Труды Конференции «Информационные технологии в управлении» (ИТУ'2012). - СПб: ЦНИИ "Электроприбор, 2012. - С. 98-107.

ДИНАМИЧЕСКОЕ УПРАВЛЕНИЕ БИЗНЕС-ПРОЦЕССАМИ НА ОСНОВЕ СОВМЕЩЕННЫХ СЕТЕЙ УПРАВЛЕНИЯ И ДАННЫХ

А.В. Яцутко, В.С. Выхованец

Институт проблем управления РАН, Москва
117997, ул. Профсоюзная, 65. +74959267784. 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.

Рассматривается новый подход в области динамического управления бизнес-процессами, основанный на использовании графической нотации с совмещёнными 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.

Введение

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 [1]. Входами процесса обычно являются выходы других процессов.

В известных системах моделирования структура процесса задаётся диаграммой, представляющей собой сеть, состоящую из процессов (узлов) и связей управления (дуг), задающих частичный порядок выполнения подчинённых процессов [2]. 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 [3]. Отсутствие явно задаваемого потока данных приводит к невозможности автоматического создания исполняемых моделей процессов, а также затрудняют актуализацию уже выполняющихся процессов.

Целью работы является улучшение качества динамического управления процессами предприятий путём использования такой их модели, которая основана на явной трассировке данных и управления между узлами сети. Это позволяет решить проблему автоматической генерации управляющей программы по её 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) [3] и стандарте IDEF0 (Integrated computer-aided manufacturing DEFinition) [4]. Эти методологии предназначены для высокоуровневого описания процессов в функциональном аспекте и состоят из диаграмм, текстов и глоссария, имеющих ссылки друг на друга. Диаграммы – главные компоненты модели, все функции и интерфейсы на них представлены как узлы (функциональные блоки) и дуги (рис. 1). Каждый узел может быть декомпозирован на другой диаграмме. Следует заметить, что нотации в методологии SADT и IDEF0 предназначены для описания только состава процессов, а не их последовательности, имеется также некоторая

сложность восприятия диаграмм и проблема согласования нескольких взаимодействующих процессов.

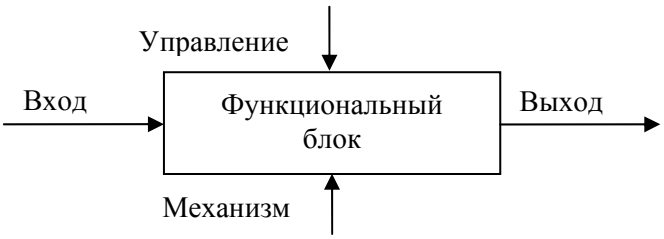

Рис. 1. Функциональный блок IDEF0

Для частичного устранения недостатков IDEF0 используется описание потоков данных в нотации DFD (Data Flow Diagram) [5]. 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 при помощи другой диаграммы или спецификации. Языки спецификации, как правило, не стандартизованы и могут варьироваться от естественного языка до визуальных языков моделирования.

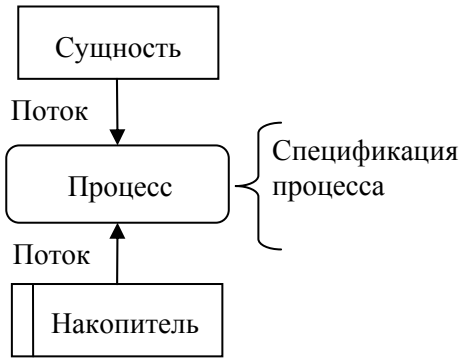

Рис. 2. Диаграмма DFD

Дальнейшее развитие методологии IDEF, а именно стандартов IDEF1 (методология моделирования информационных потоков внутри системы) и IDEF1X (методология построения реляционных структур данных) преследовало цель устранения недостатков, присущих моделям поток данных.

Следует заметить, 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. Но ввиду сложности использования этой методологии разработка соответствующего стандарта быстро прекратилась [6].

Следующим шагом в моделировании процессов стала разработка и использование стандарта IDEF3 [7], предназначенного для описания потоков работ. Модели IDEF3 использовались для детализации функциональных блоков IDEF0, не имеющих диаграмм декомпозиции. Выразительные средства стандарта IDEF3 оказались близки алгоритмическим блок-схемам (рис. 3).

Неоднозначная семантика и низкая выразительность диаграмм в стандартах IDEF и DFD привела к объединению известных методов моделирования в форме создания интегрированных методологий. Одной из таких методологий является ARIS (Architecture of Integrated Information Systems) [2]. ARIS поддерживает четыре типа моделей с множеством (около 80) их видов в каждом типе, отражающих различные аспекты исследуемых бизнес-процессов. Для каждого вида модели используется три уровня представления: описание требований, спецификация

проекта и описание реализации. Для построения моделей используются как собственные языки графического моделирования ARIS, так и другие языки, в частности, UML (Unified Modeling Language) [8].

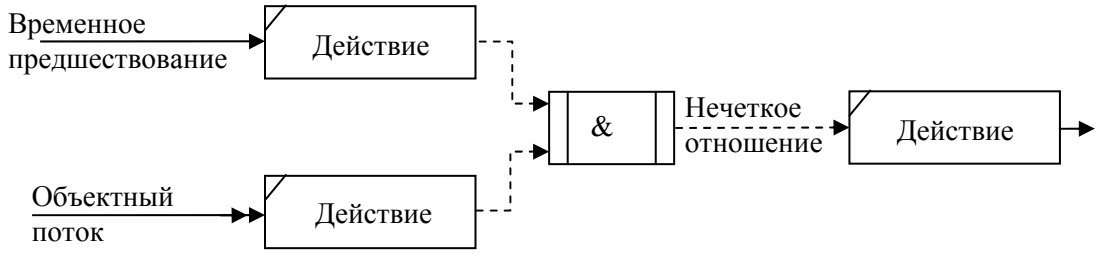

Рис. 3. Диаграмма IDEF3

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).

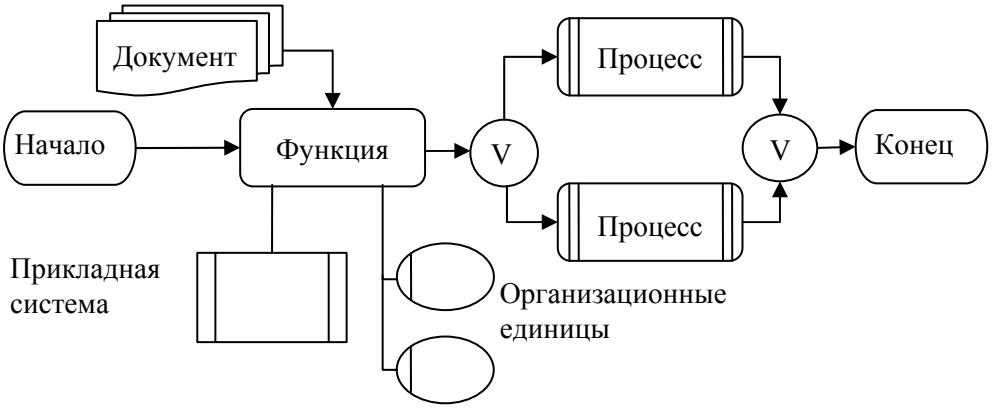

Рис. 4. Диаграмма процесса ARIS

Из-за необозримости выразительных средств в ARIS внедрён механизм методологических фильтров, позволяющих в рамках одного процесса применять только определённый набор объектов и связей. Разработка таких фильтров требует значительного времени и высокой квалификации разработчика. В конечном итоге методология ARIS позволяет описать бизнес-процессы предприятия, провести анализ полученных моделей и специфицировать требования к реализации информационной системы управления бизнес-процессами, однако эта методология порождает проблемы корректного и однозначного описания моделируемых процессов.

Необходимость не только высокоуровневого описания процессов, но и необходимость в полной или частичной их автоматизации, привела к появлению концепции потока работ Workflow [9], при которой данные и задания передаются от одного исполнителя другому для выполнения определённых действий согласно своду 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 стали основными недостатками методологий SADT, IDEF, DFD, ARIS и др., которые стимулировали разработку концепции следующего поколения – BPM (Business Process Management) [10], где 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. 6).

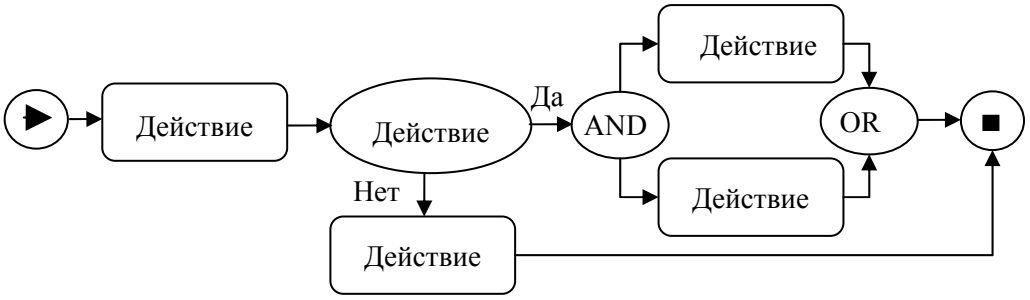

Рис. 5. Диаграмма Workflow

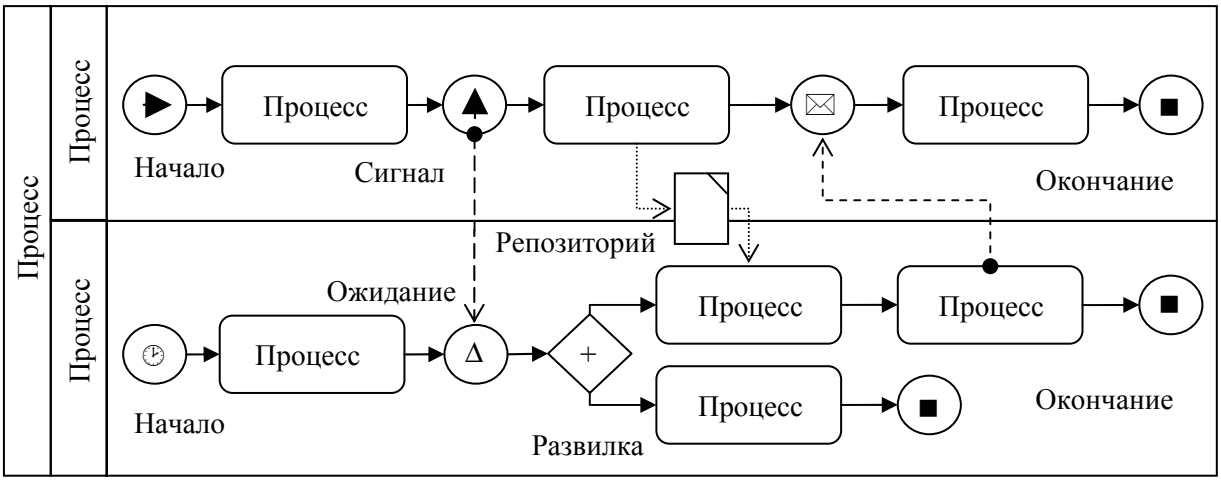

Рис. 6. Диаграмма BPMN

Автоматизация процессов производится с помощью инструментария управления процессами BPMT (Business Process Management Tool), который реализует процессное управление и состоит из трёх элементов: средство моделирования, средство исполнения, средство мониторинга. Исполнение начинается со схемы процесса, которая 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 отчёт о результатах исполнения. Мониторинг подразумевает возможность в реальном времени отслеживать прохождение процесса по этапам и исполнителям, а также позволяет формировать отчётность и оценивать результативность и показатели процесса. Информационные потоки в BPMT, как и в Workflow, по большей части представляют собой структурированные данные, содержащиеся в различных хранилищах и репозиториях.

Для разработки исполняемых моделей в BPM используют нотацию BPMN (Business Process Model and Notation) 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 к исполняемому виду. В итоге оказалось, что для массового

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

Таким образом, основной проблемой современных средств моделирования процессов является наличие труднопреодолимого семантического разрыва между графической (аналитической) и исполняемой моделью 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.

Совмещённая 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 нечёткой пространственно-временной структурированностью и несогласованностью типов. Для преодоления этих проблем предлагается использовать аналитические модели процессов, содержащие не только точно специфицируемые связи процессов по управлению, но и точно специфицируемые источники данных и пути их передачи [11]. Иными словами предлагается объединить выразительные возможности двух классов моделей – моделей для описания потоков работ и моделей для описания потоков данных.

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

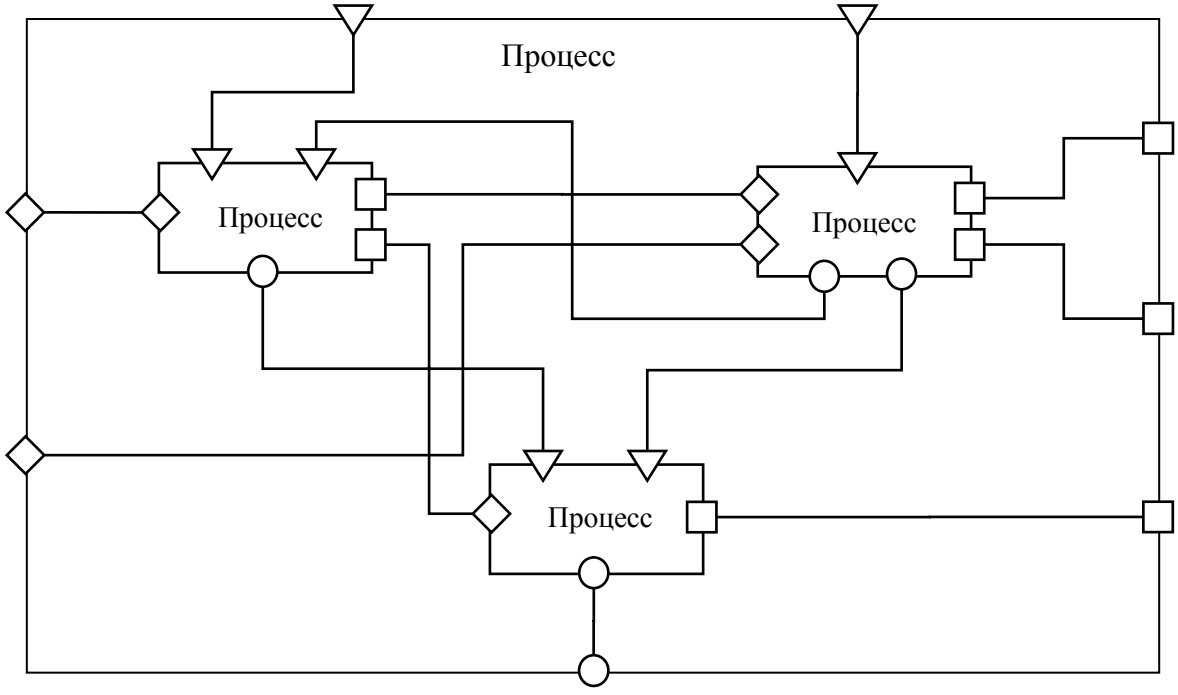

Рис. 7. Совмещённая диаграмма управления и данных

Через каждый узел сети проходит 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 передаёт 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 данному 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, то есть, декомпозируется и состоит из других блоков, 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 некоторая система программирования, виртуальная машина, внешняя библиотека, какие-либо исполнительные устройства, то есть некоторый внешний интерпретатор. В свою очередь составной узел строится из других узлов, и его вычислительная семантика однозначно определяется внутренними узлами и соединяющих их дугами [12].

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

Дифференциация входов и интеграция выходов

Особенностью исполнения совмещённых моделей является дифференциация активаторов и входных данных, и интеграция событий и выходных данных [13]. Процесс дифференциации активаторов составного узла состоит в фиксации фактов активации таким образом, что воздействие таких активаторов на связанные с ним внутренние узлы осуществляется однократно и только в момент активации. Процесс интеграции событий составного узла состоит в накоплении фактов возбуждения этих события за все время функционирования этого составного узла. Если событие составного узла было возбуждено хотя бы один раз, то это является основанием для принятия решения о передаче числа таких возбуждений внешнему узлу, включающего текущий узел. При этом число возбуждений события интерпретируется как сила активация узла или узлов, соединённых с этим событием.

Аналогичным образом процесс дифференциации входных данных заключается в их фиксации в момент активации узла. В свою очередь, интеграция выходных данных осуществляется их объединением на выходе данных. Интегрирование выходных данных осуществляется путём поимённого объединения данных, получаемых после окончания функционирования внутреннего узла, соединённого с выходом данных. Для такого объединения данные представляются именованными списками значений вида (Имя, Значение1, Значение2, …), где любым из значений может быть другой именованный список. Такую структуру данных удобно представлять деревом, у которого корнем является выходной порт, а листьями – значения данных или другие деревья (рис. 8). Например, для данных на рис. 8 именованный список представляется следующим образом:

(Выход1, (Имя1, …), (Имя2, Значение1, (Имя3, …)), Значение2, Значение3).

Следует обратить внимание на то, что рассматриваемая древовидная структура может также использоваться для спецификации интерфейсов входных и выходных данных. В этом случае при конструировании узла задаются структуры данных, минимально необходимые (непосредственно используемые) при его функционировании. Если во время исполнения такого узла на его вход поступят данные, не содержащие их минимальный объем, может фиксироваться ошибка передачи данных, что сигнализирует о неправильной работе узла – поставщика этих данных. Также 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 данных, необходимых для правильного функционирования узла – их получателя.

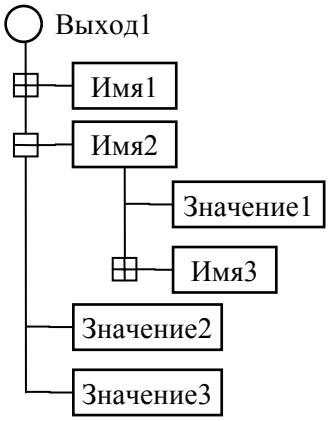

Рис. 8. Древовидная структура данных

Исполнение совмещённых моделей

Базовые узлы исполняются непосредственно тем исполнителем, который определён во время их конструирования, так как их вычислительная семантика задаётся явно. Исполнение составного узла – это исполнение всех его внутренних узлов. После дифференциации активаторов и входных данных исполняемого составного узла выполняется активация его внутренних узлов и пересылка их входных данных в соответствии с его трассировкой управления и данных. Те узлы, которые оказались активированы, передаются на исполнение. После окончания функционирования каждого из активированных узлов процесс исполнения повторяется до тех пор, пока имеются узлы, активированные в процессе предыдущего шага. При этом всякий раз, когда внутренний узел закончит функционирование, выполняется интеграция событий и выходных данных исполняемого составного узла.

Для составных узлов имеется три подхода к реализации их исполнения (рис. 9) – рекурсивный интерпретатор, циклический интерпретатор и компиляция [14].

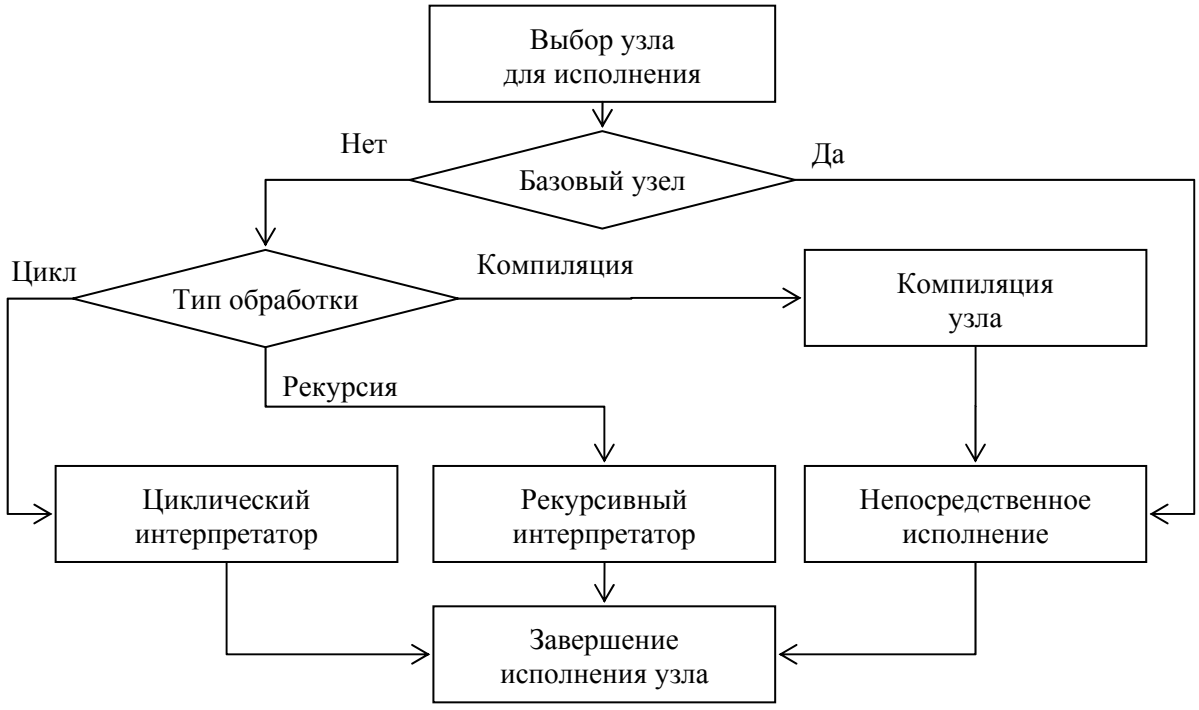

Рис. 9. Исполнение узла совмещённой модели

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

При циклической интерпретации составного узла описания активированных внутренних узлов записываются в специальную таблицу. Затем один или несколько циклических интерпретаторов выбирают из этой таблицы для исполнения самые приоритетные узлы. Приоритет отдаётся тем узлам, которые являются базовыми, имеют большую глубину вложенности или все составные узлы которого 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. Компиляция составных узлов основана на том, что вычислительная семантика как базовых, так и составных узлов полностью определяется их описаниями. Это позволяет автоматически породить текст программы на некотором языке программирования, который после компиляции реализует исполнение откомпилированных узлов. Компиляцию следует применять для часто используемых и устоявшихся составных узлов, так как актуализация этих узлов или используемых для их определения других узлов потребует больших вычислительных ресурсов.

Актуализация в процессе исполнения

Характерной чертой динамического управления процессами является возможность корректировки созданных и запущенных для исполнения процессов в реальном времени.

Как было показано ранее, при активизации узла во временную область памяти копируется его полное описание, а исполнение узла происходит в результате передачи этого описания некоторому интерпретатору или непосредственному исполнителю (рис. 9). Такая организация процесса исполнения позволяет после запуска узла изменять его описание – активация изменённого узла приведёт к копированию и исполнению нового описания. Таким образом, предлагаемый подход позволяет выполнять актуализацию моделей процессов в соответствии с изменяющимися или уточняющимися данными о моделируемых процессах.

Динамическое распараллеливание

При исполнении процесса возможна ситуация, когда составляющие его процессы могут быть выполнены параллельно. По этой причине нотации известных методологий моделирования включают в себя специальные элементы для явной (статической) организации распараллеливания процессов. Для этого, например, используется такие средства как «логическое И», многоэкземплярные циклы параллельного типа, развёрнутые процессы с параллельными блоками, шлюз «ИЛИ» с неэксклюзивным условием, и т.п. [2, 10]. Однако у явного управления распараллеливанием имеются существенные недостатки. Во-первых, снижается наглядность модели процесса, которая особенно важна на этапах его анализа и верификации. Во-вторых, для параллельного исполнения теряется большая часть запущенных процессов, распараллеливание которых по многим причинам не могут быть в явном виде указаны на диаграммах.

Моделирование процессов в формализме совмещённых сетей управления и данных позволяет реализовать метод динамического (неявного) управления распараллеливанием, основанный на использовании естественного параллелизма процессов [14]. В связи с тем, что составной узел содержит явное указание на принимаемые и передаваемые данные между внутренними узлами, то при его исполнении распараллеливанию подлежат все активированные узлы. Следует заметить, что здесь проявляется динамический характер распараллеливания, так как активация узлов зависит от конкретной траектории развития моделируемого процесса.

Наиболее эффективно динамическое распараллеливание проявляется при циклической интерпретации, где выбор узла для исполнения происходит среди множества активированных узлов, в том числе и с разной глубиной вложенности.

Таким образом, совместная и раздельная трассировка данных и управления предоставляет гибкий механизм синхронизации моделируемых процессов и позволяет реализовать естественное их распараллеливание [15].

Заключение

Применение методологии анализа и моделирования процессов на основе совмещённых сетей управления и данных позволяет преодолеть накопившиеся проблемы в области промышленных систем управления бизнес-процессами. В частности, это позволяет на основе графической модели процесса без участия разработчика породить исполняемый код, необходимый для реализации этого процесса. Следует заметить, что предлагаемая графическая нотация является такой же понятной аналитикам, как и другие зарекомендовавшие себя нотации, так как содержит небольшое число элементов и простые правила их интерпретации. Немаловажным является возможность актуализации запущенных моделей процессов при их естественном изменении в бизнес-среде и возможность организовать параллельные вычисления с использованием естественного параллелизма процессов.

ЛИТЕРАТУРА

1. ГОСТ Р ИСО 9000-2008 «Система менеджмента качества. Основные положения и словарь» (ISO 9000:2005 «Quality management systems. Fundamentals and vocabulary»).

2. Davis R. Business Process Modeling with ARIS: A Practical Guide. – New York: Springer-Verlag, 2005. – 531 p.

3. Марка Д. Методология структурного анализа и проектирования SADT / Д. Марк, К. МакГоуэна; предисловие Д.Т. Росса. – М.: МетаТехнология, 1993. – 247 с.

4. Integration Definition For Function Modeling (IDEF0). – Washington: National Institute of Standards and Technology, 1993. – 116 p.

5. Gane C., Sarson, T. Structured Systems Analysis: Tools and Techniques. – New York: Prentice-Hall, 1979. – 452 p.

6. Черемных С.В. Структурный анализ систем: IDEF-технологии. – М.: Финансы и статистика, 2001. – 208 с.

7. Information Integration for Concurrent Engineering (IICE). IDEF3 Process Description Capture Method Report / R.J. Mayer, C.P. Menzel, M.K. Painter, et al. – Ohio: Air Force Materiel Command, 1995. – 235 p.

8. Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя. – М., СПб.: ДМК Пресс Питер, 2004. – 432 с.

9. Fischer L. Workflow Handbook 2002. – Association with the Workflow Management Coalition (WfMC), 2002. – 428 p.

10. Smith H., Fingar P. Business Process Management (BPM): The Third Wave. – Meghan-Kiffer Press, 2002. – 312 p.

11. Яцутко А. В. Моделирование бизнес–процессов крупномасштабного производства // Труды III Международной конференции «Управление развитием крупномасштабных систем». – М.: ИПУ РАН, 2009. – Том II. – С. 301–303.

12. Яцутко А. В. Управление проектами на основе графического моделирования с описанием вычислительной семантики // Труды международной научно–практической конференции «Теория активных систем». – М.: ИПУ РАН, 2009. – Том II. – С. 61–64.

13. Яцутко А. В. Актуализация моделей бизнес-процессов в совмещённых сетях управления и данных // Сборник трудов Второй российской конференции с международным участием «Технические и программные средства систем управления, контроля и измерения». – М.: ИПУ РАН, 2010. – С. 1268-1272.

14. Яцутко А. В. Параллельные вычисления в методах теории управления на основе совмещенных сетей данных и управления // Доклады 5-ой Международной конференции «Параллельные вычисления и задачи управления». – М: ИПУ РАН, 2010. – С. 519-528.

15. Яцутко А.В. Синхронизация данных в автоматных сетях с раздельной трассировкой управления и данных // Сборник трудов Третьей российской конференции с международным участием «Технические и программные средства систем управления, контроля и измерения». М.: ИПУ РАН, 2012. – С. 2060-2070.