Neurol. 1978. Vol. 181. P. 567-599.

8. Kuenzel W.J., M. Masson A stereotaxic atlas of the brain of the chick (Gallus domesticus). The J. Hopkins University Press, 1988.

9. Воронов Л.Н., Шуканов А.А., Григорьев С.Г. Способ классификации нервных клеток, окрашенных по методу Ниссля. – Приоритет. Изобрет. №2002124688/13 (026086) от 21.07.2003 г.

SPECIFICS OF MORPHOTYPES OF NEURONS OF FIELD SPARROW FOREBRAIN IN A GENDER ASPECT

© 2014

S.G. Grigoriev, Doctor of Biology, Associate Professor of the Department of Economics and Management in Tourism and Social Sphere

N.M. Tabakova, Candidate of Biology, Associate Professor of the Department of Social Ecology nd Ecology Law

A.S. Roshnova, Specialist of the Faculty of Humanities and Natural-Mathematical Disciplines

Branch of Russian State Social University, Chebokasry (Russia)

Annotation: The peculiarities of sexual dimorphism of cytoarchitectonic organization of fields of endbrain of a field sparrow are given. Heterogeneity of the cellular composition of the forebrain of studied bird species depending on gender is set. The female field sparrows are marked relatively by high densities of distribution of neurons, glia, neuroglial complexes, the males, on the contrary, are characterized by reduced performance of these parameters.

Keywords: forebrain, cytoarchitectonics, sexual dimorphism, neuron, glia, neuroglial complex.

УДК 004.94

ПОСТРОЕНИЕ ОБЪЕКТНЫХ МОДЕЛЕЙ ПО МОДЕЛЯМ СИСТЕМНОЙ ДИНАМИКИ ДЖ. ФОРРЕСТЕРА

© 2014

В.И. Гурьянов, кандидат технических наук, доцент кафедры информационных систем и математики

Санкт-Петербургский государственный экономический университет, Чебоксары (Россия)

Аннотация: В статье анализируются основные концепции системной динамики. Предложен минимальный набор описательных конструкций системной динамики. Этот набор представлен на языке объектно-ориентированного моделирования. Приведен пример построения объектной модели по модели системной динамики.

Ключевые слова: имитационное моделирование, системная динамика, объектно-ориентированное моделирование.

Постановка проблемы в общем виде и ее связь с важными научными и практическими задачами. В настоящее время наблюдается возрождение интереса к имитационному моделированию на универсальных языках программирования. Это в значительной степени обусловлено потребностью научных исследований в современных информационных технологиях. В данной статье рассматриваются правила сопоставления основных конструкций системной динамики (СД) и объектно-ориентированного моделирования (ООМ). Применяя данные правила, можно строить объектные модели, опираясь на многочисленные валидные модели системной динамики. Для построения объектных моделей будем использовать язык имитационного моделирования UML SP (В.Гурянов [1]).

Анализ последних исследований и публикаций, в которых рассматривались аспекты этой проблемы и на которых обосновывается автор; выделение неразрешенных раньше частей общей проблемы. По данной проблематике большая часть работ опубликована в первом десятилетии века и почти все они обсуждают вопрос о сопоставление моделей системной динамики с агентными моделями (Д. Каталевский [2], А. Борцов [3]). В работе (А. Борцов [3]) показано, как модель СД может быть конвертирована в агентную модель. В качестве примера рассмотрена модель распространения нового продукта (Bass Diffusion model). В приложении приведено соотношение типичных конструкций СД и соответствующих агентных моделей. Хотя агентное моделирование и ООМ – это разные подходы к имитационному моделированию (они различаются методологиями: моделирование снизу-вверх и сверху-вниз), тем не менее многие агентные решения могут почти без изменений использоваться в ООМ. Приведенные соотношения, к сожалению, недостаточны для создания моделей ООМ, поскольку в них отражен только процессный аспект.

Отдельного внимания заслуживает работа (А. Духанов [4]). В книге рассматривается преобразование системно-динамической модели в код языка программирования типа Pascal или C. Показано, что основная проблема преобразования модели связана с переходом от параллельной композиции процессов к квазипараллельному моделированию. Для решения этой проблемы авторы предлагают специальную методику, основанную на анализе диаграмм зависимостей переменных. Приведенные в работе (А. Духанов [4]) результаты ориентированы на процедурное программирование, однако допускают распространения данного метода и на объектно-ориентированное моделирование. Тем не менее остается еще достаточно много нерешенных вопросов. Прежде всего – определение правил сопоставления элементов моделей СД и программных конструкций ООМ.

Изложение основного материала исследования с полным обоснованием полученных научных результатов. Системная динамика Дж. Форрестера базируется на ряде концепций (Дж. Форрестер [5]). Основные положения СД мы группируем следующим образом.

Концепция вещественных и информационных сетей. Эта концепция проводит четкое разграничение между материальными и информационными потоками. Для материальных потоков характерен закон сохранения. Напротив, для информации характерна возможность накопления и тиражирования. Информационные связи управляют вещественными потоками. В ООМ материальные потоки моделируются потоковыми классами (TThread) или классами, в которых копирующий конструктор объявлен приватным (для квазипараллельного моделирования).

Уровни и темпы. Динамку поведения сколь угодно сложного процесса можно свести к изменениям некоторых фондов, или, как говорят, *уровней*, а сами изменения регулируются *темпами* наполняющих или исчерпывающих фонды потоков. *Темп* характеризует потоки, входящие в резервуары или выходящие из резервуаров. Изменение уровня определяется только темпами потока. Мы будем исходить из того, что системная динамика в ООМ проявляется в следующем. Объект инкапсулирует структуру подсистем, предоставляя другим объектам

для взаимодействия только интерфейс. Этот интерфейс и есть отражение уровней (свойства) и потоков (методы).

Функции решений, которые управляют темпами. Величина темпов регулируется информацией об уровнях. Темп не может непосредственно воздействовать на другой темп, так же как и уровень не может воздействовать на другой уровень. Как подчеркивает Дж. Форрестер, эти функциональные зависимости обычно являются существенно нелинейными. В моделях ООМ многие подобные соотношения получаются естественным образом из алгоритмов взаимодействия объектов.

Концепция петли обратной связи. В самом общем случае петля обратной связи моделируется циклом. В теории динамических систем петле обратной связи соответствует понятие предельного цикла, причем цикл может быть как аттрактором, так и репеллером. Тем самым можно говорить о петле обратной связи с положительной или отрицательной обратной связью. Специальным случаем петли обратной связи в ООМ будет диалог – коммуникативный акт с ответом. В отличие от СД коммуникативный акт является дискретным. Моделируется вызовом метода класса с некоторым возвращаемым значением.

Запаздывание. В самом общем случае понятие запаздывания в ООМ связано с таким понятием как синхронизация. Для синхронизации потоков обычно используется приостановка вычислений потока. Однако в ряде случаев такое описание будет неадекватным и следует использовать *активное ожидание*. Поток не приостанавливается, а непрерывно проверяет выполнение некоторого условия. Когда данное условие будет выполнено, поток продолжает вычисление. Типичным примером является паттерн *Producer*/*Consumer*. Поток *Producer* пишет в буфер (буфер обычно реализован как очередь), *Consumer* читает из буфера. Если в буфере нет данных, то поток *Consumer* ожидает поступления данных, и используется для этого активное ожидание. В некоторых реализациях этого паттерна поток *Producer* также проверяет, что данные были считаны из буфера. Тем самым элемент СД *запаздывание* в ООМ реализуют такие программные конструкции, как *Producer*/*Consumer*. Запаздывание может иметь место как для материальных потоков, так и для информационных связей. Информационное запаздывание иногда можно моделировать линиями задержки.

Продемонстрируем вышесказанное на следующей типичной модели системной динамики. Гидравлическая система состоит из двух емкостей *A* и *B*, каждая из которых характеризуется уровнем воды; начальные значения l_A и l_B. Емкости соединены двумя трубами. По первой трубе вода подается насосом из емкости *A* в емкость *B*. Поток постоянен и составляет *q* литр/секунду. По второй трубе вода поступает из емкости *B* в емкость A, и поток пропорционален l_B. Коэффициент пропорциональности *d* устанавливается краном. Кроме того, предусмотрена задержка во времени на t_d секунд управляющего воздействия. Заметим, что с гидравлической аналогией необходимо обращаться осторожно. Приведенный далее код описывает работу насоса с управляемой мощностью, а не свободное истечение жидкости. Из внешней среды в емкость *A* поступает постоянный приток воды q_e литр/секунду. Требуется определить значения l_A и l_B через *t* секунд.

Для решения данной задачи использовалась система имитационного моделирования Vensim (Ventana Systems, Inc.). Соответствующая модель представлена на рисунок 1.

Обратимся теперь к объектной модели. Модель задается диаграммой классов рисунок 2; язык программирования – C++. Гидравлическая система моделируется классом Pipeline. Ограничимся моделью «толстого слоя» ($l >> 0$). Класс Surroundings моделирует окружение системы и определяет начальные и граничные условия для системы Pipeline (ExternalStream). Емкости *A* и *B* моделируются разделяемыми объектами класса Barrel – это поля barrelA и barrelB класса Pipeline. Подсистемы моделируются двумя классами FirstStream и SecondStream, объекты этого класса конфигурируются объектами barrelA и barrelB. Атомарные объекты – слой воды – моделируются экземплярами класса Drop и владеют собственными вычислительными потоками (thread). Класс обладает свойствами upper и under, которые отражают свойство связанности слоев воды друг с другом (слипание).

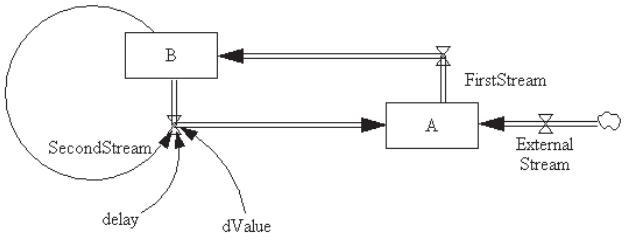

Рисунок 1 - Модель системной динамики

Обратимся теперь к объектной модели. Модель задается диаграммой классов рисунок 2; язык программирования – C++. Гидравлическая система моделируется классом Pipeline. Ограничимся моделью «толстого слоя» ($l >> 0$). Класс Surroundings моделирует окружение системы и определяет начальные и граничные условия для системы Pipeline (ExternalStream). Емкости *A* и *B* моделируются разделяемыми объектами класса Barrel – это поля barrelA и barrelB класса Pipeline. Подсистемы моделируются двумя классами FirstStream и SecondStream, объекты этого класса конфигурируются объектами barrelA и barrelB. Атомарные объекты – слой воды – моделируются экземплярами класса Drop и владеют собственными вычислительными потоками (thread). Класс обладает свойствами upper и under, которые отражают свойство связанности слоев воды друг с другом (слипание).

Прямое моделирование параллельности предполагает только синхронизацию потоков. Действительно, каждый из объектов Barrel взаимодействует с потоками FirstStream и SecondStream так, что один поток ставит в очередь, а второй поток – читает из очереди объекты Drop. Тем самым ситуации *гонок* не возникает. Единственно, что необходимо сделать – это синхронизировать интенсивность этих процессов. Для этого используется *барьер,* выполненный на основе разделяемого счетчика. Поток, выполнив цикл чтения/записи, понижает счетчик на единицу и приостанавливает себя. Когда значение счетчика становится равным нулю, поток Pipeline перезапускает потоки.

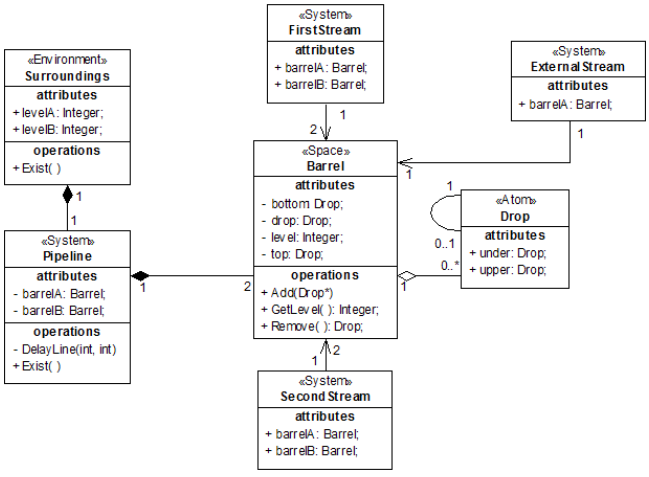

Рисунок 2 - Объектная модель

С переходом к квазипараллельному описанию модель существенно упрощается. Можно не инкапсулировать процессы течения в объектах и тем самым обойтись

без классов FirstStream, SecondStream и ExternalStream. Метод Exist класса Pipeline тогда следующим образом определит единицу дискретно-событийного времени системы:

```
Drop *d;
for (int i = 0; i < 1; i++) {
d = this->barrelA->Remove(); this->barrelB->Add(d);
};
int d_level = DelayLine(barrelB->level, delay); // запаздывание
int control = d_level / dValue; // управлеиие
for (int i = 0; i < control; i++) {
d = this->barrelB->Remove(); this->barrelA->Add(d);
}; .
```

Первый оператор for моделирует поток воды из *A* в *B*. Второй оператор for моделирует поток воды из *B* в *A*. Интенсивность второго потока определяется величиной d_level, которая вычисляется в линии задержки DelayLine. В следующей строке кода на основе информации d_level и коэффициента пропорциональности dValue вырабатывается управляющая информация control.

Сравним обе модели. Оба цикла for моделируют материальные потоки, для которых должен выполняться некоторый инвариант цикла Exist. В частности, таковым инвариантом является величина $l_A + l_B = const$ (закон сохранения количества жидкости). Этот закон выполняется, если внешний поток равен нулю. В модели системной динамики эти процессы изображены двойными стрелками. Данный инвариант используется для определения уравнений баланса имитационной модели. Кроме материальных потоков, имеют место два информационных процесса. Один процесс передает значение уровня емкости *B* в переменную управления второго цикла for. Другой информационный процесс – это процесс запаздывания. В модели системной динамики эти связи изображены сплошными стрелками. Функция решения в данном случае – вычисление величины control. Обозначается в СД как вентиль.

Выводы исследования и перспективы дальнейших изысканий данного направления. Итак, сравнительный анализ объектной модели и модели системной динамики позволяет сделать вывод о гомоморфном отображении моделей ООМ на модели системной динамики. Объектные модели требуют привлечения дополнительных предположений, что делает их более содержательными. В частности, многие «загадочные» нелинейные зависимости СД в рамках ООМ получают естественное выражение как алгоритмы работы с объектами. Напомним, что в популярной системе AnyLogic есть возможность совмещения агентных моделей и моделей системной динамики. Приведенные выше сопоставления говорят о том, что действительно возможно сопряжение обеих разновидностей моделирования, причем на глубинном теоретическом уровне.

СПИСОК ЛИТЕРАТУРЫ

1. Гурьянов В. И. Профиль UML для имитационного моделирования // Объектные системы – 2013: материалы VII Международной научно-практической конференции (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://simulation.su/uploads/files/default/object-systems-2013-proceedings.pdf

2. Каталевский Д. Ю. Системная динамика и агентное моделирование: необходимость комбинированного подхода // Устойчивое экономическое развитие: интеграция государства и бизнеса в современном обществе: материалы 14-й Междунар. науч.-практ. конф. М.: ГОУВПО ГУУ, 2009.

3. Борщев А. В. Практическое агентное моделирование и его место в арсенале аналитика // Exponenta Pro. 2008. № 3,4.

4. Духанов А. В. Имитационное моделирование сложных систем: курс лекций. Владимир: Изд-во Владимирский гос. ун-та, 2010. 115 с.

5. Форрестер Дж. Основы кибернетики предприятия (Индустриальная динамика) /перевод: Л.А. Балыков, Л.Е. Балясный, А.И. Гоман, В.Ю. Невраев, Н.А. Палатников, В.Э. Рексин. М.: Прогресс, 1970. 340 с.

THE DESIGN OF THE OBJECT MODEL FOR SYSTEM DYNAMICS MODEL

***V.I. Gurianov*,** Ph. D. in technical science, assistant professor
Saint-Petersburg State Economic University, Chebokasry (Russia)

Annotation: The article considers the basic concepts of system dynamics. The minimum set of descriptive constructions system dynamics is proposed. This set is presented in the language of object-oriented modeling. The example of design of the object model for system dynamics model is given.

Keywords: simulation, system dynamics, object-oriented modeling.