ПРАКТИЧЕСКОЕ АГЕНТНОЕ МОДЕЛИРОВАНИЕ И ЕГО МЕСТО В АРСЕНАЛЕ АНАЛИТИКА

А. В. Борщёв (Санкт-Петербург)

1. Имитационное моделирование: уровни абстракции, основные подходы

Моделирование можно рассматривать как один из способов решения проблем, возникающих в реальном мире: в технике, производстве, обслуживании, маркетинге, финансах, здравоохранении, транспорте и т.д. Оно применяется в случае, если эксперименты с реальными объектами/системами или их прототипирование невозможно или слишком дорого. Моделирование позволяет оптимизировать систему до её реализации. Процесс моделирования включает в себя отображение проблемы из реального мира в мир моделей (процесс *абстракции*), анализ и оптимизацию модели, нахождение решения и его отображение обратно в реальный мир (рис. 1). Мы различаем аналитическое и имитационное моделирование. *Аналитическая* модель допускает аналитическое решение, зависимость выхода от входа можно реализовать статически в виде, например, электронных таблиц. Это требует от аналитика владения всего лишь общепринятыми программными средствами, например Excel. Однако, к сожалению, аналитические решения не всегда возможны, а существующие не всегда просто найти. В этом случае аналитики применяют имитационное моделирование (ИМ, английский термин – simulation modeling), которое по контрасту можно назвать динамическим. *Имитационную* модель можно рассматривать как множество правил (дифференциальных уравнений, карт состояний, автоматов, сетей и т.п.), которые определяют в какое состояние система перейдёт в будущем из заданного текущего состояния. Имитация здесь – это процесс “выполнения” модели, проводящий её через изменения (дискретные или непрерывные) состояния во времени. В общем случае для сложных проблем, где время и динамика важны, имитационное моделирование представляет собой более мощное средство анализа.

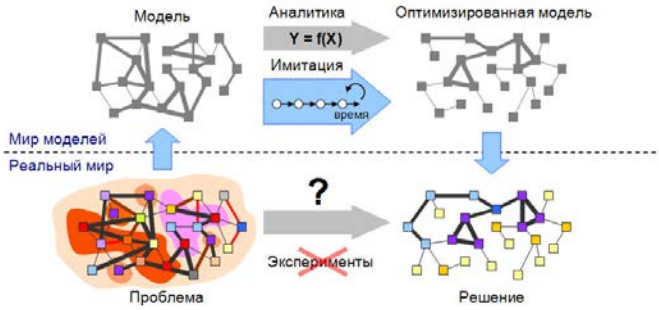


Рис. 1. Аналитическое (статическое) и имитационное (динамическое) моделирование

Уровни абстракции в имитационном моделировании На рис. 2 показан примерный (и, безусловно, неполный) круг практических задач, к которым эффективно применяется имитационное моделирование. Задачи эти расположены на шкале уровня абстракции, который используется в соответствующих моделях.

На самом детальном уровне находится так называемое “физическое” моделирование, где рассматриваются конкретные материальные объекты с их точными размерами, расстояниями, скоростями, ускорениями и временами. Таким образом, внизу нашей шкалы расположены модели систем управления, мехатронных систем, уличное и пешеходное движение, моделируемое на микро-уровне и т.д. Модели производств с конвейерами, станками, операторами расположены выше, поскольку обычно здесь есть возможность абстрагироваться от точных физических траекторий и времён и использовать их усреднённые значения или стохастические модели. То же относится к моделям складской логистики с автопогрузчиками, паллетами, стеллажами и т.п.

Рис. 2. Приложения имитационного моделирования на шкале уровня абстракции

Модели бизнес-процессов и систем обслуживания оперируют обычно лишь с временами и расписаниями, хотя физическое перемещение иногда и принимается в расчёт. Например, в здравоохранении при моделировании обычного отделения больницы в основном важны количество и график работы персонала, оборудование, поток пациентов и логика работы с ними, в то время как для отделения скорой помощи могут быть учтены также конфигурация здания, длины коридоров и т.д.

При моделировании транспортных и компьютерных сетей важны расписания, задержки, мощности и ёмкости, времена погрузки/разгрузки/обработки. Макро-уровень транспортно-пешеходно-сетевого моделирования абстрагируется от индивидуальных машин, людей и пакетов данных и рассматривает только их количества. Цепочки поставок моделируются на самых разных уровнях абстракции, так что их можно было бы расположить в любом месте шкалы от среднего до высокого уровня.

К задачам в верхней части шкалы традиционно применяют понятия влияний, обратных связей, тенденций и т.п. Вместо индивидуальных объектов, таких как клиенты, сотрудники, машины, животные, транзакции, товары, рассматривают их агрегаты, количества. Динамика систем на этом уровне описывается утверждениями типа “увеличение количества рабочих мест вызовет увеличение иммиграционного притока”.

Основные подходы в имитационном моделировании – это системная динамика (СД), дискретно-событийное моделирование (ДС), под которым здесь понимается любое развитие идей GPSS, агентное моделирование (АМ). СД и ДС – традиционные устоявшиеся подходы, АМ – относительно новый. Область моделирования динамических систем, являясь инженерной дисциплиной, останется в стороне от рассмотрения. Математически, СД и динамические системы оперируют в основном с непрерывными во времени процессами, тогда как ДС и АМ – в основном с дискретными.

Рис. 3. Подходы в имитационном моделировании на шкале уровня абстракции

Динамические системы по уровню абстракции находятся внизу шкалы (рис. 3). СД, заменяя индивидуальные объекты их агрегатами, наоборот, предполагает наивысший уровень абстракции. ДС-моделирование работает в низком и среднем диапазоне. Что же касается АМ, то оно может применяться практически на любом уровне и в любых масштабах. Агенты могут представлять пешеходов, автомобили или роботов в физическом пространстве, клиента или продавца на среднем уровне, или же конкурирующие компании на высоком.

СД, ДС и динамические системы исторически преподаются совершенно разным категориям студентов: менеджмент, инженеры по организации производства (industrial engineers) и инженеры–разработчики систем управления. В результате возникли три отдельных сообщества, которые практически никак не общаются друг с другом. АМ же до недавнего времени было академической “игрушкой”. Однако растущий спрос на глобальную оптимизацию со стороны бизнеса заставил ведущих аналитиков обратить внимание именно на АМ и его объединение с традиционными подходами с целью получения более полной картины взаимодействия сложных процессов различной природы. Отсюда спрос на программные платформы, позволяющие интегрировать различные подходы.

Системная динамика была разработана и предложена Джеем Форрестером в конце 1950-х годов как “исследование информационных обратных связей в промышленной деятельности с целью показать, как организационная структура, усиления (в политиках) и задержки (в принятии решений и действиях) взаимодействуют, влияя на успешность предприятия” [1, 2]. Приложения СД включают также социальные, урбанистические, экологические системы. Процессы, происходящие в реальном мире, в СД представляются в терминах накопителей – stocks (например, материальных объектов, знаний, людей, денег), потоков между этими накопителями – flows, информации, которая определяет величину этих потоков. СД абстрагируется от отдельных объектов и событий и предполагает “агрегатный” взгляд на процессы, концентрируясь на политиках, этими процессами управляющих. Моделируя в стиле СД, вы представляете структуру и поведение системы как множество взаимодействующих положительных и отрицательных обратных связей и задержек

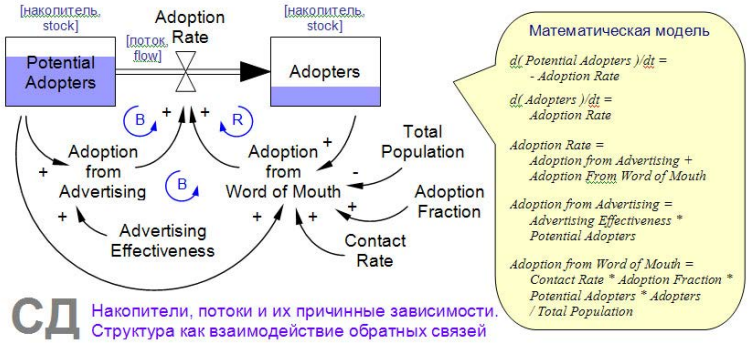


Рис. 4. Классическая модель системной динамики: Bass Diffusion в Vensim™

> **Модель Bass Diffusion.** В этой классической модели распространения нового продукта, или инновации, взятой из учебника (рис. 4, [3]), потенциальные клиенты (*Potential Adopters*) становятся клиентами (*Adopters*) со скоростью диффузии (*Adoption Rate*), которая зависит от рекламы и “устной рекламы”, т.е. общения клиентов с не-клиентами. Влияние рекламы моделируется следущим образом: некий постоянный процент потенциальных клиентов (*Advertising Effectiveness* = 0,011 в этой статье) всё время становятся клиентами. Их доля в *Adoption Rate* равна, соответственно, *Potential Adopters * Advertising Effectiveness*. Что касается устной рекламы, делается предположение, что в данной группе людей все контактируют со всеми. Количество контактов человека в единицу времени обозначено как *Contact Rate* (100). В случае, если клиент общался с потенциальным клиентом, последний становится клиентом с вероятностью *Adoption Fraction* (0,015). Таким образом, в единицу времени все клиенты обратят *Adopters * Contact Rate * Adoption Fraction * [Potential Adopters / (Potential Adopters + Adopters)]* потенциальных клиентов в клиентов. Выражение в квадратных скобках – вероятность того, что тот, с кем был контакт у клиента, ещё не клиент.

Математически системно-динамическая модель – это система дифференциальных уравнений. Важно отметить следующие особенности СД: а) поскольку модель оперирует только количествами (агрегатами), объекты, находящиеся в одном накопителе, неразличимы, лишены индивидуальности; б) аналитику предлагается рассуждать в терминах глобальных структурных зависимостей, и, естественно, ему необходимы соответствующие исходные данные. Подход СД поддерживается тремя–четырьмя инструментами, весьма похожими друг на друга.

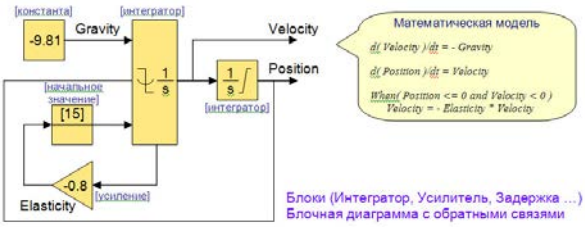


Рис. 5. Модель динамической системы: прыгающий мячик в MATLAB™ Simulink™

Динамические системы моделировались задолго до возникновения СД и являются, собственно, её прообразом. Моделирование ДС используется инженерами в механике, электронике, энергетике, химии как часть стандартного процесса разработки. На рис. 5 показана типичная блок-схема в языке MATLAB Simulink. Такими схемами пользуются при разработке систем управления; для разных областей могут применяться разные визуальные и текстовые нотации. Соответствующая математическая модель, как и в случае СД, состоит из набора переменных состояния и системы алгебро-дифференциальных уравнений над ними. В отличие от СД, здесь переменные состояния имеют прямой “физический” смысл: координата, скорость, давление, концентрация и т.д.; они, естественно, непрерывные и не являются агрегатами (количествами) дискретных объектов. Математическое разнообразие и сложность в динамических системах могут быть значительно выше, чем в СД, так что в принципе любая СД-проблема может быть решена инструментами для моделирования динамических систем, и даже с большей точностью (за счёт более совершенных численных методов). Однако такие инструменты, “заточенные” под инженерные нужды, неудобны для СД-аналитиков и не используются ими: можно сказать, *они не поддерживают их привычного образа мышления*.

Ниже динамические системы, рассматриваться более не будут, хотя отметим, что и здесь делались интересные попытки агентного моделирования, например, молекулы газа представлялись в виде отдельных взаимодействующих агентов.

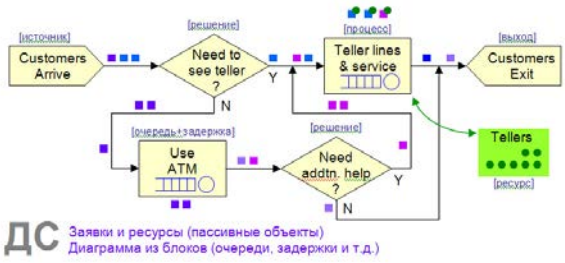


Рис. 6. Дискретно-событийная модель: отделение банка в Arena™

Дискретно-событийное моделирование. Этим термином обозначим подход, в основе которого лежит концепция заявок (транзактов, entities), ресурсов и потоковых диаграмм (flowcharts), определяющих потоки заявок и использование ресурсов. Этот подход восходит к Джеффри Гордону, который в 1960-х годах придумал и развил GPSS и реализовал её, работая в IBM [6]. Заявки (транзакты в GPSS) – это пассивные объекты, представляющие людей, детали, документы, задачи, сообщения и т.п. Они путешествуют через flowchart, стоя в очередях, обрабатываясь, захватывая и освобождая ресурсы, разделяясь, соединяя и т.д. Типичная потоковая диаграмма показана на рис. 6 в терминах Arena. Вообще существует около сотни коммерческих инструментов, так или иначе поддерживающих подобный стиль моделирования; некоторые общего назначения, большинство нацелено на определённые ниши: обслуживание, бизнес-процессы, производство, логистика и т.д. Их пользовательские интерфейсы могут существенно различаться из-за специализации, но за ними непременно стоит более или менее одинаковый дискретно-событийный “движок” (engine), который “гоняет” заявки через блоки. Для целей данного исследования важно отметить, что дискретно-событийную модель можно рассматривать как *глобальную схему обработки заявок*, обычно со стохастическими элементами.

Агентное моделирование. Под этим названием делается большое количество исследований и разработок в различных областях знания, например, в искусственном интеллекте, теории сложных систем, теории игр и т.д. Общепризнанного определения “что такое агент” не существует; до сих пор спорят о том, какими же качествами должен обладать объект, чтобы “заслужить” называться агентом: инициативность и реактивность, ориентация в пространстве, способность обучаться, общаться, “интеллект” и т.д. [11]. Мы не собираемся предлагать здесь свое определение: агенты, которые реально используются в нашей консалтинговой модельной практике, бывают самые разные. Однако есть нечто, объединяющее все агентные модели: они существенно *децентрализованы*. В отличие от системной динамики или дискретно-событийных моделей, здесь нет такого места, где централизованно определялось бы поведение (динамика) системы в целом. Вместо этого аналитик определяет поведение на индивидуальном уровне, а глобальное поведение возникает (emerges) как результат деятельности многих (десятков, сотен, тысяч, миллионов) агентов, каждый из которых следует своим собственным правилам, живёт в общей среде и взаимодействует со средой и с другими агентами. Поэтому АМ называют ещё моделированием *снизу вверх*.

Рис. 7. Типичная архитектура агентной модели. Поведение (карта состояний) в AnyLogic™

Рис. 7 можно условно рассматривать как агентную модель динамики населения страны. В этой модели один из аспектов поведения агента задан картой состояний (statechart, эта полезная конструкция объясняется ниже), а модель среды может включать жильё, рабочие места, транспортную инфраструктуру и т.д. Далее будет показано, как АМ соотносится с СД и ДС и как и когда оно может практически применяться.

2. Соотношение СД и агентных моделей

Для начала покажем, как “конвертировать” существующую СД-модель в эквивалентную агентную модель, а затем – как можно развить последнюю, чтобы учесть более сложную динамику реальной системы. Для понимания материала будет полезна следующая справка.

> **Карты состояний (statecharts).** В консалтинговой практике мы часто используем карты состояний для описания поведения агентов. Карта состояний (рис. 8) – это, фактически, тот же конечный автомат с несколькими удобными дополнениями, предложенными Давидом Харелом, принятыми мировым моделирующим сообществом и вошедшими в стандартный UML (The Unified Modeling Language, [15]). Карты состояний позволяют графически определить возможные состояния агента, переходы между ними, события, вызывающие эти переходы, временные задержки и действия, совершаемые агентом на протяжении своей жизни. Такие конструкции, как вложенные состояния, позволяют задавать режимы функционирования агента. Агент может иметь несколько параллельно активных и взаимодействующих карт состояний, каждая из которых отвечает за какой-либо аспект его жизни: например, образование и семейное положение.

Рис. 8. Карты состояний UML: язык для задания событийного и временного поведения

Будет уместно отметить следующее: в то время, как традиционные подходы в имитационном моделировании вроде СД или ДС практически не получили существенного развития на протяжении последних десятилетий, в технологии разработки программного обеспечения произошли революционные открытия, в корне изменившие принципы работы со сложными системами. Значительная часть этого опыта была аккумулирована в UML. И, хотя UML сам по себе не может использоваться как язык имитационного моделирования (в частности, не у всех конструкций есть чёткая семантика), многие его принципы, например, разделение структуры и поведения, объектная ориентированность (концепция классов и объектов, инкапсуляция) и т.п. могут сэкономить аналитику огромное количество времени и усилий при разработке имитационной модели.

Рассмотрим СД-модель как цепочку накопителей и потоков и множество “правил”, контролирующих эти потоки (рис. 9). Ключевым начальным действием будет “деагрегировать” накопители, т.е. *представить их не как “ёмкости с жидкостью”, а как, например, “ящики с шариками”*. Эти шарики и станут агентами. Мысленно поместите теперь себя в такой шарик и посмотрите на происходящее. Вы увидите, что у вас два возможных состояния *State A* и *State B*, соответствующие тому накопителю (ящику), в котором вы находитесь, и если вы находитесь в *State A*, то когда-нибудь перейдёте в *State B*. Момент времени, когда это произойдёт, очевидно, зависит от интенсивности потока. Агентная модель, которая ведёт себя так же, как эта СД-модель, будет состоять из (*Stock A* + *Stock B*) агентов, каждый из которых “выполняет” такую вот карту состояний. Переход между состояниями может быть реализован несколькими способами (два из них показаны на рис. 9): *синхронно*, когда решение о переходе принимается на каждом временном шаге *dt*, и *асинхронно*, когда время перехода рассчитывается заранее при входе в *State A* и может пересчитываться потом при изменении *Rate*. Заметим, что в последнем случае понятие временного шага в модели в принципе отсутствует. *Асинхронные агенты могут быть гораздо более эффективны вычислительно и проще в реализации, чем синхронные*. В последующих примерах будем использовать асинхронные агенты везде, где это возможно.

Модель Bass Diffusion – агентная версия. Теперь проиллюстрируем технику перехода от CD к AM на классической модели распространения инновации из знаменитого учебника Штермана по СД [3], приведённой выше (рис. 10).

Шаг 1. Для двух накопителей создаём два состояния агента: *Potential Adopter* и *Adopter*.

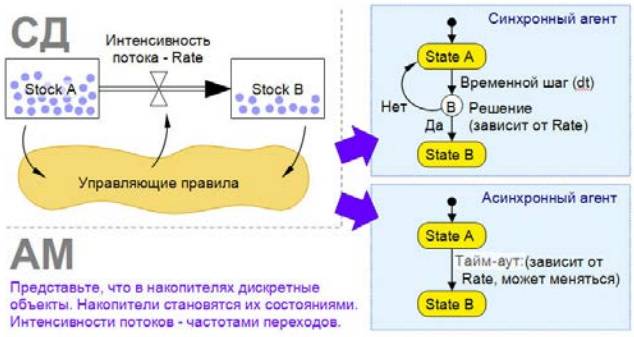
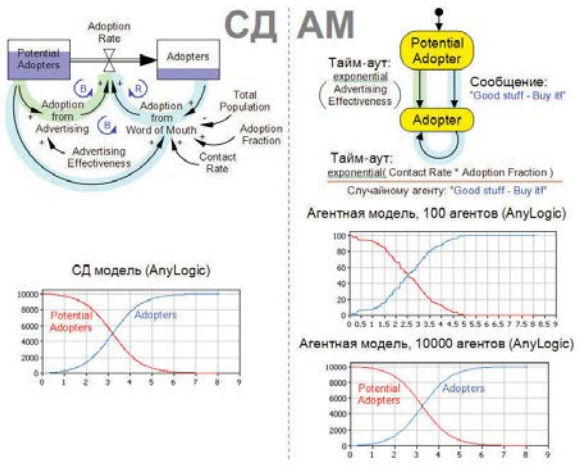


Рис. 9. “Конвертация” СД-модели в агентную модель. Общая схема

Шаг 2. Две составляющие *Adoption Rate* будут моделироваться отдельно. Для влияния рекламы *Adoption from Advertising* создаём переход из состояния *Potential Adopters* в состояние *Adopters*, который срабатывает по истечении экспоненциально распределённого (среди всех агентов) времени со средним значением *Advertising Effectiveness*. Это моделирует постоянный процент людей, становящихся клиентами под влиянием рекламы.

Шаг 3. Для "устной рекламы" *Adoption from Word of Mouth* добавляем циклический переход, периодически срабатывающий у клиентов. Этот переход будет моделировать их контакты с другими людьми, происходящие с интенсивностью *Contact Rate*. При каждом контакте агент-клиент будет посылать сообщение «хорошая вещь – купи!» (*"Good stuff – Buy it!")* другому агенту. В случае, если этот другой ещё не клиент, т.е. находится в состоянии *Potential Adopter*, он станет клиентом (перейдёт в состояние *Adopter*) с соответствующей вероятностью, отсюда появляется второй переход из *Potential Adopter* в *Adopter*. Клиенты же такое сообщение будут игнорировать. Приведённая реализация оптимизирована: мы моделируем только успешные контакты (поэтому интенсивность циклического перехода сразу умножена на вероятность *Adoption Fraction*, а эффект всегда гарантирован).

СД-модель порождает хорошо известную S-образную кривую, и подобную же кривую даёт агентная модель. Если число агентов невелико, дискретная (*и, таким образом, более реалистичная*) природа модели хорошо просматривается на графике; при увеличении числа агентов кривая будет приближаться к гладкому решению СД-модели.



Р ис. 10. Модель Bass Diffusion, конвертированная из СД в агентную

> Количество моделируемых агентов зависит от программного и аппаратного обеспечения, но практически важнее определить, *сколько агентов действительно* ***нужно*** *моделировать*. Например, если вас интересует динамика населения страны (300 000 000 человек) не нужно имитировать 300 000 000 агентов. Существуют различные методы уменьшения числа агентов, делающие моделирование вычислительно эффективным и при сохранении корректных результатов.

Процедура "конвертации" или, скорее, изменения концепции модели, которая была только что рассмотрена, имеет практический смысл только в том случае, если планируется развивать полученную агентную модель далее, например, учитывать индивидуальную память агентов, их пространственное положение и т.д. *Если же объекты, находящиеся в накопителях СД-модели, пассивны и неразличимы, то смысла в такой конвертации нет*. Например, когда в СД-модели рассматриваются денежные средства, нам не интересна индивидуальная история каждого доллара, а доллары сами по себе не проявляют активности (по крайней мере, в смысле данного исследования).

Теперь поменяем в модели некоторые предположения. Пусть эффективность устной рекламы конкретного клиента (вероятность, с которой он обращает в клиентов тех, с кем общается) зависит от того, как давно он сам приобрёл продукт. Предположим, что сразу после покупки это влияние велико, а затем оно постепенно уменьшается и стабилизируется на некотором уровне (рис. 11).

Для того чтобы это учесть, в агент добавляется переменная *Time Purchased*, в которой запоминается время покупки (см. действия на обоих переходах из состояния *Potential Adopter*), а также используется табличная функция *Influence* вместо константы *Adoption Fraction*. Естественно, полученная кривая отличается от предыдущих из-за другой, более низкой эффективности устной рекламы.

Возможно ли учесть те же предположения в СД-модели? Теперь доля каждого клиента в общей скорости распространения продукта (диффузии) *Adoption Rate* различна, к тому же она меняется со временем. Поэтому агрегирование клиентов в одном (или любом разумном числе) накопителе неизбежно приводит к искажению результатов.

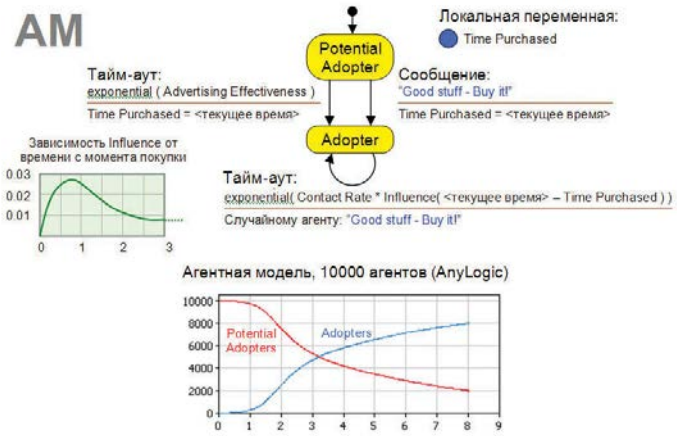


Рис. 11. Агентная модель Bass Diffusion: учитываем меняющееся влияние устной рекламы

Частичное решение (а точнее, “обход”) этой проблемы состоит в замене одного накопителя массивом ёмкостей, так что объекты с близкими свойствами попадают в одну ёмкость и перемещаются в другую при изменении этих свойств. Предположим, однако, что параметров, влияющих на свойства, не один, а несколько. Тогда массив становится многомерным, и при увеличении сложности модели ёмкостей, *элементов накопителя, может стать больше, чем собственно моделируемых объектов* [14]. Такая СД-модель, естественно, бессмысленна, в то время как агентная модель будет всегда содержать ровно столько агентов, сколько запланировано.

Хищники – жертвы (Predator Prey) – агентная версия.

СД-модель ««хищники – жертвы» (рыси и зайцы) состоит из пары дифференциальных уравнений, которые описывают динамику популяций хищников и жертв (или паразитов – носителей) в её простейшем случае (одна популяция хищников, одна – жертв). Модель была предложена независимо Альфредом Лоткой и Вито Вольтеррой (Alfred Lotka и Vito Volterra) [4], [5] в 1920-х годах; она характеризуется колебаниями в размерах обеих популяций, причём пик количества хищников немного отстаёт от пика количества жертв. В модели приняты следующие упрощённые предположения: а) жертвы всегда имеют достаточное количество ресурсов и погибают только будучи съеденными хищниками; б) жертвы – единственный источник пищи для хищников, и хищники умирают только от голода; в) хищники могут поглощать неограниченное количество жертв и г) пространство обитания не имеет размерностей, т. е. любой хищник может встретить любую жертву (рис. 12).

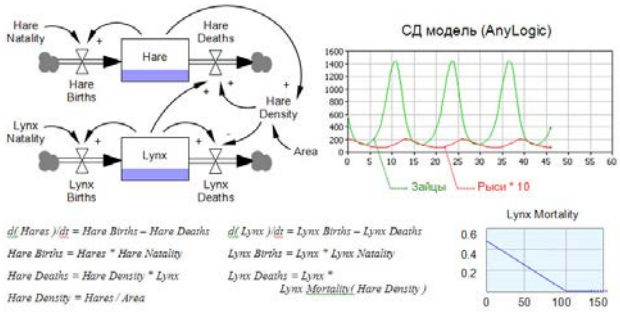


Рис. 12. Классическая СД-модель «хищники– жертвы» (Predator Prey)

На этот раз вместо конвертирования СД-модели агентная модель будет построена непосредственно “по постановке задачи” с предположениями даже более близкими к реальности, чем перечисленные выше. В агентной модели: а) зайцы (hares) и рыси (lynx) имеют конечную продолжительность жизни, т. е. они умирают также и от старости, а не только будучи съеденными или от голода; б) зайцы

голода; б) зайцы и рыси живут в двумерном пространстве (в терминологии агентного моделирования “*space-aware*”); в) плотность зайцев ограничена (например, неким пищевым ресурсом), так что зайцы размножаются, только если вокруг достаточно свободного места; г) рысь может поймать зайца только поблизости от места её обитания; д) рысь охотится периодически; е) если во время охоты заяц не пойман, рысь перемещается; и ж) если рысь так и не находит зайца в течение определённого времени, она умирает (рис. 13).

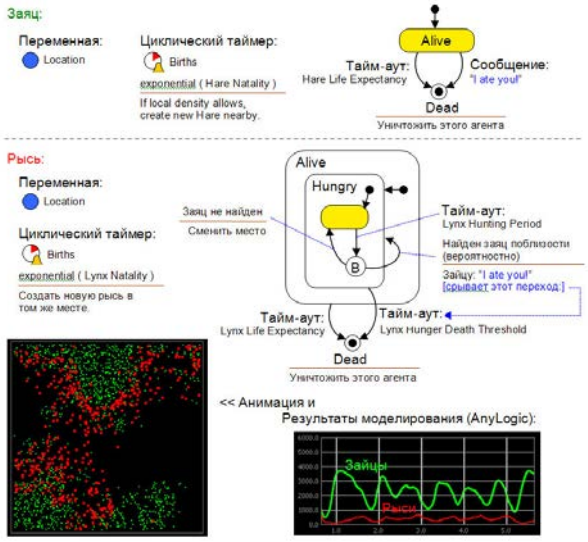


Рис. 13. Агентная модель хищников и жертв (Predator Prey): более реалистичные предположения

Агент-рысь и агент-заяц оба имеют переменные *Location* их текущего местоположения; вначале оно случайное. Оно меняется при перемещении агентов и влияет на их поведение. У рысей и у зайцев с определённой частотой появляются рысята и зайчата. Это моделируется циклическими «таймерами рождений» *Births*, которые создают новых агентов, причём в случае зайцев это зависит от их локальной плотности. Карта состояний зайца состоит всего из двух состояний: жив *Alive* и мёртв *Dead* и двух переходов между ними, соответствующих двум различным причинам смерти: возраст и съедение рысью (последнее моделируется сообщением, которое рысь напрямую посылает зайцу). У рыси поведение более сложное. Рысь охотится через каждые *Lynx Hunting Period* и, если она не находит зайца (это вероятностно зависит от их локальной плотности), она перемещается (изменяет *Location*), оставаясь в голодном состоянии *Hungry*. В случае, если заяц убит (рысь посылает ему сообщение «я тебя съела!», “*I ate you!*”), она выходит и тут же входит опять в состояние *Hungry*, что (в соответствии с семантикой карт состояний) вызовет “перезапуск” «тайм-аута голодной смерти» *Lynx Hunger Death Threshold*. Таким образом, рыси нужен как минимум один заяц каждые *Lynx Hunger Death Threshold* единиц времени.

Имитация агентной модели даёт гораздо более богатый выход, чем СД. Действия разворачиваются на плоскости: видны атаки рысей, их вымирание там, где съедены все зайцы, и быстрое заполнение зайцами свободного от рысей пространства. На агрегатном (количественном) уровне модель показывает колебательное поведение, похожее на поведение СД-модели (пики популяции рысей следуют за пиками популяции зайцев). В зависимости от параметров рыси могут полностью вымереть (иногда вместе с зайцами), чего никогда не случается в СД-модели из-за её непрерывности. Осцилляции стохастичны из-за стохастического характера модели.

> Один из часто задаваемых вопросов: как калибровать агентные модели? В СД зависимости переменных и параметров друг от друга явно определены, что позволяет более или менее понятным образом проводить калибровку. В агентных моделях параметры определены на локальном уровне, в то время как калибруют модель относительно глобальной статистики. Поэтому *агентную модель можно калибровать как (стохастический) чёрный ящик*, хотя на практике обычно понятно, какие из параметров на что влияют.

3. Соотношение ДС и агентных моделей

В ДС-модели уже есть индивидуальные объекты (заявки), что облегчает задачу ее «конвертации» в АМ: эти объекты, естественно, и станут агентами. Однако в ДС-моделировании заявки пассивны; ими управляют правила, определённые в блоках потоковой диаграммы (flowchart). Таким образом, задача состоит в том, чтобы посмотреть на процесс с точки зрения заявки и попытаться децентрализовать некоторые из правил. Опять же, всё это имеет смысл, только если планируется учесть в агентной модели какие-то индивидуальные поведения, не выражаемые в терминах ДС.

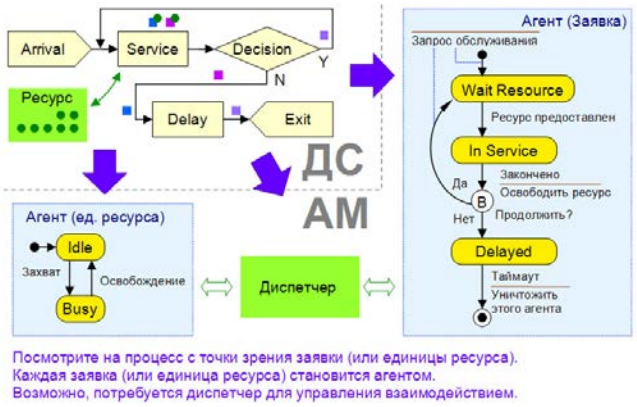


Рис. 14. “Конвертация” ДС-модели в агентную модель. Общая схема

Рассмотрим простую систему обслуживания (рис. 14), где объекты (люди, транзакции, и т.д.) входят в систему, обслуживаются (для этого требуется ресурс) один или несколько раз, в зависимости от каких-то их свойств, задерживаются на некоторое время и выходят. Эти объекты станут агентами. Событие генерации очередного объекта (заявки) будет соответствовать созданию нового агента. После создания агент запрашивает обслуживания, но не обязательно сразу же его получает, так что у агента есть состояние ожидания *Wait Resource* (соответствует заявке, стоящей в очереди в блоке *Service*). Когда ресурс предоставлен, агент переходит в состояние обслуживания *In Service* (соответствует заявке, получившей ресурс и задержанной в том же блоке *Service*) и, по окончании обработки, решает, повторить ли процедуру или перейти в состояние задержки *Delayed*. По выходе из *Delayed* агент уничтожает себя, что соответствует заявке, выходящей из потоковой диаграммы.

Ресурсы также можно моделировать агентами, если в этом есть смысл (например, ресурсы – операторы, персонал с каким-либо индивидуальным поведением). В этом простом примере ресурсы могут иметь два состояния: свободен *Idle* и занят *Busy*. Для координации доступа к ресурсам будет необходим какой-то механизм, реализованный либо централизованно (диспетчер), либо децентрализованно через взаимодействие агентов.

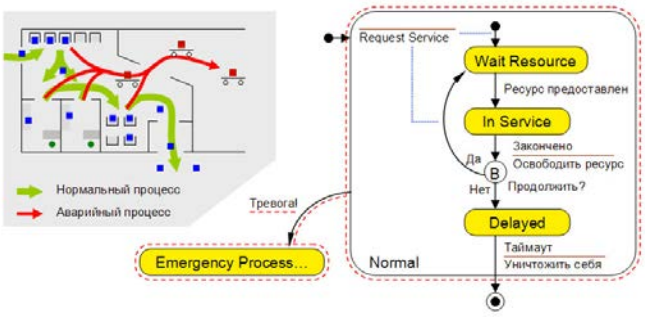


Рис. 15. Агентная модель системы обслуживания: реакция агента на особое событие

Теперь, как и в примере Bass Diffusion, немного усложним задачу. Пусть в любой момент времени нахождения объекта в системе может произойти нечто, что заставит его немедленно покинуть систему независимо от стадии обработки (рис. 15). Это может быть телефонный звонок, сердечный приступ, сигнал тревоги и т.п. В агентной модели это очень легко учесть: на карте состояний агента добавляется составное состояние, охватывающее

все состояния, ранее нами определённые. Назовём это состояние *Normal*, оно будет соответствовать нормальному режиму. Из него определим переход в новое состояние *Emergency Process*, соответствующее процессу “аварийного покидания” системы. Переход будет срабатывать по особому событию. (При выходе из нормального процесса, возможно, придётся освободить захваченные ресурсы). В ДС-подходе такое поведение моделировать непросто, придётся вставлять специальный код во все блоки; в некоторых инструментах это вообще невозможно сделать.

4. Инструменты. AnyLogic – поддержка нескольких подходов

Практически все присутствующие на рынке инструменты имитационного моделирования разработаны для поддержки одного определённого подхода, см. Рис. 16. Для системной динамики есть всего четыре инструмента. Дискретно-событийное же моделирование поддерживается десятками различных инструментов. Это объясняется просто: ДС как дисциплина определена не так строго, как СД, существует масса “диалектов”, созданных под конкретные приложения. В мире динамических систем доминирует MATLAB Simulink. Для агентного моделирования до последнего времени не существовало ни одного коммерческого инструмента, только библиотеки на Java или C++, разработанные в различных университетах.

Рис. 16. Инструменты имитационного моделирования

AnyLogic [16] – инструмент, который использует компания XJ Technologies и наши партнёры, исторически разрабатывался специалистами не по имитационному моделированию, а по computer science, в частности по распределенным системам. Поэтому за основу не был взят ни один из подходов ИМ; вместо них в основе AnyLogic лежат языки и методы, принятые в практике разработки сложных информационных систем. СД-диаграммы накопителей и потоков, как и ДС-схемы, естественным образом ложатся на объектно-ориентированный язык AnyLogic, и даже тем, кто моделирует, оставаясь в рамках этих традиционных подходов, инструмент даёт значительный выигрыш: компактное представление структуры, гибкое определение данных и т.д. Однако наиболее существенным преимуществом AnyLogic является возможность быстрого создания профессиональных агентных моделей в той же самой графической среде. AnyLogic поддерживает языковые конструкции для задания поведения агентов, их взаимодействия, моделирования среды, а также имеет богатейшие анимационные возможности. И, наконец, AnyLogic позволяет описывать разные части больших гетерогенных систем, используя разные подходы, объединяя СД, ДС и АМ в одной модели.

5. Примеры практических приложений АМ

Динамика употребления алкоголя. В этой модели, разработанной нами совместно с Research Triangle Institute International [см. также 8, 9], исследуется отношение людей к алкоголю, продолжительность жизни и связанные с этим расходы на здравоохранение. Рассмотрим упрощенный вариант модели. Мы различаем четыре состояния у человека: не употребляет вообще *Never User*, употребляет время от времени *Recreational User*, алкоголик *Addict* и бросивший пить *Quitter* (рис. 17). Переходы между состояниями – это стохастические тайм-ауты. Например, возраст, когда человек начинает

пить (“инициируется”) – это реализация случайной величины с распределением *Initiation Time Distribution*. Распределение построено на базе имеющихся статистических данных, а именно – вероятностей инициации для различного возраста. То же относится к продолжительности жизни, которая распределена по закону *Death Time Distribution*, но это распределение может изменяться в зависимости от отношений человека с алкоголем. В этой модели агенты не взаимодействуют друг с другом.

Рассматриваются две группы людей: одна с “естественной” алкогольной динамикой *Normal* (контрольная группа), другая, подвергшаяся вмешательству *Intervened*. Вмешательство (это могут быть изменения в законодательстве, “социальные” рекламные кампании и т.п.) моделируется как изменения в вероятностях инициации и отказа от алкоголя.

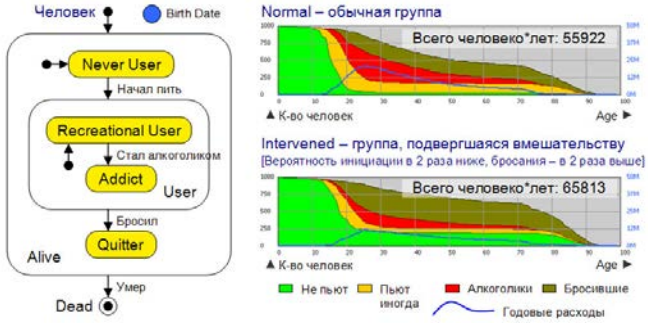


Рис. 17. Агентная модель динамики употребления алкоголя

Пример результатов моделирования (количество непьющих, употребляющих, алкоголиков и бросивших пить в зависимости от возраста) показаны в виде стековых графиков на рис. 17. Группа, подвергшаяся вмешательству, здесь имеет вероятность инициации в два раза ниже, а вероятность бросить в два раза выше, чем контрольная группа. В группе *Intervened* люди живут в среднем дольше, и ресурсов на их медицинское обслуживание уходит меньше. Подобного рода модели используются для поддержки принятия решений (decision support) при разработке федеральных, муниципальных или корпоративных политик.

Моделирование поведения и ассимиляционной динамики испаноязычного населения США. Эта модель [10] разработана консалтинговой компанией Decisio Consulting [17] для альянса крупных корпораций Synthesis Alliance. По последним опубликованным данным Американского бюро переписи населения (US Census), испаноязычное население (“Hispanics”) стало наибольшим по размерам национальным меньшинством в США. При помощи имитационного моделирования исследуются структурные силы, формирующие характеристики этой группы.

Смоделирована испаноязычная группа, чей уровень ассимиляции в основное население динамически меняется в зависимости от индивидуального выбора. Модель использует как агентный, так и системно-динамический подход. Испаноязычная группа деагрегирована до уровня индивидуальных агентов. Каждый агент принимает решения

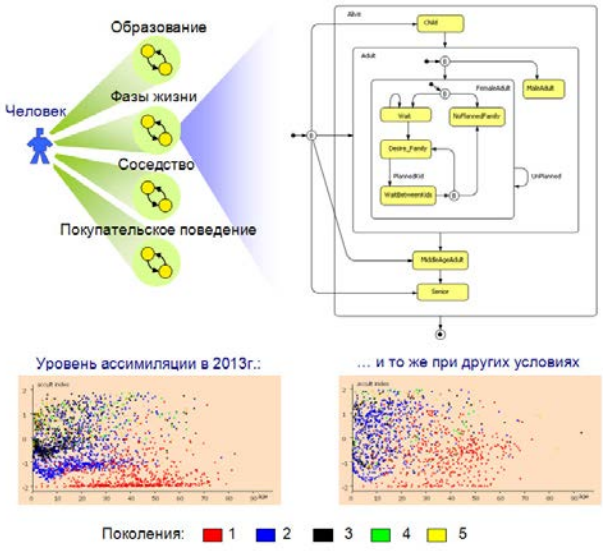


Рис. 18. Агентная модель поведения и ассимиляционной динамики испаноязычного населения

вероятностно, в зависимости от своего текущего состояния и внешней среды. Понятия “испаноязычность”, “ассимилированность” определены в культурных атрибутах личности и проявляются через миграцию, выбор соседства и другие механизмы.

Некоторые компоненты состояния агента представлены дискретно, некоторые – как “мягкие” непрерывные переменные, опирающиеся на проверенные СД-концепции и моделирующие накопление и утерю агентом культурных атрибутов. Также в традициях СД определены глобальные структуры обратных связей, которые в конечном счёте определяют поведение на индивидуальном уровне. Динамика моделируемой системы визуализируется с использованием инновационных средств AnyLogic.

Модель помогает увидеть, что система имеет достаточно сложную динамику, в частности появляются временно стабильные сегменты внутри испаноязычной группы (рис. 18). Динамика, порождающая эти сегменты, как и сами сегменты, представляет интерес для тех, кому важно глубокое понимание испаноязычного населения, в частности власть и бизнес. Применённые методы моделирования, естественно, могут быть легко использованы в других исследованиях по динамике населения и динамике культур.

Заключение. Какой подход использовать?

Мы увидели, что во многих случаях АМ позволяет легче отобразить в модели многие явления реального мира, чем СД или ДС-моделирование. Это, однако, не означает, что АМ – абсолютная замена традиционных подходов. Для большого числа приложений СД и ДС позволяют эффективно строить адекватные модели и получать достоверные результаты. Более того, в таких случаях попытки применить агентное моделирование могут быть менее продуктивными: агентные модели труднее строить. Таким образом, если задача хорошо подпадает под один из традиционных подходов, его нужно использовать без колебаний. Для этого под рукой имеется большое количество коммерческих инструментов, AnyLogic – один из них.

Агентное моделирование – для тех, кто хочет выйти за рамки ограничений, присущих системной динамике и дискретно-событийному моделированию (см. также [14]). Оно особенно эффективно при моделировании систем, содержащих большие количества активных объектов. Для таких систем AnyLogic поможет разработать агентную модель с минимальными усилиями, а также полностью или частично перейти от существующей СД или ДС модели к агентам.

Рис. 19. Комбинированные (многоподходные) модельные архитектуры

Существует также возможность использовать разные подходы для разных частей модели. Некоторые примеры комбинированных модельных архитектур приведены на рис. 19. Верхняя часто встречается в литературе по моделированию цепочек (сетей) поставок [12]: процессы внутри компании моделируются СД-диаграммой, а взаимодействие компаний друг с другом – полностью дискретно. В середине агенты (граждане или семьи) живут в среде (рабочие места, жильё, инфраструктура), динамика которой описана в

технике СД. Архитектура, показанная в нижней части рис. 19, может быть применена, скажем, при моделировании медицинских учреждений, если есть необходимость отслеживать более длительные периоды жизни пациентов и сотрудников.

Литература

1. **Forrester J.** *Industrial Dynamics: A Major Breakthrough for Decision Makers*///Harvard Business Review. – 1958.Vol. 36. No. 4. – P. 37–66.
2. **Forrester J.** *Industrial Dynamics*. Cambridge, MA: MIT Press, 1961.
3. **Sterman J.** *Business Dynamics: Systems Thinking and Modeling for a Complex World*. McGraw Hill, 2000.
4. **Lotka A. J.** *Elements of physical biology*. Baltimore: Williams & Wilkins Co, 1925.
5. **Volterra V.** *Variazioni e fluttuazioni del numero d'individui in specie animali conviventi*//Mem. R. Accad. Naz. dei Lincei. Ser. VI. Vol. 2. 1926.
6. **Gordon G.** *A General Purpose Systems Simulation Program*. McMillan NY//Proceedings of EJCC. – Washington D.C. 1961. – P. 87–104.
7. **Schelling T.** *Micromotives and Macrobehavior*. W. W. Norton and Co, 1978.
8. **Bobashev G., Zule W., Root E., Wechsberg W., Borshchev A. and Filippov A.** *Geographically-Enhanced Mathematical Models of HIV Dynamics*//NIDA Symposium on AIDS, Cancer and Related Problems. Russia. St. Petersburg, 2004.
9. **Bobashev G., Zule W., Root E., Wechsberg W., Borshchev A. and Filippov A.** *Scalable Mathematical Models for Substance Use: From Social Networks to the Whole Populations*//The College on Problems of Drug Dependence 66th Annual Meeting, Puerto Rico. San Juan, 2004.
10. **Wallis L., Paich M. and Borshchev A.** *Agent Modeling of Hispanic Population Acculturation and Behavior*//The 3rd International Conference on Systems Thinking in Management (ICSTM 2004), Philadelphia, Pennsylvania, USA, 2004.
11. **Schieritz N. and Milling P.** *Modeling the Forest or Modeling the Trees - A Comparison of System Dynamics and Agent-Based Simulation*//The 21st International Conference of the System Dynamics Society, New York, USA, 2003.
12. **Schieritz N. and Grosler A.** *Emergent Structures in Supply Chains – A Study Integrating Agent-Based and System Dynamics Modeling*//The 36th Annual Hawaii International Conference on System Sciences, Washington, USA, 2003.
13. **Solo K. and Paich M.** *A Modern Simulation Approach for Pharmaceutical Portfolio Management*//International Conference on Health Sciences Simulation (ICHSS'04), San Diego, California, USA, 2004 [Available from] http://www.simnexus.com/SimNexus.PharmaPortfolio.pdf.
14. Keenan P. and Paich M. *Modeling General Motors and the North American Automobile Market*//The 22nd International Conference of the System Dynamics Society, Oxford, England, 2004.
15. UML – The Unified Modeling Language. http://www.uml.org.
16. AnyLogic. http://www.anylogic.com.
17. Decisio Consulting. http://www.decisio.net.