**Моделирование систем и языки моделирования.**

Алгоритмические языки при моделировании систем служат вспомогательным аппаратом в разработке машинной реализации и анализа характеристик моделей. *Основная задача* – это выбор языка.

Каждый язык имеет свою систему абстракций, лежащую в основе формализации функционирования сложных систем.

Для программирования модели могут использоваться следующие языки:

1. Универсальные алгоритмические языки высокого уровня.
2. Специализированные языки моделирования: языки, реализующие событийный подход, подход сканирования активностей, языки, реализующие процессно-ориентированный подход.
3. Проблемно-ориентированные языки и системы моделирования.

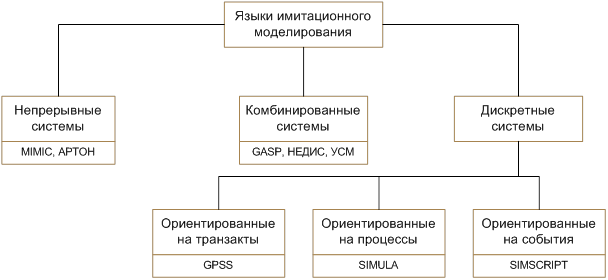
Качество языков моделирования характеризуется:

1. Удобство описания процесса функционирования;
2. Удобство ввода исходных данных, варьирования структуры, алгоритмов работы и параметров модели;
3. Эффективностью анализа и вывода результатов моделирования;
4. Простотой отладки и контроля работы моделирующей программы;
5. Доступностью восприятия и использования языка.

В большинстве своем языки моделирования определяют поведение систем во времени с помощью модифицированного событийного алгоритма. Как правило, он включает в себя список текущих и будущих событий.

*Классификация языков имитационного моделирования.*

Основа классификации – принцип формирования системного времени.



Непрерывное представление систем сводится к представлению дифференциальных уравнений, с помощью которых устанавливают связь между входной и выходной функциям. Если выходные переменные модели принимают дискретные значения, то уравнения являются разностными.

GASP – комбинированный, в основе лежит язык FORTRAN. Предполагается, что в системе могут наступать события двух типов:

* события, зависящие от состояния
* события, зависящие от времени.

Состояние системы описывается набором переменных, причем некоторые из них меняются непрерывно. При таком подходе пользователь должен составлять процедуры, описывающие условия наступления событий. Законы изменения непрерывных переменных, правила перехода от одного состояния к другому, т.е. реализуется классический принцип ДУ.

Группы языков моделирования, ориентированные на дискретное время, используется при построении именно имитационных моделей, но при этом используются разные способы описания динамического поведения исследуемого объекта.

*Формальное описание динамики моделируемого объекта.*

Будем считать, что любая работа в системе совершается путем выполнения *активностей*. Т.е. *активность* является наименьшей единицей работы и её рассматривают как единый дискретный шаг. Следовательно, активность является, единым динамическим объектом, указывающим на совершение единицы работ. *Процесс* – это логически связанный набор активностей.

Пример: активность установки головки жесткого диска, активность передачи информации с жесткого диска.

Активности проявляются в результате свершения событий. *События* – это мгновенное изменение состояния некоторого объекта системы. Рассмотренные объекты (активности, процессы, события) являются *конструктивными элементам* для динамического описания поведения системы. На их основе строятся языки моделирования системы. В то время, когда динамическое поведение системы формируется в результате выполнения большого числа взаимодействующих процессов, сами эти процессы образуют относительно небольшое число классов. Чтобы описать поведение системы, достаточно описать поведение каждого класса процессов и задать значение атрибутов для конкретных процессов.

Задачи построения модели.

Построение модели состоит из решения двух основных задач:

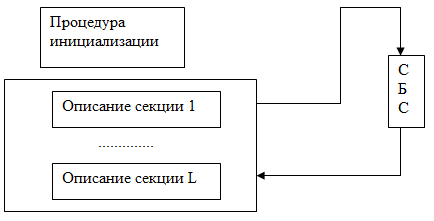
1. Первая задача сводится к тому, чтобы описать правила, описывающие виды процессов, происходящих в системе.
2. Вторая задача заключается в том, чтобы описать правила задания атрибутов или задать правила генерации этих значений. При этом система определяется на конкретном уровне детализации в терминах множества описаний процессов, каждый из которых в свою очередь включает множество правил и условий возбуждений активности. Такое описание системы может быть детализировано на более подробном или более иерархическом уровне представления с помощью декомпозиции процессов (в идеальном случае в активности). Это обеспечивает многоуровневое исследование системы.

Т.к. система в общем случае служит для описания временнОго поведения, то язык моделирования дискретных систем должен обладать средствами, отображающими время. В реальной системе совместно выполняются несколько активностей, принадлежащим как связанным, так и не связанным процессам. Имитация их действий должна быть строго последовательной. Таким образом, модель системы можно рассматривать как модель описаний, активностей, событий или процессов. Отсюда и деление языков моделирования.

*Языки, ориентированные на события.*

Моделирующая программа организованна в совокупности в виде секций событий (процедуры отражающие события). Процедура состоит из набора операций, которые выполняются после выполнения какой-либо активности. Синхронизация происходит с помощью списка будущих событий.

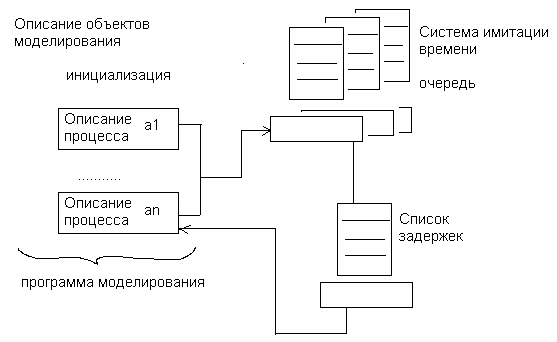
Структуру рассмотрим на примере языка SIMSCRIPT.



*Языки, ориентированные на процессы.*

Моделирующая программа организуется в виде набора описаний процесса. Каждый из которых описывает одни класс. Описание процесса функционирования устанавливает атрибуты и активности всех процессов. Синхронизация операций во времени реализуются так же с помощью списка будущих событий, который содержит точку возобновления конкретного процесса (точка прерывания).

На примере языка SIMULA:

**

Результаты экспертных оценок сравнения различных языков при моделировании большого класса систем.

Критерии:

1. *Возможность языка*. Выше всех находится SIMULA -> SIMSCRIPT -> GPSS -> C -> PASCAL
2. *Простота применения*: GPSS -> SIMSCRIPT -> SIMULA -> C -> PASCAL
3. *Предпочтение пользователей*: GPSS -> SIMSCRIPT -> SIMULA -> PASCAL -> C

GPSS (Лабу которую уже выдали, реализована на этом языке):

GEN 3,1

QUEUE 1

SEIZE 1

DEPART 1

ADVANCE 3, FN$XPDIS

RELEASE 1

START 100

// говорит здесь и ошибка есть

При использовании универсальных алгоритмических языков программист имеет практически неограниченные возможности по созданию имитационной модели, наилучшим образом используя ресурсы системы, особенности ОС и т.д. Но в то же самое время требует больших трудозатрат, работы программистов высокой квалификации, взаимодействие специалистов разных профилей (программисты, эксперты предметной области и т.д.). В результате модель получается узко направленной и, как правило, использование этой модели в других разработках невозможно.

При использовании языков имитационного программирования снижается гибкость и универсальность. Модель создается в несколько раз быстрее и не требует присутствие системы от программистов. Модели обладают свойством концептуальной выразительности. Используются специальные термины языка в области, исследуемой  задачи.

Алгоритм сканирования активности включает в себя цикл по всем действиям активностей, дальше срабатывает проверка выполнения действия. И если условие выполняется, то выполняется действие, изменяющее состояние модели. Иначе цикл по всем действиям заканчивается.

*Сравнение универсальных и специализированных языков программирования при моделировании:*

|  |  |
| --- | --- |
| Преимущества | Недостатки |
| Универсальные |  |
| Минимум ограничений на выходной формат | Значительное время, затрачиваемое на программирование |
| Широкое распространение | Значительное время, затрачиваемое на отладку |
| Специализированные |  |
| Меньше затраты времени на программирование | Необходимость точно придерживаться ограничений на форматы данных |
| Более эффективные методы выявления ошибок | Меньшая гибкость модели |
| Краткость, точность понятий, характеризующих имитируемые конструкции |  |
| Возможность заранее строить стандартные блоки, которые могут использоваться в любой имитационной модели |  |
| Автоматическое формирование определенных типов данных, необходимых именно в процессе имитационного моделирования |  |
| Удобство накопления и представления выходной информации |  |
| Эффективное использование ресурсов |  |

*Основные концепции языка РДО (Ресурсы, действия, операции).*

Язык придуман в МГТУ.

Причина создания РДО:

* Универсальность имитационного моделирования относительно класса исследуемых систем и процессов.
* Легкости модификации моделей
* Моделирование сложных систем управления совместно с управляемым объектом (включая использования имитационного моделирования в режиме реального времени)

Язык РДО является реализацией т.н. *интеллектуального подхода* к имитационному моделированию. При этом подходе стараемся отойти от жесткого алгоритмического подхода в процессе принятия решения при моделировании для удобства автоматизирования той части процесса, где используются знания человека, и сделать процесс моделирования максимально гибким по способам представления информации об объекте. В основе языка РДО лежит продукционная система, которая состоит из трех элементов: классов и отношений, правил, управляющей структуры.

*Классы и отношения* трактуются как база данных, содержащая декларативные знания. Процедуры представляют собой набор модифицированных продукционных правил: ЕСЛИ – ТО ДЕЙСТВИЕ.

*Управляющая структура* – это интерпретатор правил, управляющий выборкой правил.

*Условие* – это проверка состояние базы данных.

*Действие* – изменяет  некоторым образом базу данных.

Достоинства системы основанной на продукциях:

* Простота создания и понимания, отдельных правил
* Легкость модификации

Недостатки:

* Неясность взаимных отношений правил
* Сложность оценки целостного образа знаний
* Низкая эффективность обработки

Для имитационного моделирования основным недостатком системы продукции является отсутствие времени, поэтому такие продукционные правила применимы для анализа статических объектов, т.е. правила описывают причинно следственные связи и не описывают динамику процесса. Поэтому при интеллектуально моделировании, т.е. для моделирования динамики системы, на основе знаний о протекающих процессах в РДО используют модифицированное продукционное правило.

Как оно выглядит:

*ЕЛСИ <УСЛОВИЕ>, ТО <СОБЫТИЕ 1> … ЖДАТЬ (отведенный интервал) … ТО <СОБЫТИЕ 2>*

СОБЫТИЕ 2 наступает через некоторый интервал времени.

Если в течение указанного времени происходит нерегулярное событие, затрагивающее релевантное данному событию ресурсы, то событие 2 может не наступить или наступить в другой момент времени.

В РДО сложная дискретная система представляется в виде множества взаимодействующих между собой *ресурсов*.

*Ресурс* – это элемент сложной системы, внутренней структурой которого можно пренебречь, в то время как наличие и свойства его, как целого, важны и существенны для описания.

Каждый ресурс модели должен быть описан и должен иметь свое уникальное имя.

Ресурсы могут быть двух типов:

1. *Постоянные*. Они всегда присутствуют в системе.
2. *Временные*. Поступают в систему и покидают её в процессе функционирования. Причем они могут быть и результатом работы, т.е. здесь наличие обратной связи.

Все ресурсы системы образуют некоторое множество: R(t) = {ri: i = 1, …, N(t)}

Где Ri – i-ый ресурс сложной дискретной системы

N(t) – число ресурсов в текущий момент времени.

Каждый ресурс описывается множеством параметров, которые могут быть следующих типов:

1. *Описательные*, представляющие факты внутренне присущие каждому ресурсу.
2. *Указывающие*, используемые для дачи имени или обозначения ресурса, проще говоря, идентификаторы.
3. *Вспомогательные*, используемые для связи различных ресурсов, накопления статистики, графического вывода при имитации и т.д.

Состояние ресурса в момента времени t: Ci(t) = {Cij(t): j = 1…Mi}

Где Cij – значение j-ого параметра i-ого ресурса

Mi – число параметров i-ого ресурса

Состояние всей системы является совокупностью состояний всех ресурсов.

Ресурсы, принадлежащие к одному типу, наследуют общие свойства этого типа. Отношения наследования может использоваться как для отображения общности ресурсов, так и для идентификации ассоциативных связей.

Ресурсы взаимодействуют друг с другом в соответствии с определенным алгоритмом. Каждое действие связано с изменением состояния системы, которое связано с конкретным событием. *Все события должны быть определены и зафиксированы в модели*. Они могут быть *внешними* и *внутренними* по отношению к системе.

В РДО событие происходит в счетные моменты времени, которые фиксируются в модели с помощью независимой переменной. Эта переменная изменяется дискретно и служит базой для определения различий в наблюдении одного и того же свойства. Все события делятся на *регулярные* и *нерегулярные*.

*Регулярные* – это события, вызываемые штатным функционированием ресурсов. Они выражают логику взаимодействия ресурсов между собой.

*Нерегулярные события* происходят либо при нештатной работе (поломка или отказ) ресурса, либо из-за внешних по отношении к системе причин, т.е. в систему пришел новый временный ресурс.

В отличие от регулярных событий, нерегулярные носят стахостическийхарактрет.

Ресурсы в процессе функционирования выполняют определенные действия. С каждым действием связано два события:

1. Время начала

2. Время окончания

*Действие* представляет собой целенаправленное мероприятие, выполняемое под управлением некоторой подсистемы и направленное на достижение определенной цели. Поэтому действие планируется и может находиться в следующих состояниях:

· Запланировано

· Начато

· Окончено

· Прервано по какой-либо причине

Действие с нулевой длительностью представляет собой *событие*.

Процесс функционирования сложной дискретной системы строится как временная последовательность действий и нерегулярных событий. Действие может начинаться только в том случае, если значение параметров его релевантных ресурсов удовлетворяет необходимому условию.

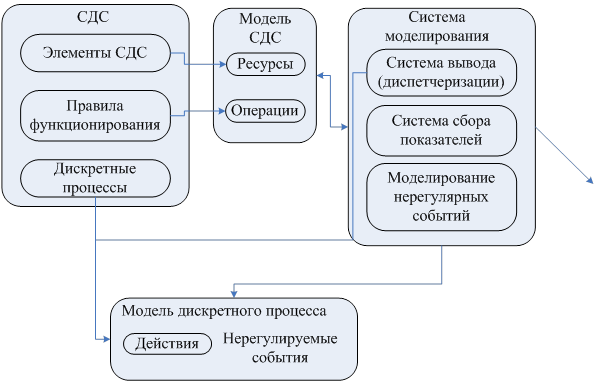
В языке используется понятие *виртуального действия*, которое не привязано ни к началу, ни к окончанию, а привязано только к необходимому условию начала действия.

Множество виртуальных действий может быть разбито на небольшое число подмножеств, имеющих одинаковую природу, т.е. одинаковую логику взаимодействия ресурсов и различаются лишь конкретно участвующими в них ресурсами. Для формального описания логики однотипных виртуальных действий используются понятия *операций*.

*Операция* – это фактически процедура, у которой формальные параметры условия выполнения и алгоритмы, а в качестве фактических параметров выступают подмножества ресурсов. При задании фактических ресурсов элементами соответствующих подмножеств однотипных ресурсов из операции получается виртуальное действие. Т.е. операция отражает логику взаимодействия ресурсов. Всякий раз, когда состояние соответствует условию срабатывания виртуального действия, происходит действие описывающее соответствующей операции с различными временами начала и окончания.

Следовательно, *операция* описывает, как происходит действие или виртуальное действие и с какими множествами релевантных ресурсов. Т.е. что может произойти в сложной системе при определенных условиях. А *действие* – что произошло, происходит, произойдет и в какое время.

Представление сложной дискретной системы в РДО методе.



СДС – сложная дискретная система

Основные моменты этого метода:

Все элементы сложной дискретной системы представлены как ресурсы.

Основные положения РДО-метода формируются следующим образом:

1. Каждый ресурс определенного типа описывается одними и теми же параметрами.
2. Состояние ресурса определяется вектором значений всех его параметров. Состояние системы в целом – совокупностью всех его параметров.
3. Процесс протекающий в сложной системе описывается как последовательность целенаправленных действий и нерегулярных событий, изменяющих состояние ресурсов. Действия ограничены во времени событиями начала и конца.
4. Нерегулярные события описывают изменение состояния сложной системы не предсказуемые в рамках продукционной модели системы. Моменты наступления нерегулярных событий случайны.
5. Действия описываются операциями, которые представляют собой модифицированные правила, учитывающие временные параметры. Операция описывает предусловия, которым может удовлетворять состояние, участвующих в операциях ресурсов. И правила изменения состояния ресурсов в начале и конце соответствующего действия.
6. Множество ресурсов и множество операций образуют модель сложной дискретной системы.

Лабораторная работа №5.

Создать концептуальную модель функционирования метро «Бауманская» в час пик. Структура включает в себя:

Равномерное прибытие поездов, как в противофазе, так и вместе.

3 эскалатора (Супербабушка может включить два верх, один вниз и наоборот)

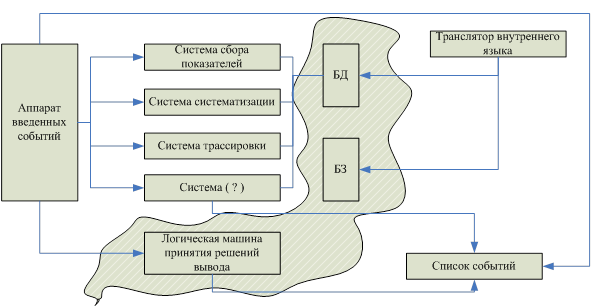
Выходные двери от 4 до 1.

Вся статистика: сколько времени стоишь в очереди и т.п.

Определить, куда лучше втискиваться в очередь, т.е. где лучше всего находиться…

Модель сложной дискретной системы в РДО представляет собой динамическую продукционную систему. *Базой данных* этой продукционной системы является *множество ресурсов*, *базой знаний* – *множество операций*. Параметрическая настройка в конкретной системе заключается в формализованном описании ресурсов и операций на внутреннем языке и введении в соответствующую базу знаний и базу данных.

Структура РДО имитатора:



АВС – аппарат введения событий

*Основными элементами* РДО имитатора является: динамическая продукционная система и аппарат событий. Действия имитируются самой системой вывода, а нерегулярные события имитируются специальным блоком.

При имитации состояний системы изменяются в соответствии с описанием нерегулируемого события, либо действия, которое началось и завершилось. После любого изменения состояния, т.е. при каждом событии вызывается система вывода. Она просматривает в базе знаний все операции и проверяет у предусловия, могут ли они начаться. При нахождении таких операций инициируются события начала соответствующих действий. Продукционная система (БД, БЗ и Система Вывода, т.е. принятия решений), система имитаций нерегулярных событий и аппарат ведения событий совместно осуществляют процесс построения модели. На основании результатов анализа имитации, вычисляются различные показатели функционирования модели. Система трассировки вводит подробную информацию о событиях в специальный файл, по содержимому которого принимается решение о проведении математического эксперимента.

Система анимации позволяет отображать на экране поведение системы во время моделирования.

РДО имитатор создавался как средство создания имитационных моделей систем планирования, игр и тренажеров. Кроме того, на языке РДО могут быть реализованы экспертные системы, а так же гибридные системы включающие в себя экспертную составляющую, имитирующую модель и алгоритмы оптимизации.

В языке используются следующие понятия:

1. Модель – совокупность объектов РДО языка, определяющих реальный объект, собираемый в процессе имитации показателей, кадры анимации, различные графические элементы, результаты трассировки.
2. Прогон – единая неделимая точка имитационного эксперимента. Характеризуется совокупностью объектов представляющих собой исходные данные и результаты.
3. Проект – один или более прогонов, объединенных какой-либо общей целью.
4. Объект – совокупность информации предназначенной для определенных целей и имеющей смысл для имитационной программы. Состав объектов обусловлен РДО методом, определяющим парадигму представления сложной дискретной системы на языке РДО. Описание объекта в зависимости от типа разделяются по разным модулям.

Объекты исходных данных:

· типы ресурсов,

· образцы операций,

· операции,

· точки принятия решений,

· константы,

· функции,

· последовательности,

· кадры анимации,

· требуемая статистика,

· результаты трассировки.

1. Комментарий прогона – произвольный текст, предназначенный для хранения сопроводительной информации прогона.
2. Комментарий проекта – текстовая информация, характеризующая проект.

Подготовка исходных данных моделирования и анализ результатов производится с помощью интегрированной среды РДО, которая имеет следующие возможности:

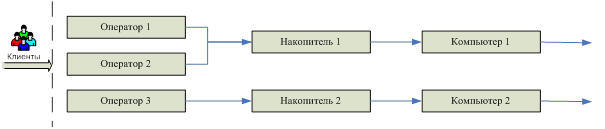
* 1. Реализует графический объектно-ориентированный интерфейс
  2. Позволяет структурировать информацию по имитационным исследованиям
  3. Автоматически поддерживает целостность информации
  4. Режимы работы эксперта и пользователя

В рамках одной среды объединены как средства описания исходных данных, так и средства моделирования и анализа результатов. Предусмотрена возможность комментирования отдельных имитационных исследований и прогонов.

Лабораторная работа №6.

В информационный центр приходят клиенты через интервал времени 10±2 минуты. Если все три имеющихся оператора заняты, клиенту отказывают в обслуживании. Операторы имеют разную производительность и могут обеспечивать обслуживание среднего запроса пользователя за 20 +- 5; 40 +- 10; 40 +- 20. Клиенты стремятся занять свободного оператора с максимальной производительностью. Полученные запросы сдаются в накопитель. Откуда выбираются на обработку. На первый компьютер запросы от 1 и 2-ого операторов, на второй – запросы от 3-его. Время обработки запросов первым и 2-м компьютером равны соответственно 15 и 30 мин. Промоделировать процесс обработки 300 запросов.

Необходимо для этого создать концептуальную модель в терминах СМО, определить Эндогенные и Экзогенные переменные и уравнения модели. За единицу системного времени выбрать 0,01 минуты.



В процессе взаимодействия клиентов с информационным центром возможно:

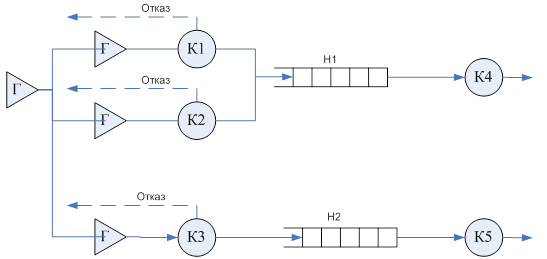
1) Режим нормального обслуживания, т.е. клиент выбирает одного из свободных операторов, отдавая предпочтение тому у которого меньше номер.

2) Режим отказа в обслуживании клиента, когда все операторы заняты

Переменные и уравнения имитационной модели.

Эндогенные переменные: время обработки задания i-ым оператором, время решения этого задания j-ым компьютером.

Экзогенные переменные: число обслуженных клиентов и число клиентов получивших отказ.





Вместе с этой лекцией читают "[4. Экономико-географическое направление](https://studizba.com/lectures/33-nauki-o-zemle/602-socialno-ekonomicheskaya-geografiya/11437-4-ekonomiko-geograficheskoe-napravlenie.html)".

*AnyLogic™*

AnyLogic™ имитационного моделирования которой позволяет моделировать при помощи визуальных компонент как стандартных, так и разработанных пользователем. Программировать иерархические структуры на разных уровнях абстракции.

Создавать интерактивные 2 и 3D анимации визуально отображающие результаты работы модели в реальном времени.

Увеличить жизненный цикл модели.

Использовать средства анализа и оптимизации непосредственно из среды разработки модели.

Достаточно просто интегрировать модель открытой архитектуры с офисными и корпоративными программными продуктами (Электронные таблицы, БД и БЗ, CRM и т.д.)