МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

#### Учреждение образования

Гомельский государственный университет имени Франциска Скорины

##### Математический факультет

Кафедра математических проблем управления

Имитационная модель автоматизированного участка обработки деталей

ДИПЛОМНАЯ РАБОТА

##### Гомель 2007

Оглавление

1. Имитационное моделирование

1.1 Понятие сложной системы

1.2 Понятие математической модели сложной системы

1.3 Классификация математических моделей сложной системы

1.4 Предпосылки для имитационного моделирования сложной системы

1.5 Технологические этапы машинного моделирования сложной системы

1.6 Представление динамики модели при имитационном моделировании

2 СРЕДСТВА РЕАЛИЗАЦИИ ИМИТАЦИОННОЙ МОДЕЛИ

2.1 Табличный процессор Excel

2.2 Visual Basic for Application

3 РАЗРАБОТКА ИМИТАЦИОННОЙ МОДЕЛИ автоматизированного участка обработки деталей

3.1 Концептуальная модель

3.2 Формальное описание модели

Алгоритм активностей

4 ВЕРИФИКАЦИЯ ИМИТАЦИОННОЙ МОДЕЛИ

4.1 Контроль за выполнением порядка активностей

1. Имитационное моделирование

##  Понятие сложной системы

Решение современных задач управления, проектирования и исследования технических, экономических, организационных и других систем требует привлечения специалистов разных профилей. Их эффективное сотрудничество возможно лишь при условии наличия общей методологии, в рамках которой проводится исследование. Такая методология носит звание «системный анализ». Объектом его изучения является «сложная система», а один из важнейших его инструментов есть моделирование на ЭВМ.

Термин «система» появился в научной литературе давно и является фактически таким же неопределенным, как «множество» или «совокупность». Определим понятие система, как множество компонентов, объединенных в единое целое некоторой формой регулярного взаимодействия или взаимозависимости для выполнения определенной функции. При этом компоненты будем подразделять на подсистемы, также имеющие внутреннюю структуру, как и сама система, и элементы, которые являются неделимыми с точки зрения исследователя сложной системы. Компоненты имеют определенные характеристики (признаки), которые могут принимать дискретные или непрерывные значения в процессе функционирования системы и ее взаимодействия с внешней средой. Воздействие внешней среды выражается через входные (экзогенные) переменные. С другой стороны, результат работы системы фиксируется через выходные (эндогенные) переменные. Если они характеризуют внутреннюю динамику функционирования системы, то это переменные состояния. Выходные воздействия работы системы на внешнюю среду отражаются через переменные, называемые откликами.

Системой вследствие присущих ей свойств могут устанавливаться ограничения, представляющие собой пределы изменения значений входных переменных или условия, при которых наблюдаются определенные значения. Ограничения могут также вводиться разработчиком сложной системы. Ни одна задача изучения сложной системы не может быть решена без введения целевой функции (критерия эффективности), которая представляет собой точное отображение целей или задач системы и необходимых правил оценки их выполнения.

Наиболее широко термин «система» первоначально использовался в механике, где обозначал материальную систему, т. е. совокупность материальных точек, подчиненных определенным связям. Подобные системы рассматриваются в основном в задачах динамики. Законы динамики были получены длительным индуктивным путем. Выдвигаемые гипотезы проверялись на многочисленных опытах. Проверялись также и многочисленные следствия выдвигаемых гипотез. Все это было реализовано благодаря возможности ставить «чистые опыты», т.е. устранять многочисленные мешающие факторы – сводить трение к минимуму, ставить опыты в вакууме, проводить достаточно точные измерения и т. п. Кроме того, условия опытов могли быть воспроизведены с весьма большой точностью в другое время и в другом месте.

Новый этап начался с момента, когда ученые приступили к исследованию систем, названных впоследствии «сложными», динамика которых во многом зависит от человека и принимаемых им решений. Перечислим наиболее характерные особенности сложных систем (СС).

1. Уникальность. Аналогичные по назначению системы имеют ярко выраженные специфические свойства, во многом определяющие их поведение.

2. Слабая структурированность теоретических и фактических знаний о системе. Так как изучаемые системы уникальны, то процесс накопления и систематизации знаний о них затруднен. Сюда же следует отнести слабую изученность ряда процессов, связанную с обычными для сложных систем изменениями их технической и технологической баз, значительным влиянием человеческого фактора, невозможностью или ограниченностью «натурного эксперимента».

Следствием этого, в частности, является необходимость использования ансамбля моделей при анализе системы. Различные модели могут отражать как разные стороны функционирования системы, так и разные уровни отображения исследователем одних и тех же процессов.

3. Составной характер системы. Уже на самом первом этапе изучения системы исследователь вынужден использовать понятие подсистемы как некоторой достаточно автономной части всей системы. Разделение СС на подсистемы, т.е. ее декомпозиция, как правило, зависит от принятых технических решений, целей создания системы и взглядов исследователя на нее. При декомпозиции существенны следующие факторы:

* рассматриваемая система может быть разделена (не обязательно единственным образом) на конечное число подсистем; каждая подсистема в свою очередь может быть разделена на конечное число более мелких подсистем и т.д. - до получения, в результате конечного числа шагов, таких частей, называемых элементами сложной системы, относительно которых имеется договоренность, что в условиях данной задачи они не подлежат дальнейшему разделению на части;
* элементы СС функционируют не изолированно друг от друга, а во взаимодействии, при котором свойства одного в общем случае зависят от условий, определяемых поведением других элементов, и влияния внешней среды;
* свойства СС в целом определяются не только свойствами элементов, но и характером взаимодействия между элементами.

4. Разнородность подсистем и элементов, составляющих систему. Составляющие систему элементы и подсистемы разнородны в самых различных смыслах. Во-первых, это – физическая разнородность. Во-вторых, это – разнородность математических схем, описывающих функционирование различных элементов.

Удобно разделить модели подсистем и элементов на две категории: внешние и внутренние. Названия эти условны и имеют следующий смысл.

Вследствие недостатка знаний о функционировании элемента, из-за необходимости понизить размерность модели, а также по другим причинам часто используют модели типа «вход-выход». При этом не интересуются динамикой состояний элементов, а лишь описывают их внешнее поведение. Примерами моделей подобного типа служат различные регрессионные модели, поверхности отклика, функциональные зависимости и т. п. Такие модели назовем внешними (черный ящик).

В отличие от внешних для внутренних моделей характерным является описание механизмов, управляющих динамикой их состояний, которое может базироваться на нашем представлении и гипотезах относительно истинного поведения моделей. В известном смысле идеальным случаем является формирование указанного механизма на базе уже выявленных и экспериментально проверенных закономерностей. Примерами могут служить модели, описываемые дифференциальными уравнениями, марковскими процессами и др.

5. Случайность и неопределенность факторов, действующих в системе. Примерами подобных факторов могут служить погодные условия, случайные отказы оборудования, транспорта и т. д. Учет этих факторов приводит к резкому усложнению задач и увеличивает трудоемкость исследований (необходимость получения представительных наборов данных).

6. Многокритериальность оценок процессов, протекающих в системе. Невозможность однозначной оценки диктуется следующими обстоятельствами: наличием множества подсистем, каждая из которых, вообще говоря, оценивается по своим критериям; множественностью показателей (иногда противоречивых), характеризующих работу всей системы (например, форсирование темпов, как правило, приводит к ухудшению качества работ); наличием неформализуемых критериев, используемых при принятии решений (в случае, когда решения основаны, например, на практическом опыте лиц, принимающих решения).

7. Большая размерность системы. Эта особенность системы обусловливает потребность в специальных способах построения и анализа моделей.

## Понятие математической модели сложной системы

Составной характер сложной системы диктует представление ее модели в виде тройки <A, S, Т>, где А – множество элементов (в их число включается также внешняя среда); S – множество допустимых связей между элементами (структура модели); Т — множество рассматриваемых моментов времени. Эти понятия могут быть формализованы разными способами. В качестве Т обычно выбирают множество [0, Т0) или [t0; T0), T0<∞. В каждый момент t∈Т в множестве А выделяется конечное подмножество Аt = (A1t, A2t, ..., Akt)∈A элементов, из которых в этот момент состоит модель, а в множестве S – подмножество St⊂S, указывающее на то, какие именно связи реализованы в момент t. Следовательно, допускается как переменность состава сложной системы, так и переменность ее структуры.

Основной задачей теории СС считается разработка методов, позволяющих на основе изучения особенностей функционирования и свойств отдельных элементов, анализа взаимодействия между ними получить характеристики системы в целом. Приведенная выше общая модель отвечает данной задаче – она построена в виде совокупности моделей элементов и связей между ними. Рассмотрение объекта материального мира как системы, состоящей из взаимодействующих элементов, построение математической модели для нее и исследование ее свойств методом моделирования составляет сущность системного подхода. Таким образом, системный анализ представляет собой научную дисциплину, содержащую совокупность методов и приемов построения, исследования и эксплуатации математических моделей СС.

В области естественных наук наиболее распространенными являются два вида моделирования – физическое и математическое.

Процесс физического (аналогового) моделирования состоит в изучении системы посредством анализа некоторого макета, сохраняющего физическую природу системы. Примером является модель летательного аппарата, исследуемая в аэродинамической трубе. Параметры эксперимента при этом выбирают из соотношений подобия. Аналоговое моделирование основано на указанных выше возможностях описывать разнородные явления и процессы одними и теми же уравнениями. Эти уравнения воспроизводятся обычно с помощью специально подобранных (в соответствии с уравнениями) схем, чаще всего, электрических. Искомые характеристики для исследуемой системы получаются путем измерения на модели соответствующих величин. Переработка информации в такой модели носит параллельный характер и реализуется в форме процесса, происходящего в собранной схеме.

Однако модели физического типа имеют ограниченную сферу применения. Не для всяких явлений и объектов могут быть по­строены физические аналоги. Достаточно указать на радиолокационные станции, вычислительные центры, организационные системы, производственные процессы и т. п.

Математическое моделирование основано на том факте, что различные изучаемые явления могут иметь одинаковое математическое описание. Хорошо известным примером служит описание одними и теми же уравнениями электрического колебательного контура и пружинного маятника. Математическая модель концентрирует в себе записанную в форме математических соотношений совокупность наших знаний, представлений и гипотез о соответствующем объекте или явлении. Поскольку знания эти никогда не бывают абсолютными, а в гипотезах иногда намеренно не учитывают некоторые эффекты (например, влияние силы трения в механике, потери на тепло в электротехнике и т. п.), модель лишь приближенно описывает поведение реальной системы.

Основное назначение модели – сделать возможными некоторые выводы о поведении реальной системы. Наблюдения над реальной системой (натурные эксперименты) в лучшем случае могут дать материал лишь для проверки той или иной гипотезы, той или иной модели, поскольку они представляют собой источник информации ограниченного объема о прошлом этой системы. Модель допускает значительно более широкие исследования, результаты которых дают нам информацию для прогнозирования поведения системы, характера ее траектории. Правда, чтобы обеспечить эти и другие возможности, приходится решать проблему соответствия (адекватности) модели и системы, т. е. проводить дополнительное исследование согласованности результатов моделирования с реальной ситуацией.

Математические модели строят на основе законов и закономерностей, выявленных фундаментальными науками: физикой, химией, экономикой, биологией и т. д. В конечном счете, ту или иную математическую модель выбирают на основе критерия практики, понимаемого в широком смысле. Математическая модель отображает записанную на языке математических отношений совокупность наших знаний, представлений и гипотез о соответствующем объекте или явлении. Для сложных систем нельзя получить абсолютно подобных математических моделей. Путем формализации системы получается упрощенная модель, отражающая основные ее свойства и не учитывающая второстепенных факторов.

Таким образом, математическая модель CC – это совокупность соотношений (формул, неравенств, уравнений, алгоритмов), определяющих выходные характеристики состояний системы в зависимости от ее входных параметров и начальных условий. Другими словами, ее можно рассматривать как некоторый оператор, ставящий в соответствие внутренним параметрам системы совокупность внешних откликов. После того, как модель построена, необходимо исследовать ее поведение.

С усложнением изучаемых объектов использование аналитических методов для построения и анализа моделей возможно лишь в мало интересных для практики случаях. Выход состоит в переходе к машинным реализациям математических моделей (машинным моделям). При этом на компьютер возлагается как работа по воспроизведению динамики изучаемой модели (имитация ее траекторий), так и по проведению экспериментов с ней.

Таким образом, в процессе моделирования исследователь имеет дело с тремя объектами: системой (реальной, проектируемой, воображаемой); математической моделью системы; машинной (алгоритмической) моделью. В соответствии с этим возникают задачи построения математической модели, преобразования ее в машинную и программной реализации машинной модели. В процессе решения этих задач исследователь получает белее полное и структурированное представление об изучаемой системе, разрабатывает различные варианты модели, отвечающие разным сторонам функционирования системы и их структурных преобразований. Однако основные проблемы исследования систем на машинных моделях сводятся к получению качественной картины поведения модели, а также необходимых количественных характеристик. При этом исследователь вправе использовать не только информацию, содержащуюся в машинной модели, но и информацию, полученную им на этапе создания модели.

## 1.3 Классификация математических моделей сложной системы

Математические модели можно классифицировать по различным признакам. Если исходить из соотношений, которые выражают зависимости между состояниями и параметрами СС, то различают следующие модели:

* детерминированные, когда при совместном рассмотрении этих соотношений состояние системы в заданный момент времени однозначно определяется через ее параметры, входную информацию и начальные условия;
* стохастические, когда с помощью упомянутых соотношений можно определить распределения вероятностей для состояний системы, если заданы распределения вероятностей для начальных условий, ее параметров и входной информации.

По характеру изменения внутренних процессов выделяют

* непрерывные модели, в которых состояние СС изменяется в каждый момент времени моделирования;
* дискретные модели, когда СС переходит из одного состояния в другое в фиксированные моменты времени, а на (непустых) интервалах между ними состояние не изменяется.

По возможности изменения во времени своих свойств различают

* динамические модели, свойства которых изменяются во времени;
* статические модели, не изменяющие своих свойств во времени.

Если при классификации исходить из способа представления внутренних процессов для изучения СС, то модели разделяются на аналитические и имитационные.

Для аналитических моделей характерно, что процессы функционирования элементов СС записываются в виде некоторых математических схем (алгебраических, дифференциальных, конечно-разностных, предикатных и т.д.). Аналитическая модель может исследоваться одним из следующих способов: аналитическим, когда стремятся получить в общем виде явные зависимости для искомых величин; численным, когда, не имея общего решения, удается найти частное решения или некоторые свойства общего решения, например, оценить устойчивость, периодичность, и т.п.

В имитационных моделях (ИМ) моделирующий алгоритм приближенно воспроизводит функционирование элементов СС во времени, причем элементарные явления, составляющие динамический процесс, имитируются с сохранением логической структуры и последовательности протекания во времени. Сущность этого метода моделирования обеспечивается реализацией на ЭВМ следующих видов алгоритмов: отображения динамики функционирования элементов СС, обеспечения взаимодействия элементов СС и объединения их в единый процесс; генерация случайных факторов с требующимися вероятностными характеристиками; статистической обработки и графической презентации результатов реализации имитационного эксперимента (ИЭ). Моделирующий алгоритм позволяет по исходным данным, содержащим сведения о начальном состоянии процесса и его параметрах, получать информацию о состоянии СС в произвольный момент времени.

Имитационные модели в большинстве случаев – это динамические (обязательно), стохастические, дискретные модели.

## 1.4 Предпосылки для имитационного моделирования сложной системы

Большинство исследователей считает, что следует выбрать имитационный метод для изучения сложных систем по следующим причинам.

Не существует законченной постановки задачи исследования. Каждый коллектив разработчиков математической модели определяет объект собственных исследований. Каждый раз по-новому вносятся предположения о природе взаимодействующих процессов, обсуждаются факторы, не учитываемые в модели, строится критерий качества функционирования. Как правило, для ИМ задача ставится значительно шире.

Сложность и трудоемкость аналитического аппарата. Для описания отдельных элементов системы подходит различный математический аппарат: теория массового обслуживания, конечно-разностные схемы, булева алгебра в контексте теории графов. Однако возможное количество исходных уравнений и неравенств представляется чрезмерно большим для удовлетворительного решения. Кроме того, известно мало случаев одновременного использования нескольких математических методов в рамках одной задачи.

Необходимость наблюдения за поведением компонентов системы в динамике. Специалистам недостаточно иметь усредненные оценки характеристик функционирования системы. Представляет также интерес временная последовательность возникновения узких мест, эффективность мероприятий по их ликвидации, внесение различных управлений в работу системы и т.д.

Экономическая нецелесообразность постановки натурных экспериментов. Любое исследование сложной системы является дорогостоящим мероприятием. При внесении неудачных изменений в работу реального объекта могут пострадать люди, что вызывает социальную напряженность. Проектирование же новых систем связано с большими материальными затратами. Поэтому в большинстве случаев решения принимаются на основе опыта специалистов предметной области без сотрудничества с научными коллективами. Имитация должна служить для предварительной проверки новых стратегий перед принятием решения в реальной системе.

Необходимость точного отображения функционирования компонентов системы. Аналитическая модель, как правило, не соответствует структуре сложной системе. Следовательно, изучение некоторого периода ее работы потребует разработки отдельной аналитической модели. При имитационном моделировании подобная задача тривиально решается путем сохранения и восстановления промежуточного состояния системы во внешней памяти ЭВМ.

Использование ИМ в качестве тренажера. При подготовке специалистов и освоении правил принятия решений на ИМ может обеспечиваться возможность приобретения новых навыков в управлении системой.

Однако при имитационном моделировании сложных систем несмотря на перечисленные достоинства может возникнуть ряд существенных проблем. Разработка хорошей ИМ часто обходится дороже создания аналитической модели и требует наличия квалифицированных специалистов и больших затрат времени. Иногда может показаться, что ИМ точно отражает реальное положение дел в моделируемой системе, а в действительности это не так. Причем к неверному решению могут привести свойственные именно имитации особенности. При использовании ИМ требуется применять нестандартные методы статистического анализа данных, что усложняет исследование. Преодоление перечисленных выше проблем лежит на пути создания программно-технологического инструментария, позволяющего автоматизировать этапы построения ИМ и тем самым ускорить сроки их исследования.

## 1.5 Технологические этапы машинного моделирования сложной системы

*1. Составление содержательного описания объекта моделирования*

Представляет собой выполнение следующих действий. Вначале определяется объект имитации, достаточный для изучения тех сторон его функционирования, которые представляют интерес для исследователя. Устанавливаются границы изучения функционирования объекта. Составляется возможный список ограничений модели, которые допустимы при организации имитации или при наличии которых ещё имеет смысл имитация функционирования СС. Перед разработчиками ИМ ставятся вполне конкретные цели моделирования и формулируются основные критерии эффективности, по которым предполагается проводить сравнение на модели вариантов организации СС. Результатом работ на данном этапе является содержательное описание объекта моделирования с указанием целей имитации и аспектов функционирования объекта моделирования, которые необходимо изучить на ИМ. Обычно оно представляет собой техническое описание объекта моделирования, описание внешней среды, с которой он взаимодействует, и временную диаграмму этого взаимодействия.

*2. Построение концептуальной модели*

На основании содержательного описания детализируется задача моделирования, определяется процедура и график её решения. Уточняется методика всего ИЭ в зависимости от наличных ресурсов, выделенных для имитации. Необходимо провести декомпозицию СС на составные компоненты, описать процессы их функционирования и взаимосвязи между ними. Общая задача моделирования разбивается на ряд частных задач. Устанавливаются приоритеты решения этих задач. Обосновываются требования в ресурсах ЭВМ. Выполняются такие работы, как выбор параметров и переменных СС, представляющих интерес для моделирования; уточнение критериев эффективности вариантов СС; выбор типов аппроксимации отдельных компонентов модели. Проводятся также предварительный анализ требований к модели СС; определение необходимых математических уравнений, описывающих реальные процессы; поиск возможных методов проверки правильности функционирования модели. Одновременно с этим исследователь должен выбрать способ будущей формализации процессов в объекте моделирования. Результатом выполнения работ являются концептуальная модель, выбранный способ формализации и организации имитации. В состав концептуальной модели входят: уточнённое содержательное описание, свободное от всего того, что не представляет интереса для имитации поведения СС, список параметров и переменных моделирования; критерии эффективности функционирования вариантов системы; список используемых методов обработки результатов имитации и перечисление способов представления результатов моделирования. При создании небольших ИМ данный этап работ совмещается с этапом составления содержательного описания моделируемой системы. Только с усложнением объекта моделирования и задач имитации появляется необходимость определения способа формализации, который подходит для решения конкретной задачи исследования СС.

*3. Формализация объекта моделирования*

В зависимости от сложности СС могут использоваться три вида формализации: аппроксимация явлений функциональными зависимостями, алгоритмическое описание процессов в СС, смешанное представление в виде последовательности формул и алгоритмических записей. В зависимости от принятого способа имитации используются свои способы формализации (активностями, событиями, процессами, транзактами, агрегатами, элементами системной динамики и др.). При составлении формального описания СС исследователю рекомендуется такая последовательность действий: уточнение декомпозиции системы, алгоритмизация компонентов модели, уточнение взаимодействия с управляющей программой моделирования, документация этапа.

*4. Программирование и отладка модели*

На данном этапе выполняются следующие действия. Во-первых, составляется план создания и использования программной модели. В плане указывается тип ЭВМ, средство автоматизации моделирования, примерные затраты памяти и времени на создание ИМ. Во-вторых, приступают к программированию алгоритмов ИМ. Не существует существенных отличий создания программы ИМ от других видов программного обеспечения на языках программирования. Однако опыт, накопленный в области моделирования, потребность в моделировании все более сложных систем и возможности современных ЭВМ обусловили появление новых взглядов на архитектуру и функции программного обеспечения системного моделирования.

Появление имитационных систем — закономерное следствие общей тенденции развития программного обеспечения вычислительных машин, которая заключается во все более полной автоматизации действий по программированию и подготовке задач к решению на ЭВМ.

В свое время стремление пользователей переложить на ЭВМ как можно большую часть рутинных операций привело к появлению универсальных и различных проблемно-ориентированных языков программирования, среди которых были многочисленные языки моделирования. Следует заметить, что некоторые языки моделирования, например СИМУЛА-67, СИМСКРИПТ-2 и др., нисколько не уступают по своим алгоритмическим возможностям универсальным языкам и называются проблемно-ориентированными лишь постольку, поскольку содержат ряд специальных средств, ориентированных на программирование моделей.

Построение на базе языков моделирования систем программирования позволило во многом облегчить разработку и реализацию моделей. Кроме того, многие языки моделирования содержат стандартные средства для подготовки и выполнения машинных прогонов, накопления, обработки и представления результатов имитации. Однако набор этих стандартных средств обычно довольно ограничен (генераторы псевдослучайных чисел, автоматический расчет средних и дисперсий, печать таблиц и графиков).

Модель сложной системы не всегда удается концептуально выдержать в рамках одного языка моделирования. Отдельные элементы и подсистемы могут быть описаны, например, обыкновенными дифференциальными уравнениями, другие — конечными автоматами, третьи — в терминах теории массового обслуживания и т. д. В этом случае использование какого-либо одного языка моделирования может привести к потере точности описания реаль­ных систем или к усложнению программ.

Программирование моделей сложных систем осложняется тем, что их разработка и эксплуатация обычно выполняются разными лицами. Поэтому к модели как к конечному программному продукту предъявляются высокие требования. Разработчик должен организовать удобное взаимодействие пользователя с моделью, для чего необходимо программировать специальные модули, обеспечивающие ввод-вывод информации в удобной для пользователя форме. Модель должна быть составлена с учетом возможности ее дальнейшей модификации, поэтому разработчик должен предусмотреть обоснованные стандарты на организацию межмодульного интерфейса в модели.

Требование быстрого и правильного составления модели обусловливает необходимость:

* замены программирования конструированием из готовых элементов;
* разработки проблемно-ориентированных библиотек элементов;
* отдельной трансляции элементов для выявления синтаксических ошибок в их описаниях;
* автономной отладки элементов;
* автоматической компоновки элементов в моделирующий алгоритм.

Нетривиальными для сложных моделей становятся процедуры подготовки машинных экспериментов, сбора, хранения и обработки результатов машинных экспериментов. Появляется необходимость в использовании готовых или разработке новых прикладных программ, реализующих более сложные по сравнению со стандартными возможностями языков моделирования процедуры планирования машинных экспериментов и обработки данных. Пользователь может потребовать от разработчика предусмотреть стандартные средства для подключения таких программ к модели, что обеспечит пользователю возможность самостоятельного выбора этих программ при имитации.

Таким образом, собственно модель сложной системы дополняется множеством программ, обеспечивающих ее эффективное использование.

*5. Испытание ИМ*

Включает два аспекта:

1. необходимо убедиться в правильности динамики развития алгоритма моделирования компонентов ИМ (верификация);
2. определить совпадение с заданной точностью векторов характеристик поведения объекта моделирования и ИМ (адекватность).

При отсутствии адекватности проводят калибровку ИМ ("подправляют" характеристики алгоритмов компонентов модели). Наличие ошибок во взаимодействии компонентов ИМ возвращает исследователя к этапу создания ИМ на бумаге. Возможно, что в ходе формализации исследователь слишком упростил процессы и исключил из рассмотрения ряд важных сторон функционирования СС, что привело к неадекватности ИМ. В этом случае исследователь должен вернуться к этапу формализации СС. В тех случаях, когда выбор способа формализации оказался неудачным, исследователю необходимо повторить этап составления концептуальной модели с учётом новой информации и появившегося опыта. Наконец, когда у исследователя оказалось недостаточно информации об объекте, он слишком упростил моделируемые явления, исключил из рассмотрения важные стороны функционирования СС и т.д., то необходимо вернуться к этапу составления содержательного описания СС и уточнить его с учётом результатов испытания предыдущей ИМ СС.

*6. Исследование свойств ИМ*

На данном этапе решаются так называемые тактические проблемы постановки ИЭ: оцениваются точность имитации явлений, необходимый объем выборки, длина реализация прогона ИМ, устойчивость результатов моделирования, чувствительность критериев качества к изменению параметров ИМ, стационарность режима моделирования и др.. Получить эти оценки в ряде случаев бывает весьма сложно, однако без успешных результатов этой работы доверия к ИМ не будет. Точность имитации явлений обычно представляет собой оценку влияния стохастических элементов на функционирование ИМ СС. Устойчивость результатов моделирования характеризуется сходимостью контролируемого отклика моделирования к определённой величине при изменениях параметров модели СС. Стационарность режима моделирования характеризует собой некоторое установившееся равновесие процессов в модели СС, когда дальнейшая имитация бессмысленна, поскольку новой информации из ИМ исследователь не получит и продолжение имитации приведёт к увеличению затрат машинного времени. Поэтому необходимо разработать процедуру проверки момента достижения стационарного режима имитации. Чувствительность ИМ представляется величиной минимального приращения выбранного критерия качества, вычисляемого по статистикам моделирования, при последовательном варьировании параметров моделирования на всём диапазоне их изменения.

*7. Эксплуатация ИМ*

Исследователь СС, выяснив условия и найдя характеристики, при которых должен выполняться имитационный эксперимент, переходит к решению стратегических задач – расчету выборочных средних значений (дисперсии и моментов более высокого порядка) откликов модели и различных зависимостей на их основе.

Этап эксплуатации ИМ начинается с составления плана эксперимента, позволяющего исследователю получить максимум информации при минимальных усилиях на вычисление. Составляется статистическое обоснование плана эксперимента. Планирование эксперимента представляет собой процедуру выбора числа и условий проведения опытов, необходимых и достаточных для решения поставленной задачи с требуемой точностью. Стремятся минимизировать общее число опытов на ИМ с одновременным варьированием всеми переменными. Выбирают такую стратегию ИЭ, которая позволяет принимать обоснованную стратегию с помощью процедур принятия решений после каждой серии экспериментов на ИМ.

*8. Анализ результатов моделирования*

Данный этап завершает технологическую цепочку этапов создания и использования ИМ. Получив результаты моделирования, исследователь приступает к интерпретации результатов. Возможно, что в ходе интерпретации результатов исследователь установил наличие ошибок либо при создании модели, либо при формализации объекта моделирования. В этих случаях осуществляется возврат на этап построения описания ИМ или на составление концептуальной модели СС соответственно. Результатом этапа интерпретации данных ИЭ являются рекомендации по проектированию или модификации СС. На их основе исследователи приступают к принятию решений. На интерпретацию результатов ИЭ оказывают существенное влияние изобразительные возможности средств моделирования на ЭВМ. При анализе результатов ИЭ рекомендуется использовать пакеты OLAP (On-Line Analytical Processing), включающие текстовый процессор, средства статистической обработки и графической презентации данных, принятия решений.

В конечном итоге после выполнения всех перечисленных выше итерационных этапов имитации исследователь либо окажется удовлетворённым результатами моделирования и будет их учитывать при проектировании СС, либо забракует проектируемую систему и сформулирует техническое задание на разработку новой архитектуры СС.

## 1.6 Представление динамики модели при имитационном моделировании

Будем характеризовать каждый компонент Ki сложной системы множеством состояний zi (переменных, содержащих информацию, необходимую для прогноза будущей динамики элемента). В каждом элементе Ki в результате выполнения функциональных действий происходят события eij. Время наступления события τij и его содержание полностью определяется состоянием zi элемента Ki. Для каждого Ki введем понятие локального времени ti. В сложной системе все ti изменяются одновременно, однако характер этих изменений различен и определяется последовательностью временных интервалов τij. При построении ИМ СС функциональные действия аппроксимируются некоторыми алгоритмами αij при неизменном значении ti, а затем уже отображается изменение ti на величину τij, инициируя таким образом появление события eij. Пару (αij, τij) назовем активностью и обозначим aij.

Если СС состоит из одного элемента K1, то динамика ее ИМ представляла бы собой последовательную смену событий e1j на временной оси t1 посредством реализации активностей a1j (рис.1).



Рисунок 1.1 — Динамика поведения элемента СС.

Чтобы обеспечить имитацию модели СС, состоящей более, чем из одного элемента, вводят глобальную переменную t0 , называемую модельным временем. Опишем динамику такой модели следующим образом.

Пусть в начальный момент времени t0 = 0 определены состав элементов и структура системы S0 . Для каждого элемента Ki известны состояния zi и времена τi0 наступления событий ei0. Введем переменные  и положим ti = tim с таким номером m, что выполняется условие ti,m-1≤t0tim. Найдем величину T0=min{ti}, равную времени до наступления самого раннего события и пусть i0=argT0 (возможной неоднозначностью номера i0 пренебрежем). В момент времени T0 в элементе происходит событие , содержание которого задается состоянием .В этот момент осуществляется взаимодействие элементов модели, в результате чего формируется новый состав элементов и структура системы S1. Реакцией наступления события  является корректировка модельного времени (t0=T0) и реализация активности , т.е. выполнение алгоритма  и модификация локальной временной координаты (ti0=ti0+τi00). Далее вновь определяется время T1 наступления самого раннего события  в компоненте , корректируется модельное время (t0=T1) и реализуется активность  и т.д. (см. рис. 2)

Моменты T0, T1, T2, ... смены состава элементов и/или их состояний будем называть особыми. Переходы, происходящие в эти моменты, полностью определяют динамику модели. Поэтому при машинной имитации достаточно воспроизводить лишь эти изменения (в общем случае случайные).

Различают два типа ИМ, связанных со способом формализации реальных процессов функционирования СС. В моделях одного типа события eij отражают фактические события, происходящие в системе. В этом случае динамика ИМ по существу повторяет динамику СС, т.е. переход от одного события к другому. Говорят, что такая ИМ носит событийный (дискретный) характер.

В моделях другого типа события вводят искусственно вследствие необходимости представить некоторый непрерывный процесс в дискретной ЭВМ. Характерным примером является любой численный метод интегрирования дифференциальных уравнений. В подобных методах особые моменты определяются шагом интегрирования. Эти методы называются пошаговыми. При их использовании динамика модели является дискретным приближением реальных непрерывных процессов.

Вследствие последовательного характера обработки информации в однопроцессорном компьютере параллельные процессы, происходящие в СС, преобразуются в ИМ с помощью вышеприведенного механизма в последовательные. Такой способ представления носит название квазипараллельного процесса, а алгоритм называется алгоритмом организации квазипараллелизма в ИМ. Противоречие между параллельностью модельных процессов и последовательным характером квазипараллельного процесса является коренной причиной их неполного соответствия.

Рисунок 1.2 — Динамика взаимодействия элементов СС.

2 СРЕДСТВА РЕАЛИЗАЦИИ ИМИТАЦИОННОЙ МОДЕЛИ

## 2.1 Табличный процессор Excel

Имитационная модель автоматизированного участка обработки деталей реализована при помощи табличного процессора MS Excel и встроенной среды программирования Visual Basic for Application.

Microsoft Excel - приложение, предназначенное для работы с электронными таблицами. Excel - это простой, удобный и эффективный инструмент, позволяющий проанализировать данные и, при необходимости, проинформировать о результате заинтересованную аудиторию, используя электронную почту или Интернет.

Табличный процессор – это интерактивная компьютерная программа, которая работает с данными (электронными таблицами), представляющими собой набор строк и столбцов.

Наиболее мощные возможности Excel, выделяющие данный табличный процессор среди данного класса программ заключаются в следующем:

* проведение различных вычислений с использование мощного аппарата функций и формул;
* исследование влияния различных факторов на данные;
* решение задач оптимизации;
* получение выборки данных, удовлетворяющих определенным критериям;
* графическое отображение данных в виде диаграмм, линий, поверхностей и т.п.;
* статистический анализ данных.

Документы (файлы), которые обрабатывает программа Excel, называются рабочими книгами (или просто книгами). Рабочая книга, как и любая книга, состоит из листов. Любой из листов представляет собой таблицу, состоящую из строк и столбцов. Максимально рабочая книга Excel включает до 255 листов, каждый из которых разделен на 256 столбцов и 16 384 строк, что достаточно практически для любой задачи.

Горизонтальные строки и вертикальные столбцы составляют двумерную таблицу, а листы добавляют третью составляющую. Строки нумеруются от 1 до 65 536, столбцы обозначаются от A до IV (вначале от А до Z, затем от АА до АZ, затем от ВА до ВZ и так вплоть до IV). Листам присваиваются имена Лист1, Лист2, Лист255 или любые другие имена, состоящие не более чем из 31 символа.

Переход от одной открытой книги к другой выполняется с помощью команд меню Window (Окно), а переход с листа на лист — щелчком на ярлычке, после чего соответствующий лист переносится на передний план.

В Excel используются листы двух типов. Их назначение следующее:

* Sheet (Лист) — применяется для ввода и обработки числовых данных;
* Chart (Диаграмма) — предназначен для создания и хранения диаграмм, не внедренных в рабочие листы.

Область на пересечении строки и столбца таблицы называют ячейкой. Ячейку, на которой стоит курсор, называют активной.

Каждой ячейке соответствует свой адрес, определяющийся строкой и столбцом, в которых помещены данные ячейки. При записи адреса в Excel сначала указывается столбец, затем строка. Ячейке, которая находится в верхнем левом углу, присвоен адрес А1, а ячейке, которая находится в самом нижнем правом углу, адрес IV65536.

Чтобы указать дополнительно лист, следует вначале адреса поставить имя этого листа и восклицательный знак «!» перед координатами столбца и строки. Выделяют 4 типа диапазонов на листе:

* одна ячейка;
* одна или более строк;
* один или более столбцов;
* прямоугольная область ячеек.

Для указания диапазонов в формуле или в команде используются ссылки. В качестве ссылки для одной ячейки выступает ее адрес, например D14. Чтобы сослаться на множество строк применяют номера начальной и конечной строк, разделенные двоеточием. Аналогично, чтобы указать несколько столбцов используют имена начального и конечного столбцов. Для задания произвольного прямоугольного диапазона ячеек указываются адреса левой верхней и правой нижней ячеек через двоеточие.

Работать с рабочими книгами можно после того, как в них будут введены данные. Все данные, которые вводятся в таблицу, размещаются и хранятся в ячейках. В таблице насчитывается большое количество ячеек, а каждая ячейка может включать до 255 символов. Обычно в одну ячейку вводят одно число или короткую текстовую строку. В одной ячейке может находиться либо число, либо текст. Поэтому, прежде чем вводить в ячейку данные, следует решить, к какому типу они относятся.

В любую ячейку рабочей таблицы Excel можно ввести как значение, так и формулу. Значения – это числа, текст или последовательности символов даты или времени. Формулы – это комбинации значений, данных, содержащихся в ячейках, и операторов.

Основное различие между числами и текстом заключается в том, что с числами выполняются различные арифметические операции, а с текстом нет.

По ходу ввода Excel определяет, является ли вводимый элемент числом или текстом. Если вы вводите только числа: 0,1,2,3,4,5,6,7,8,9 или символы, используемые при записи чисел: +, -, (, ), ,, р., %, ., \*, /, Е, е, то Excel рассматривает введенный элемент как число. В том случае, если вводится значение даты или времени в одном из встроенных в Excel форматов, то и значение даты, и времени будут тоже рассматриваться как число. Правильно построенная формула, результатом которой является число, также обрабатывается как число. Все остальное считается текстом.

Excel предоставляет возможность вводить числа в различном формате. Можно вводить десятичные числа, денежные единицы, проценты и представлять число в экспоненциальном виде. Excel преобразует введенное значение в число с точностью до 15 знаков. Затем можно представить это число совершенно в другом формате.

Текстовые записи могут представлять собой любые последовательности букв, цифр и специальных символов.

Значения даты и времени необходимо вводить в определенном формате. Пользовательские форматы даты и времени собраны в диалоговом окне Формат ячеек, вызываемого из меню Формат по команде Ячейки. В Excel дата и время рассматриваются как числа, поэтому с ними можно выполнять различные арифметические операции и использовать при различных вычислениях.

Любой ввод данных, начинающийся со знака «=», интерпретируется в Excel как формула. В строке формул отображается сама формула, а в ячейке виден результат вычисления по этой формуле.

Редактирование данных можно выполнять, используя клавиши <Backspace> и <Esc>, а также клавишу <F2>. Если при редактировании данных нажимается клавиша <Backspace>, то будут удалены символы слева от точки вставки, если клавиша <Esc>, то будет удален весь введенный в ячейку фрагмент.

Для редактирования содержимого ячейки после того, как ввод завершен, необходимо выделить ее и нажать клавишу <F2>. В строке состояния появится индикатор режима Правка, а справа от элемента в ячейке будет зафиксирована точка вставки. С помощью клавиш управления курсором можно перемещать точку вставки в ячейке и изменять отдельные символы.

При вводе данных часто возникает проблема вставки пустой строки или пустого столбца в таблицу. Для этого следует установить маркер в какой-нибудь ячейке столбца, перед которым нужно вставить новый столбец и выполнить команду Вставка/Столбец. Аналогично для вставки строки выполняется команда Вставка/Строка.

В Excel позволяют изменять размеры строк и столбцов команды Столбец и Строка в меню Формат. После выбора нужной команды открывается подменю, в котором выбирается элемент Ширина или Высота соответственно. В появившемся диалоговом окне задается ширина столбца в символах или высота строки в пунктах.

Можно изменить ширину столбцов с помощью мыши. Для этого указатель мыши необходимо поместить на вертикальную линию, разделяющую заголовки столбцов (вид указателя изменится) и при нажатой левой кнопке перетащить ее в нужном направлении. Изменение высоты строк выполняется аналогично. Указатель мыши помещается на разделительную линию между заголовками строк, и эта линия перетаскивается вверх или вниз.

Основным достоинством Excel является наличие мощного аппарата формул и функций.

Формула - последовательность символов, начинающихся со знака «=». В эту последовательность символов могут входить постоянные значения, ссылки на ячейки, имена, функции или операторы. Если значения в ячейках, на которые есть ссылки в формулах, меняются, то результат изменится автоматически. В строке формул отражается содержимое ячейки, в которой расположен курсор, а, следовательно, эта ячейка является текущей или активной.

Функция – это специально созданная формула, которая выполняет операции над заданным значением или значениями. Используются функции для выполнения стандартных вычислений в рабочих книгах. Значения, которые используются для вычисления функций, называются аргументами. Значения, возвращаемые функциями в качестве ответа, - результатами.

При вводе формул следует соблю­дать 3 правила:

* сначала должна быть активизирована та ячейка, в которой будет находиться результат.
* каждая формула начинается со знака равенства.
* при написании формулы используются адреса (имена) ячеек, а не их содержимое.

Ссылки указывают на то, в каких ячейках находятся значения, которые нужно использовать в качестве аргументов формулы. С помощью ссылок можно использовать в формуле данные, находящиеся в различных местах рабочего листа, а также использовать значение одной и той же ячейки в нескольких формулах. Можно ссылаться на ячейки, находящиеся на других листах рабочей книги, в другой рабочей книге или даже на данные другого приложения.

Ссылки на ячейки используют адреса ячеек, т.е. заголовки соответствующих строк и столбцов рабочего листа.

В формулах используются относительные и абсолютные ссылки.

По умолчанию все ссылки – относительные. При копировании они преобразуются и соответствуют новому расположению формулы.

Абсолютные ссылки применяются в случаях, когда ссылка на конкретную ячейку не должна изменяться при копировании формул. Для указания абсолютной ссылки устанавливается знак доллара «$» перед ссылкой. Комбинация абсолютных и относительных ссылок образует смешанную ссылку.

Функции в Excel не только облегчают ввод данных, но и выполняют специальные расчеты. Они могут использовать координаты диапазонов, именованные диапазоны и обычные числовые значения.

Каждая функция состоит из имени функции и аргумента.

Имя функции (например, СУММ, СРЗНАЧ, МИН, МАКС) указывает на ее назначение.

Аргумент (например, В2:В12) сообщает Excel какие адреса ячеек задействованы в данной функции.

Связывание – это процесс использования ссылок на ячейки из внешних рабочих книг в целях получения данных для своей рабочей таблицы.

Общий синтаксис для формулы с внешней ссылкой выглядит следующим образом:

=[ИмяРабочейКниги]ИмяЛиста!АдресЯчейки.

Если формула содержит ссылки на ячейки из другой рабочей книги, то эта книга необязательно должна быть открыта. Если рабочая книга закрыта и не находится в текущей папке, то необходимо добавить к ссылке полный путь.

Если рабочая книга связана с несколькими рабочими книгами, то может возникнуть необходимость просмотреть список всех исходных рабочих книг. Для этого используют команду Правка⇒ Связи. В результате появляется диалоговое окно Связи, в котором перечислены все исходные рабочие книги, а также другие типы связей с другими документами. Это диалоговое окно используется и для обновления связей. Для этого в нем выбирают соответствующую исходную рабочую книгу и команду Обновить.

Иногда при вводе формулы Excel выдает значение, которое начинается с символа решетка (#). Это говорит о том, что данная формула возвращает ошибочное значение. В подобном случае следует исправить формулу (или ссылку на ячейку содержащуюся в этой формуле), чтобы избавиться от сообщения об ошибке. Если ячейка полностью заполнена знаками решетки (#), это означает, что столбец недостаточно широк, чтобы отобразить некоторую величину. В этом случае нужно или расширить столбец, или изменить числовой формат ячейки.

Большие массивы информации обычно хранятся в виде базы данных и обрабатываются специальными программами. В Excel аналогом простой базы данных является список. Он представляет собой группу строк, содержащих связанные данные. Отличительная особенность списка заключается в том, что каждый его столбец содержит однотипные данные. Если провести аналогию между списком и базой данных, то можно сказать, что столбцы списка являются полями базы данных, а его строки — записями.

Сценарий представляет собой набор сохраненных программой Excel значений, которые она может автоматически подставить в ячейки рабочего листа. Сценарии можно использовать для прогноза результатов моделирования и расчетов. Существует возможность создать отчет по результатам работы сценария, который содержит как исходные, так и итоговые значения, что позволяет прослеживать зависимости между данными в таблице [11].

Значение диаграмм как графической формы представления числовых данных трудно переоценить. Они позволяют не только повысить наглядность излагаемого материала, но и отобразить соотношение различных величин или динамику изменения показателей. В Excel предусмотрены средства работы с диаграммами различных типов.

В Excel существует возможность красочно оформлять рабочие листы, используя для этого различные графические объекты. Причем такие объекты можно создавать непосредственно в самой программе Excel, а можно внедрять из других приложений. Для создания графических объектов в Excel предназначен встроенный графический редактор, доступный при наличии панели инструментов Drawing (Рисование).

|  |  |
| --- | --- |
| Ошибочное значение | Описание |
| # ДЕЛ / 0! | 1) Формула пытается выполнить деление на 0. 2) Формула пытается выполнить деление на содержимое пустой ячейки |
| #ИМЯ? | В формуле используется имя, которое Excel не может распознать. Например, удалено имя, используемое в формуле, или при вводе текста где-то пропущена кавычка |
| #Н/Д | Неопределенные данные. Формула ссылается (прями или косвенно) на ячейку, содержащую функцию, ссылающуюся на недопустимый тип данных |
| #ЧИСЛО!  | Проблема связана со значением; например, задано отрицательное число там, где необходимо положительное.  |
| #ССЫЛКА! | Недопустимая ссылка; например, формула ссыпается на ячейку, удаленную из рабочей таблицы. |
| #ЗНАЧ! | В формулу включен аргумент недопустимого типа.  |

Рис. 1.4.4.1. Типы ошибок в формулах Excel

В таблице перечислены типы ошибочных значений, которые могут появиться в ячейке, содержащей формулу. Формулы могут возвращать ошибки и в том случае, если ячейки, на которые в них есть ссылки, содержат ошибочные значения. Этот эффект носит название цепной реакции – когда единственное ошибочное значение может породить ошибки во многих других ячейках, содержащих формулы со ссылкой на эту ячейку.

##

## 2.2 Visual Basic for Application

VBA является единой средой разработки приложений во всех программах Microsoft Office 2003. Основные принципы работы в среде VBA и в среде Visual Basic 2003 идентичны. Перечислим главные преимущества, получаемые в результате применения языка VBA.

Во-первых, с помощью VBA можно записывать последовательности повторяющихся команд, характерных для какого-либо приложения, и назначать правила (условия), в соответствии с которым эти последовательности (процедуры VBA) будут вызываться в приложении. Например, после нажатия некоторой кнопки, рабочий лист Excel будет отформатирован с учетом ваших требований, записанных на VBA и сохраненных в рабочей книге или шаблоне.

Во-вторых, применив средства VBA, можно обеспечить нестандартный диалог с пользователем, создавая диалоговые формы и обрабатывая реакцию пользователя на событие в приложении.

Наконец, использование единого языка программирования способствует более тесному взаимодействию Excel с продуктами Microsoft Office. С помощью VBA можно разрабатывать приложения, одновременно применяющие компоненты нескольких программ [12].

В Excel существует возможность создавать собственные макросы и таким образом автоматизировать выполнение основных операций. Макрос представляет собой последовательность команд и функций, записанных в модуле VBA. Насколько сложными могут быть макросы, можно судить по входящим в пакет поставки Excel надстройкам, при создании которых использовались средства макропрограммирования.

Существует два способа создания макроса: записывать свои действия во время работы с листом Excel или же написать макрос, используя редактор Visual Basic.

Наиболее простым способом создания макрокоманды является ее запись с помощью макрорекордера, и для этого вовсе не нужны глубокие знания по программированию. Работа макрорекордера во многом напоминает работу обычного магнитофона, осуществляющего запись и воспроизведение звука. Макрорекордер протоколирует все выполняемые пользователем действия и представляет их в виде VBA-кода.

Однако следует помнить, что «собственноручно написанные макросы позволяют создавать гораздо более мощные и гибкие программы, чем макросы, записываемыми с помощью макрорекордера, но, конечно, записать макрос гораздо проще, чем создать его программный код [10].

Процесс записи макроса с помощью макрорекордера можно условно разбить на три следующих этапа:

* запуск макрорекордера;
* присвоение макросу имени;
* выполнение записываемых операций;
* завершение записи.

Редактирование VBA-модулей выполняется в среде Visual Basic for Applications, в которой пользователю предоставляются средства для разработки программ, анализа связей между объектами (VBA является объектно-ориентированным языком программирования), отладки программ и их организации в единый программный комплекс (проект).

В VBA текст макроса оформляется в виде процедуры Sub. Данная процедура создается на основе событий (таковым может быть, например, изменение размера окна или щелчок на кнопке) и методов. Каждому событию поставлены в соответствие методы, которые задают необходимость выполнения определенных операций при наступлении данного события. Так, при щелчке на ярлычке рабочего листа событием является изменение активного листа. Благодаря связанному с этим событием методу лист, на ярлычке которого был произведен щелчок, активизируется и отображается на переднем плане. В зависимости от типа активизированного листа изменяются и некоторые команды меню. В VBA пользователь может задавать собственные события и ставить им в соответствие методы.

Visual Basic для приложений

Visual Basic для приложений — это объектно-ориентированный язык макропрограммирования высокого уровня, встроенный во все программы пакета Office 2003. Включение в указанный пакет современного полнофункционального языка программирования, инструментов для написания на нем программ, их редактирования и отладки, а также обширных библиотек стандартных программ позволяет создавать программные продукты, способные с достаточной полнотой автоматизировать решение конкретных задач пользователя.

Основные понятия языка Visual Basic for Applications.

VBA является объектно-ориентированной средой, содержащей большой набор объектов, каждый из которых обладает множеством свойств и методов. Количество этих объектов может напугать новичка, но, к счастью, для того чтобы начать работать, знать их всех вовсе не обязательно. Сам объект и все инструменты, необходимые для его создания, отображения или, скажем, уничтожения, относятся к определенному классу. Имя класса указывает на тип объектов, которые он содержит. Классы можно рассматривать как основу для создания других объектов этого же типа. Поэтому все объекты одного класса, например класса TextBox, будут «действовать» одинаково. В других языках программирования вы можете сгруппировать связанные друг с другом данные и ссылаться на них, используя общее имя, предположим Employees или Рroducts, но в соответствии с объектно-ориентированной методологией класс обязательно должен включать в себя и исполняемый программный код.

Свойства и методы являются членами класса. С помощью свойств описывается, как выглядит объект, в частности дается информация о приемах форматирования текста, цвете и размере шрифта. Методы являются процедурами, или, другими словами, множеством осуществляющих определенную задачу операторов, которые могут быть выполнены для объекта (процедуры создания и удаления объекта, процедуры событий, определяющие принцип взаимодействия объекта с пользователем, и т. д.) [12].

Объекты

При объектно-ориентированном программировании практически все компоненты среды разработки являются объектами. В Excel VBA-объектом считается любой элемент приложения — ячейка, лист, рабочая книга, диаграмма. Фактически объектом является и само приложение Excel. Объекты могут включать области ячеек, рамки ячеек, окна, сценарии, стили, и этот список можно продолжить. Каждый класс объектов имеет свое множество свойств, функций и событий.

Когда вы добавляете объект на рабочий лист, создается экземпляр класса объектов, и объекту присваиваются те значения свойств, которые определены в этом классе. По умолчанию экземпляры одного и того ж класса имеют имя, которое состоит из имени родительского класса и порядкового номера. Свойства каждого экземпляра одного и того же класса инициализируются одними и теми же значениями, за исключением свойства Name. Так как VBA присваивает значения всем элементам, которые в этом нуждаются, пользователю действительно не нужно много знать, чтобы начать работать в данной среде.

Свойства

Свойство является атрибутом объекта, описывающим, как объект выглядит (его цвет, размер и местоположение) и как он действует (является ли видимым, ссылается ли на другой объект). Когда вы создаете объект, Excel выполняет процедуру создания экземпляра этого объекта. Данная процедура, хранимая в классе объектов, присваивает значения всем свойствам, что позволяет вам сразу же работать с объектом. Если вы захотите изменить несколько свойств, то обнаружите, что в большинстве своем они заданы именно так, как вам нужно, поэтому вносить изменения вам не придется. Для того чтобы в макросе VBA присвоить свойству новое значение, необходимо лишь создать оператор присваивания, в котором слева от знака равенства будут указаны имя и свойство объекта (разделенные точкой), а справа — новое значение. К примеру, указанный ниже оператор заменит присвоенное по умолчанию имя листа Sheetl именем Accounts (Счета):

*SheetI.Name- "Accounts"*

Свойство листа Name отображается на его ярлычке. Причем данное свойство входит также во множество других классов.

Методы

Метод — это действие, которое может быть выполнено над объектом. Методы реализуются посредством выполнения процедуры, которая является членом класса объектов. Синтаксис вызова метода таков; имя объекта и имя метода, разделенные точкой. К примеру, для того чтобы использовать метод Protect для защиты объекта Sheetl от внесения изменений, вам следует ввести такую команду:

*Sheetl.Protect*

События

Всякий раз, когда пользователь взаимодействует с определенным объектом на рабочем листе, совершается событие. Каждый класс объектов имеет собственную группу событий, на которые объекты данного класса реагируют. Так, рабочая книга имеет событие NewSheet, которое выполняется всякий раз, когда пользователь добавляет в нее новый лист; событие SelectionChange происходит при выборе пользователем другой ячейки или области ячеек. При создании процедуры события Excel автоматически добавляет в ее код первую и последнюю строки; при желании вы можете самостоятельно написать программу для обработки события, на которое должны реагировать объекты.

Взаимодействие объектов, свойств и методов

Когда вы создаете лист в рабочей книге, функцией-членом в классе Worksheet создается объект Worksheet. Значения свойств нового объекта присваиваются по умолчанию (например, значение свойства Name, объект стиля ссылок А1 или R1C1), и с объектом можно сразу же работать. К этому объекту можно применить любой из встроенных в класс методов, таких как Printout или Save, и создать программу реагирования на такие события, как SheetCaleulate или, скажем, Open.

Использование свойств и методов, возвращающих объекты

Если свойство или метод возвращает объект, то доступ к свойствам и методам этого объекта можно получить лишь после указания его имени. В качестве примера рассмотрим такую строку программы:

*Workbooks{"Bookl").ActlveSheet.Name= "Expenses"*

Метод Workbooks("Bookl").ActiveSheet возвращает объект Worksheet, который содержит активную на данный момент таблицу, находящуюся в рабочей книге Bookl. Свойство Name является свойством объекта Worksheet, а не объекта ActiveSheet. Так как лист Sheetl считается активным, то этот оператор присваивания эквивалентен следующему:

*Sheetl.Name= "Expenses"*

Первый оператор является более общим, так как дает возможность присвоить новое имя любому активному листу, в то время как второй оператор однозначно выбирает рабочий лист Sheetl и присваивает ему имя вне зависимости от того, активен он или нет.

Свойство ActiveCel1 возвращает объект Range — одну или несколько ячеек, обращение к которым необходимо произвести одновременно. Это свойство является членом классов Application (активное приложение, в данном случае — Excel) и Window (активное окно) и возвращает область ячеек, которая содержит активную ячейку, находящуюся в активном окне активного приложения. Вот как это выглядит в коде:

*Application.ActlveWlndow.ActiveCell.Borders.Color = vbRed*

Так как в каждый момент времени активной может быть только одна ячейка, то свойства ActlveCell объектов Application и Window ссылаются на одну и ту же область ячеек. Таким образом, этот оператор можно переписать по-другому — так:

*ActiveWindow.ActiveCell. Borders. Color = vbRed*

или так:

*Application.ActiveCell.Borders.Color = vbRed*

А в связи с тем, что возвращаемый объект Range определяется однозначно, оператор можно переписать и следующим образом:

*ActiveCell.Borders.Color – vbRed*

Свойство Borders тоже относится к числу свойств, возвращающих объект. Часть оператора ActlveCellBorders возвращает объект Border (рамку области ячеек), а затем свойству Color (цвет) этого объекта присваивает значение red (красный). Префикс vb, предшествующий значению red, указывает на то, что это значение является константой Visual Basic, в отличие от переменных, которые нужно описывать в программе.

Объект, возвращаемый свойством ActiveCell, - это объект Range. Таким образом, свойство Borders задает прямоугольную рамку вокруг группы ячеек, определенных в объекте Range.

Оператор Application.ActiveWIndow.ActiveCell.Borders.Color = vbRed

станет более понятным, если прочитать его справа налево: «Цвет расположенной в активном окне приложения рамки области ячеек, которая содержит активную ячейку, становится красным [10].

Области в Excel — объекты VBA

Объекты Range определяют ячейки или области ячеек, которым можно присваивать значения, записывать в них формулы и с которыми можно работать вручную. В частности, действие оператора ActiveCell.Borders.Color = vbRed эквивалентно команде Cells (Ячейки) меню Format (Формат) и изменению значения в поле Color (Цвет) вкладки Border (Граница) диалогового окна Format Cells (Формат ячеек).

Кроме того, существует свойство Range, которое возвращает объект Range. Данное свойство позволяет использовать ссылку на ячейку или область ячеек, обращаясь к ней как к аргументу. Правда, здесь имеется одно ограничение: ссылки должны записываться с применением относительной или абсолютной адресации только в стиле А1:

Range(Al")

Range("$ASl")

Свойство Range возвращает объект Range, поэтому все свойства и методы, связанные с этим объектом, могут быть установлены посредством данного свойства. Свойство Range также может использоваться для возвращения объекта Range, который применяет ссылку на ячейку А1 (свойство Value возвращает значение ячейки А1):

Range('Al"). Value

Если ячейка А2 содержит формулу, то результат можно вычислить, воспользовавшись методом Calculate объекта Range, который возвращает свойство Range:

Range("A2").Calculate

Кроме того, воспользовавшись свойством Range, можно получить доступ к несмежным областям ячеек, а затем посредством одного оператора присвоить значение всем ячейкам. Это особенно удобно, если нужно присвоить всем ячейкам одинаковое значение:

Range("A3:B5.C3:D8").Value = 0

Свойство Range также можно использовать с двумя аргументами, которые указывают на ячейки, находящиеся в разных углах области ячеек, расположенных на одной диагонали. В частности приведенный ниже оператор выбирает ячейки области А1:С4:

Workbooks("Bookl").Sheets("Sheetl") Range{'Al","C4"}.Select

Процедуры и функции VBA

В VBA поддерживается следующая программная структура. На высшем уровне иерархии стоит приложение, далее идут проекты, связанные с фактическими документами этого приложения. На третьем уровне находятся модули (модули приложения, модули пользователя, модули класса, модули форм и модули ссылок), а на последнем — их процедуры и функции. Схематически такая иерархия показана на рисунке 1.1 [12].

Приложение

Проект 1

Проект 2

Проект N

…

Модуль 1

Модуль 2

Модуль N

…

Процедура N

Процедура 2

Процедура 1

Объявление

…

Рисунок 1.1 – Программная структура в VBA

Модуль — это часть программы, оформленная в виде, допускающем ее независимую трансляцию. Модуль состоит из двух разделов: раздела объявлений (Declaration) и раздела процедур и функций. В первом разделе описываются глобальные переменные, типы, определенные пользователем, и перечисляемые типы, во втором — процедуры и функции. Вы уже знаете, как создаются и используются макросы. Все они сохраняются в документах или шаблонах программ Office 2003 в виде процедур, написанных на языке VBA. Процедурой называется фрагмент кода (минимальная семантически законченная программная конструкция), заключенный между операторами Sub и End Sub. Синтаксис процедуры следующий:

[Private | Public | Friend] [Static] Sub имя\_процедуры (аргунент\_1, аргумент\_2. ...аргумент\_n)

<оператор VBA>

<оператор VBA>

…

<оператор VBA>

End Sub

Синтаксис функций несколько отличается от синтаксиса процедур, так как для функции необходимо указать тип возвращаемого ею значения:

[Private | Public | Friend] [Static] Function имя\_функции (аргумент\_1, аргумент\_\_2, ...

аргумент\_n As тип\_возвращаемого\_значения

<оператор VBA>

<оператор VBA>

…

имя - вычисленное\_значение

…

End Function

Ключевое слово Private задает следующую область видимости для процедуры - модуль, в котором она описана. Значит, ее могут вызывать только процедуры этого же модуля. Ключевое слово Public, наоборот, объявляет процедуру доступной для всех модулей проекта. С помощью ключевого слова Friend процедура или функция становится видимой только в том проекте, где был описан класс, членом которого она является. По умолчанию любая процедура общедоступна, то есть имеет статус Public. За операторами объявления следует необязательное ключевое слово Static, объявляющее все локальные переменные статическими (что такое статическая переменная, объясняется ниже).

Выполнение программного кода процедуры или функции происходит при передаче ей управления. Передача управления (вызов) может осуществляться различными способами. В общем случае подпрограмма вызывается из программного кода с помощью специального оператора Са11 , в котором кроме данного ключевого слова указываются имя процедуры и фактические параметры вызова, список которых заключается в круглые скобки.

Очень важную роль в программировании на VBA играет такая разновидность процедуры, как процедура без параметров. Процедуры данного типа могут выступать в роли командных макросов и процедур обработки событий. Схема работы с командными макросами не отличается от традиционной схемы работы с макросами. После создания подобного макроса с помощью редактора VBA ему можно назначить комбинацию клавиш для вызова, сопоставить его с командой меню или кнопкой на панели инструментов. Процедуры обработки событий представляют собой основу механизма связи событий с программным кодом для их обработки [8].

Функции в VBA, как правило, содержат списки параметров. Они не могут использоваться для обработки событий или, скажем, в качестве программных макросов. Вызывать же их можно как с помощью оператора Са11 , так и непосредственно в процессе расчета значения выражения VBA (из кода других процедур или функций) или формулы рабочего листа Excel.

Как правило, программа на языке VBA состоит из нескольких процедур и функций. Описание этих программных единиц хранится в модулях. Модули, в свою очередь, как составные части входят в проекты. Проекты сохраняются в файлах рабочих книг Excel 2003, но работать с ними можно лишь при наличии отдельного приложения — редактора Visual Basic. Если нужно вызвать процедуру или функцию, описание которой находится в другом модуле, ее имя при вызове указывается следующим образом: имя\_модуля.имя\_процедуры. Но если описание процедуры или функции содержится в другом проекте, при ее вызове следует использовать иной синтаксис: имя\_проекта. имя\_модуля. имя\_процедуры.

Переменные в VBA

Описанные в процедурах VBA действия выполняются над переменными или объектами. Переменная — это поименованная область памяти, которая используется для хранения данных в течение работы процедуры. Переменные VBA аналогичны переменным других языков программирования. Использованию переменной обычно предшествует ее объявление посредством оператора D1 т, в котором указываются имя переменной и тип данных, для хранения значений которого она предназначена. Хотя такой оператор не является обязательным (VBA способен определять тип данных переменной по последнему символу ее имени: !, #, $, %, &, @), тем не менее, хороший стиль программирования требует предварительного описания всех используемых переменных. Имена переменных могут иметь длину до 255 символов (используются и символы кириллицы) и не должны содержать пробелов, знаков препинания и перечисленных выше специальных символов (за исключением символа @). Они не могут совпадать с ключевыми словами VBA и именами стандартных объектов.

Управляющие конструкции VBA

Управляющие конструкции языка программирования — это инструкции и группы инструкций, применение которых позволяет изменять по мере необходимости последовательность выполнения других инструкций программы. В современных языках программирования управляющие конструкции подразделяются на ветвления и циклы. Ветвлением называется управляющая конструкция, позволяющая пропускать при выполнении те или иные группы инструкций в зависимости от значения условия (If...Else, If...Then...Else, If...Then...ElseIf...Else и Select Case).

Цикл — это управляющая конструкция, предоставляющая возможность многократно выполнять группы инструкций, до наступления какого-либо события.

*Конструкция If...Then*

Конструкция If...Then дает VBA указание принять простейшее из решений: если условие If истинно, нужно выполнить следующий за ним оператор (или операторы); если же условие ложно, нужно прейти к строке, расположенной непосредственно за условной конструкцией. Данная конструкция может располагаться как в одной, так и в нескольких строках. Однострочная конструкция If...Then выглядит так:

If условие Then оператор[ы]

Если условие выполняется, VBA обрабатывает оператор или операторы, которые следуют за условием; в противном случае эти операторы пропускаются. Конструкция If...Then, занимающая несколько строк (известна также как блок If), выглядит следующим образом:

If условие Then

оператор

[операторы]

End If

Если условие истинно, VBA, опять-таки, выполняет оператор или операторы, а в противном случае передает управление строке, расположенной под строкой End If.

Конструкция If...Then...Else

Конструкция If ...Then — прекрасный инструмент для выполнения одного действия, зависящего от условия. Но если приходится выбирать одно из двух возможных направлений, в которых будут производиться действия, следует воспользоваться конструкцией If...Then...Else. С помощью этой конструкции выбирается одно направление деятельности, если условие истинно, и другое, если оно ложно.

Например, конструкция If...Then...Else является очень удобным инструментом для работы с окнами сообщений, содержащими по две кнопки.

Синтаксис рассматриваемой конструкции таков:

If условие Then

операторы\_1.

Else

операторы\_2

End If

Если условие истинно, VBA выполняет первую группу операторов — onepaторы\_1,но если условие ложно, осуществляется переход к строке Else, а затем выполняется вторая группа операторов — операторы\_2 [12].

*Конструкция Select Case*

Вместо нескольких операторов Elself можно применить конструкцию Select Case, которая позволит представить ту же задачу принятия решения в виде более сжатой и эффективной программы.

Конструкцию Select Case рекомендуется использовать в том случае, если решение, которое нужно принять в программе, зависит от одной переменной иди от выражения, имеющего не менее трех-четырех значений. Такая переменная (или выражение) называется тестируемым случаем.

Конструкцию Select Case легче читать, чем сложные операторы If ...Then, главным образом потому, что она более компактна. Кроме того, в нее легче вносить изменения: когда нужно изменить одно или несколько используемых значений, приходится просматривать меньшее количество операторов. Синтаксис оператора Select Case таков:

Select Case проверяемое\_выражение

Case выражение\_1

операторы\_1

[Case выражение\_2

операторы\_2]

[Case Else

операторы]

End Select

На первый взгляд данный синтаксис может показаться сложным, но это не так. Оператор Select Case начинает конструкцию, а оператор End Select завершает ее. Проверяемое выражение определяет, какой из операторов Case будет выполнен, а выражение\_1, выражение\_2 и т. д. - это те выражения, с которыми VBA сравнивает проверяемое. Предположим, необходимо проверить, кнопку с каким номером выбрал пользователь в диалоговом окне или в форме. Значит, проверяемое выражение будет связано с событием выбора кнопки. Если это первая кнопка, значение проверяемого выражения совпадет с выражением\_1. В таком случае будут выполнены операторы, находящиеся в строках, следующих за оператором Case выражение\_1. Если это вторая кнопка, то значение проверяемого выражения совпадет с выражением\_2, а следовательно, будут выполнены операторы, расположенные в строках, следующих за оператором Case выражение\_2. То же можно сказать и об остальных операторах Case. Оператор Case Else подобен ключевому слову Else в операторе If. Он представляет собой необязательный оператор, который, будучи указанным, выполняется в тех случаях, когда ни одно из предложенных выражений не подходит.

3 РАЗРАБОТКА ИМИТАЦИОННОЙ МОДЕЛИ автоматизированного участка обработки деталей

## 3.1 Концептуальная модель

Для реализации ИМ предлагается рассмотреть функционирование автоматизированного участка обработки деталей, состоящего из входного конвейера, транспортного робота и двух станков для обработки деталей. Транспортный робот выполняет весь цикл переноски деталей двух типов на два различных станка. Каждая деталь в зависимости от типа переноситься разное количество времени, т.к. обслуживающие их станки находятся на разном расстоянии от входного конвейера. Детали могут образовывать очереди как перед перевозкой на транспортном роботе, так и перед обработкой на станках. После переноски деталей и последующей их обработки детали поступают на выходной конвейер (рисунок 2.1).



Рисунок 2.1 – Структурная схема объекта моделирования

Цель моделирования:

Изучение влияния интенсивности поступления деталей на загрузку транспортного робота и станков обработки.

1. Декомпозиция сложной системы.

Декомпозиция системы осуществляется просто: отдельными элементами модели будут очереди, транспортный робот, станки (в дальнейшем этапы обработки).

1. Выбор параметров и переменных.

В качестве параметров системы будет интенсивность поступления деталей на транспортировку и обработку на станках () ().

Переменными модели системы являются функция распределения длительности перевозки или обработки детали на i-том этапе .

В качестве статистик моделирования будут выступать:

* - коэффициенты загрузки этапов обработки ();
* - количество обработанных деталей на автоматизированном участке () и на i-м этапе обработки ();
* - размер очереди к каждому этапу обработки ();
* - общее время обработки деталей на i-ом участке обработки ().
1. Уточнение критериев эффективности.

Так как структура модели проста, то в качестве критериев эффективности могут выступать коэффициент загруженности этапов обработки () и среднее значение времён обработки деталей ().

В результате имитационного моделирования нужно найти следующие функциональные зависимости:  и .

1. Аппроксимация реальных процессов математическими величинами.

При задании функций распределения  длительность обработки детали на i-ом этапе обработки достаточна аппроксимация ступенчатыми функциями.

1. Выдвижение гипотез и предположений.

Выдвигаем гипотезу, что  и  имеют вид полиномов, порядок и значение коэффициентов, которых необходимо определить в ходе имитационных экспериментов.

1. Установление основной структуры моделирования.

С помощью таких эмпирических зависимостей можно предсказывать характеристики загрузки этапов обработки () и времена обработки деталей на этапе обработки () в зависимости от изменяющихся характеристик входного потока деталей () при заданных  [1].

##

## 3.2 Формальное описание модели

##

## Алгоритм активностей

Активность поступления деталей

1. Поиск места для детали:

'Ищется место для детали. Либо новая строка, либо на место ушедшей.

i = 2

While Cells(i, 1).Value > 0

i = i + 1

Wend

Cells(i, 1).Value = ModelTime

lsum = lsum + (ModelTime - tlprev) \* l

l = l + 1

tlprev = ModelTime

1. Генерация поступления и определение типа детали

 'Генерируем поступление деталей в зависимости от параметров табличного распределения и определяем тип детали в зависимости от значения параметра равномерного распределения.

tActions(0) = tActions(0) + RndN1(k, min, max, v)

If tActions(0) > tRun Then

 tActions(0) = TBIG

End If

i = 2

While Worksheets("Elements").Cells(i, 2).Value > ""

 i = i + 1

Wend

Sheets("Elements").Select

det = RndP(Rp)

If det = 1 Then

 Cells(i, 2) = 1

 Else

 Cells(i, 2) = 2

End If

Активность перевозки детали

1 Поиск первой детали в очереди

'найти первую деталь в очереди, т.е. с минимальным значением в столбце 1 листа Elements

last = Range("A30000").End(xlUp).Row

tinmin = TBIG

For i = 2 To last

If (Cells(i, 3).Value = "") And (Cells(i, 1).Value > 0) And (Cells(i,1).Value < tinmin) Then

 inmin = Cells(i, 1).Value

 imin = i

 End If

Next i

lsum = lsum + (ModelTime - tlprev) \* l

l = l - 1

tlprev = ModelTime

1. Перевозка детали транспортным роботом

'присвоение состоянию робота значение 1, т.е. робот занят

'начало перевозки детали роботом

state = 1

tkprev = ModelTime

tActions(1) = ModelTime

Cells(imin, 3) = tActions(1)

If Cells(imin, 2) = 1 Then

 tActions(2) = tActions(1) + Tdown

 Else

 tActions(3) = tActions(1) + Tup

End If

'окончание перевозки детали роботом, в зависимости от типа детали на перевозку затрачивается различное количество времени

'если тип детали 1

If (tActions(2) <= ModelTime) And (Cells(imin, 2) = 1) And (state = 1) Then

 ksum = ksum + (ModelTime - tkprev)

 tkprev = ModelTime

 l1 = l1 + 1 'увелечение очереди на 1 к первому станку

 Cells(imin, 4) = tActions(2) 'окончание перевозки детали роботом

 tActions(1) = tActions(2) + Tdown 'возвращение робота от первого станка

 tActions(2) = TBIG

End If

'если тип детали 1

If (tActions(3) <= ModelTime) And (Cells(imin, 2) = 2) And (state = 1) Then

 ksum = ksum + (ModelTime - tkprev)

 tkprev = ModelTime

 l2 = l2 + 1 'увелечение очереди на 1 ко второму станку

 Cells(imin, 4) = tActions(3) 'окончание перевозки детали роботом

 tActions(1) = tActions(3) + Tup 'возвращение робота от второго станка

 tActions(3) = TBIG

End If

'изменение состояния робота на 0, т.е. незанет

If (state = 1) And (tActions(1) <= ModelTime) Then

 state = 0

 tActions(1) = TBIG

End If

Активность начала обработки детали на станке

'обработка детали первого типа

If (state1 = 0 And l1 > 0) Then

'найти первую деталь в очереди, т.е. с минимальным значением в столбце 4 листа Elements

 last = Range("D30000").End(xlUp).Row

 tinmin = TBIG

 For i = 2 To last

 If (Cells(i, 5).Value = "") And (Cells(i, 2) = 1) And

 (Cells(i, 1).Value > 0) And (Cells(i, 1).Value < tinmin) Then

 tinmin = Cells(i, 1).Value

 imin1 = i

 End If

 Next i

 i1 = 1

 state1 = 1 'станок занят

 l1 = l1 – 1 'уменьшение очереди к станку на 1

 If ModelTime > Cells(imin1, 4) Then

 tActions(4) = ModelTime

 Else

 tActions(4) = Cells(imin1, 4)

 End If

 Cells(imin1, 5) = tActions(4)

 tActions(6) = tActions(4) + one 'время обработки детали типа на станке

End If

обработка детали второго типа

If (state2 = 0 And l2 > 0) Then

'найти первую деталь в очереди, т.е. с минимальным значением в столбце 4 листа Elements

 last = Range("D30000").End(xlUp).Row

 tinmin = TBIG

 For i = 2 To last

 If (Cells(i, 5).Value = "") And (Cells(i, 2) = 2) And

 (Cells(i, 1).Value > 0) And (Cells(i, 1).Value < tinmin) Then

 tinmin = Cells(i, 1).Value

 imin2 = i

 End If

 Next i

 i2 = 0

 state2 = 1

 l2 = l2 - 1

 If ModelTime > Cells(imin2, 4) Then

 tActions(5) = ModelTime

 Else

 tActions(5) = Cells(imin2, 4)

 End If

 Cells(imin2, 5) = tActions(5)

 tActions(7) = tActions(5) + two

End If

Активность окончания обработки детали на станке

'окончание обработка детали первого типа

If (tActions(6) <= ModelTime) Then

NextModelTime = False

'откорректировать значения для среднего времени пребывания детали в системе

Nb = Nb + 1

tbuysum = tbuysum + (ModelTime - Cells(imin1, 1).Value)

If i1 = 1 Then

 state1 = 0

End If

Cells(imin1, 6) = tActions(6)

tActions(6) = TBIG

End If

'окончание обработка детали второго типа

If (tActions(7) <= ModelTime) Then

 NextModelTime = False

'откорректировать значения для среднего времени пребывания детали в системе

 Nb = Nb + 1

 tbuysum = tbuysum + (ModelTime - Cells(imin2, 1).Value)

 If i2 = 0 Then

 state2 = 0

 End If

 Cells(imin2, 6) = tActions(7)

 tActions(7) = TBIG

 Call Trace("Finish", imin2)

End If

4 ВЕРИФИКАЦИЯ ИМИТАЦИОННОЙ МОДЕЛИ

## 4.1 Контроль за выполнением порядка активностей

Порядок выполнения активностей рассмотрим на примере одной детали при первом прогоне. Arrival, Begin, Processing, Finish – это активности. В первой активности происходит генерация деталей. Первая активность появляется согласно закону нормального табличного распределения с параметрами k, min, max, v(i), где i=. Поэтому первая активность появляется случайным образом. На рисунке 4.1 видно, что деталь после ее появления начинают обрабатывать во второй, третьей, а затем и в четвертой активности.



Рисунок 4.1 - Изменение откликов на листе Trace