УДК 303.094.7

С. А. Даденков, Е. Л. Кон
Пермский национальный исследовательский политехнический университет

Анализ моделей и методов агентного и дискретно-событийного имитационного моделирования

Выполняется анализ соотношения агентного и дискретно-событийного методов имитационного моделирования. В результате анализа основных этапов разработки модели – проектирования, моделирования, сопровождения – выделяются преимущества и недостатки методов.

Имитационное моделирование, дискретно-событийный подход, агентный подход, анализ методов

В настоящее время имитационное моделирование (ИМ) активно применяется во всех сферах человеческой деятельности. Актуальность применения и распространенность систем имитационного моделирования показана во многих источниках [1], [2]. При выборе систем моделирования принято выделять системы проблемно-ориентированного (специализированные) и общецелевого назначения (универсальные). Специализированные имеют узкие области применения, например: OPNET (SteelCentral), COMNET – вычислительные сети; eMPlant, DELMIA – машиностроение и судостроение; ISSOP – производство и логистика; PTV Vision Vissim – транспортные потоки и дорожное движение. Для построения моделей в таких системах используются готовые наборы типовых элементов моделируемого объекта. Общецелевые системы моделирования содержат лишь базовые элементы, из которых строятся модели. Чаще всего это элементы систем массового обслуживания (СМО) – заявки, обслуживающие устройства, ресурсы. Примером последних являются распространенные системы AnyLogic, GPSS (General Purpose Simulation System), Arena, которые применяются практически во всех областях промышленности и народного хозяйства [1], [2]. В настоящее время развиваются комбинированные (смешанные) системы ИМ, в которых возможно общецелевое проектирование, а также предлагаются готовые библиотеки объектов систем. Указанным свойством характеризуется система AnyLogic, содержащая пешеходную и железнодорожную библиотеки. Достоинством универсальных систем моделирования, в том числе рассматриваемых в статье, является гибкость, масштабируемость, интерактивность разрабатываемых моделей [3].

Основу функционирования систем имитационного моделирования (СИМ) составляют способы представления модельного времени – дискретный и непрерывный [3], [4]. Для дискретного характерно использование дискретно-событийного (discrete-event simulation, DES) и агентного (agent-based simulation, ABS) методов (подходов) моделирования; для непрерывного способа – методов системной динамики (system dynamics simulation, SDS) и динамических систем (dynamic systems simulation, DSS). Наиболее важный этап разработки модели – этап планирования, одной из актуальных задач которого является выбор метода моделирования [4]. Важность адекватного выбора метода моделирования и последствия иного выбора отмечаются в работах [5], [6]. Анализ публикаций и литературных источников, посвященных описанию и сравнению методов моделирования, показывает недостаточность уровня проработанности вопросов корректного выбора подхода(ов) к моделированию [7]–[9]. Во многом это обусловлено наличием в литературе множества только частных примеров моделирования, без объяснения необходимости применения того или иного метода. Кроме того, предлагаемая литература в большей степени ориентирована на опытных разработчиков и не позволяет неподготовленному читателю, знакомому с каким-либо одним методом моделирования, понять достоинства и недостатки других методов. В связи с этим проведем сравнительный анализ традиционного для разработчиков дискретно-событийного и «нового» агентного подходов. Результаты анализа должны позволить разработчику на качественном уровне понять эффективность подходов и необходимость

их использования посредством выделения достоинств и недостатков проектирования и применения моделей; взять на вооружение метод агентного моделирования для решения задач определенного класса. Наконец, предлагаемый в статье подход к анализу соотношения методов отличается оригинальностью и простотой, предоставляя результаты, понятные разработчикам, знакомым с основами дискретно-событийного моделирования, основами программирования.

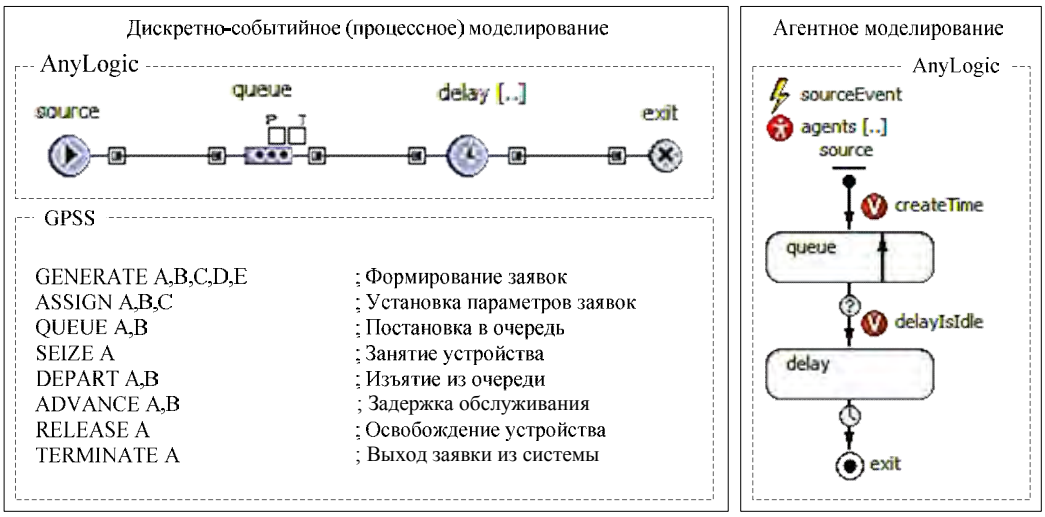
Для решения поставленной задачи материал в статье систематизирован следующим образом. Вначале, в необходимом для анализа объеме, рассматривается эволюция развития дискретно-событийного подхода на примере систем моделирования GPSS и AnyLogic. Показывается современное состояние дискретно-событийного подхода как полноценного объектно-ориентированного подхода к моделированию. Анализируются возможности методов моделирования. С позиции анализа методологий проектирования модели показывается простота применения агентного подхода для решения задач класса динамически взаимодействующих объектов (заявок). В результате анализа основных этапов разработки модели – проектирования, моделирования, сопровождения – выделяются преимущества и недостатки методов, а также предлагаются рекомендации к их применению в общем виде.

Анализ развития методов моделирования сложных систем и процессов. В настоящее время разработчики моделей преимущественно используют мощности ставшего традиционным дискретно-событийного метода моделирования. При этом лишь немногие начинают применять «новый» агентный подход к построению моделей, интуитивно понимая его преимущества. Сложность применения агентного подхода обусловлена следующим основными положениями [10], [11]: отсутствует общепризнанное определение «агента»; до сих пор идут споры о качествах объекта, присущих агенту; отсутствует критериальная оценка достоинств и недостатков подхода к моделированию. В [10] производится противопоставление агентов и заявок в системе, приводится обоснование разницы между ними на уровне их поведения (взаимодействия с другими заявками). В работе [10] автор настаивает на разнице в уровне абстракции проектируемых моделей. В [8] приводится мнение об отличительных чертах агента – внутренней анимации в виде его состояний. В других работах предлагается некорректное, как будет показано по результатам анализа, разделение подходов по возможностям (мощностям) разработки модели. Таким образом, существует сложность понимания достоинств и недостатков и, как следствие – сложность выбора и применения агентного моделирования.

Авторы считают, что наиболее понятный и простой путь анализа методов моделирования – анализ развития моделей и методов моделирования с лежащими в их основе языками программирования. В связи с этим далее приводится анализ особенностей эволюционирования дискретно-событийного метода к агентному с позиции развития языков программирования от процессно-ориентированного к объектно-ориентированному; показывается необходимость и целесообразность развиваемого агентного моделирования; выделяются преимущества и недостатки подходов. На каждом этапе развития методов моделирования авторы обращают внимание на объектный (агентный) подход разработки модели как необходимость для построения сложных и адекватных моделей.

Представленный в статье анализ методов, по мнению авторов, носит универсальный характер, т. е. полученные результаты справедливы для большинства известных систем моделирования. Анализ развития методов моделирования выполняется на примере двух наиболее распространенных в России и за рубежом систем имитационного моделирования (СИМ) – GPSS World и AnyLogic [7]. Противопоставление мощностей языковой (GPSS) и графической (блочно-языковой) (AnyLogic) систем, с точки зрения авторов, позволит оценить возможности и сложности реализации (проектирования и моделирования) моделей, а в итоге – сформулировать корректные выводы.

Исторически первый этап становления систем моделирования (GPSS – 1961 г.; AnyLogic (COVERS) – 1990 г.) основан на принципах структурного программирования. Модель представляется в виде иерархической структуры (см. рисунок) (программного кода в GPSS, графических блоков в AnyLogic), объединяющей структуры: выбора (условия), повторений (циклы), следования (последовательности действий), транзакта (далее используется общепринятый в СМО термин – заявки), обслуживания. Уже на данном этапе становления дискретно-событийного моделирования разработчики работают с объектами



системы, реализующими указанные структуры. Объектами являются заявки, характеризуемые набором присущих им индивидуальных параметров (приоритетов, вероятностных и временных параметров), и обслуживающие устройства (программные, графические блоки), характеризуемые набором методов работы с заявками (задержка, выделение ресурса, перенаправление, проверка условий), методов сбора статистики. Методология проектирования дискретно-событийной модели осуществляется посредством определения глобального процесса, представляемого в модели хронологической последовательностью отдельных процессов, реализуемых блоками программы (совокупности процессов образуют потоковые диаграммы, flowcharts). Процессные модели простейшей СМО представлены на рисунке.

Глобальный процесс в модели децентрализован среди множества программных блоков, функционал которых условно независим (автономен и определяется собственными дисциплинами обслуживания), а зависимости (взаимовлияния) между процессами могут осуществляться через индивидуальные параметры каждой заявки и/или глобальные «переменные». Далее такой вид зависимости (поведения) в модели будем называть статическим. Примером статического поведения может быть зависимость времени обслуживания заявки на текущем участке (этапе) от затраченного времени в другом (предшествующем) процессе для этой же заявки. Согласно методологии проектирования моделей дискретно-событийным подходом, а также ввиду сохранности его принципов для агентного далее в статье дискретно-событийный подход целесообразно именовать процессным.

Дальнейший этап развития дискретного подхода связан с необходимостью упрощения сложных моделей посредством уменьшения объема однотипного (однофункционального) кода, используемого на различных участках модели. Происходит становление процедурной парадигмы программирования. Отметим, что данный этап является промежуточным и не влияет на развитие агентного моделирования, рассматривается только для сохранности хронологии этапов эволюционирования систем моделирования и программирования. Например, в системе GPSS World (1993) добавление процедур выполнено с помощью надстройки встроенного языка программирования низкого уровня PLUS. В СИМ AnyLogic процедуры поддерживались, начиная с первых выпусков программного обеспечения.

Следующий этап становления процессного моделирования может быть проиллюстрирован на примере системы AnyLogic, построенной на базе объектно-ориентированного языка программирования Java. В модель вводятся классы объектов. Классом называется дополнительная структура объекта, описывающая индивидуальные переменные (поля) и функции (методы) работы с ними. Использование классов существенно облегчило написание программного продукта, сокращая объемы однотипного кода. В GPSS World отсутствуют пользовательские классы (userClass) объектов, что наряду с отсутствием графического интерфейса позволяет считать модели, разработанные в AnyLogic, более гибкими, масштабируемыми, наконец, просто понятными пользователям [7].

Дальнейшая ступень объектной ориентации методов моделирования, в частности системы AnyLogic, вызвана необходимостью моделирова-

ния более сложных распределенных систем, составные процессы которых характеризуются динамическими взаимосвязями, определяемыми поведением объектов модели – заявок и обслуживающих устройств системы [11]. Под поведением в статье понимается последовательность процессов и сопутствующих им событий, в которые вовлечена заявка в модели. Таким образом, в отличие от статического динамическое поведение характеризуется наличием зависимости не только от текущих и нескольких предыдущих индивидуальных событий заявки, но и зависимости от поведения других элементов системы. Иными словами, динамическое поведение заявок в системе обусловлено зависимостью поведения одних групп объектов (чаще – заявок) в системе от других групп. Примером динамического поведения является диалог группы людей, в результате которого часть людей изменяет свое мнение, а другая остается при своем. Как правило, динамическое поведение сопровождается лавинообразным (каскадным) появлением в системе большого числа событий, в рамках которых каждая заявка «мгновенно» пересматривает свое участие и местоположение в процессах модели. Лавинообразность событий объясняется тем, что помимо простой зависимости от события заявки могут иметь дополнительные признаки взаимозависимости, так называемое родство, при котором изменяется не только поведение элементов в рамках одного процесса, но и поведение объектов, находящихся на различных его стадиях (участках процесса). Напомним, что до этого процессное моделирование рассматривалось исключительно как совокупность условно независимых статичных процессов. В связи с этим моделирование систем с динамическими взаимосвязями становится весьма затруднительным. В частности, при традиционном процессном моделировании для задания динамических взаимосвязей разработчик должен обладать подробнейшей информацией обо всех заявках в системе, их параметрах, распределенности по программным блокам модели, состояниях их обслуживания. Таким образом, становится необходимым получение доступа на чтение и изменение системных списков запланированных в модели событий. Ввиду сложности системных списков, а также «закрытости» методов работы с ними в СИМ предлагаются различные инструментарии. В системе GPSS в распоряжение разработчика предоставляются списки пользователя (блок LINK) [12], которые должны анализироваться при каждом событии в модели и дополняться структурами выбора, повторений, следования для сортировки и перенаправления заявок. Отметим, что количество операций со списками событий возрастает в геометрической прогрессии с ростом сложности модели, определяемой количеством блоков и соответственно списков. В системе AnyLogic реализован иной подход, основанный на сохранении единого списка всех активных заявок в модели (принимающих участие в моделировании). Таким образом, в распоряжении разработчика в каждый момент времени находится массив всех заявок, участвующих во всех процессах системы, доступ к которому организован с помощью процедуры getDefaultPopulation. Разработчик в AnyLogic, в отличие от GPSS, может анализировать как отдельные элементы списка, так и целиком список заявок, управляя ими с помощью структур выбора и следования. Таким образом, становится понятнее и целесообразнее работа с заявками в системе как с некоторыми ее объектами. Трудность применения данного подхода в рамках процессного моделирования может быть обусловлена сложностью инфраструктуры модели, характеризуемой наличием в системе заявок различных классов, большого числа элементов, реализующих процессы обслуживания, что затрудняет доступ к заявкам, требует выполнения работ, направленных на определение классов заявок в списках, места их нахождения в системе (определение блоков). Наконец, требуется определение множества различных структур управлений в блоках программы. Сложность реализации является значимой характеристикой СИМ не только на этапе проектирования, но и при сопровождении (модернизации) модели. Так, внесение изменений (например, добавление блоков) в процессную структуру может потребовать изменения каждого блока, принимающего участие в динамическом управлении процессом.

В связи с этим следующим этапом становления методов моделирования становится полнофункциональный объектно-ориентированный, или агентный подход, в рамках которого разработчику модели предоставляются эффективные и простые методы работы с объектами модели. В агентном подходе система представляется не как совокупность процессов в системе (процессное моделирование), а в виде набора объектов различных классов – агентов, представляющих собой заявки, ресурсы, обслуживающие системы. Каждый агент системы может иметь свою программную реализацию, традиционно в ней описываются процессы жизненного цикла объекта в виде конечного автомата или диаграммы состоя-

ний (statechart). Именно возможность программной реализации и простота работы с динамическим взаимодействием объектов, по мнению авторов, отличает агента от заявок (объектов) процессного моделирования. Таким образом, методология проектирования дискретно-событийных моделей отходит от принципов процессного моделирования «от процесса к объектам» и переходит к агентным принципам «от объектов к процессам». При этом сохраняется принцип децентрализации процесса по объектам модели, но уровень децентрализации, как правило, становится ниже и может изменяться в зависимости от реализации модели. В общем случае агент может содержать полный жизненный цикл или лишь его часть, а реализация другой части (общей с другими агентами) выносится в отдельные элементы (агенты). С позиции обозначенного, децентрализацию процессного моделирования можно охарактеризовать наиболее высокой (по отдельным очередям, обслуживающим устройствам), а уровень децентрализации агентного подхода – более низким (весь процесс или его большая часть реализованы в агенте) (таблица). Таким образом, работа с объектами различного типа наиболее удобна в агентном представлении, поскольку требует лишь выбора нужного класса агентов и внесения изменений, характерных только для этого класса. И наоборот, для процессного представления удобна работа с одним типом заявок либо различными, но с одинаковым статическим поведением. Ввиду отмеченного, агентные модели отличаются от процессных простотой масштабирования, изменения производятся локально, для уточняемого объекта, практически независимо от других элементов модели.

Анализ преимуществ и недостатков методов моделирования. Для решения задачи выбора подхода к моделированию был выполнен их сравнительный анализ по наиболее важным критериям [3]: сложность проектирования модели и вычислительная сложность реализации процессов функционирования модели, обусловливающие время моделирования.

Проанализировав преимущества и недостатки процессного и агентного подходов к моделированию, основываясь на различных методологиях проектирования и анализе функционирования множества моделей, авторы пришли к выводам, представленным в таблице.

Сложность проектирования моделей традиционно определяется необходимостью использования дополнительных объемов программного кода и, как следствие, увеличением времени проектирования (время прямо пропорционально сложности). Для процессных моделей дополнительный код применяется в редких случаях, поскольку большинство возможных событий и их реализаций уже предусмотрено системными разработчиками в программных блоках моделей, а проектировщику модели требуется выбрать дисциплины обслуживания, указать параметры процесса обслуживания и события системы. Для более «сложных» процессных моделей проектировщику требуется работать с классами заявок и методами работы со списками заявок. В отличие от процессной агентная модель характеризуется практической сложностью реализации. Обусловлено это необходимостью реализации с помощью диаграмм состояний алгоритмов постановки, выборки, сортировки агентов в очереди-состоянии, программной реализации дисциплин функционирования и выбора каналов обслуживания. Агентные модели требуют программной реализации элементов сбора статистики, в то время как для объектов диаграммы процессов разработаны «системные» методы сбора вероятностно-временных характеристик. Программная реализация агентных моделей требует от разработчика углубленных знаний объектно-ориентированного программирования, методов работы с диаграммами состояний, и поэтому сложность проектирования агентных систем выше, чем процессных. Предлагаемая авторами оценка ограничена сверху высоким уровнем моделирования систем при реализации не в системах моделирования, а с помощью языков программирования, например *C++*. Об-

Критерий	Моделирование	
	процессное	агентное
Методология проектирования модели	От процесса к объектам	От объектов к процессам
Децентрализация процессов и событий	По процессам (объектам обслуживания)	По объектам-агентам (заявкам, устройствам)
Уровень децентрализации	Высокий	Низкий
Сложность проектирования и сопровождения (статич./динамич.)	Низкая / Высокая	Средняя
Вычислительная сложность	Высокая	Средняя

ратной стороной относительной сложности проектирования агентных моделей является простота ее сопровождения, поскольку уточнения вносятся на локальном уровне по мере накопления данных об объекте.

Распространенная оценка вычислительной сложности является функцией объема работы, выполняемой алгоритмом моделирования в зависимости от исходных данных. Объем работы определяется временем и вычислительными ресурсами (объемом затраченной в расчетах оперативной памяти). Анализ структуры реализации обслуживающих элементов процессной модели на примере AnyLogic показывает для каждого процессного блока большое число (более 20) переменных (параметров), что свидетельствует о высоких затратах вычислительных ресурсов на организацию процесса в системе. Такой же объем процессов в агентном исполнении занимает гораздо меньший объем памяти. Объем памяти, требуемый для реализации заявок в процессной и агентной средах, одинаков ввиду практического отсутствия отличий в структурах заявок. Ряд экспериментов с простыми и сложными моделями показывает практическое равенство времени моделирования процессного и агентного подходов. Тем не менее, заметим, что в некоторых источниках [11] приводится информация о более высоком времени моделирования агентных систем.

Одним из результатов, продемонстрированных в настоящей статье и идущих в разрез с рядом публикаций, является вывод о неделимости процессных и агентных моделей по признаку возможностей. На основе исследованных подходов могут быть построены модели любой сложности. Как показано, отличие заключается в сложности этапов проектирования, моделирования и сопровождения моделей. Используя предложенные критерии и результаты анализа основных этапов имитационного моделирования, разработчик может отдать предпочтение определенному этапу работы с моделью, тем самым определив метод моделирования. В более общем смысле, полученные результаты объясняют рациональность известных высказываний [10], [13]: «При выборе между процессным дискретно-событийным и агентным моделированием надо иметь в виду, что агентные модели обычно более трудоемки в построении и калибровке, поэтому, если система хорошо описывается в виде последовательности операций над (пассивными) объектами, разделяющими ресурсы, нет смысла строить агентную модель»; «Агентное моделирование является эффективным при моделировании систем, содержащих большие количества активных объектов и динамические взаимосвязи между ними».

Кроме того, отметим, что наиболее эффективными являются инструменты, объединяющие в себе лучшие черты исследуемых подходов к моделированию [11]. Уже сегодня известен пример такой мощной системы имитационного моделирования – AnyLogic, предоставляющей разработчику возможность взаимоувязанного использования в рамках одной модели дискретно-событийного («процессного») и агентного подходов.

СПИСОК ЛИТЕРАТУРЫ

1. Области применения системы имитационного моделирования AnyLogic. Многоподходное моделирование AnyLogic. URL: http://www.anylogic.ru/application-areas.

2. Проекты по разработке имитационных приложений компании ООО «Элина-Компьютер». URL: http://elina-computer.ru/static/proekty.html.

3. Шеннон Р. Имитационное моделирование систем – искусство и наука. М.: Мир, 1978. 418 с.

4. Кельтон В., Лоу А. Имитационное моделирование. Классика CS. 3-е изд. СПб.: Питер; Киев: Издательская группа BHV, 2004. 847 с.

5. Борщёв А. В. Как строить простые, красивые и полезные модели сложных систем // Материалы конф. «Имитационное моделирование. Теория и практика» ИММОД, Казань, 2013. URL: http://www.anylogic.ru/articles/kak-stroit-prostye-krasivye-i-poleznye-modeli-slozhnykh-sistem.

6. Егоров С. Выбор языка имитационного моделирования, или не заколачивайте гвозди микроскопом / Блог компании AnyLogic. URL: http://www.anylogic.ru/blog?page=post&blog=blog_RU&id=173.

7. Боев В. Д. Компьютерное моделирование: пособие для практических занятий, курсового и дипломного проектирования в AnyLogic 7: СПб.: ВАС, 2014. 432 с.

8. Wai Kin Victor Chan, Young-Jun Son, Charles M. Macal, Agent-based simulation tutorial – simulation of emergent behavior and differences between agent-based simulation and discrete-event simulation // Proc. of the Winter Simulation Conf., Baltimore, Maryland, Dec. 05-08. 2010.

9. Борщёв А. В. От системной динамики и традиционного ИМ – к практическим агентным моделям: причины, технология, инструменты. URL: http://www.gpss.ru/paper/borshevarc.pdf.

10. Борщев А. В. Практическое агентное моделирование и его место в арсенале аналитика // Exponenta PRO. 2004. № 3–4 (7–8). С. 38–47.

11. Девятков В. В. Методология и технология имитационных исследований сложных систем: современное состояние и перспективы развития. М.: Вузовский учебник; ИНФРА-М, 2014. 448 с.

12. Шрайбер Т. Дж. Моделирование на GPSS / пер. с англ.; под ред. М. А. Файнберг. М.: Машиностроение, 1980. 592 с.

13. Борщев А. В. Применение имитационного моделирования в России – состояние на 2007 г. // сб. докл. III Всерос. науч.-практ. конф. «Имитационное моделирование. Теория и практика» (ИММОД-2007), СПб., 17–19 окт. 2007. Т. I. С. 11–16.

S. A. Dadenkov, E. L. Kon
Perm national research polytechnical university

COMPARISON OF AGENT-BASED AND DISCRETE-EVENT SIMULATION MODELS AND METHODS

In this paper agent-based and discrete-event modeling approach are compared. Through analysis of basic steps in simulation model development process, such as designing, testing and completion, the pros and cons of aforementioned methods are revealed.

Simulation modeling, discrete-event simulation, agent-based simulation, analysis methods simulation

УДК 004.43

А. В. Малов
Motorola Solutions

С. В. Родионов
Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

К вопросу классификации современных парадигм программирования

Рассматриваются основные парадигмы и подходы к программированию, недостатки некоторых из существующих классификаций парадигм программирования. Подчеркивается, что множество современных подходов к программированию определяется не характеристиками языков программирования, а способами проектирования и организации программ и программных комплексов. Предлагается подход к классификации современных парадигм программирования.

Парадигмы программирования, языки программирования, классификация парадигм программирования

В настоящее время развитие информационных технологий идет очень динамично. Множество существующих языков программирования постоянно пополняется новыми языками [1], обновляются стандарты для существующих языков программирования. В основе каждого языка лежит одна либо несколько парадигм программирования. Следует отметить, что современные языки программирования, базирующиеся на какой-либо парадигме, часто обладают свойствами и механизмами управления, относящимися к другим парадигмам программирования. Как правило, целью сочетания нескольких парадигм в одном языке являются удобство и упрощение решения требуемого множества задач с использованием данных языков. В настоящей статье представлен обзор существующих парадигм и подходов к программированию, рассматривается возможный подход к их классификации и построению иерархии.