УДК 004.5

Динамическое управление бизнес-процессами на основе совмещенных сетей управления и данных

А.В. Яцутко [1], В.С. Выхованец [1, 2]

[1] Институт проблем управления РАН, Москва, 117997, Россия

[2] МГТУ им. Н.Э. Баумана, Москва, 105005, Россия.

Рассмотрен новый подход в области динамического управления бизнес-процессами, основанный на использовании графической нотации с совмещенными сетями управления и данных. Благодаря явному заданию потока данных появляется возможность автоматического преодоления семантического разрыва, возникающего между графической и исполняемой моделью бизнес-процессов. Последнее приводит к повышению качества управления производственными и технологическими процессами на предприятиях.

E-mail: valery@vykhovanets.ru

Ключевые слова: *моделирование процессов, динамическое управление процессами, совмещенные сети управления и данных, семантический разрыв, методы исполнения и актуализации процессных моделей, динамическое распараллеливание.*

Процессно-ориентированный подход, заключающийся в представлении объекта управления как многоуровневого набора взаимосвязанных процессов, широко используется в современных системах производственного и организационного управления. Бизнес-процесс, или просто процесс, рассматривается как совокупность взаимосвязанных или взаимодействующих видов деятельности, преобразующих некоторые входы в выходы и направленные на достижение максимальной эффективности производственно-хозяйственной и финансово-экономической деятельности предприятия. Входами одного процесса обычно являются выходы других процессов.

В известных системах моделирования структура процесса задается диаграммой, представляющей собой сеть, которая состоит из процессов (узлов) и связей управления (дуг), задающих частичный порядок выполнения подчиненных процессов [1]. Данные между процессами, как правило, передаются неявно — через атрибуты процессов, через специальные объекты внутри процесса или через общую для всех процессов шину данных. Такая организация работы с данными исторически наследуется с самых первых методологий анализа. Это обусловлено тем, что выделение потока данных сложно реализовать и методически, и технически [2]. Отсутствие явно задаваемого потока данных приводит к невозможности автоматического создания исполняемых моделей процессов, а также затрудняет актуализацию уже выполняющихся процессов.

Целью работы является улучшение качества динамического управления процессами предприятий путем использования такой их модели, которая основана на явной трассировке управления и данных между узлами сети. Это позволяет решить проблему автоматической генерации управляющей программы по ее описанию в виде иерархически организованных диаграмм процессов.

Проблемы моделирования процессов. Современное моделирование бизнес-процессов возникло на основе методологии структурного анализа и проектирования SADT (Structured Analysis and Design Technique) [2] и стандарте IDEF0 (Integrated computer-aided manufacturing DEFinition) [3]. Эти методологии предназначены для высокоуровневого описания процессов в функциональном аспекте; они состоят из диаграмм, текстов и глоссария, имеющих ссылки друг на друга. Диаграммы — главные компоненты модели, все функции и интерфейсы в них представлены как узлы (функциональные блоки) и дуги (рис. 1). Каждый узел может быть декомпозирован на другой диаграмме. Следует отметить, что нотации в методологии SADT и IDEF0 предназначены для описания только состава процессов, а не их последовательности. Имеются также некоторая сложность восприятия диаграмм и проблема согласования нескольких взаимодействующих процессов.

Рис. 1. Функциональный блок IDEF0

Для частичного устранения недостатков IDEF0 используется описание потоков данных в нотации DFD (Data Flow Diagram) [4]. Цель такого представления — продемонстрировать, как каждый процесс преобразует свои входные данные в выходные. Основными компонентами диаграмм потоков данных являются внешние и внутренние сущности, процессы, накопители данных и потоки данных (рис. 2). Каждый процесс может быть детализирован с помощью другой диаграммы или спецификации. Языки спецификации, как правило, не стандартизованы и могут варьироваться от естественного языка до визуальных языков моделирования.

Дальнейшее развитие методологии IDEF, а именно стандартов IDEF1 (методология моделирования информационных потоков внутри системы) и IDEF1X (методология построения реляционных структур данных), преследовало цель устранить недостатки, присущие моделям потока данных.

Рис. 2. Диаграмма DFD

Следует отметить, что модели в ранее рассмотренных нотациях являются статическими по определению и не позволяют отразить функционирование моделируемых объектов во времени. Проблему создания динамических диаграмм пытались решить с помощью IDEF2 (Simulation Model Design) — методологии динамического моделирования развития. Однако ввиду сложности использования этой методологии разработка соответствующего стандарта быстро прекратилась [5].

Следующим шагом в моделировании процессов стала разработка и использование стандарта IDEF3 [6], предназначенного для описания потоков работ. Модели IDEF3 применяли для детализации функциональных блоков IDEF0, не имеющих диаграмм декомпозиции. Выразительные средства стандарта IDEF3 оказались близки алгоритмическим блок-схемам (рис. 3).

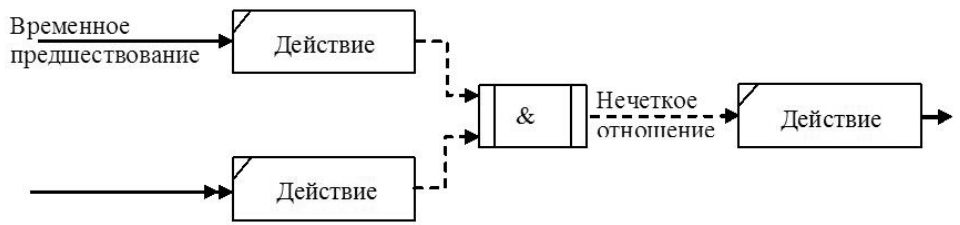

Рис. 3. Диаграмма IDEF3

Неоднозначная семантика и низкая выразительность диаграмм в стандартах IDEF и DFD привела к объединению известных методов моделирования в форме создания интегрированных методологий, одной из которых является ARIS (Architecture of Integrated Information Systems) [1]. Методология ARIS поддерживает четыре типа моделей с множеством (около 80) их видов в каждом типе, отражающих различные аспекты исследуемых бизнес-процессов. Для каждого вида модели используются три уровня представления: описание требова-

ний, спецификация проекта и описание реализации. Для построения моделей применяют как собственные языки графического моделирования ARIS, так и другие языки, в частности UML (Unified Modeling Language) [7].

Модель процесса в ARIS является в некотором смысле расширением IDEF3: процесс в ARIS выражается в виде диаграмм и представляет собой поток последовательно выполняемых работ, расположенных в порядке их выполнения (рис. 4).

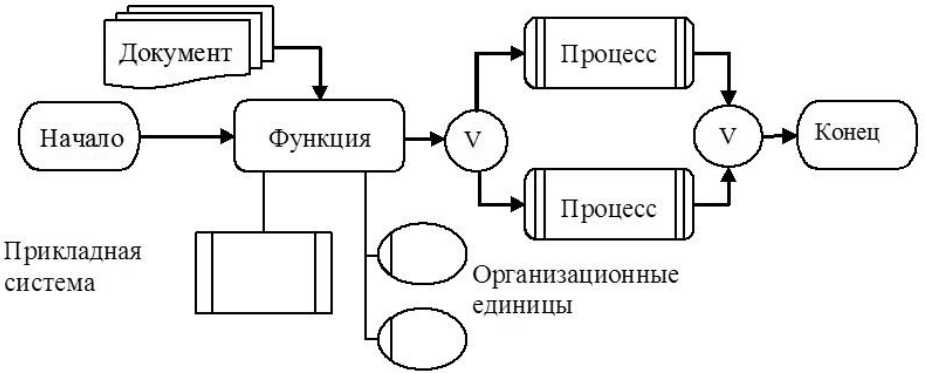

Рис. 4. Диаграмма процесса ARIS

Ввиду многообразия выразительных средств в ARIS внедрен механизм методологических фильтров, позволяющих в рамках одного процесса применять только определенный набор объектов и связей. Для разработки таких фильтров требуются значительное количество времени и разработчик высокой квалификации. В конечном итоге методология ARIS дает возможность описать бизнес-процессы предприятия, провести анализ полученных моделей и определить требования к реализации информационной системы управления бизнес-процессами. Однако эта методология не обеспечивает корректного и однозначного описания моделируемых процессов.

Необходимость не только описать процессы на высоком уровне, но и полностью или частично их автоматизировать привела к появлению концепции потока работ Workflow [8], согласно которой данные и задания передаются от одного исполнителя другому для выполнения определенных действий в соответствии со сводом процедурных правил (рис. 5).

Следует обратить внимание на то, что в Workflow данные не перемещаются вместе с управлением, а содержатся в глобально доступных переменных и локальных переменных блоков. Последнее вызывает значительные трудности синхронизации данных.

Негибкость создаваемых моделей, их неспособность обеспечить оперативное реагирование на постоянные изменения в среде являются основными недостатками методологий SADT, IDEF, DFD,

ARIS и др., которые стимулировали разработку концепции следующего поколения — BPM (Business Process Management) [9]: на смену радикальному реинжинирингу приходит динамическое управление, заключающееся в возможности корректировки уже автоматизированных процессов в ответ на изменения в структуре и организации предприятия. В этом случае инструменты моделирования позволяют создавать и внедрять новые процессы «на лету», а само моделирование больше похоже на моделирование в Workflow.

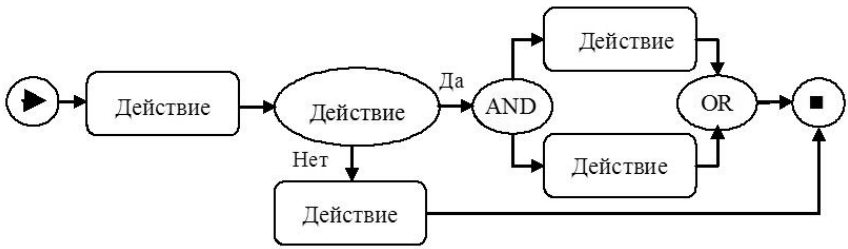

Рис. 5. Диаграмма Workflow

Автоматизация процессов проводится с помощью инструментария управления процессами BPMT (Business Process Management Tool), реализующего процессное управление, который состоит из трех средств: моделирования, исполнения, мониторинга. Исполнение начинается с того, что схема процесса загружается в среду исполнения, где и происходит запуск процесса. При этом каждый исполнитель, задействованный в процессе, получает запланированное для него задание, которое необходимо выполнить, и выводит отчет о результатах исполнения. Мониторинг подразумевает возможность в реальном времени отслеживать прохождение процесса по этапам и исполнителям, а также позволяет формировать отчетность и оценивать показатели процесса. Информационные потоки в BPMT, как и в Workflow, представляют собой структурированные данные, содержащиеся в различных хранилищах и репозиториях.

Для разработки исполняемых моделей в BPM используют нотацию BPMN (Business Process Model and Notation) (рис. 6) и соответствующие языки моделирования: язык моделирования процессов BPML (Business Process Modeling Language), язык исполнения процессов BPEL (Business Process Execution Language) и язык определения процессов XPDL (XML Process Definition Language). Однако построение моделей непосредственно на этих языках неудобно для разработчиков по причине их синтаксической и семантической сложности. В этой связи большое внимание разработчики программных средств BPM уделяют конвертированию графических моделей процессов в исполняемые формы. Однако до настоящего времени применение BPEL в основном заключается в автоматизированном создании шаблона для «ручной доводки» модели до исполняемого вида. В итоге оказалось,

что для массового распространения методологии BPM пока не хватает технологических возможностей, позволяющих создавать и автоматически исполнять модели автоматизируемых процессов.

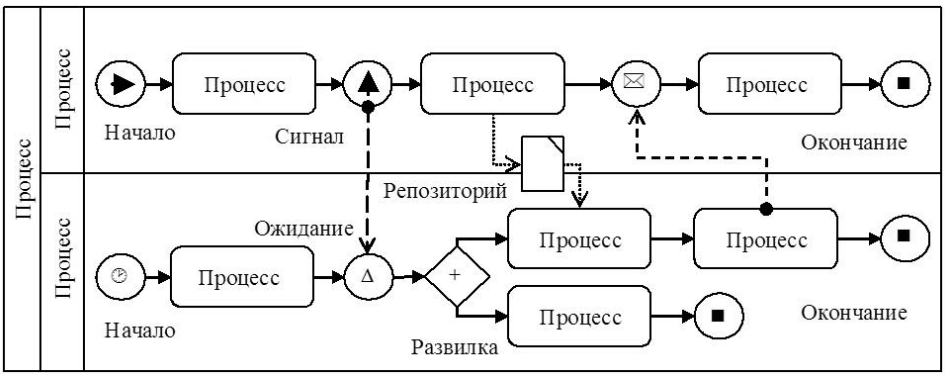

Рис. 6. Диаграмма BPMN

Таким образом, основной проблемой современных средств моделирования процессов является наличие труднопреодолимого семантического разрыва между графической (аналитической) и исполняемой моделями процесса. Здесь под семантическим разрывом понимается различие между принципами (подходами), используемыми для аналитического описания процессов, и принципами (подходами), необходимыми для реализации исполняемых моделей этих процессов. Семантический разрыв проявляется в том, что понятия, объекты и структуры данных, которыми оперирует аналитик, не совпадают, а порой даже не согласуются с понятиями, объектами и структурами данных, которые должен использовать разработчик исполняемой модели. Для преодоления указанного семантического разрыва требуется обеспечить автоматическое преобразование аналитической модели процесса в его исполняемую форму.

Совмещенная сеть управления и данных. Появление проблем, свойственных современным средствам автоматизированного управления технологическими и производственными процессами, следует связать с недостаточностью данных, отражаемых на аналитических диаграммах процессов, их нечеткой пространственно-временной структурированностью и несогласованностью типов. Для преодоления этих проблем предлагается использовать аналитические модели процессов, содержащие не только связи процессов по управлению, но и источники данных и пути их передачи [10]. Иными словами, предлагается объединить выразительные возможности двух моделей — моделей для описания потоков работ и моделей для описания потоков данных.

Графическая нотация, основанная на совмещенных сетях управления и данных, позволяет явно указать на одной диаграмме как потоки управления, так и потоки данных (рис. 7).

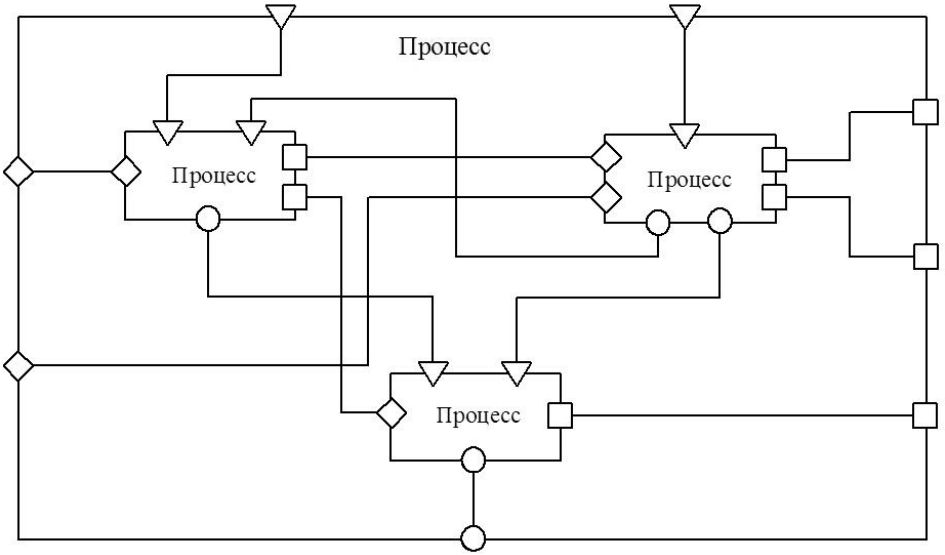

Рис. 7. Совмещенная диаграмма управления и данных

Через каждый узел сети (показан в виде *прямоугольника*) проходят поток управления и поток данных. Поток управления, входящий в узел, называется активатором (*ромб*). Получение активации означает начало работы узла. Выход потока управления из узла называется событием (*квадрат*). Событие, созданное узлом, может активировать один или несколько других узлов. Поток данных передает узлу структурированные входные данные (*треугольник*). После того как узел закончил работу, в поток данных поступают структурированные выходные данные (*окружность*), которые могут передаваться следующему узлу в качестве входных. При срабатывании одного или нескольких активаторов узел активируется, считываются входные данные и выполняются действия, привязанные к этому узлу. Затем формируются одно или несколько событий и выходные данные.

Часть узлов, которые не имеют диаграммы, называются базовыми. Составной узел отличается от базового узла тем, что не является последним в иерархии, т. е. декомпозируется в совмещенную сеть и состоит из других узлов, каждый из которых, в свою очередь, может быть либо базовым, либо составным.

К каждому базовому узлу привязывается вычислительная семантика в виде последовательности действий (операций) на языке некоторого исполнителя. В качестве исполнителя может выступать некоторый внешний интерпретатор — система программирования, виртуальная машина, внешняя библиотека, какие-либо исполнительные устройства. В свою очередь, составной узел строится из других узлов, а его вычислительная семантика однозначно определяется внутренними узлами и соединяющими их дугами [11].

При активации одного или нескольких активаторов составного узла в соответствии с заданными связями активизируются внутренние узлы. После окончания функционирования внутренних узлов их события по внутренним связям активизируют другие внутренние узлы. Функционирование составного узла завершается, когда все внутренние узлы перестают активироваться и заканчивают свое функционирование. Аналогично связям активации происходит передача данных между внутренними узлами и формирование выходных данных составного узла.

Дифференциация входов и интеграция выходов. Особенностью интерпретации совмещенных моделей является дифференциация активаторов и входных данных, а также интеграция событий и выходных данных [12]. Процесс дифференциации активаторов составного узла заключается в фиксации фактов активации таким образом, что воздействие этих активаторов на связанные внутренние узлы осуществляется однократно и только в момент активации. Процесс интеграции событий составного узла состоит в накоплении событий, поступающих от внутренних узлов за все время их функционирования. Если событие составного узла было возбуждено хотя бы один раз, это является основанием для принятия решения о передаче их числа внешнему узлу, включающему текущий узел в качестве составного. При этом число возбуждений события интерпретируется как сила активации узла или узлов, активаторы которых соединены с этим событием.

Аналогичным образом процесс дифференциации входных данных заключается в их фиксации в момент активации узла. В свою очередь, интеграция выходных данных осуществляется объединением данных на выходе данных. Интегрирование выходных данных осуществляется путем поименного объединения частей данных, получаемых после окончания функционирования каждого внутреннего узла, соединенного с выходом данных. Для такого объединения данные представляются именованными списками значений вида (Имя, Значение 1, Значение 2, …), где любым из значений может быть другой именованный список. Такую структуру данных удобно представлять деревом, у которого корнем является выходной порт, а листья — значения данных или другие деревья (рис. 8). Например, для данных на рис. 8 именованный список представляется так:

(Выход 1, (Имя 1, …), (Имя 2, Значение 1, (Имя 3, …)), Значение 2, Значение 3).

Следует обратить внимание на то, что рассматриваемая древовидная структура может использоваться для спецификации интерфейсов входных и выходных данных узлов. В этом случае при конструировании узла задаются структуры данных, минимально необходимые (непосредственно используемые) при его функционировании. Если во время активации такого узла на его вход поступят данные, не содержащие их минимальный объем, может быть зафиксирована ошибка

передачи данных, что сигнализирует о неправильной работе узла — поставщика этих данных. В свою очередь, во время конструирования узлов могут быть выданы предупреждающие сообщения о том, что выполненная трассировка данных не является корректной, так как выходной порт узла — поставщика данных не содержит данные, необходимые для правильного функционирования узла — их получателя.

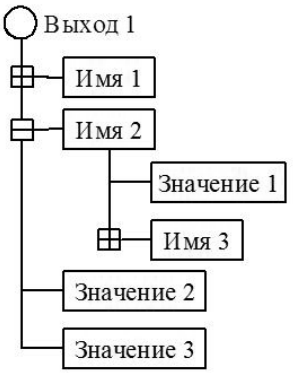

Рис. 8. Древовидная структура данных

Исполнение совмещенных моделей. Базовые узлы исполняются непосредственно тем исполнителем, который определен во время их конструирования, так как их вычислительная семантика задается явно. Исполнение составного узла означает исполнение всех его внутренних узлов. После дифференциации активаторов и входных данных исполняемого составного узла выполняется активация его внутренних узлов и пересылка входных данных в соответствии с внутренней трассировкой. Те узлы, которые активированы, передаются на исполнение. После окончания функционирования каждого из активированных узлов процесс исполнения повторяется до тех пор, пока имеются узлы, активированные на предыдущем шаге. При этом всякий раз, когда внутренний узел окончит функционирование, выполняется интеграция событий и выходных данных.

Для составных узлов существует три подхода к реализации их исполнения (рис. 9) — рекурсивный интерпретатор, циклический интерпретатор и компиляция [13].

При рекурсивной интерпретации составного узла после определения активированных внутренних узлов происходят непосредственное исполнение активированных базовых узлов и рекурсивный вызов того же интерпретатора для каждого активированного составного узла. Рекурсивную интерпретацию следует применять для быстро исполняемых составных узлов с небольшим уровнем вложенности, так как при рекурсивной обработке на все время исполнения составного узла резервируется полный объем вычислительных ресурсов, необходимый для исполнения всех вложенных узлов.

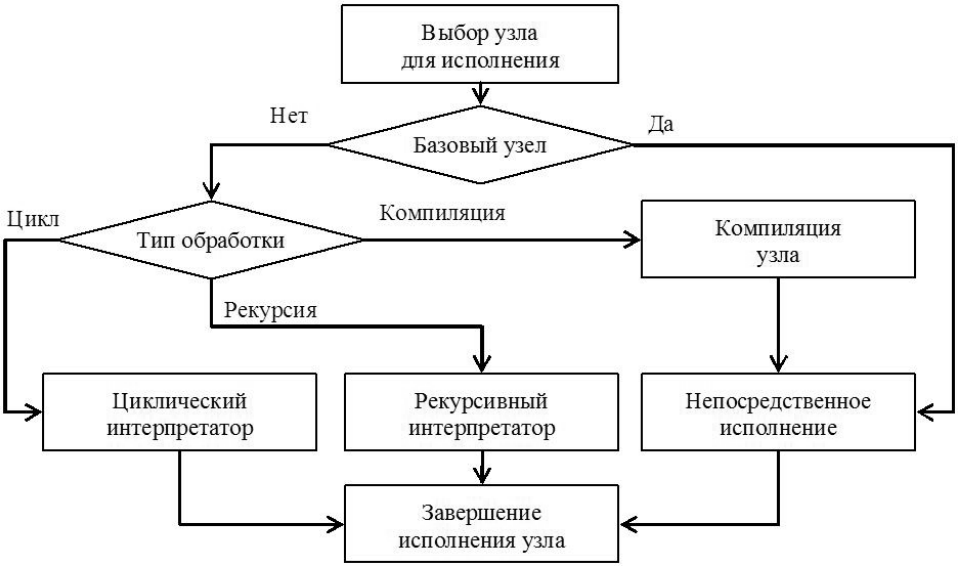

Рис. 9. Исполнение узла совмещенной модели

При циклической интерпретации составного узла описания активированных внутренних узлов записываются в специальную таблицу. Затем один или несколько циклических интерпретаторов выбирают из этой таблицы самые приоритетные описания. Приоритет отдается тем узлам, которые являются базовыми, имеют большую глубину вложенности или все составные узлы которых оказались исполненными. Циклическую интерпретацию следует применять при исполнении составных узлов с большим временем функционирования, вызванным, как правило, ожиданием реакции на некоторые внешние события.

Третий способ реализации исполнения составных узлов — их компиляция, основанная на том, что вычислительная семантика базовых и составных узлов полностью определяется их описаниями. Это позволяет автоматически породить программу на некотором языке программирования, которая после компиляции реализует исполнение откомпилированных узлов. Компиляцию следует применять для простых и часто используемых составных узлов, так как для их актуализации требуются большие вычислительные ресурсы.

Актуализация и реанимация узлов. Актуализация процесса — это внесение изменений в его описание, вызванное изменениями во внешней среде. Характерной чертой рассматриваемого подхода является возможность актуализации созданных и запущенных для исполнения процессов в режиме реального времени.

Действительно, при активации узла во временную область памяти копируется его полное описание, а исполнение узла происходит в результате передачи этого описания некоторому интерпретатору (см. рис. 9). Такая организация процесса исполнения позволяет после

запуска узла изменять его исходное описание. В этом случае активация измененного узла приведет к копированию и исполнению нового исходного описания.

При существенной перестройке интерфейса узла, когда изменяется число активаторов и событий, а также число входов и выходов данных или структуры данных, описанный выше способ актуализации может оказаться неприменимым. Тогда для актуализация узла требуется перепроектирование и тех составных узлов, которые используют его в качестве внутреннего. Понятно, что такое перепроектирование эквивалентно реинжинирингу моделируемого процесса и должно осуществляться путем создания его нового описания.

Однако на стадии исполнения может оказаться некоторое число копий актуализированного узла, поэтому желательно привести запущенный узел в актуализированное состояние в процессе его реанимации. Реанимация узла основана на знании текущего его состояния и состояний всех внутренних узлов, включая значения входных и выходных данных, ее осуществляют путем редактирования копии описания узла и его данных в соответствии с новым понимаем моделируемого процесса.

Таким образом, предлагаемый подход позволяет выполнять актуализацию моделей процессов в соответствии с изменяющимися или уточняющимися данными о моделируемых процессах.

Динамическое распараллеливание. Нотации многих известных методологий моделирования процессов включают в себя специальные элементы для явной (статической) организации параллельного их выполнения. Для этого, например, используются такие средства, как «логическое И», многоэкземплярные циклы параллельного типа, развернутые процессы с параллельными блоками, шлюз «ИЛИ» с неэксклюзивным условием и т. п. [1, 9]. Однако у явного управления распараллеливанием процессов имеются существенные недостатки. Во-первых, качественно снижается наглядность модели, которая особенно важна на этапах ее анализа и верификации. Во-вторых, для параллельного выполнения теряется большая часть запускаемых процессов, распараллеливание которых по многим причинам не может быть в явном виде указано на диаграммах. В-третьих, разработка диаграммы с параллельным выполнением процессов является, как правило, сложной и непроизводительной.

Моделирование процессов с помощью совмещенных сетей управления и данных позволяет реализовать метод динамического (неявного) управления распараллеливанием, основанный на использовании естественного параллелизма процессов [12]. В связи с тем, что составной узел содержит явное указание на принимаемые и передаваемые данные между внутренними узлами, при его исполнении распараллеливанию подлежит исполнение всех активированных узлов. Следует обратить внимание на то, что именно здесь проявляется динамический

характер распараллеливания, заключающийся в том, что в каждый конкретный момент времени и в зависимости от предыстории развития моделируемого процесса принимается решение о том, какие узлы подлежат параллельному исполнению, а какие нет [14]. Может оказаться так, что при одинаковом описании узла в один момент времени параллельному исполнению подлежит одно подмножество внутренних узлов, а в другой момент времени — другое.

Наиболее эффективно динамическое распараллеливание проявляется при циклической интерпретации узлов, когда выбор узла для исполнения происходит среди множества активированных узлов, в том числе и принадлежащих различным моделируемым процессам. В этом случае реализуется динамическая балансировка загрузки интерпретаторов (процессоров), проявляющаяся в том, что в зависимости от траекторий развития моделируемых процессов более нуждающийся в обслуживании процесс получает больше вычислительных ресурсов, а менее нуждающийся — меньше.

Таким образом, совместная и раздельная трассировка данных и управления между узлами сети предоставляет гибкий механизм синхронизации моделируемых процессов, позволяет реализовать естественное их распараллеливание и эффективное выполнение.

В заключение отметим, что применение методологии анализа и моделирования процессов на основе совмещенных сетей управления и данных позволяет преодолеть накопившиеся проблемы в области промышленных систем управления бизнес-процессами. В частности, на основе графической модели процесса появляется возможность породить его исполняемый код без участия разработчика. При этом предлагаемая графическая нотация содержит небольшое число исходных элементов и простые правила их интерпретации, является такой же понятной аналитикам, как и другие зарекомендовавшие себя нотации. Немаловажной является возможность актуализации запущенных моделей процессов при их естественном изменении в бизнес-среде, благодаря чему удается существенно улучшить качество управления современным динамически изменяющимся предприятием.

Использование структурированных входных и выходных данных процессов создает новое выразительное средство, облегчающее верификацию создаваемых моделей и отсутствующее у современных средств моделирования, а именно спецификацию интерфейсов входов и выходов данных, которая позволяет обнаружить и устранить ошибки трассировки данных на этапе моделирования.

При автоматическом динамическом распараллеливании процессов добиваются высокой эффективности исполнения получаемых моделей. Реализуемая параллельность при этом близка к естественному параллелизму моделируемых процессов. Исполнение процессов с использованием динамического распараллеливания отличается возможностью масштабирования на различное число параллельно работающих интерпретаторов (процессоров, параллельных потоков).

Более того, интерпретаторы автоматически подстраиваются под характеристики запущенных на исполнение процессов, реализуя динамическую балансировку своей загрузки.

СПИСОК ЛИТЕРАТУРЫ

1. Davis R. Business process modeling with ARIS: A practical guide. — N.Y.: Springer-Verlag, 2005. 531 p.
2. Марка Д. Методология структурного анализа и проектирования SADT. М.: Метатехнология, 1993. 247 с.
3. Integration definition for function modeling (IDEF0). Washington: National Institute of Standards and Technology, 1993. 116 p.
4. Gane C., Sarson, T. Structured systems analysis: Tools and techniques. N.Y.: Prentice-Hall, 1979. 452 p.
5. Черемных С.В. Структурный анализ систем: IDEF-технологии. М.: Финансы и статистика, 2001. 208 с.
6. Information integration for concurrent engineering (IICE) / R.J. Mayer, C.P. Menzel, M.K. Painter, et al. // IDEF3 Process Description Capture Method Report. Ohio: Air Force Material Command, 1995. 235 p.
7. Буч Г., Рамбо Д., Джекобсон А. Язык UML: Руководство пользователя. М.; СПб.: ДМК «Пресс Питер», 2004. 432 с.
8. Fischer L. Workflow handbook 2002. Association with the Workflow Management Coalition (WfMC), 2002. 428 p.
9. Smith H., Fingar P. Business process management (BPM): The Third Wave. Meghan-Kiffer Press, 2002. 312 p.
10. Яцутко А.В. Моделирование бизнес-процессов крупномасштабного производства // Тр. III Междунар. конф. «Управление развитием крупномасштабных систем». М.: ИПУ РАН, 2009. Т. II. С. 301–303.
11. Яцутко А.В. Управление проектами на основе графического моделирования с описанием вычислительной семантики // Тр. Междунар. науч.-практ. конф. «Теория активных систем». М.: ИПУ РАН, 2009. Т. II. С. 61–64.
12. Яцутко А.В. Актуализация моделей бизнес-процессов в совмещенных сетях управления и данных // Сб. тр. Второй Российской конф. с междунар. участием «Технические и программные средства систем управления, контроля и измерения». М.: ИПУ РАН, 2010. С. 1268–1272.
13. Яцутко А.В. Параллельные вычисления в методах теории управления на основе совмещенных сетей данных и управления // Докл. 5-й Междунар. конф. «Параллельные вычисления и задачи управления». М.: ИПУ РАН, 2010. С. 519–528.
14. Яцутко А.В. Синхронизация данных в автоматных сетях с раздельной трассировкой управления и данных // Сб. тр. Третьей Российской конф. с междунар. участием «Технические и программные средства систем управления, контроля и измерения». М.: ИПУ РАН, 2012. С. 2060–2070.

Статья поступила в редакцию 25.10.2012