ПРОГРАММНАЯ ПОДСИСТЕМА ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ ДИСКРЕТНЫХ ПРОИЗВОДСТВЕННЫХ СИСТЕМ

С. А. Лазарев (Орел)

Одной из важнейших и наиболее сложных частей общей системы управления предприятием является система управления производством. Одним из основных факторов повышения эффективности и качества функционирования системы управления производством является разработка и внедрение комплексных автоматизированных систем планирования и диспетчирования. При этом основу данной системы должна составлять интегрированная имитационная модель функционирования производства. Данные автоматизированные системы, как правило, представляют собой мощные программные комплексы, реализованные на языках высоко уровня [1].

Машинная же реализация имитационной модели возможна с применением двух различных программных средств построения модели:

- специализированных систем (языков) моделирования (GPSS, Dinamo, AnyLogic);
- универсальных языков программирования высокого уровня (Delphi, C++, Java).

Специализированные системы имитационного моделирования ориентированы, в основном, на построение имитационной модели как отдельного объекта для последующего анализа. Основной акцент в них сделан на визуализацию процесса и результатов моделирования. Они также имеют определенные сложности интеграции построенной модели в разрабатываемый программный комплекс. Специализированные языки не обеспечивают возможность создания графического интерфейса пользователя, отвечающего требованиям программного комплекса. Они также не имеют мощных средств работы с базами данных.

В свою очередь универсальные языки не имеют характерных для языков моделирования средств формализации и описания модели, необходимых структур хранения данных.

Кроме того, при разработке программного и технического обеспечения функционирования системы моделирования следует учитывать их совместимость с имеющимися программными и техническими средствами АСУ промышленных предприятий.

Исходя из вышеизложенного, в контексте создания комплексной автоматизированной системы планирования и диспетчирования возникает задача создать специализированный инструментарий, функционирующий в среде разработки высокого уровня для построения имитационной модели производственной системы. Данный программный инструментарий должен отражать специфику и характер производства, а также обеспечивать полную интеграцию имитационной модели в программный комплекс с целью повышения эффективности его разработки, дальнейшей адаптации и функционирования.

В этой связи наиболее оптимальным подходом является специализированный язык моделирования, созданный на основе универсального языка в виде отдельного программного модуля. Модуль реализует в себе все необходимые структуры данных и средства формального описания модели.

Основные принципы моделирования

Производственная система с позиций имитационного моделирования представляется как программная модель сложной системы, в которой отражены структура, алгоритмы развития и протекания процессов во времени, временные характеристики от-

дельных элементов. Имитация имеет своей основной целью моделирование динамики, т. е. изменение состояния системы во времени.

Как и всякому формализованному подходу, имитационному моделированию присущи свои понятия и атрибуты [2, 3].

Время моделирования – это временной интервал, на котором имитируется поведение системы, т.е. в сущности, имитационное представление реального времени.

Активность – наименьшая единица работы при выбранном уровне представления моделируемой системы, которая рассматривается как единый дискретный шаг.

Процесс – совокупность логически связанных активностей.

Событие – мгновенное изменение состояния некоторого объекта системы, который может быть активным либо пассивным. События можно разделить на две категории: события следования (управляют инициализацией активностей внутри данного процесса) и события изменения состояния (управляют выполнением активностей, относящихся в общем случае, к независимым процессам).

Транзакты – динамические объекты, представляющие собой поток элементов обслуживания и являющиеся конкретной реализацией процессов.

Средства – функционально-ориентированные объекты, которые соответствуют элементам оборудования или рабочим, обслуживающим транзакты.

Очереди можно рассматривать как статические объекты, позволяющие оценить поведение системы. В очередь попадают транзакты, которые задержаны в какой-то момент времени до тех пор, пока не выполнится условие, необходимое для их продвижения.

Необходимо отметить, что при моделировании процессы, события и активности целиком зависят от потоков и траекторий движения транзактов: транзакт, попадая в моделируемую систему, занимает определенные блоки, вызывая при этом события. Наступление событий должно планироваться соответствующими средствами моделирования. При выполнении определенных условий событие вычеркивается из системы моделирования, а на смену ему должны приходить следующие события.

При событийном моделировании производственной системы выделяют узловые моменты динамики в виде событий. В процессе моделирования осуществляется переход (скачок во времени) от предыдущего события к последующему. Каждое событие выполняется мгновенно во времени, модельное время затрачивается только на переход от события к событию. Реализация событий во времени напоминает цепную реакцию: при отработке любого события планируется одно или несколько последующих (будущих) событий.

Общая структура языка EML

В общем, случае динамическое поведение системы можно представить как выполнение большого числа взаимодействующих процессов, которые сводятся к некоторому небольшому числу классов. Тогда, чтобы описать поведение системы, достаточно описать поведение каждого класса процессов и задать значения атрибутов для конкретных процессов (т.е. транзактов). Поскольку процесс представляет собой совокупность логически связанных активностей, описание процесса должно включать в список активностей, входящих в процесс, порядок следования этих активностей и условия выполнения каждой активности.

Модель системы на языке EML можно рассматривать как набор событий, где события представлены завершениями активностей, и, следовательно, описание события состоит из совокупности операций, которые должны выполняться после совершения соответствующей активности. Поведение же системы во времени отображается порядком следования этих событий.

Упорядочение событий достигается с помощью списка событий. При планировании события указывается номер, время события и номер транзакта. Чтобы получить момент времени моделирования, в который должно произойти некоторое событие, время выполнения события (точнее, время выполнения работы, представленной данной активностью) складывается с текущим временем. Это время, номер события и номер транзакта затем помещаются в список событий, который упорядочивается в порядке возрастания значений времени. Затем из списка вычеркивается событие, которое должно завершиться первым, время моделирования становится равным времени завершения этого события (т.е. продвижение по оси времени осуществляется на величину очередного интервала между событиями), выполняются действия, соответствующие завершению данного события, и планируется очередное событие. Если при этом оказывается, что для какого-то транзакта очередное событие не может быть спланировано из-за занятости оборудования, то данный транзакт помещается в очередь. Транзакты, входящие в очередь, упорядочиваются по значению единственного атрибута (приоритета), а в случае равенства этих значений – в порядке поступления.

Каждый транзакт представляет собой деталь, сборку или изделие и имеет уникальный номер, используемый в модели для идентификации каждого компонента изделия.

Транзакты, относящиеся к определенному изделию, находятся в иерархической подчиненности между собой согласно принципу вхождения деталей в сборки, сборок в узлы, узлов в изделие. Следовательно, завершение обработки транзакта, представляющего собой изделие, невозможно, пока не будут обработаны все транзакты, представляющие детали, сборки и узлы, входящие в изделие. На этом принципе основана логика функционирования модели – восхождение от основания иерархической структуры изделия к ее вершине. Поиск же оптимального с точки зрения производства пути и времени прохождения данного этапа и составляет основную задачу имитационного моделирования.

Все переменные языка EML, включая время, являются целочисленными. Единица времени моделируемой системы определяется пользователем, и все временные интервалы, разделяющие моменты наступления запланированных событий, должны быть заданы в этих единицах.

Внутренняя организация языка EML

Основой внутренней организации являются объекты, описывающие устройства (рабочих), очереди, события. Они представляют собой реляционные таблицы, взаимосвязанные на уровне программного модуля, которые легко интегрируются в любой СУБД реляционного типа. Поля соответствующих таблиц соответствую определенным атрибутам и характеристикам данных объектов, а записи отдельным экземплярам данного класса.

Для каждого устройства (рабочего) моделируемой системы строится заголовок очереди и собственно очередь к данному средству. Элемент очереди соответствует одному транзакту в состоянии ожидания, т.е. в частном случае очередь может быть пустой.

Для работы с устройствами и очередями определены специальные методы, которые осуществляют резервирование и освобождение устройства, постановку и извлечение транзактов из очереди.

Для обработки динамического списка событий, наступающих в системе, определено методы планирования новых событий, отмены ранее запланированных и чтения текущего события.

Каждый вызов функции Schedule (планировать событие) сопровождается включением в список событий нового элемента. Этот список автоматически упорядочивается по возрастанию времени совершения событий.

Для обработки событий в системе EML применен метод регистрации событий. Каждому типу событий сопоставлен адрес процедуры его обработки. Реализация этот механизм с помощью объекта процедурного типа (TRegiteredEvents).

Во всех описанных списках используется понятие номер транзакта. Эта характеристика вводится для того, чтобы отличать элементы списка, представляющие разные реализации процессов. Другими словами, транзакты, вызванные двумя разными запросами, должны иметь и разные номера.

Все перечисленные выше структуры данных объединены в объект *TModel*, являющийся основой для построения модели. Он содержит в себе следующие методы:

- *Run* – осуществляет процесс моделирования;
- *RegisterEvents* – регистрирует процедуры обработки событий;
- *Error* – выдает сообщение об ошибке в процессе моделирования;
- *Setup* – настраивает параметры модели (вызывается конструктором *Create*);
- *Report* – создает отчет о результатах моделирования.

При построении моделирующей программы необходимо переопределить соответствующие методы объекта *TModel* для придания им необходимых свойств.

Для хранения текущего времени служит глобальная переменная *_Time*, которой присваивается нулевое значение, используемое в качестве начального значения времени моделирования.

В языке EML имеются три статистические функции, которые возвращают случайное число, распределенное равномерно, экспоненциально, нормально.

Для контроля над ходом выполнения программы в системе EML предусмотрена возможность создания файла трассировки, в котором отражается каждое обращение к процедуре или функции языка EML.

Особенности моделирующей программы на языке EML

Имитационная модель и реализующая ее программа, построенная с использованием языка EML, должна состоять из ряда блоков:

- инициализации – в его функции входит возбуждение системы имитации модели во времени, описание моделируемой системы и генерация первого события;
- управления – определяет логику функционирования модели, изменяет текущее время моделирования и инициирует обработку событий. Как только время достигнет предельного значения, работа программы приостанавливается;
- обработки событий – выполняет действия, связанные с окончанием данного события, и планирует новые события, которые могут происходить в данные момент;
- генерации отчетов – формирует для пользователя необходимые отчеты на основе накопленной статистической информации.

Выполнение моделирующей программы может быть прервано из-за ошибок, допущенных при реализации алгоритма функционирования моделируемой системы средствами языка EML. Для своевременного обнаружения ряда ошибок и облегчения их поиска в языке предусмотрены процедуры контроля ошибочных ситуаций, обеспечивающие приостановку работы программы.

Заключение

Описанный выше язык моделирования может быть использован для имитационного моделирования дискретных производственных систем с малой и средней размерностью на ЭВМ.

Используя особенности объектно-ориентированного программирования, методы, реализованные в модуле языка EML, могут быль переопределены и модифицированы в каждой конкретной реализации имитационной модели. Это обуславливает некоторую универсальность подсистемы моделирования и легкость адаптации подхода, реализованного в нем, к особенностям реальных задач построения имитационных моделей производства в контексте создания комплексных автоматизированных систем планирования и диспетчирования.

Такая программная подсистема имеет определенные преимущества, сочетая в себе основные достоинства универсальных и специализированных языков, а также обеспечивая легкость интеграции построенных с его помощью моделей в программные комплексы. Основным ее недостатком является невозможность использования моделирующего модуля с другими языками программирования без "переписывания" его на эти языки.

Литература

1. **Савина О. А.** Управление промышленными предприятиями с использованием систем поддержки решений. М.: Изд-во МАИ, 2000. 256 с.
2. **Шеннон Р.** Имитационное моделирование систем – искусство и наука/Пер. с англ. М.: Мир, 1978. 418 с.
3. **Федорович О. В., Шевелева О. А.** Исследование гибких дискретных производств методами имитационного моделирования. Харьков. Изд-во ХАИ, 1985.