От системной динамики и традиционного ИМ – к практическим агентным моделям: причины, технология, инструменты

Андрей Борщёв
ООО «Экс Джей Текнолоджис» XJ Technologies www.xjtek.com
и Санкт-Петербургский Государственный Политехнический Университет
Офис 101, дом 21 Политехническая улица, Санкт-Петербург 194021 Россия
Тел: +7 812 2471674 Факс: +7 812 2471639
andrei@xjtek.com

Аннотация Рассматривайте эту статью как практическое руководство для аналитиков-“симуляционистов”, которые хотели бы добавить к своему инструментарию агентное моделирование, ныне вполне для этого созревшее. Предполагается, что читатель имеет базовые знания в системной динамике или дискретно-событийном имитационном моделировании. Мы рассмотрим моделирование в приложении к системам, содержащим большие количества активных объектов (людей, животных, машин, предприятий или даже проектов, активов, товаров, и т.п.), которые объединяет наличие элементов индивидуального поведения, от сложных (цели, стратегии) до самых простых (временные ограничения, события, взаимодействия). Мы сравним, как три основные подхода, сложившиеся в современном имитационном моделировании – системная динамика, дискретно-событийное моделирование и агентное моделирование – предлагают анализировать такие системы. Мы покажем, как построить агентную модель по существующей системно-динамической или дискретно-событийной, и продемонстрируем, как просто она может быть расширена для учёта более сложных поведений, зависимостей и взаимодействий, чтобы предоставить вам более глубокую и точную информацию о моделируемой системе. На протяжении всей статьи используются классические примеры; все модели сформулированы на графическом языке пакета AnyLogic™. Мы рассматриваем агентное моделирование ни в коем случае не как замену традиционным подходам, а как весьма полезное дополнение к системной динамике и дискретно-событийному моделированию; мы также предлагаем несколько комбинированных модельных архитектур.

Ключевые слова: имитационное моделирование, агентное моделирование, системная динамика, дискретно-событийное моделирование, AnyLogic

1. Имитационное моделирование: уровни абстракции, основные подходы

Чтобы убедиться, что мы одинаково понимаем используемую базовую терминологию, пожалуйста, посмотрите на Рис 1. Моделирование – один их способов решения проблем, возникающих в реальном мире. Моделирование применяется в случае, если эксперименты с реальными объектами/системами или их

прототипирование невозможно или слишком дорого. Моделирование позволяет нам оптимизировать систему до её реализации. Моделирование включает в себя отображение проблемы из реального мира в мир моделей (процесс *абстракции*), анализ и оптимизацию модели, нахождение решения, и отображение решения обратно в реальный мир. Мы различаем аналитическое и имитационное моделирование. В *аналитической* модели выход функционально зависит от входа (набора параметров), и в этом смысле она – статическая; такую модель можно реализовать в виде электронных таблиц. Это требует от аналитика владение всего лишь общепринятыми программными средствами, например, Excel. Однако, к сожалению, аналитические решения не всегда существуют, а существующие не всегда просто найти. И тогда аналитики применяют имитационное моделирование (ИМ, английский термин – simulation modeling), которое мы по контрасту можем назвать динамическим. *Имитационную* модель можно рассматривать как множество правил (дифференциальных уравнений, карт состояний, автоматов, сетей и т.п.), кторые определяют в какое состояние система перейдёт в будущем из заданного текущего состояния. Имитация здесь – это процесс "выполнения" модели, проводящий её через (дискретные или непрерывные) изменения состояния во времени. В общем случае, для сложных проблем, где время и динамика важны, имитационное моделирование представляет собой более мощное средство анализа.

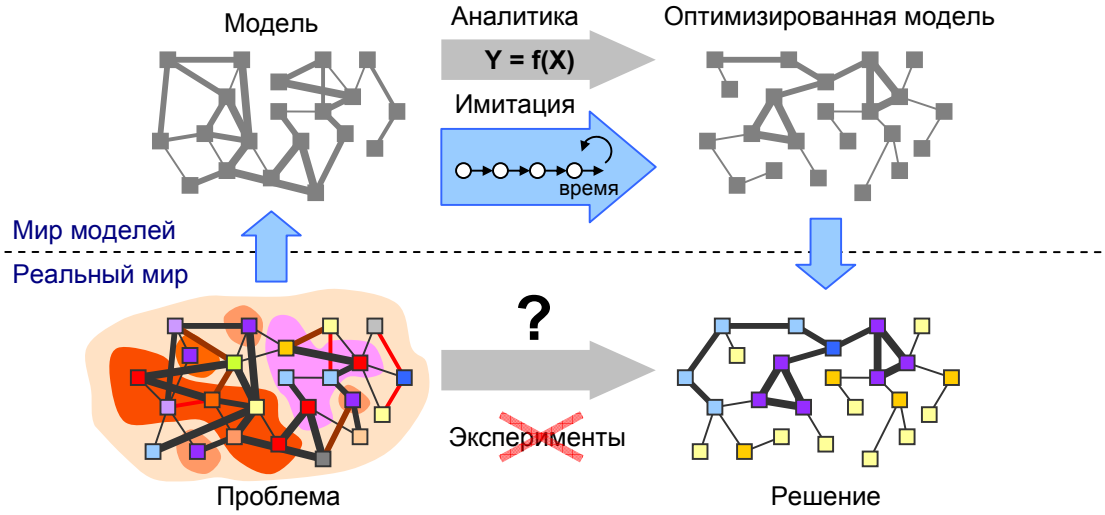


Рис 1: Аналитическое (статическое) и имитационное (динамическое) моделирование

Уровни абстракции в имитационном моделировании На Рис 2 показан примерный (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 мы имеем 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, станциями, операторами расположены выше, поскольку обычно здесь мы можем себе позволить абстрагироваться от точных физических траекторий и времён и использовать их усреднённые или стохастические значения. 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 и, естественно, поток пациентов и логика работы с ними, в то время как для отделения скорой помощи может быть учтена 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”.

Рис 3: Подходы в имитационном моделировании на шкале уровня абстракции

Основные подходы в имитационном моделировании На Рис 3 показаны основные подходы в имитационном моделировании: системная динамика (СД), дискретно-событийное моделирование (ДС), под которым мы понимаем 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 нашего рассмотрения. Математически, СД и динамические системы оперируют в основном с непрерывными во времени процессами, тогда как ДС и АМ – в основном с дискретными.

Динамические системы, естественно, находятся внизу шкалы. 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 (три “мира”), 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 походами с целью получения более полной картины взаимодействия сложных процессов различной природы. Отсюда спрос на программные платформы, позволяющие интегрировать различные подходы.

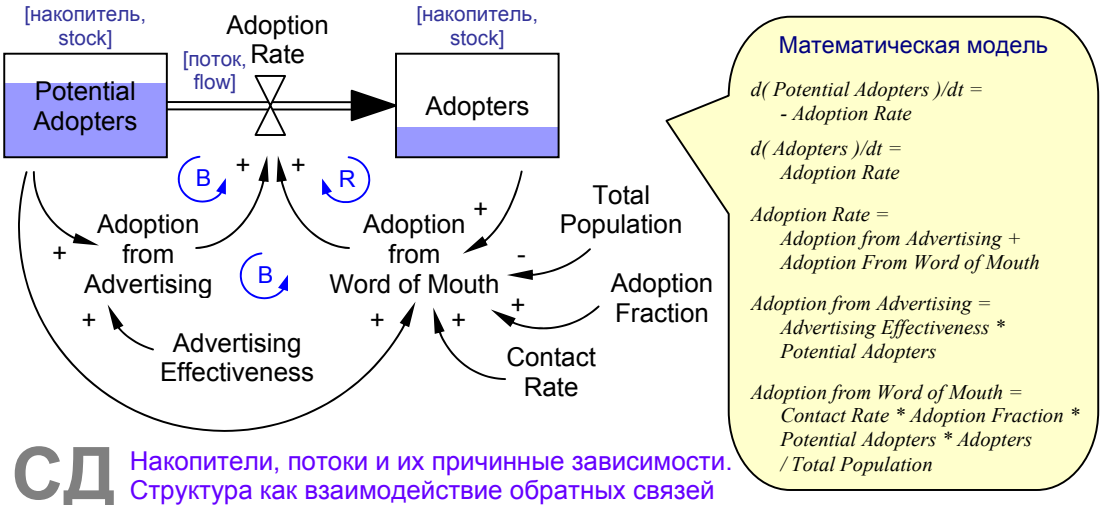


Рис 4: Классическая модель системной динамики: Bass Diffusion в Vensim™

Системная динамика Этот подход был разработан и предложен Джеем Форрестером в конце 1950х 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” [Forrester 1958 и 1961]. 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, и 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, как показано на Рис 4.

> **Модель Bass Diffusion** В этой классической модели распространения нового продукта, или инновации, взятой из учебника (Рис 4, [Sterman, 2000]) потенциальные клиенты (*Potential Adopters*) становятся клиентами (*Adopters*) со скоростью диффузии, распространения (*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, мы делаем предположение, что в нашей группе людей все контактируют со всеми. Количество контактов человека в единицу времени обозначим 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. Для нас важно отметить следующие моменты СД: а) поскольку модель оперирует только количествами, агрегатами, объекты, находящиеся в одном накопителе, неразличимы, лишены индивидуальности, и б) аналитику предлагается рассуждать в терминах глобальных структурных зависимостей и, соответственно, ему необходимы соответствующие данные. Подход СД поддерживается 3-4 инструментами весьма похожими друг на друга.

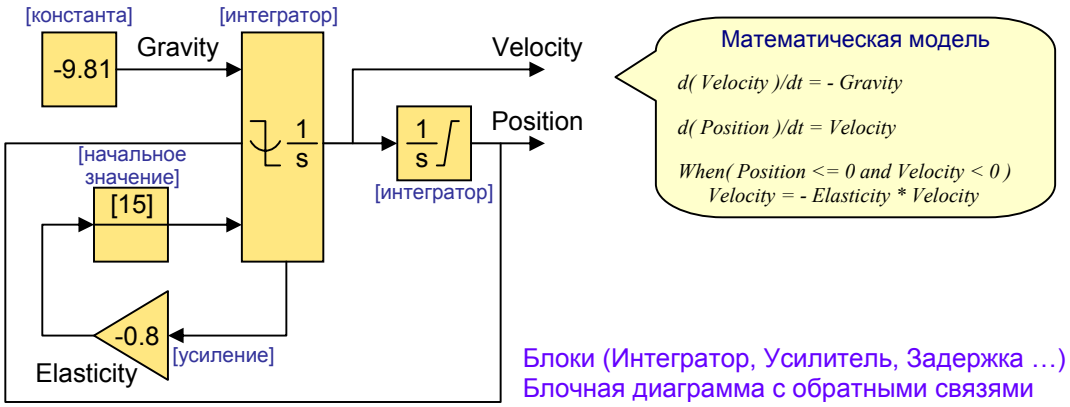


Рис 5: Модель динамической системы: прыгающий мячик в MATLAB™ Simulink™

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 использоваться разные визуальные и текстовые нотации. Соответствующая математическая модель, как и в случае СД, будет состоять 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 лучшей точностью (за счёт более совершенных численных методов). Однако, такие инструменты, “заточенные” под инженерные нужды, не являются удобными для СД-аналитиков и не используются ими: можно сказать, *они не поддерживают их привычного образа мышления*.

В нашем дальнейшем исследовании мы не будем касаться более динамических систем, хотя отметим мимоходом, что делались небезуспешные попытки моделировать молекулы газа в виде отдельных взаимодействующих агентов.

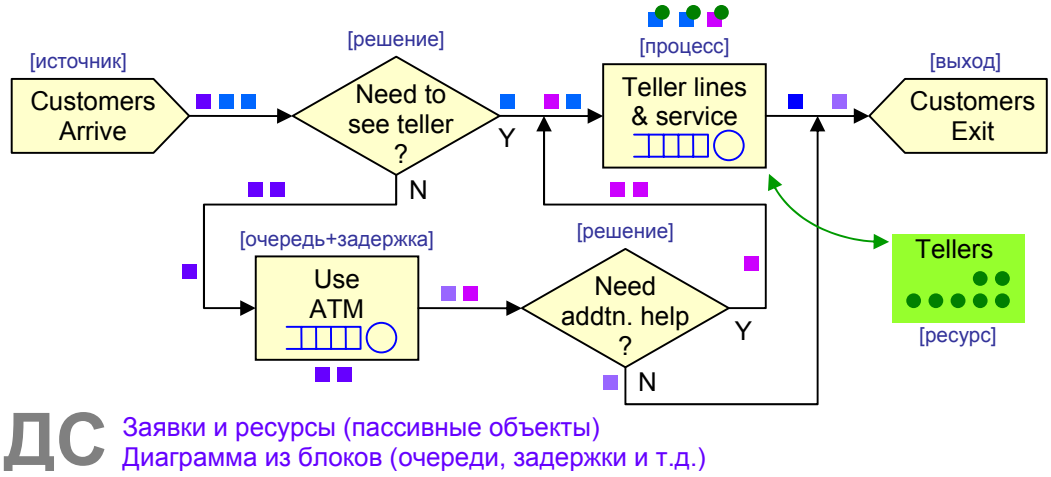


Рис 6: Дискретно-событийная модель: отделение банка в Arena™

Дискретно-событийное моделирование Мы зарезервируем этот термин за подходом, 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 1960х придумал и развил GPSS и реализовал её, работая в IBM [Gordon 1961]. 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, соединясь 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 нашего исследования важно отметить, что дискретно-событийную модель можно рассматривать как *глобальную схему обработки заявок*, обычно со стохастическими элементами.

Агентное моделирование Под этим лозунгом 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; люди 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. [Schieritz и Milling 2003]. Мы не собираемся предлагать здесь ещё одно определение агентов: агенты, которые реально используются нами в нашей консалтинговой модельной практике, бывают самые разные. Однако, есть нечто, что объединяет 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 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 в этой статье мы покажем, как АМ соотносится с СД и ДС, и как и когда оно может практически применяться.

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

Соотношения между подходами Мы теперь ограничимся рассмотрением систем, содержащих большие количества активных объектов (людей, животных, машин, предприятий или даже проектов, активов, товаров и т.п.), которые объединяет наличие элементов индивидуального поведения, от сложных (цели, стратегии) до самых простых (временные ограничения, события, взаимодействия). Мы собираемся продемонстрировать, что для систем подобного типа *агентное моделирование является подходом более универсальным и мощным, так как оно позволяет учесть любые сложные структуры и поведения*. Другое важное преимущество агентного моделирования в том, что *разработка модели возможна в отсутствии знания о глобальных зависимостях*: вы можете знать очень немного о том, как вещи влияют друг не друга на глобальном уровне, или какова глобальная последовательность операций, и т.п., но, понимая индивидуальную логику поведения участников процесса, вы сможете построить агентную модель и вывести из неё глобальные поведение. Таким образом, иногда, даже если в принципе и существует, скажем, СД модель системы, построить агентную модель может быть проще. И наконец,

агентную модель проще поддерживать: уточнения обычно делаются на локальном уровне и не требуют глобальных изменений.

2. Соотношение СД и агентных моделей

В этом разделе мы исследуем соотношение системно-динамических и агентных моделей. Для начала мы покажем, как “конвертировать” существующую СД модель в эквивалентную агентную модель, а затем – как можно развить последнюю, чтобы учесть боле сложную динамику реальной системы. Для понимания материала будет полезна нижеследующая справка.

> **Карты состояний (statecharts)** В нашей практике мы постоянно используем карты состояний для описания поведения агентов. Карта состояний – 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, [UML]). 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.
>
> Будет уместно сделать здесь вот какое замечание. В то время, как традиционные подходы в имитационном моделировании вроде СД или ДС практически не получили никакого существенного развития на протяжении последних десятилетий, в технологии разработки программного обеспечения произошли революционные изменения, в корне изменившие принципы работы со сложными системами. Значительная часть этого опыта бала 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, объектно-ориентированность (концепция классов и объектов, инкапсуляция) и т.п. могут сэкономить аналитику огромное количество времени и усилий, будучи применены при разработке имитационной модели.

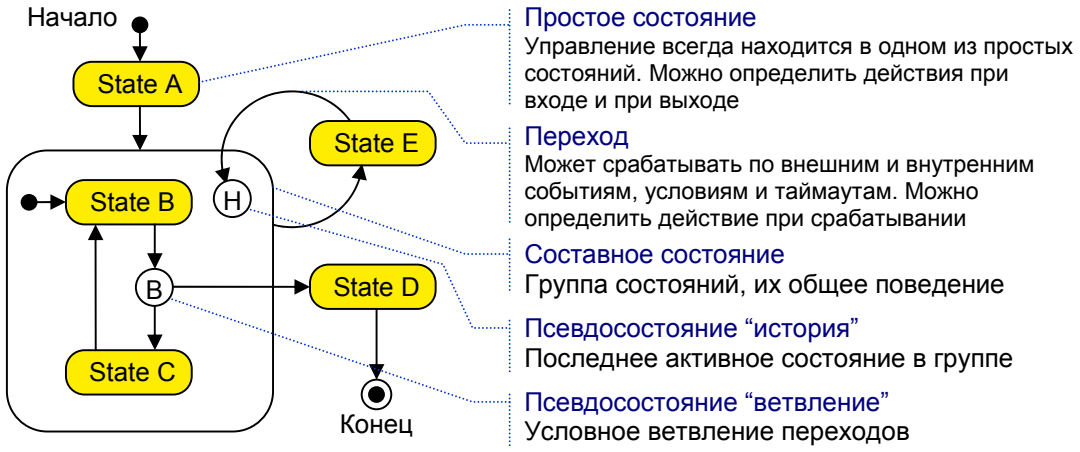


Рис 8: Карты состояний UML: язык для задания событийного и временного поведения

Давайте рассмотрим СД модель как цепочку накопителей и потоков и множество “правил”, контролирующих эти потоки, см. 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. 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*, вы 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*. Заметьте, 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 мы будем стараться использовать асинхронные агенты везде, где это возможно. Для более детальной информации о вышеописанной технике обратитесь к Приложению А.

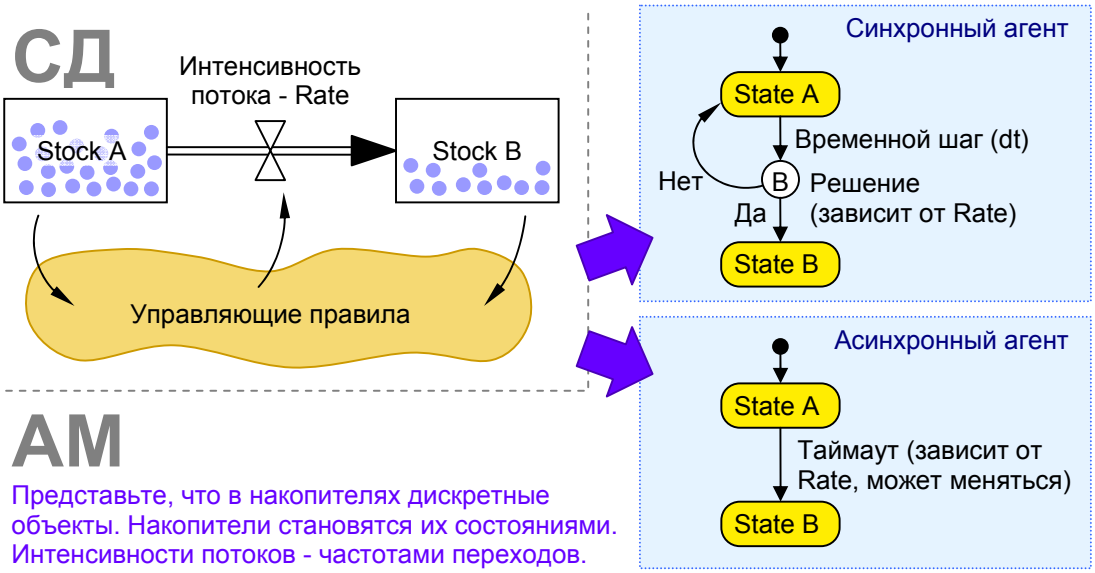


Рис 9: “Конвертация” СД модели в агентную модель. Общая схема

Модель Bass Diffusion – агентная версия Теперь мы проиллюстрируем эту технику на классической модели распространения инновации из знаменитого учебника Штермана по СД [Sterman, 2000], которую мы привели выше. Всё очень просто, посмотрите на Рис 10:

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

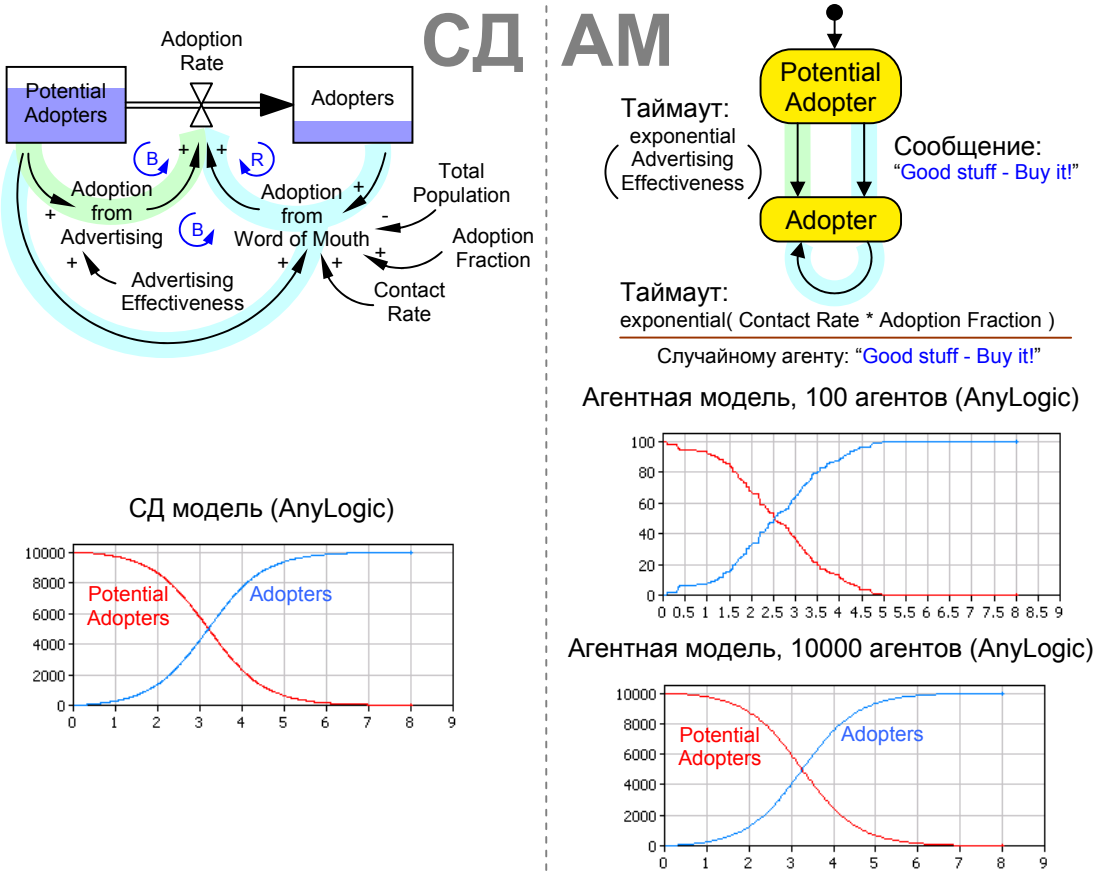


Рис 10: Модель Bass Diffusion, конвертированная из СД в агентную

Шаг 2. Две составляющие *Adoption Rate* будут моделироваться отдельно. Для влияния рекламы *Adoption from Advertising* мы 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 (см. также Приложение).

Шаг 3. Для "устной рекламы" *Adoption from Word of Mouth* мы добавим 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 попросту игнорировать. Наша 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.

> Вот один из часто задаваемых вопросов *"Сколько агентов я могу эффективно моделировать?"* Это, естественно, зависит от программного и аппаратного обеспечения, но более интересен и практически важен несколько другой вопрос: *"Сколько агентов мне действительно **нужно** моделировать?"* Например, если вас интересует динамика населения страны (300,000,000 человек), нужно ли вам имитировать 300,000,000 агентов? Ответ таков: в большинстве случаев это необязательно! Существует несколько методов уменьшения числа агентов, делающих моделирование вычислительно эффективным, и при этом сохраняющих корректные результаты. Однако, эта интереснейшая тема выходит за рамки данной статьи.

Процедура "конвертации" или, скорее, изменения концепции модели, которую мы только что рассмотрели, имеет практический смысл только в том случае, если вы собираетесь 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, вы, скорее всего, ничего не выиграете от такой конвертации*. 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 нашего исследования :).

Теперь поменяем в нашей 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, как показано на графике табличной функции *Influence* на Рис 11.

Для того, чтобы это учесть, в агентной модели требуется только одно небольшое изменение: мы добавляем в агент переменную *Time Purchased*, в которой запоминаем время покупки (см. действия на обоих переходах из состояния *Potential Adopter*), а также используем табличную функцию *Influence* вместо константы *Adoption Fraction*. Мы получаем поведение, показанное на графике на Рис 11. Естественно, кривая отличается от предыдущих из-за другой, более низкой эффективности устной рекламы.

Теперь вопрос: сможете ли вы учесть те же предположения в СД модели? Теперь доля каждого клиента в общей скорости распространения продукта (диффузии) *Adoption Rate* различна, да ещё и меняется со временем. Поэтому, агрегирую клиентов в одном (или любом разумном числе) накопителей, вы неизбежно исказите результат.

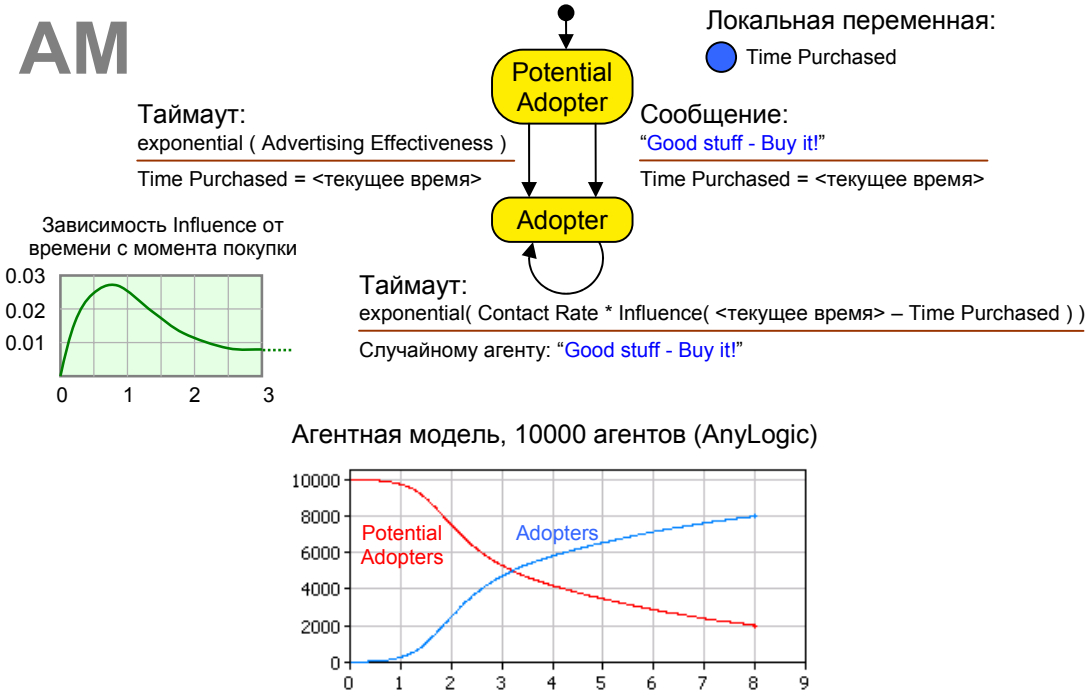


Рис 11: Агентная модель Bass Diffusion: учитываем меняющееся влияние устной рекламы

> Конечно же, системная динамика не сейчас столкнулась с подобными проблемами. Они были осознаны давно, и существует даже методика их частичного решения (а точнее, "обхода"), состоящая в замене одного накопителя массивом "вёдер", так что объекты с близкими свойствами попадают в одно "ведро" и перемещаются из одного ведра в другое при изменении этих свойств. Представьте себе, однако, что параметров, влияющих на свойства не один, а несколько. Тогда массив становится многомерным, и при увеличении сложности модели мы будем наблюдать грустное явление: "вёдер", *элементов накопителя станет больше, чем собственно моделируемых объектов* [Keenan и Paich 2004]. 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 Теперь мы используем другой, практически более значимый, подход для создания агентной версии другой знаменитой СД модели – хищники и жертвы (рыси и зайцы) [Lotka 1925 и Volterra 1926], см. Рис 12.

> **Модель хищники - жертвы** 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) в 1920х; 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) среда обитания не имеет размерностей, то есть любой хищник может встретить любую жертву.

Мы не будем на этот раз конвертировать СД модель, а построим агентную модель непосредственно “по постановке задачи” с предположениями, более близкими к реальности, чем перечисленные выше. В нашей агентной модели а) зайцы (hares) и рыси (lynx) имеют конечную продолжительность жизни, так что они умирают также и от старости, а не только будучи съеденными или от голода; б) зайцы и рыси живут в двумерном пространстве (в терминологии агентного моделирования говорят, что агенты “*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.

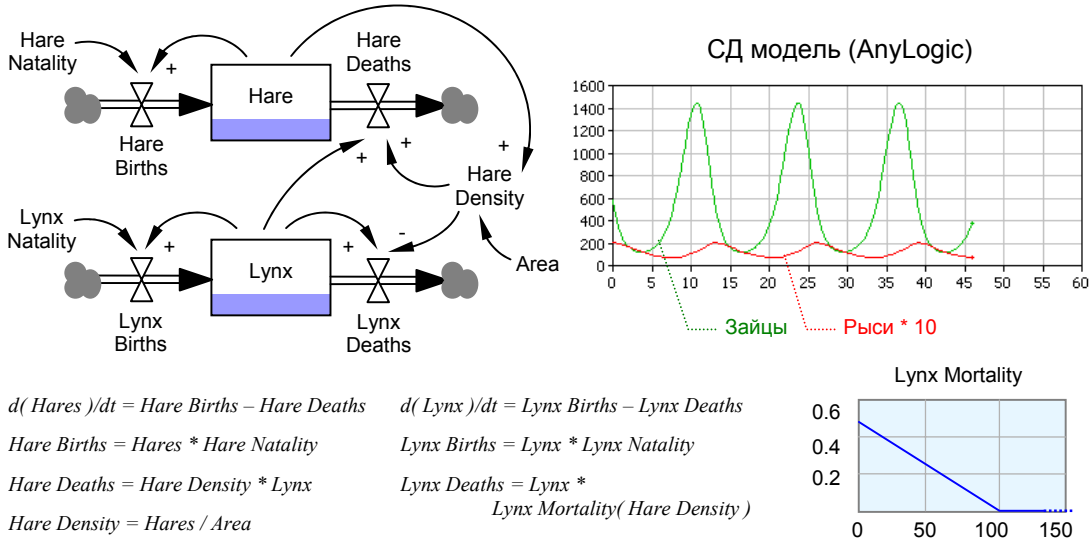


Рис 12: Классическая СД модель хищники - жертвы (Predator Prey)

Агент-рысь и агент-заяц на Рис 13 оба имеют переменные *Location*, где хранится их текущее местоположение; 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: возрасту и съедением рысью (последнее моделируется сообщением, которую 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” «таймаута голодной смерти» *Lynx*

Hunger Death Threshold. Таким образом, рыси нужен как минимум один заяц каждые *Lynx Hunger Death Threshold* единиц времени.

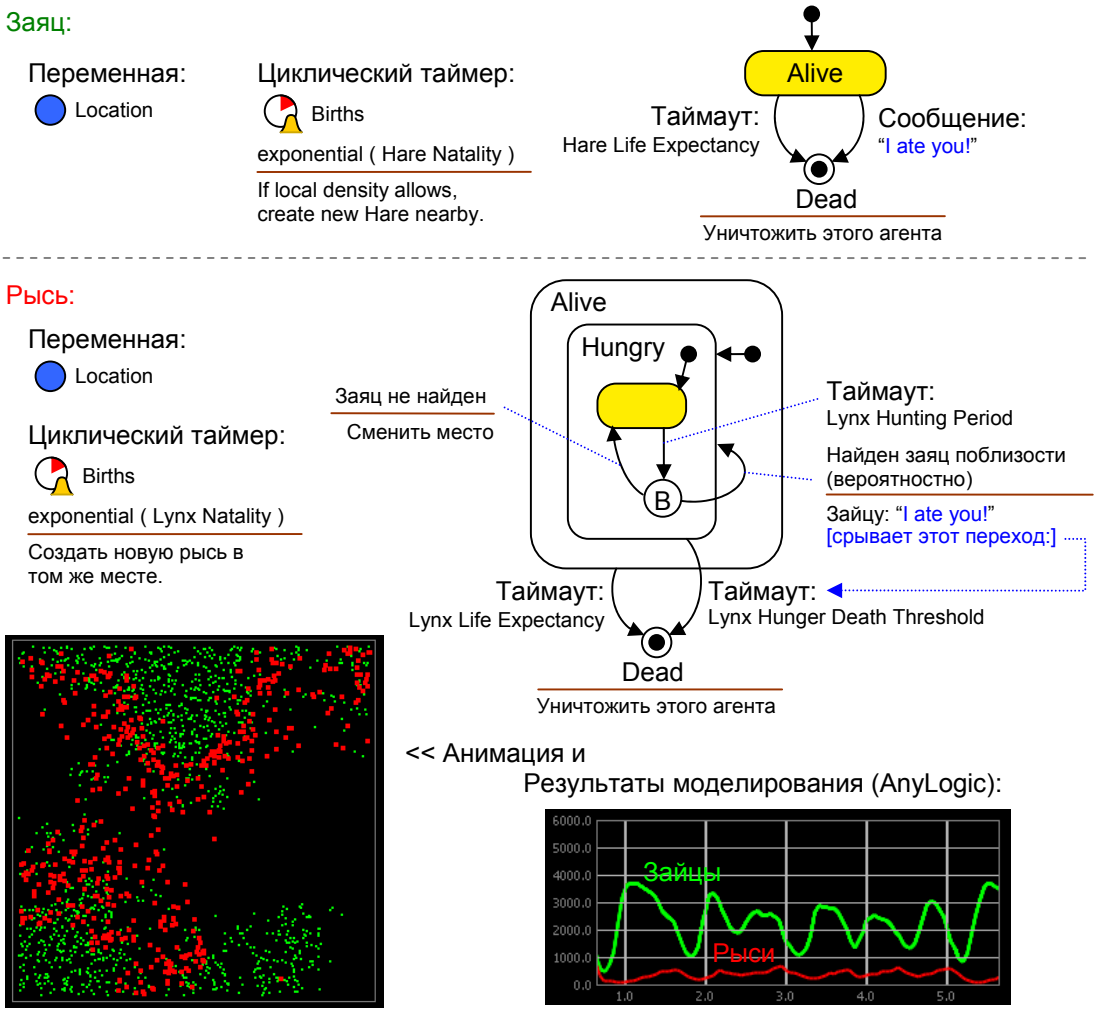


Рис 13: Агентная модель хищников и жертв (Predator Prey): более реалистичные предположения

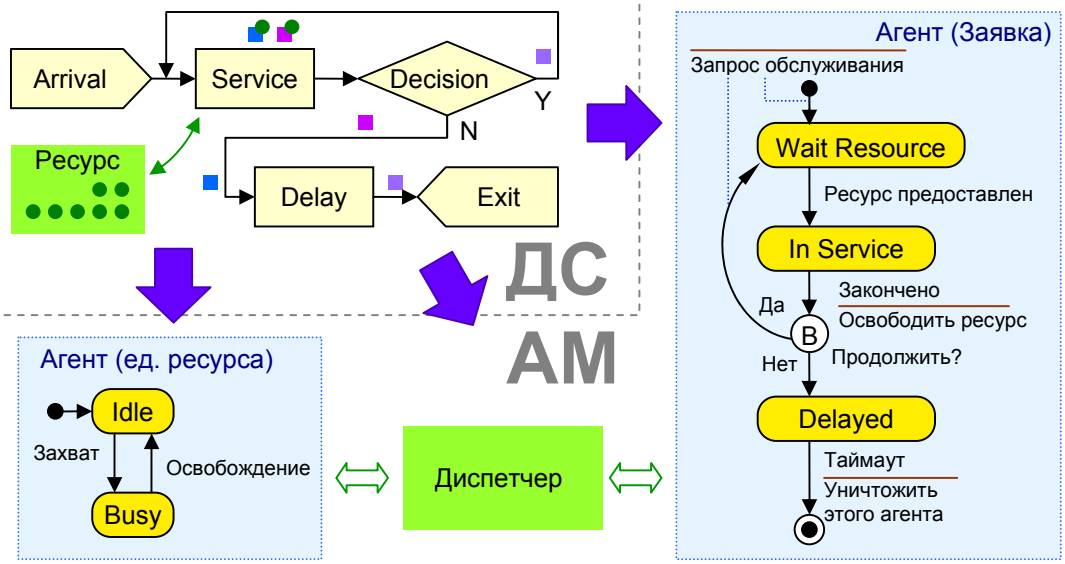
Имитация агентной модели даёт гораздо более богатый выход, чем СД. Действия разворачиваются на плоскости: видны атаки рысей, их вымирание там, где съедены все зайцы, и быстрое заполнение зайцами свободного от рысей пространства. На агрегатном (количественном) уровне модель показывает колебательное поведение, 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

Давайте теперь предпримем (чуть менее детальное) исследование соотношения дискретно-событийного и агентного моделирования. В ДС модели у нас уже есть индивидуальные объекты (заявки), что облегчает задачу: они естественно и станут агентами. В ДС-моделировании, однако, заявки пассивны; ими управляют правила, определённые в блоках потоковой диаграммы (flowchart). Таким образом, "упражнение" 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 вы собираетесь учесть в агентной модели какие-то индивидуальные поведения, не выражаемые в терминах ДС.



Посмотрите на процесс с точки зрения заявки (или единицы ресурса).
Каждая заявка (или единица ресурса) становится агентом.
Возможно, потребуется диспетчер для управления взаимодействием.

Рис 14: "Конвертация" ДС модели в агентную модель. Общая схема

Рассмотрим простую систему обслуживания на 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 нашем 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.

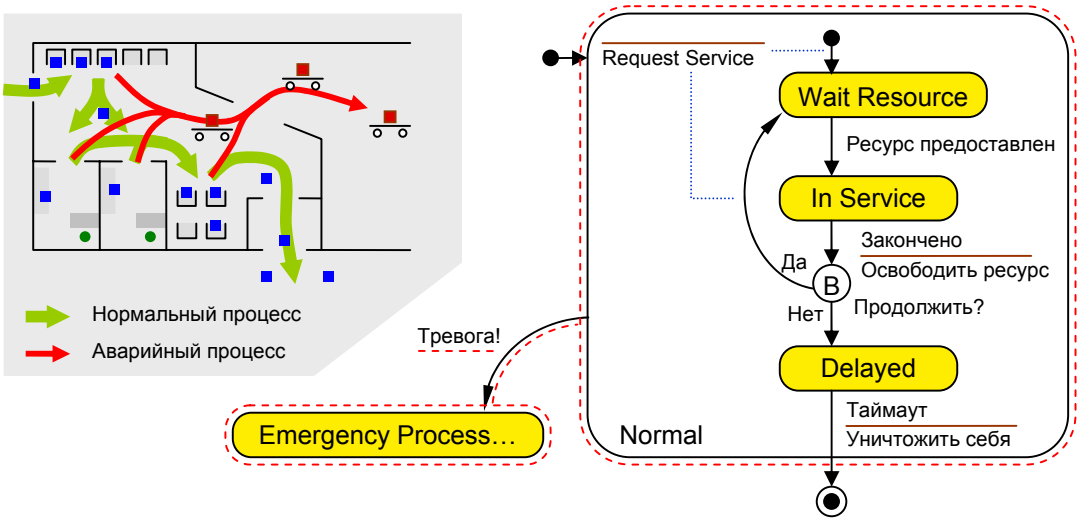


Рис 15: Агентная модель системы обслуживания: реакция агента на особое событие

Теперь, как и в примере Bass Diffusion, мы немного усложним постановку задачи. Пусть в любой момент во время нахождения объекта в системе может произойти нечто, что заставит его немедленно покинуть систему независимо от стадии обработки, см. 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 or C++, разработанные в различных университетах.

AnyLogic™ [AnyLogic] – инструмент, который используем мы 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 этого 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.

Рис 16: Инструменты имитационного моделирования

5. Примеры практических приложений АМ

Динамика употребления алкоголя В этой модели, разработанной совместно с Research Triangle Institute International [см. также Bobashev и др. 2004], мы исследуем отношение людей к алкоголю, продолжительность жизни и связанные с этим расходы на здравоохранение. Для этой статьи модель упрощена. Мы различаем четыре состояния у человека: не употребляет вообще *Never User*, употребляет время от времени *Recreational User*, алкоголик *Addict* и бросивший *Quitter*, см. Рис 17. Переходы между состояниями – это стохастические таймауты. 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.

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

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

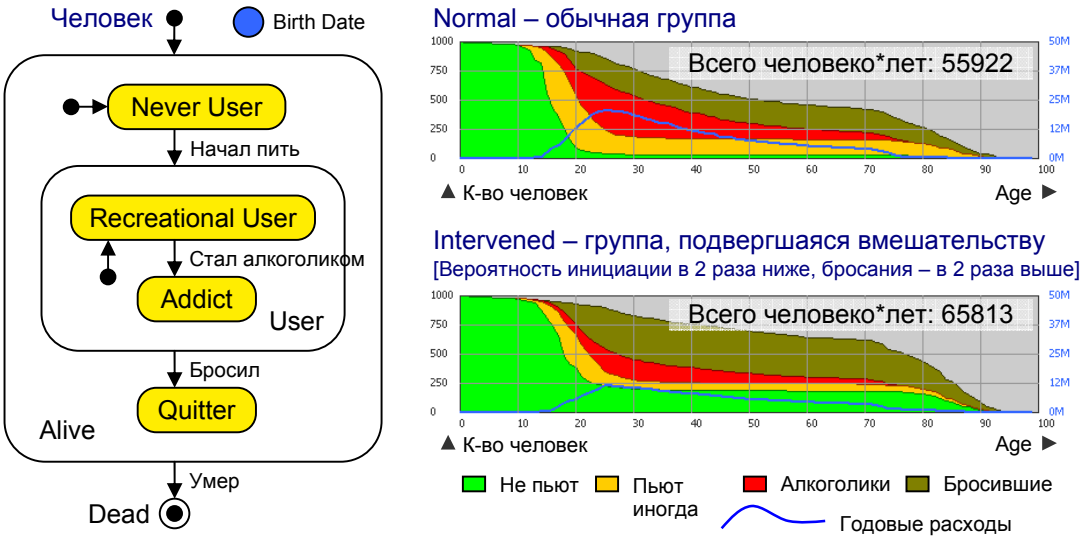


Рис 17: Агентная модель динамики употребления алкоголя

Моделирование поведения и ассимиляционной динамики испаноязычного населения США Эта модель [Wallis, Paich и Borshchev 2004] разработана консалтинговой компанией Decisio Consulting [Decisio] 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.

В модели имеется 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 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.

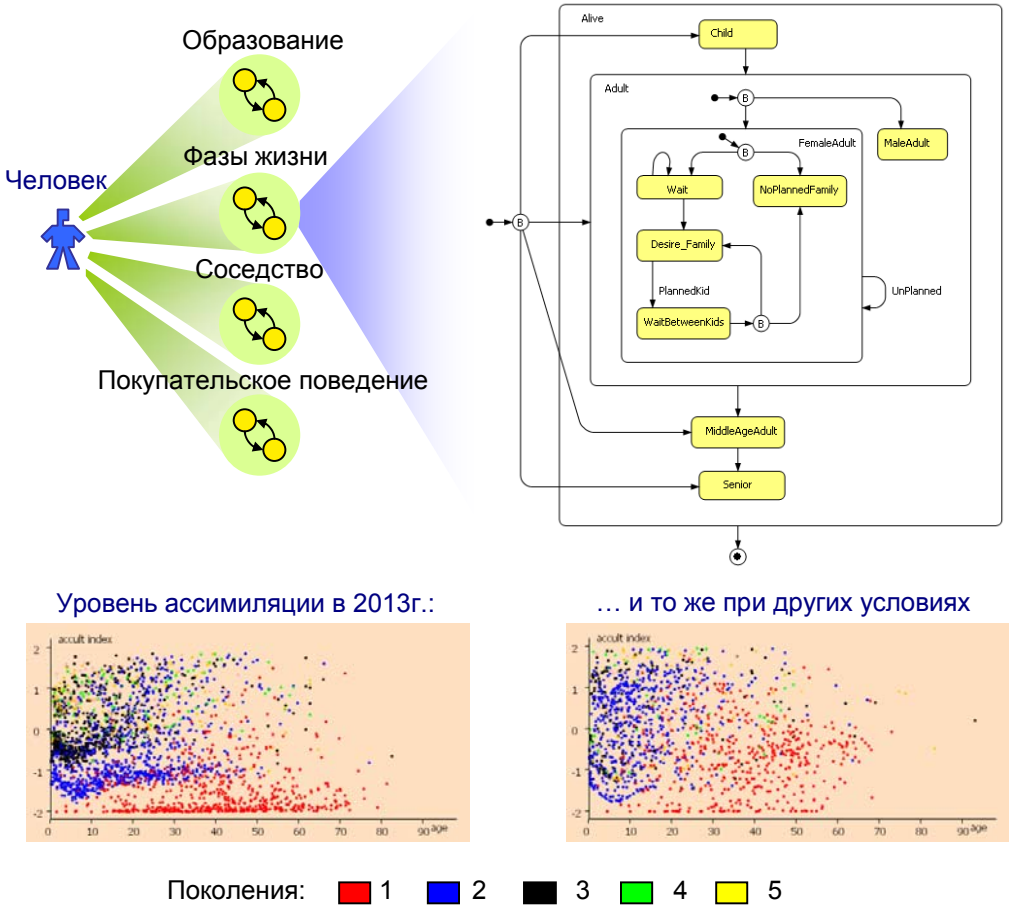


Рис 18: Агентная модель поведения и ассимиляционной динамики испаноязычного населения

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. 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.

6. 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 вы чувствуете, что задача хорошо подпадает под один из традиционных подходов, используйте его без колебаний. В этом вам поможет 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 Keenan и Paich 2004]. Оно особенно эффективно, как мы видели, при моделировании систем, содержащих большие количества активных объектов. Для таких систем AnyLogic поможет вам разработать агентную модель с минимальными усилиями, а также полностью или частично перейти от существующей СД или ДС модели к агентам.

Вы также не должны забывать о возможности использовать разные подходы для разных частей модели. Некоторые примеры комбинированных модельных архитектур приведены на Рис 19. Верхняя часто встречается в литературе пл моделированию цепочек (сетей) поставок [Schieritz и Grosler 2003]: 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 внизу может быть применена, скажем, при моделировании медицинских учреждений, если есть необходимость отслеживать более длительные периоды жизни пациентов и сотрудников.

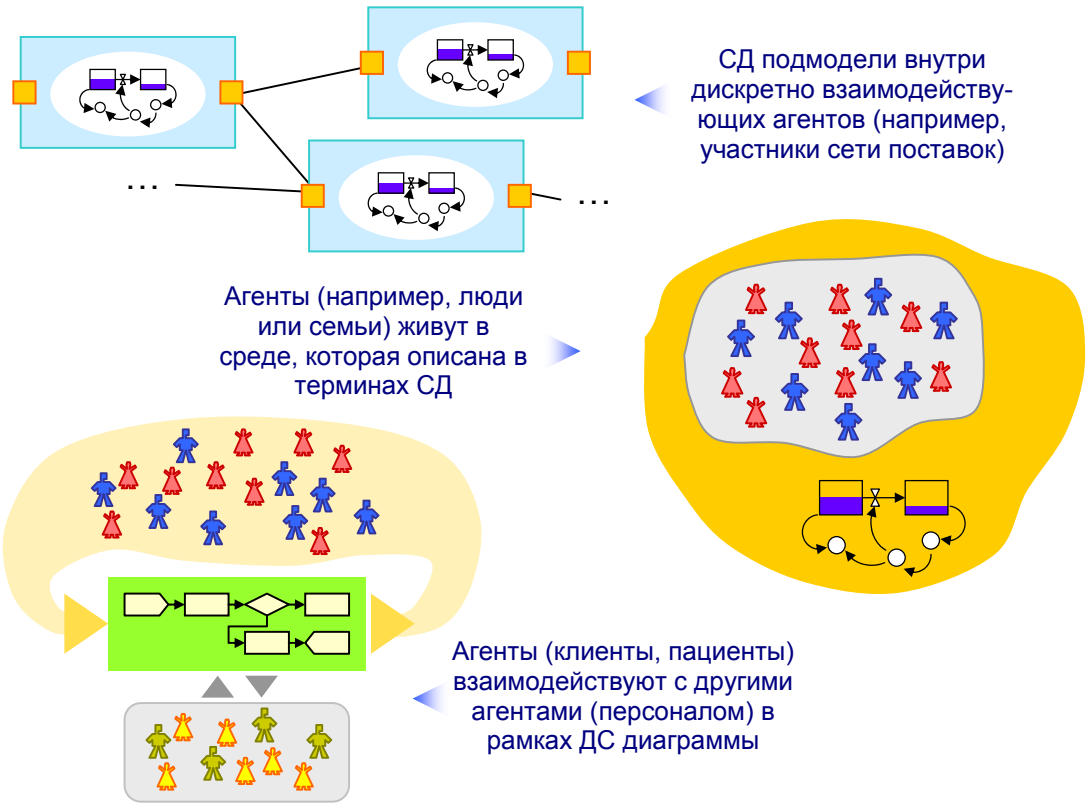


Рис 19: Комбинированные (многоподходные) модельные архитектуры

7. Литература

Forrester, Jay. 1958. *Industrial Dynamics: A Major Breakthrough for Decision Makers*. Harvard Business Review, Vol. 36, No. 4, 37-66.

Forrester, Jay. 1961. *Industrial Dynamics*. Cambridge, MA: MIT Press.

Sterman, John. 2000. *Business Dynamics: Systems Thinking and Modeling for a Complex World*. McGraw Hill.

Lotka, Alfred J. 1925. *Elements of physical biology*. Baltimore: Williams & Wilkins Co.

Volterra, Vito. 1926. *Variazioni e fluttuazioni del numero d'individui in specie animali conviventi*. Mem. R. Accad. Naz. dei Lincei. Ser. VI, vol. 2.

Gordon, Geoffrey. 1961. *A General Purpose Systems Simulation Program*. McMillan NY, Proceedings of EJCC, Washington D.C., 87-104.

Schelling, Thomas. 1978. *Micromotives and Macrobehavior*. W. W. Norton and Co.

Bobashev, Georgiy, Zule, William, Root, Elizabeth, Wechsberg, Wendee, Borshchev Andrei, and Filippov, Alexei. 2004. *Geographically-Enhanced Mathematical*

Models of HIV Dynamics. NIDA Symposium on AIDS, Cancer and Related Problems, St. Petersburg, Russia.

Bobashev, Georgiy, Zule, William, Root, Elizabeth, Wechsberg, Wendee, Borshchev Andrei, and Filippov, Alexei. 2004. *Scalable Mathematical Models for Substance Use: From Social Networks to the Whole Populations*. The College on Problems of Drug Dependence 66th Annual Meeting, San Juan, Puerto Rico.

Wallis, Lyle, Paich, Mark, and Borshchev, Andrei. 2004. *Agent Modeling of Hispanic Population Acculturation and Behavior*. The 3rd International Conference on Systems Thinking in Management (ICSTM 2004), Philadelphia, Pennsylvania, USA.

Schieritz, Nadine, and Milling, Peter. 2003. *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.

Schieritz, Nadine, and Grosler, Andreas. 2003. *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.

Solo, Kirk, and Paich, Mark. 2004. *A Modern Simulation Approach for Pharmaceutical Portfolio Management*. International Conference on Health Sciences Simulation (ICHSS'04), San Diego, California, USA. Available from http://www.simnexus.com/SimNexus.PharmaPortfolio.pdf.

Keenan, Philip, and Paich, Mark. 2004. *Modeling General Motors and the North American Automobile Market*. The 22nd International Conference of the System Dynamics Society, Oxford, England

UML – The Unified Modeling Language. http://www.uml.org.

AnyLogic. http://www.anylogic.com.

Decisio Consulting. http://www.decisio.net.

Приложение. Соотношение СД и агентных моделей

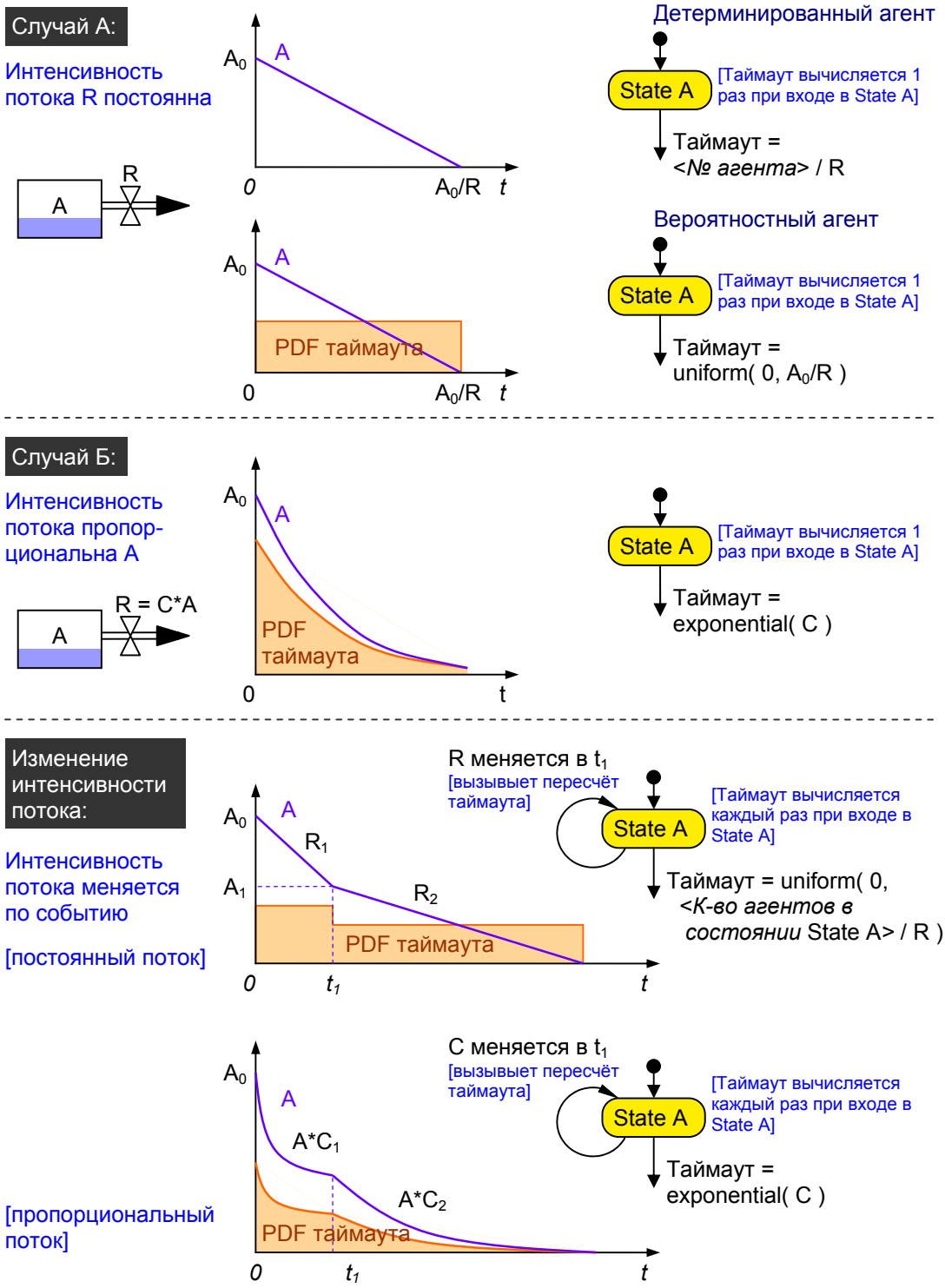


Рис 20: Соотношение СД и агентных моделей. Постоянные и пропорциональные потоки

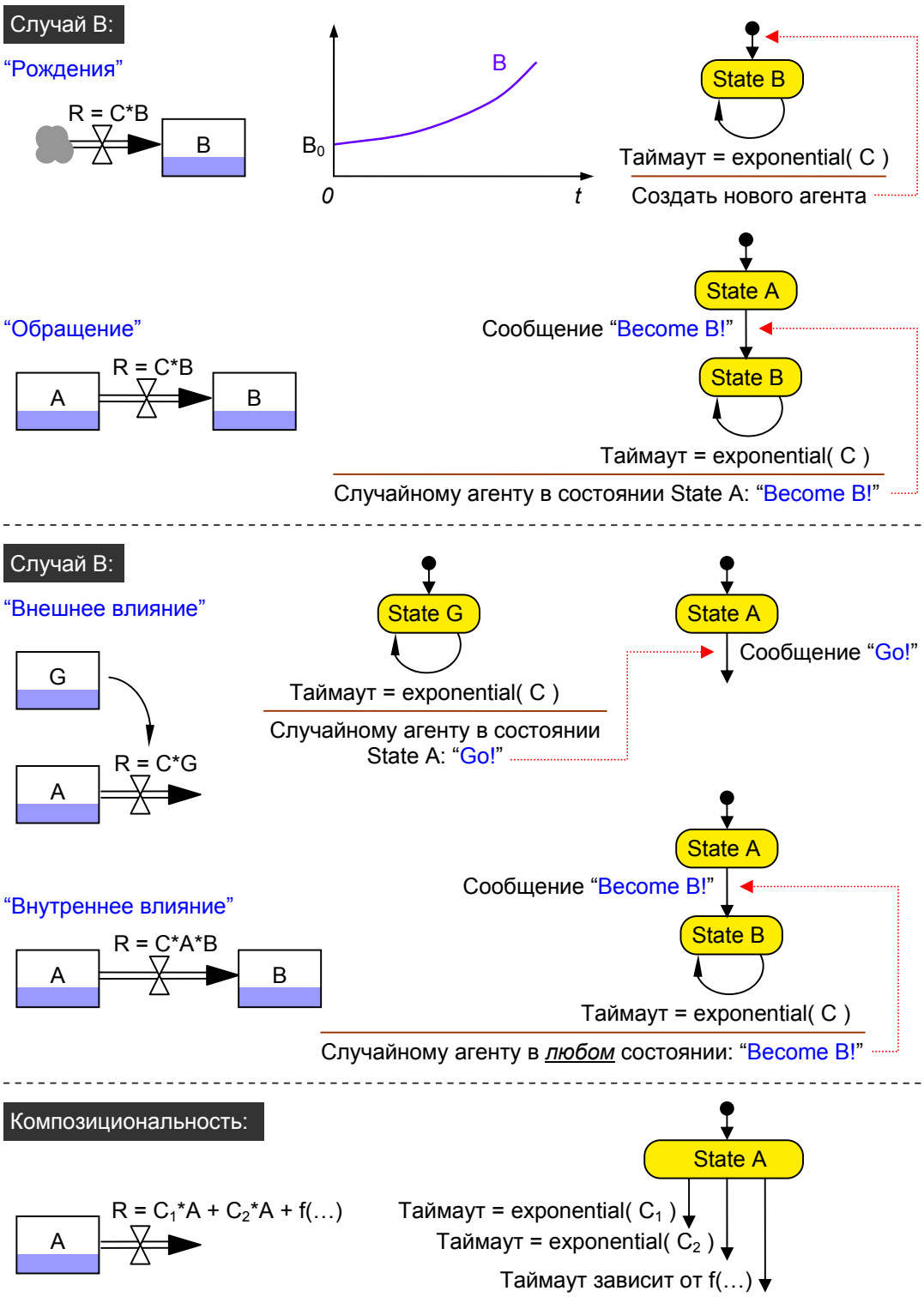


Рис 21: Соотношение СД и агентных моделей. Несколько накопителей. Композициональность