УДК 004.421.2

ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ В МЕТОДАХ ТЕОРИИ УПРАВЛЕНИЯ НА ОСНОВЕ СОВМЕЩЕННЫХ СЕТЕЙДАННЫХ ИУПРАВЛЕНИЯ

А.В. Яцутко
Институт проблем управления им. В.А. Трапезникова РАН
Россия, 117997, Москва, Профсоюзная ул., 65
E–mail: aleksundra@yatsutko.net

Рассматривается задача эффективного выделения параллельных подпроцессов при выполнении процесса в рамках автоматизированной системы управления процессами. Для решения этой задачи необходимо использовать графическую нотацию на основе совмещённых сетей с явно выделенным потоком данных. Описываются методы построения моделей и их обработки. Показано, что при исполнении таких моделей используется естественный параллелизм процесса, что не требует затрат на явное выделение параллельных подпроцессов с помощью дополнительных графических примитивов и повышает эффективность системы управления процессами.
Ключевые слова: параллельные вычисления, системы управления процессами, совмещённые сети, методы исполнения графических моделей.

PARALLEL COMPUTING IN MANAGEMENT THEORY`S METHODS ON BASIS OF CONTROL AND DATA COMBINED NET

/ A.V. Yatsutko (V.A. Trapeznokov Institute of Control Sciences, 65 Profsoyuznaya, Moscow 117997, Russia). This paper considers the problem of effective separation of parallel threads during process execution within the bounds of computer–aided business process management system. The graphic notation with marked out data–flow on basis of combined net is needed to solve this problem. Methods of modeling and model`s processing are described. During the executing of such a model the process`s natural parallelism is used, and this method requires no costs for evident separation of parallel threads with additional graphic items and increases business process management system efficiency.
Key words: parallel computing, process management systems, combined net, methods of graphic model`s executing.

1. Введение

Процессно–ориентированный подход, заключающийся в представлении действующей системы как многоуровневого набора взаимосвязанных процессов, широко используется в современных системах управления. Промышленные реализации автоматизированных систем управления бизнес–процессами – BPMS (Business Process Management System) или BPM–системы – предоставляют инструмент для создания, запуска, мониторинга, контроля и учёта динамических моделей процессов управления [1].

Для каждого реального процесса создаётся модель, которая точки зрения целей управления достаточно полно его описывает: исполнители (люди или исполнительные устройства), привлекаемые ресурсы, входящие и исходящие элементы данных, условия запуска и условия выполнения процессов, а так же выполняемые при этом операции. Один процесс может состоять из множества других процессов, понимаемых, в свою очередь, как элементарные операции.

Базовой формой динамического моделирования таких процессов являются диаграммы, создаваемые в одной из стандартных графических нотаций. Эти диаграммы представляют

собой схемы процессов управления, где для каждого элемента диаграммы или их наборов задаётся вычислительная семантика, необходимая для последующего преобразования диаграммы в исполняемую модель процесса управления (программу управления). Тем самым диаграммы связываются с объектами управления и реализуют требуемые процессы управления.

Однако реальные процессы управления допускают не только последовательное, но и параллельное выполнение. Отсюда следует, что графические модели должны позволять выявлять параллелизм моделируемых процессов. Но тут возникают проблемы, связанные с автоматическим выделением групп параллельно выполняющихся процессов (подпроцессов).

Очевидно, что управлять параллельностью подпроцессов можно двумя способами: явно (на этапе создания графической модели) и неявно (на этапе работы исполняемой модели).

Графическая модель процесса управления с явным указанием на параллельность содержит специальные примитивы, с помощью которых процессы и подпроцессы распараллеливаются. Метод явного управления параллельностью применяются в современных BPM–системах. Они используют такой инструментарий графических нотаций (ARIS, BPMN) как «логическое И», многоэкземплярные циклы параллельного типа, развёрнутые подпроцессы с параллельными блоками, шлюз «ИЛИ» с неэксклюзивным условием.

Но у явного управления параллельностью есть существенные недостатки. Во–первых, в угоду простоты распараллеливания снижается наглядность схем процесса управления, особенно важная для пользователя на этапах анализа и верификации диаграмм. Во–вторых, теряются для параллельного выполнения части реальных процессов, которые не могут быть указаны на схеме как параллельно выполнимые по разным причинам: по причине невидимости всех возможных параллельных подпроцессов на одной диаграмме, неочевидностью взаимосвязей подпроцессов по причине непрозрачности передачи данных между подпроцессами и неоднозначности появления возможности распараллеливания подпроцессов, которая может зависеть от исходных данных или результатов выполнения связанных подпроцессов.

Метод неявного управления параллельностью видится более перспективным. Он заключается в том, что подпроцессы, выявленные на этапе создания диаграммы, должны выполняться в таком же порядке, как реальные бизнес–процессы. При создании диаграммы в неё закладывается общая логика процесса, его внутренние правила и ограничения. Неявное управление параллельностью подразумевает использование естественной параллельности процесса.

При таком подходе важна достоверность и точность диаграмм, степень их соответствия реальным процессам [2]. Современные графические нотации не имеют возможности использовать неявное управление параллельностью, потому что не выделяют отдельный поток данных [3, 4]. Это приводит к тому, что на основе диаграммы становится невозможным определить без явного указания, какие процессы могут быть выполнены параллельно, а какие нет.

Таким образом, решение задачи выделения групп параллельных подпроцессов сводится к задаче создания такой графической нотации, которая позволит выявлять параллельные подпроцессы.

Для решения этой задачи предлагается использовать совмещённые сети, в которых реализована синхронизация не только по управлению, но и по данным.

2. Совмещённые сети управления и данных

Графическая нотация, основанная на совмещённых сетях управления и данных, позволяет явно выделить как отдельные потоки управления, так и потоки данных [5]. Это даёт возможность синхронизации процессов как по управлению, так и по данным в рамках одного подхода и на одном уровне. Такой подход позволяет определять параллельные подпроцессы без явного графического указания на их параллельность.

2.1. Графические примитивы

Предлагаемая графическая нотация содержит один графический примитив – «блок» и два типа связи – по управлению (горизонтально) и по данным (вертикально). Схема базового блока представлена на рис. 1.

Рис. 1. Базовый блок

Блок – это графическое представление процесса. Активатор – это точка входа в блок потока управления, событие – точка выхода. Активаторов, событий, входных и выходных данных у одного блока может быть любое количество. Данные на диаграмме передаются сверху вниз, а управление – в любую сторону в горизонтальной плоскости. К каждому базовому блоку привязывается вычислительная семантика в виде последовательности действий (операций) на языке исполнителя базового блока. В качестве исполнителя может выступать некоторая система программирования, виртуальная машина, внешняя библиотека, исполнительное устройство, человек, то есть некоторый внешний интерпретатор для заданной последовательности действий, запускаемой активатором и параметризируемой входными данными.

Другой используемый тип графического примитива – составной блок (рис. 2).

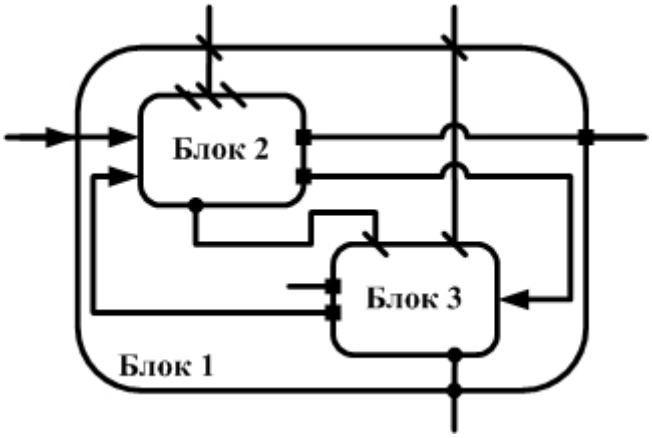

Рис. 2. Составной блок

Составной блок не имеет собственной вычислительной семантики. Он состоит из других блоков, а его вычислительная семантика задается внутренними блоками и их связями.

Составной блок может содержать как базовые блоки, так и блоки, тоже являющиеся составными. Глубина вложенности не регламентируется и определяется на этапе анализа и декомпозиции предметной области перед созданием модели процесса.

2.2. Пример диаграммы

На рис. 3 представлен пример схемы организационного процесса управления «Создание комплекта документов».

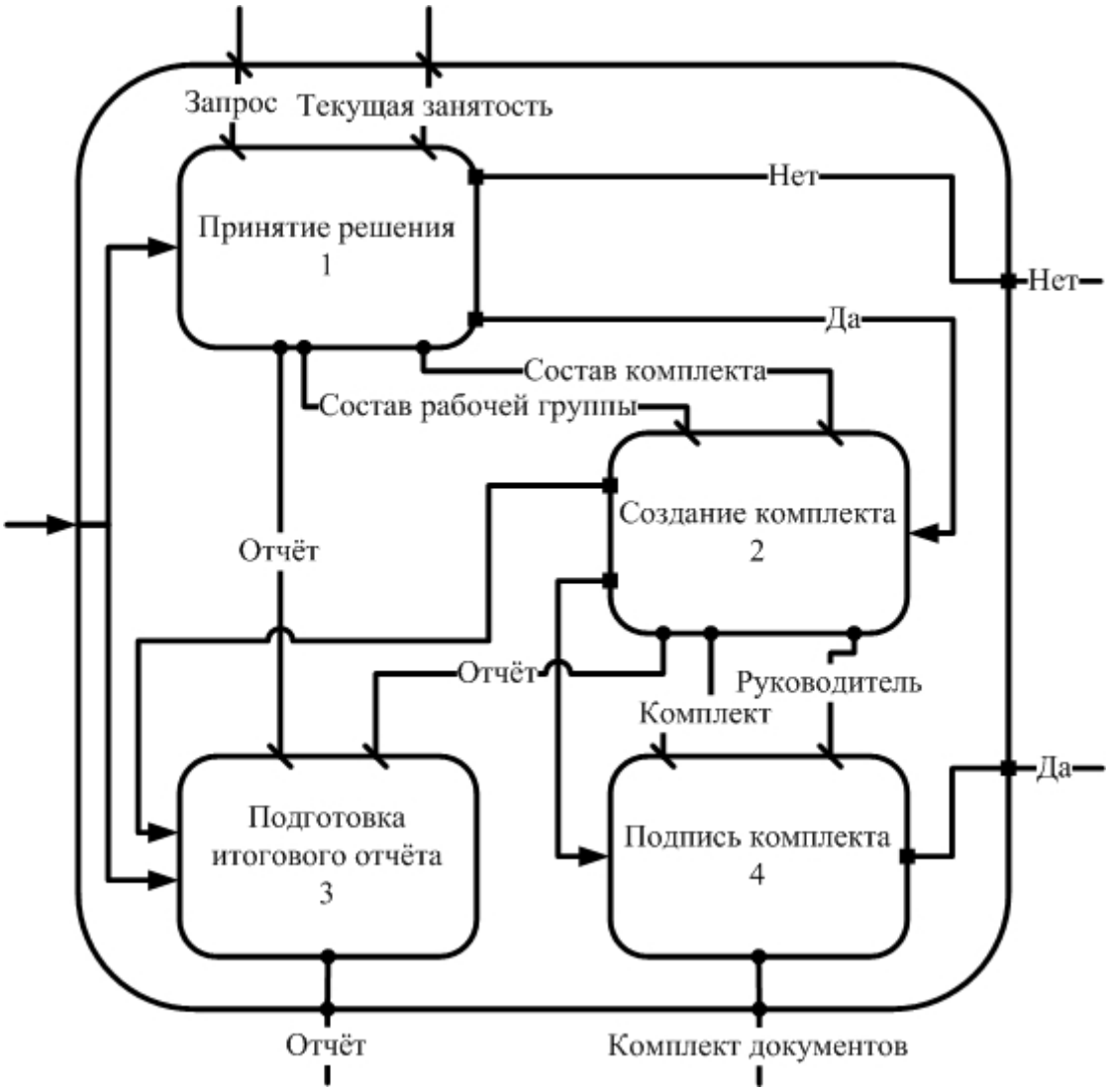

Рис. 3. Схема процесса «Создание комплекта документов»

Основной процесс «Создание комплекта документов» представлен в виде составного блока, которые содержит внутренние блоки – подпроцессы «Принятие решения» (1), «Создание комплекта» (2), «Подготовка итогового отчёта» (3) и «Подпись комплекта» (4).

При активации составного блока сигнал активации передаётся блокам 1 и 3. Входные данные, переданные составному блоку (запрос на выполнения работы по созданию комплекта документов и информация о текущей занятости сотрудников), передаются блоку 1. После того, как будет выполнен блок 1, он сгенерирует одно из событий – «Нет» (в случае, если было принято решение не выполнять работу) или «Да» (в случае, если работа будет выполняться). Блок 3, получив данные (отчёт о принятии решения) и сигнал активации на второй вход, начинает работу и выдаёт итоговый отчёт на порт выходных данных.

Если блок 1 сгенерировал событие «Нет», никакие другие блоки не окажутся активными и, в итоге, получим завершённый процесс «Создание комплекта документов», итогом исполнения которого будет событие (сообщающее о том, что комплект сформирован не был) и итоговой отчёт.

Если блок 1 сгенерировал событие «Да», происходит активация процесса создания комплекта на основе полученных от блока 1 данных (состав комплекта и список членов рабочей группы).

Если создание комплекта прошло с ошибками и не завершилось, событие передаётся на первый выход и активирует блок 3, который в то же время получает по потоку данных отчёт о произошедшем. После завершения работы блока 3 активных блоков не останется и процесс будет завешён.

Если создание комплекта прошло успешно, события подаются на оба выхода: активируются и подготовка отчёта, и блок 4 – «Подписание комплекта документов». При этом блок 4 получает в качестве исходных данных сам комплект для подписи и сведения о том, кто должен его подписать. В результате работы блока 3 на выходе получается итоговый отчёт всего процесса, а блок 4 генерирует событие, означающее успешное создание комплекта документов и передаёт по потоку данных подписанный комплект.

Каждый из внутренних блоков (1–4) может в свою очередь являться составным блоком и содержать более детальную схему этой части общего процесса.

3. Параллельные вычисления в совмещённых сетях

Совмещённые сети данных и управления позволяют использовать параллельные вычисления без дополнительных затрат на явное указание параллельности подпроцессов.

3.1. Исполнение блоков

Основой исполнения совмещённых моделей является то, что активаторы дифференцируются, а события – интегрируются. То есть, сигнал активации, пришедший на входной порт блока, может сработать только один раз, а события на выходных портах блока накапливаются в процессе его исполнения. В описании блока содержится информация о том, какие активаторы и сочетания активаторов могут запустить блок и какие именно операции будут выполняться в том или ином случае.

Исполнение блока начинается при поступлении на один или несколько выходных портов сигнала об активации. В этот момент считываются входящие данные, создаются необходимые внутренние переменные (активаторы, события, входные и выходные данные).

Если блок базовый, его вычислительная семантика передаётся указанному в описании блока исполнителю. После того, как исполнитель закончил работу с блоком, этот блок считается исполненным. Если блок составной, его исполнение – это исполнение всех его внутренних блоков. Составной блок считается неисполненным до тех пор, пока у него есть хоть один не завершивший работу внутренний блок. Как только ни один внутренний блок не оказывается активированным, исполнение составного блока считается завершённым.

После завершения исполнения блока на выходные порты передаются данные и события. Эти данные могут быть исходными для других блоков, а события, в свою очередь, могут активировать другие блоки.

Так как потоки управления и данных являются сквозными для всей диаграммы, исполнение любого составного блока рано или поздно сводится к исполнению набора

базовых блоков. Такой подход позволяет исполнять параллельно не только внутренние блоки, принадлежащие составному, но и блоки разных уровней декомпозиции.

3.2. Обработка блоков

Алгоритм исполнения совмещённых моделей представлен на рис. 4.

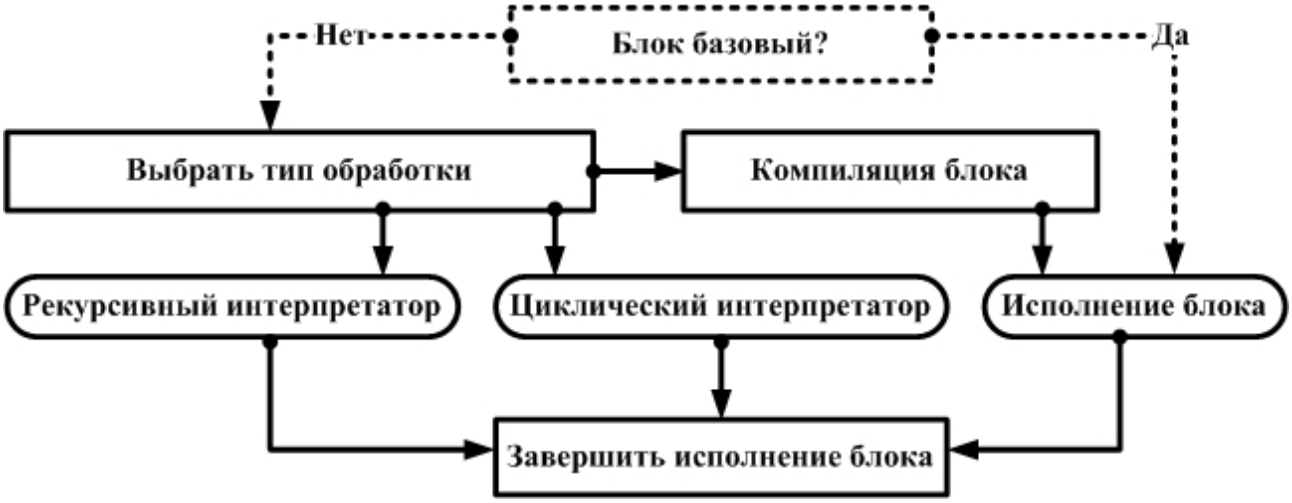

Рис. 4. Обработка блоков

Базовые блоки исполняются сразу, так как их вычислительная семантика задана явно. Для составных блоков имеется три подхода к реализации их исполнения – рекурсивный интерпретатор, циклический интерпретатор и компиляция.

3.2.1. Рекурсивный интерпретатор

Рекурсивный интерпретатор (рис. 5) исполняет сразу весь составной блок целиком.

Рис. 5. Рекурсивный интерпретатор

Процедура «Выбрать блок» последовательно выбирает все активные блоки, входящие в состав основного блока. Если все внутренние блоки оказываются базовыми, происходит их последовательное исполнение и работа основного блока завершается.

Если выбран составной блок, он запускается на исполнение: дифференцируются его активаторы и пересылаются его данные. Те внутренние блоки, которые входят в его состав, получают сигналы активации и входные данные. После этого снова запускается рекурсивный интерпретатор начиная с процедуры «Выбрать блок», но теперь последовательный выбор происходит на множестве тех внутренних блоков, которые входят в уже выбранный ранее составной блок и оказались активированы.

После того, как исполнились все внутренние блоки выбранного составного блока, происходит интегрирование событий и принятие данных этого блока. В случае, если какие–то внутренние блоки снова оказались активированы с помощью сгенерированных событий, для них повторяется последовательный выбор и запуск рекурсивного интерпретатора.

После того, как больше не окажется активных внутренних блоков, ранее выбранный составной блок считается исполненным и происходит выбор следующего блока из множества активных, входящих в состав основного блока.

Этот способ разумно применять только при исполнении несложных составных блоков, так как при рекурсивной обработке требуется много машинной памяти. При работе рекурсивного интерпретатора параллельные вычисления возможны только в том случае, если в составе исполняемого составного блока есть другие блоки, которые передаются другим параллельно работающим рекурсивным интерпретаторам или исполнителям.

3.2.2. Циклический интерпретатор

Циклический интерпретатор (рис. 6) – это основной метод исполнения составных блоков.

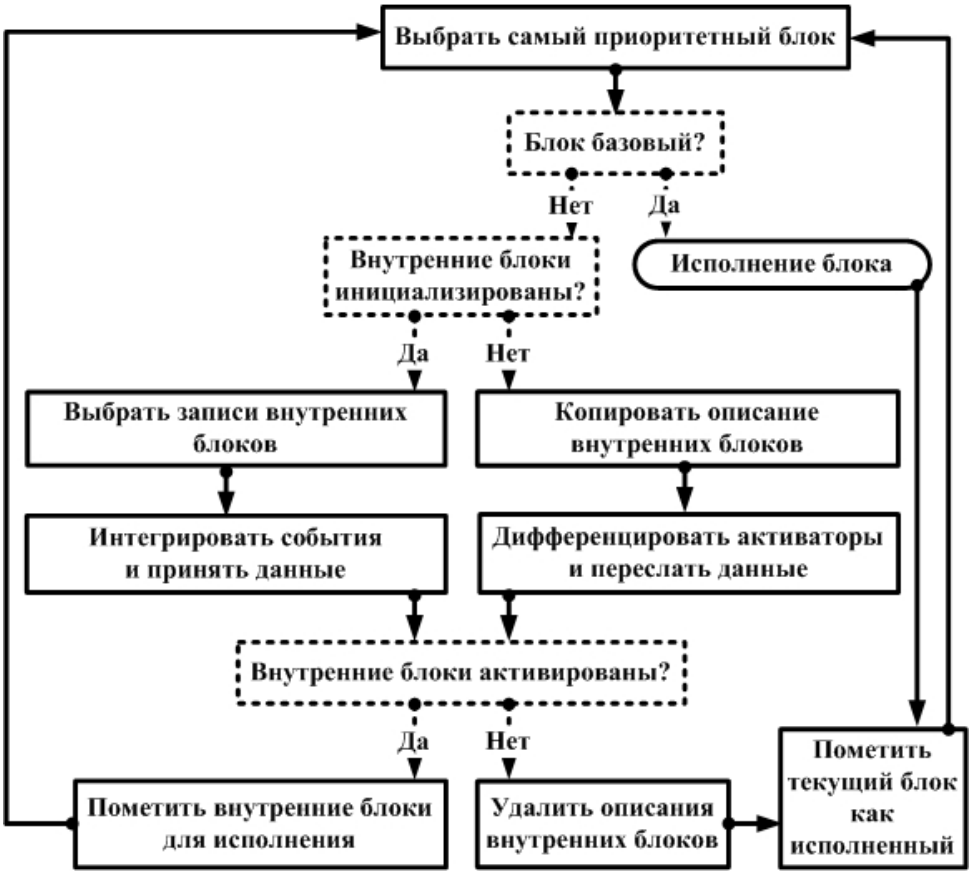

Рис. 6. Циклический интерпретатор

При активации основного блока в специальную таблицу заносятся описания всех его внутренних блоков. У каждого из них есть показатель приоритетности, который зависит от многих факторов (тип внешнего или внутреннего блока, входные данные и т.д.) и специальный флаг, который указывает, помечен ли блок для исполнения (то есть, активирован и должны начать исполняться). Процедура «Выбрать самый приоритетный блок» работает со всем множеством записей таблицы исполняемых блоков, которые помечены для исполнения.

Если выбранный блок оказался базовым, происходит его исполнение и выбирается следующий блок. Если выбранный блок оказался составным, происходит проверка – были ли его внутренние блоки инициализированы (то есть, внесены в таблицу исполняемых блоков). Если не были, описания внутренних блоков вносятся в эту таблицу, а затем происходит дифференциация активаторов и передача данных выбранного (внешнего по отношению к только что внесённым в таблицу) блока. Если в результате дифференциации активаторов ни один внутренний блок не получил сигнала об активации, выбранный блок считается исполненным. Если какие–то внутренние блоки были активированы, они помечаются для исполнения. То есть, включаются в множество блоков, с которыми работает процедура «Выбрать самый приоритетный блок».

Если после проверки на инициализацию внутренних блоков оказывается, что их описания уже были скопированы в таблицу исполняемых блоков ранее, происходит интегрирование событий и передача данных для всех внутренних блоков, входящих в состав изначально выбранного блока. До тех пор, пока хоть один из них оказывается активированным после очередного интегрирования событий, все активированные блоки помечаются для исполнения (то есть, становятся доступны для выбора с учётом обозначенного приоритета). Когда все внутренние блоки становятся исполненными, их описания удаляются из таблицы исполняемых блоков, а выбранный составной блок считается исполненным.

Таким образом, процедура «Выбрать самый приоритетный блок» работает с множеством блоков разных уровнях декомпозиции. Исполняются те блоки, которые могут быть исполнены в данный момент с учётом синхронизации по управлению и по данным, то есть, в рамках одной вычислительной системы происходит параллельное выполнение процессов и подпроцессов разных уровней.

Этот способ исполнения составных блоков является универсальным и позволяет максимально использовать естественный параллелизм процесса с учётом его текущего состояния. Кроме того, такой способ экономит аппаратные ресурсы: если окажется, что некоторые внутренние блоки завершат работу раньше других, освободившиеся вычислительные ресурсы направляются на параллельное выполнение оставшихся блоков.

3.2.3. Компиляция

Третий способ исполнения составных блоков заключается в том, что их вычислительная семантика создаётся сразу, на основе вычислительной семантики внутренних блоков и в виде откомпилированного машинного кода для заранее заданного исполнителя. И исполнение блока сводится к исполнению этого кода. Такой способ разумно применять для часто используемых и устоявшихся составных блоков с несложными внутренними связями. При этом способе параллельные вычисления реализуются средствами выбранного исполнителя и повторяют по своей структуре естественную параллельность реализуемого блока.

4. Заключение

Научная новизна предлагаемого подхода заключается в выделении отдельного потока данных, что даёт возможность использования естественного параллелизма задачи, который получается непосредственно из графического описания процесса. Все необходимые для выявления параллельности данные закладываются в схему уже на этапе проектирования, но для этого, в отличие от широко используемого подхода явного выделения параллельных подпроцессов, не надо прилагать дополнительные усилия и использовать специальные инструменты. Кроме того, гибкий подход к исполнению блоков (процессов) позволяет использовать параллельные вычисления наиболее эффективным способом.

Более того, предлагаемый подход не только улучшает наглядность схем и упрощает их создание, но и повышает эффективность работы системы управления процессами.

Список литературы

1. Stefanie Rinderle-Ma, Shazia Sadiq, Frank Leymann (Eds.). Business Process Management Workshops. Springer, 2005.
2. Яцутко А. В. Управление проектами на основе графического моделирования с описанием вычислительной семантики. Труды международной научно–практической конференции «Теория активных систем – 2009», том II. – М.: ИПУ, 2009. С. 61–64.
3. Davis R. Business Process Modelling with ARIS: A Practical Guide. Springer, 2005. P. 21–25.
4. White S.A., Miers D. BPMN Modeling and Reference Guide. Lighthouse Point: Future Strategies Inc., 2008. P. 20–22.
5. Яцутко А. В. Моделирование бизнес–процессов крупномасштабного производства. Труды III Международной конференции «Управление развитием крупномасштабных систем» (MLSD'2009), том II. – М.: ИПУ, 2009. С. 301–303.