воспользуемся паттерном *Bridge*. На роль *Abstraction* подходит класс Route, тогда класс Place будет играть роль *Implementor* рис.1. Предполагая типовую двухслойную архитектуру, разложим классы по основным пакетам модели анализа. Классы Transport№etwork, Bus и STripA1 поместим в пакет «*World*», остальные классы разместим в пакете «*Ontology Entity*» так, как показано на рис. 2. Заметим, что на рис. 2 нет необходимости указывать стереотипы классов, поскольку пакеты и группируют классы по категориям. Мы приводим стереотипы для большей наглядности. После того как выполнено разделение классов по пакетам, становится возможным определение их стереотипов рис.1. Из рис. 2 видно, что классификация зависит от выбора классов пакета «*Meronomy*», т.е. от способа определения функтора. Например, в нашем случае в качестве альтернативной мерономии можно было бы в качестве миронов выбрать дороги, через которые проходит маршрут. Таксономия не изменилась бы, но классификация была бы уже другой.

Пример 2. В качестве иллюстрации фасетной системы классификации рассмотрим объектную имитационную модель химического элемента. Напомним, что химические свойства атомов определяются строением внешней электронной оболочки и их можно проследить по изменению высшей валентности. Высшая валентность определяется количеством электронов, которые может принять атом в молекулах с ковалентной связью, и связана с группой периодической таблицы Д.И.Менделеева. Номер периода определяется количеством электронных оболочек. Таким образом, имитационная модель должна моделировать процесс присоединения валентного электрона и процессы перехода электронов между оболочками. Атом химического элемента моделируется конкретным классом ChemicalElementAtom, который является подклассом абстрактного класса ChemicalElement. Конструктор класса принимает параметр z – порядковый номер химического элемента. Класс определяет метод join, который моделирует процесс присоединения валентного электрона, и объявлен в абстрактном классе CEGrupe. Свойство characteristic_spectrum объявлено в классе CEPeriod и моделирует характеристический спектр химического элемента. Класс ChemicalElement получается из класса CEGrupe в результате подмешивания класса CEPeriod. Мы ограничимся чисто фасетной структурой, рассматривая только малые периоды периодической системы химических элементов.

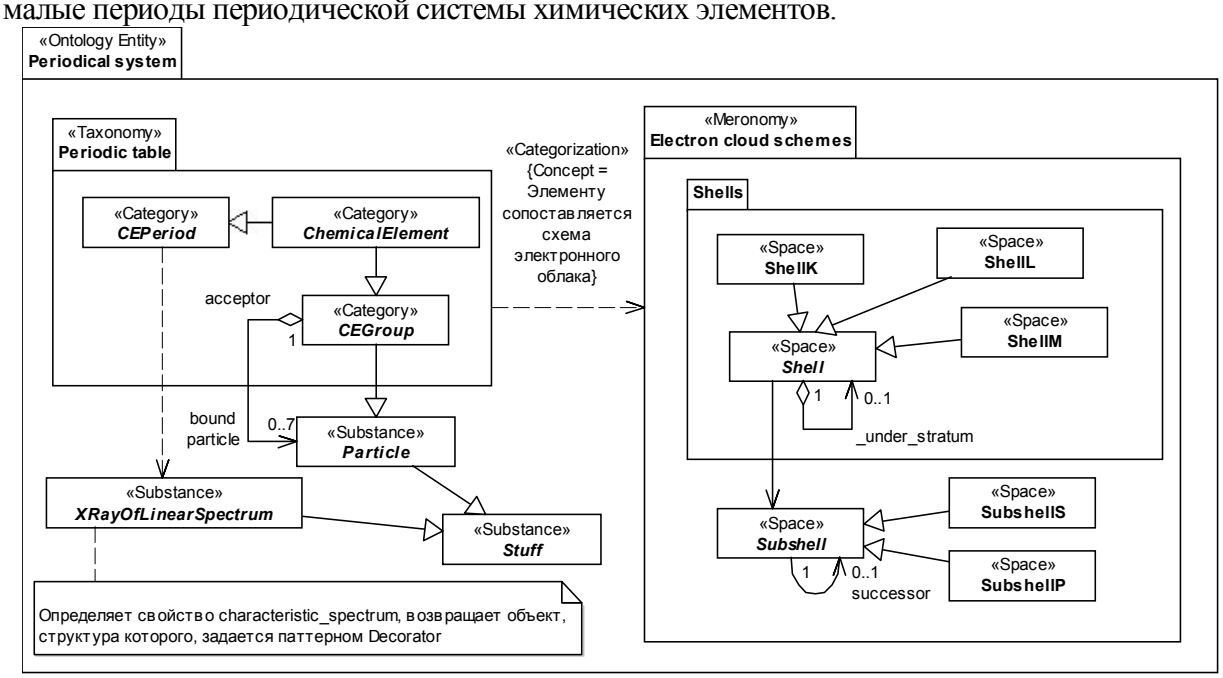

Рис. 3. Пример модели фасетной системы классификации

Рассмотрим ассоциацию классов как паттерн *Bridge*. На роль *Abstraction* выберем класс CEGroup. Указатель на объект типа *Implementor* хранится в поле _electron_cloud и

имеет класс Shell. Подклассы Shell моделируют три первые электронные оболочки химических элементов и сами состоят из подуровней. Для того чтобы оболочка заполнялась в соответствии с правилами Хунда, для создания композиции этих классов использован паттерн *Chain of Responsibility* [4]. В пакет «*Ontology Entity*» поместим часть диаграммы классов так, как это показано на рис. 3.

В заключение сделаем следующее существенное замечание. С точки зрения вычислительной семантики, классы «*Substance*» – это абстрактные классы архитектурного паттерна. С точки зрения предметной семантики, концепт этого стереотипа может быть определен посредством такого математического понятия, как *каркас* [3], а также близкого понятия «системы, нарисованные на системах» В.А. Лефевра). Классы «*Substance*» могут образовывать иерархию наследования, что моделируется вложенными пакетами. В целом, если обратиться к семиотике, пакет «*Ontology Entity*» есть описание некоторой теории в знаках UML.

Выводы. Язык имитационного моделирования UML SP построен на основе обобщения опыта разработки имитационных моделей. В данной статье рассмотрена теория классификации Мейена-Шрейдера, которая позволяет обосновать выбор стереотипов категории «классификация». Тем не менее проблема обоснования UML SP по-прежнему остается актуальной. Дальнейшее развитие метамодели UML SP видится на основе следующих подходов. Общеметодологической базой UML SP является понимание информатики как науки об *информационном взаимодействии* [5]. Уточнение стереотипов возможно на основе современного понимания системного подхода, прежде всего, опираясь на методологии и технологии системного моделирования [6], а также работы научной школы С.П. Никанорова (методология концептуального анализа и проектирования). Большое значение имеет опыт создания методологии моделирования, разрабатываемой в СПИИ РАН [7].

Литература

1. Гурьянов В.И. Профиль UML для имитационного моделирования // Объектные системы – 2013: материалы VII Международной науч.-практ. конф. (Ростов-на-Дону, 10–12 мая 2013 г.) / Под общ. ред. П.П. Олейника. – Ростов-на-Дону: ШИ (ф) ЮРГТУ (НПИ), 2013. – С.39–45. URL: http://objectsystems.ru/files/2013/Object_Systems_2013_Proceedings.pdf (дата обращения: 25.08.2013).

2. Труды Симпозиума «Онтологическое моделирование». г. Звенигород, Московской обл. 19 – 20 мая 2008 г. / Под общ. ред. Л. А. Калиниченко. – М.: ИПИ РАН, 2008. – 303 с.

3. Шрейдер Ю.А., Шаров А.А. Системы и модели**. –** М: Радио и связь, 1982 – 152 с. (Кибернетика)

4. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. Паттерны проектирования. – СПб: Питер, 2010. – 368 с.

5. Юсупов Р. М., Юсупов Ю.В. Состояние и перспективы развития информатики // *Тр. СПИИ РАН*, **5** (2007), – С. 10–45.

6. Аврамчук Е.Ф., Вавилов А.А., Емельянов С.В. и др. Технология системного моделирования / Под общ. Ред. С.В. Емельянова. – М.: Машиностроение; Берлин: Техник, 1988. – 520 с.

7. Иванищев В. В., Марлей В. Е. Введение в теорию алгоритмических сетей. – СПб.: Изд-во СПбГТУ, 2000. – 179 с.