МЕТОД ОТОБРАЖЕНИЯ ИСПОЛНЯЕМОЙ МОДЕЛИ БИЗНЕС-ПРОЦЕССА В СЕТИ ПЕТРИ

УДК 004

Игорь Григорьевич Фёдоров,
к.т.н., проф., Московский государственный университет экономики, статистики и информатики (МЭСИ)
Эл. почта: IFedorov@mesi.ru,

Исполняемые модели бизнес-процессов, как и программы, нуждаются в доказательстве бездефектного завершения. Сегодня широко используются методы, основанные на формализме сетей Петри. Бизнес-процесс представляется сетью Петри, а его свойства устанавливаются путем анализа свойств сети. Целью работы является исследование методов отображения исполняемой модели бизнес-процесса в сети Петри. Анализ свойств полученной модели позволяет доказать ряд важных свойств: она является сетью свободного выбора и чистой, не имеет зацикливаний.

Ключевые слова: верификация бизнес-процессов, сети Петри, BPMN.

Igor G. Fedorov,
PhD in Technical Sciences, Professor, Moscow State University of Economics, Statistics and Informatics (MESI)
E-mail: IFedorov@mesi.ru

METHOD OF DISPLAYING AN EXECUTABLE BUSINESS PROCESS MODELS INTO PETRI NETS

Executable business process models, as well as programs, require evidence of a defect-free finish. The methods based on the formalism of Petri nets are widely used. A business process is a network of dishes, and its properties are set by the analysis of the properties of the network. The aim is to study the methods of displaying an executable business process model in a Petri net. Analysis of the properties of the resulting model allows us to prove a number of important properties: it is a network of free choice and clean without looping.

Keywords: verification of business processes, Petri nets, BPMN.

1. Введение

Исполняемые модели бизнес-процессов, как и программы, нуждаются в доказательстве бездефектного завершения. Единожды стартовав, процесс должен завершиться за конечное число шагов, для этого – не содержать ловушек и бесконечных циклов. Вопросы анализа бездефектного завершения является очень актуальным, поскольку сложность моделей постоянно возрастает, а встроенные в среду моделирования средства проверки пока являются далеко не совершенными. Поэтому основная нагрузка по отладке модели ложится на плечи аналитика, который не имеет достаточной математической подготовки, что бы анализировать процесс инженерными методами [1].

Большое число работ, посвященных анализу бездефектного завершения бизнес-процесса, [2], [3], [4], [5], [6], используют подход, основанный на отображении бизнес-процесса в сети Петри (СП) с целью дальнейшего его анализа формальными математическими методами. Базовые концепции этих работ формировались, когда на рынке доминировали системы управления потоком работ. Сейчас большую популярность приобрели системы управления бизнес-процессами (BPMS). Вследствие некоторых отличий этих систем, критерии бездефектного завершения нуждаются в уточнении. Данная работа ставит перед собой цель – обосновать способы отображения модели процесса в нотации BPMN в СП, выявить свойства модели процесса, важные для дальнейшего анализа.

2. Сети Петри и их свойства

Сети Петри есть помеченный двудольный ориентированный граф, состоящий из вершин двух типов – позиций и переходов, соединённых между собой дугами, причем вершины одного типа не могут быть соединены непосредственно (см. Рисунок 1). В позициях могут размещаться маркеры, способные перемещаться по дугам через переходы вдоль СП. Распределение маркеров по позициям называется маркировкой, она определяет состояние СП в любой момент времени. Дуги имеют кратность, которая обозначает количество маркеров перемещаемых по данной дуге в результате срабатывания перехода.

Математически сеть представляется четверкой $N = (P, T, F, M_0)$, где $P = \{p_i \mid i = 1,n\}$ есть непустое множество, элементы которого называются позициями, а $T = \{tj \mid j = 1,m\}$ – непустое множество переходов, причем $P \cap T = 0$, множество дуг $F \subseteq (P \times T) \cup (T \times P)$ описывает бинарное отношение инцидентности между местами и переходами, наконец маркировкой сети $M: P \rightarrow$ □ называют отображение множества позиций на множество СП называется сильно связной, если из любого узла сети можно пройти в любой другой, передвигаясь вдоль направленных дуг.

Поведение СП принято рассматривать в терминах изменения ее состояния, происходящего в результате срабатывания перехода по следующим правилам. Что бы переход был подготовлен, каждая его входная позиция должна содержать достаточное для срабатывания количество маркеров – большее чем кратность связывающей их дуги. Т.о. переход $t \in T$ считается разрешенным при маркировке M, если $M \geq \bullet t$. В результате срабатывания из всех входных позиций перехода забирается число маркеров, определяемое кратностью дуг, направленных в переход, а в каждую выходную позицию помещается количество маркеров, равное кратности соответствующих исходящих дуг. При этом общее число забранных маркеров и количество помещенных в выходные позиции может не совпадать. Если M есть исходная маркировке, срабатывание перехода t приведет к новой маркировке M´, которая вычисляется по правилу: $M' = M - \bullet t + t\bullet$, где $\bullet t$ есть входящая в переход дуга, а $t\bullet$ – исходящая. Срабатывание перехода t обозначается как $M [t\rangle M'$.

Мертвым называется переход $t_i \in T$, для которого нет достижимой маркировки $M' \in R(M_0)$, в которой переход t_i разрешен и может сработать. Соответственно, маркировка, в которой есть мертвый или неразрешенный переход, называется неживой. Маркировка называется мертвой, если в ней нет ни одного живого перехода, который может сработать. Можно сказать, что всякий мертвый переход

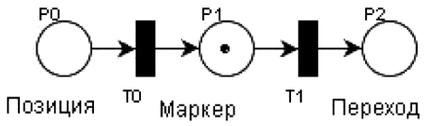

Рис. 1. Сеть Петри

является неживым, но обратное неверно. Неживой переход может оказаться не мертвым.

Переход $t_j \in T$ называется живым, если существует маркировка $M' \in R(M_0)$, достижимая из начальной, в которой он разрешен $M'[t > M''$, где $M'' \in R(M_0)$ другая достижимая маркировка. Сеть называют живой, по отношению к некоторой начальной маркировке, если для $\forall t_j \in T$ существует маркировка $M' \in R(M_0)$, в которой t_j разрешен. Наконец, маркировка $M' \in R(M_0)$ называется живой, если в ней существует переход $t_j \in T$, который может сработать. Сеть (N,M_0) называют живой если в ней нет мертвых переходов.

Ловушка в СП это непустое множество позиций и переходов, в котором каждый переход, который забирает маркер из множества, возвращает его обратно. Попав в ловушку, маркер больше не может ее покинуть и находится в ней бесконечно долго. Непустая ловушка может поглотить любое количество входящих в нее маркеров. Пример (см. Рисунок 2 А) показывает сеть, в которой переход T_0 может захватить маркер, если позиции P_1 изначально содержит хотя бы один маркер. Затем T_0 направит маркер по маршруту: T_0-P_2-T_1-P_1-T_0. Покинуть ловушку маркер не сможет. Если изначально ловушка не содержит маркера, то переход T_0 окажется тупиком, он никогда не сможет сработать.

Сифоном называют множество позиций и переходов, в котором каждый переход, который помещает маркер в это множества, забирает его обратно, поэтому маркер никогда не может в него попасть. Рассмотрим пример (см. Рисунок 2 Б), если позиции P_0 и P_1 изначально не содержат ни одного маркера, то маркер не сможет попасть в сифон ни при каких условиях.

Не следует путать ловушку или сифон с зацикливанием. Ловушка захватывает маркер при любых маркировках сети, а цикл только при определенных. Пример (см. Рисунок 2 В) показывает зацикливание. Позиция P_0 и два перехода T_1 и T_2 образуют логическую функцию «ИЛИ». Если сработает T_1, маркер двинется по процессу, а если сработает T_2, попадет в цикл. Маркер будут находиться в цикле до тех пор, пока не сработает переход T_1.

Количество маркеров, циркулирующих в сети, может изменяться. Некоторые конфигурации СП могут генерировать маркеры в сети. Говорят, что сеть ограничена, если число маркеров в любой позиции конечно. Если же количество маркеров в любой позиции не может превышать одного, то сеть называют безопасной. Сеть, которая не изменяет общее количество маркеров, которые присутствовали в начальной маркировке, называют строго сохраняющей. Сеть которая имеет одинаковое число маркеров в начальной маркировке M_0 и некоторой конечной маркировке $M_к$, тогда как в других маркировках число маркеров может увеличивается и уменьшается, называют сохраняющей относительно некоторого вектора взвешивания. Вектор взвешивания это вектор-столбец, его размер равен числу позиций сети. Любой элемент этого вектора есть вес, ассоциируемый с одной из позицией сети. Он выбирается таким образом, что бы для всех позиций сети произведение числа маркеров в позиции на вес были постоянны для всех позиций сети. Если m_0 число маркеров в начальной позиции, а m_i – в произвольной достижимой позиции, то произведение числа маркеров на соответствующие веса является константой для всех позиций сети $m_0w_0 = m_iw_i$ для любой позиции сети $p_i \in P$.

СП, в которой позиции могут содержать только конечное число маркеров, называется ограниченной. Если же в каждой позиции может одновременно располагаться не более одного маркера, сеть называется 1-ограниченной или безопасной. Сеть называется вполне завершаемой, если всякий раз, когда ее выполнение завершается: (1) маркер достигает позиции, обозначенной как конечная, (2) в сети не остается других маркеров [7].

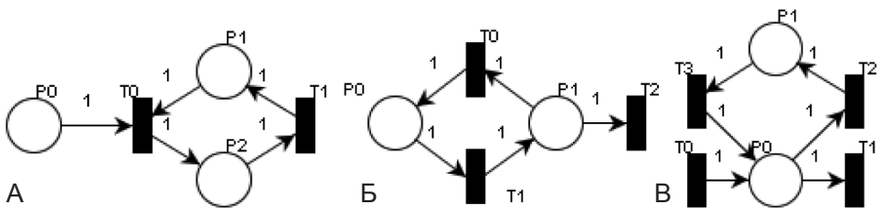

Рис. 2. Ловушка (А), Сифон (Б) и Цикл (В)

3. Поведенческие и структурные свойства сетей Петри

Различают поведенческие и структурные свойства СП [8]. Свойство называют поведенческим, если оно справедливо при определенной начальной маркировки сети. Одноименное свойство называют структурным, если оно справедливо для любой допустимой маркировки сети. Поведенческие и структурные свойства различаются способом доказательства. Первые доказываются путем построения дерева достижимости СП, которое представляет из себя граф, вершинами которого являются все допустимые маркировки сети. Алгоритмы построения графа достижимости известны, однако трудоемкость их построения растет в экспоненциально относительно числа позиций сети. Структурные свойства проверяются с применением матричных методов с использованием матрицы инцидентности, путем поиска неотрицательных P- и T-инвариантов [8].

В большинстве работ предполагается анализ поведенческих свойств модели. Например, свойства живости и безопасности, являются поведенческими. Это накладывает на способы отображения исходного процесса в СП требование поведенческой эквивалентности. В работе [6] анализируются сложности, возникающие при таком отображении. К сожалению, таким образом, удается отобразить в СП только набор элементов, соответствующий предыдущей версии BPMN 1.0.

Однако возникает вопрос о целесообразности поведенческого моделирования процесса в нотации BPMN с помощью СП. Во-первых, СП не моделирует данные, т.ч. нельзя определить направление, по которому логический оператор маршрутизирует поток управления. Это не позволяет анализировать поведение отдельного экземпляра процесса. Во-вторых, системы BPMS позволяют проводить динамический анализ процесса непосредственно в среде моделирования.

Вместе с тем, доказав структурное свойство моделируемого процесса, мы сможем утверждать, что оно выполняется при любых сочетаниях данных

процесса. Отказавшись от необходимости обеспечить поведенческую эквивалентность СП и бизнес-процесса, мы существенно упростим отображение модели процесса в СП. Мы сосредоточимся на анализе структурных свойств модели бизнес-процесса.

4. Отображение модели процесса в нотации BPMN в сеть Петри

Нотация BPMN применяется для разработки исполняемой модели процесса. Она включает богатый набор элементов для описания процессов разных типов, но мы рассмотрим только их ограниченное подмножество, используемое для моделирования процессов оркестровки [9]. В качестве узлов на диаграмме выступают объекты потока управления, включающие: операции, логические операторы и события. Отображение основных конструкций BPMN на СП иллюстрирует Рисунок 3 [6]. Отличие предлагаемого подхода в том, что бы отказаться от поведенческой эквивалентности, это позволит упростить метод отображения.

Договоримся различать узлы, которые производят изменение объекта управления, приводящее к смене его состояния и узлы, которые его не меняют а маршрутизируют. Операции трансформируют объект процесса, они ассоциируется с переходом. Дуги определяют порядок исполнения операций процесса. Последовательное выполнение операций процесса моделируется цепью чередующихся позиций и переходов, связанных простым маршрутом (см. Рисунок 3 А).

Логические операторы объект не изменяют, но маршрутизируют синхронно с потоком управления. Логический оператор ветвления «И» моделируется переходом, который имеет одну входную позицию и несколько выходных (Рисунок 3 Б). В результате срабатывания подготовленного перехода во всех выходных позициях образуются маркеры. В дальнейшем, параллельные ветви выполняются независимо друг от друга. Логический оператор слияния «И» моделируется переходом, который имеет несколько входных позиций и одну выходную (Рисунок 3 В). Что бы переход сработал, во всех входных позициях должны находиться маркеры. После срабатывания перехода происходит синхронизация потоков, только один маркер продолжит движение по сети. Логический оператор ветвления «ИЛИ» моделируется позицией, которая имеет несколько выходных переходов (Рисунок 3 Г). Срабатывание любого из них определяет направление маршрутизации маркера. Логический оператор слияния «ИЛИ» моделируется позицией, которая имеет несколько входных переходов (Рисунок 3 Д).

Нотация BPMN допускает «сокращенную» форму записи, когда один графический элемент объединяет сразу и операцию и логический оператор. Например, можно изобразить операцию, которая имеет два выхода, безусловный и условный. Но для целей анализа правильнее изобразить отдельно операцию и логический оператор «ИЛИ». В рассуждениях мы будем всегда разделять операции процесса и логические операторы.

Циклическое исполнение операции моделируется парой переходов (Рисунок 3 Е). Первый из их выполняет собственно работу, а второй служит для проверки условия. Иногда ошибочно пытаются объединить оба перехода в один, что приводит к самозацикливанию СП. Разделяя эти переходы, мы исключаем зацикливания.

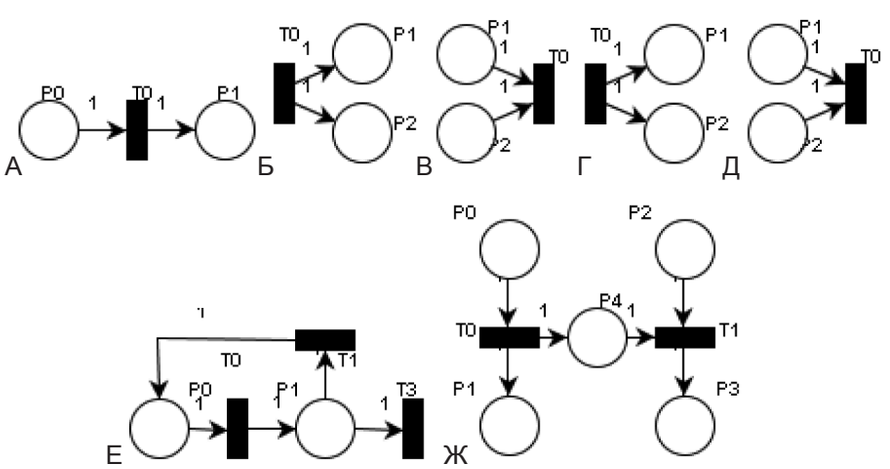

Рис. 3. Отображение модели процесса в нотации BPMN в СП

События и маршрутизируют поток управления и изменяют его. Рассмотрим подробнее разные типы событий. Стартовой событие представляет работу, инициирующее исполнение экземпляра процесса. Когда процесс стартует по получении, например сообщения или сигнала, эта работа очевидна. В других ситуациях, например, когда процесс начинается с нетипизированного события, работа не видна, тем не менее, она есть – пользователь запускает процесса через экранный интерфейс. Завершающее событие представляет собой работу, выполняемую при завершении процесса. Если завершающее событие есть отправка сообщения или сигнала, работа очевидна. Если же это простое нетипизированное завершающее событие, то работа не видна, но выполняется, поскольку в завершающий процесс возвращается статус завершения. Будем моделировать стартовое и завершающее события переходом СП.

События, размещаемые в потоке, поток не трансформируют и не маршрутизируют, иногда приостанавливают. Когда поток управления достигает генерирующего события, размещенного в потоке, то немедленно происходит обработка этого события (отправляется сообщение, сигнал и т.д.), после чего поток управления немедленно покидает текущий элемент и продолжает движение далее по процессу. Когда поток управления достигает знака обрабатывающего события, исполнение останавливается до тех пор, пока не произойдет соответствующее событие (например, получено сообщение, сигнал и т.д.). После этого поток управления покидает элемент обработки события. Поскольку потоки данных СП не моделируются, промежуточные события, размещаемые в потоке можно изобразить переходом (Рисунок 3 Ж).

События, прикрепляемые к границам операций, бывают только обрабатывающими. Наступление события изменяет нормальный сценарий исполнения, возникает дополнительный поток управления, который указывает на обработчик события. Дальнейшее выполнение операции, к которой прикреплено событие, зависит от типа прикрепленного события: прерывающее приостанавливает исполнение, а непрерывающее продолжает её работу. Таким образом, прерывающее событие

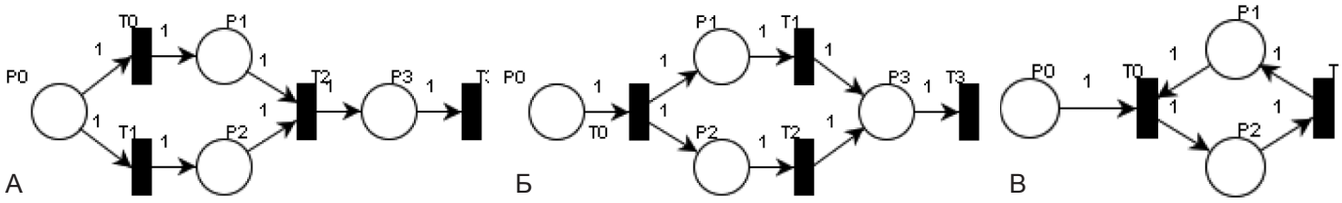

Рис. 4. Мертвая точка (А), генерация маркеров (Б), ловушка (В), сифон (Г)

моделируются логическим оператором «ИЛИ», а непрерывающее - оператором «И».

Позициям СП нет прямого соответствия на диаграмме BPMN. После того как была выполнена очередная операция, но до того как будет выполнена следующая, задание находится в списке заданий, готовых к исполнению, где оно ждет, что сотрудник приступит к его исполнению. В этот момент задание находится на середине пути, связывающего две операции. Можно представить себе, что этой точке между двумя операциями соответствует позиция СП.

5. Коллизии, возникающих в исполняемой модели бизнес-процесса

Рассмотрим, какие ситуации в модели процесса могут вызывать коллизии при его исполнении. Мертвой точкой модели на процесса будем называть логический оператор, который не может сработать ни при каких сочетаниях значений данных процесса, потому что условие никогда не может быть выполнено. Если поток управления достигнет мертвой точки, он остановится и не сможет продвинуться дальше. Возникает явная ошибка логики процесса. Например, представим себе поток управления, который сперва разделяется на две ветви с использованием ЛО «ИЛИ», а затем оба потока соединяются вместе при помощи ЛО «И» (Рисунок 4 А). Синхронизация невыполнима, поскольку после ветвления «ИЛИ» поток управления будет направлен только по одной из ветвей, а узел «И» ждет потоки из обеих.

Мертвой зоной модели называется группу операций, которые не могут быть исполнены ни при каких сочетаниях значений данных процесса, потому что поток управления никогда не может их достигнуть. Либо модель содержит лишние и ненужные элементы, которые можно без ущерба удалить, либо существует ошибка в логике процесса и управление не попадает в нужную точку. Например, если в примере выше после узла слияния «И» есть другие операции (переход T_3 на рисунке 4 А), то они не будут выполнены никогда.

Генератор маркеров есть группа операций и логических операторов, которая в ответ на получение потока управления, генерирует на выходе некоторое число (в т.ч. бесконечное) потоков управления. Например, представим себе поток управления, который сперва разделяется на две ветви с использованием ЛО «И», а затем оба потока соединяются вместе при помощи ЛО «ИЛИ» (Рисунок 4 Б). При этом на выходе возникнут два сигнала, поскольку ЛО «ИЛИ» пропустит потоки из обеих ветвей. Возникает потенциальная опасность – клиент обратился к нам за получением кредита, мы проводим параллельную обработку запроса несколькими службами, но после проверки в процессе движутся две заявки на получение кредита вместо одной.

Дискриминатор потока управления это группа операций и логических операторов, которая в ответ на получение на вход нескольких потоков управления пропустит на выход меньшее их количество, чем поступило на вход. Например, мы проводим голосование участников, для принятия решения достаточно квалифицированного большинства, остальные ответы нас могут не интересовать. Дискриминатор помогает нам отсчитать нужное число ответов, а остальные игнорировать. Дискриминатор обычно используется совместно с генератором. Но если он используется отдельно, он может поглотить все маркеры, так что ни один не достигнет конца процесса.

Ловушкой называется группа некоторого конечного числа операций и операторов процесса, такая, что выход из ловушки одновременно является входом в нее (Рисунок 4 В). Попав в ловушку, маркер не может покинуть ее ни при каких обстоятельствах. Ловушку следует отличать от зацикливания, последняя удерживает процесс только при определенном сочетании значений данных процесса, при выполнении заданных условий маркер сможет покинуть цикл.

Модель процесса не должна содержать висячие и оборванные цепи. Например, недопустим оператор, который не имеет входных дуг, он никогда не получит управления, т.е. является мертвой зоной. Так же недопустим оператор, который не имеет выходного потока. Поток всегда должен заканчиваться завершающим событием, а если его нет, то маркер не сможет покинуть операцию. Т.о. оборванная связь эквивалентна ловушке.

Нотация BPMN допускает использование в одном процессе нескольких точек старта, однако предупреждает, что они должны быть альтернативными, так что наступление первого из них инициирует исполнение экземпляра процесса. Так же допускается существование в одной модели нескольких завершающих событий, однако они должны быть альтернативными, поскольку процесс не может завершиться сразу с несколькими статусами.

6. Свойства СП, эквивалентной модели процесса в нотации BPMN

Покажем, что СП является чистой (pure), в ней не может быть самозацикливаний. Рассмотрим четыре возможные комбинации элементов (Рисунок 5):

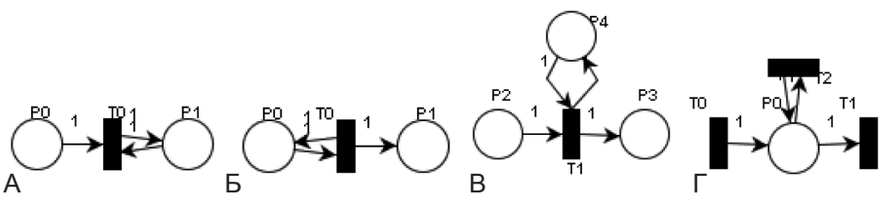

Рис. 5. Варианты самозацикливания СП

А. Позиция связана циклом с последующим переходом. Это «нереальная» ситуация, т.к. операция процесса не может отозвать задание, выполнение которого на данном шаге уже завершено, а позиция пассивна и не может вернуть задание.

Б. Позиция связана циклом с предшествующим переходом. Этот вариант описывает ситуацию, когда исполнитель начал работу над заданием, но прервал работу, что бы продолжить ее позднее, задание возвращается во входную очередь. Этот сценарий является штатным для BPMS, но не моделируется на схеме BPMN, поэтому мы можем исключить его из рассмотрения на модели СП.

В. Переход связан с позицией, которая лежит не на пути от начала к концу процесса. Аналогично предыдущему случаю, работа приостановлена, задание переведено в служебное состояние. Этот сценарий так же является штатным для BPMS, но оно не моделируется на схеме BPMN, поэтому мы можем исключить его из рассмотрения на модели СП.

Г. Переход расположен не на пути от начала к концу процесса. Эту конструкцию неправильно трактуют как исполнение операции в цикле. Разделив собственно операцию и проверку условия, мы превращаем зацикливание в обычный цикл.

Покажем, что СП, используемая для моделирования бизнес-процесса, является сетью свободного выбора, у которой каждая дуга, выходящая из позиции, является либо единственным выходом из нее, либо единственным входом в следующий за ней переход (мы будем рассматривать сеть свободного выбора в наиболее строгой формулировке, соответственно результат окажется справедлив для расширенного класса сетей свободного выбора (EFC) [10]).

Рассмотрим пример (Рисунок 6 А). Нотация BPMN допускает «сокращенную» форму записи ветвления «ИЛИ» и слияния «И» без использования логических операторов. Эквивалентная BPMN схема, показывающая логические операторы ветвления и слияния, изображена на рисунке Б. Преобразуем схему в СП, пользуясь сформулированными выше правилами. При этом будем разделять служебные переходы, используемые для изображения логических операторов, не будем объединять их друг с другом (рисунок В). Результирующая модель есть сеть свободного выбора. Некоторые авторы не разделяют переходы, соответствующие операциям и логическим элементам, при этом редуцируют сеть так, что полученная сеть не соответствует критериям свободного выбора (рисунок Г). Следует помнить, что не все правила редукции сохраняют структурные свойства, поэтому не применимы к данной ситуации [10].

Докажем важное свойство бизнес-процессов - одно инициирующее стартовое событие создает ровно один отклик на его выходе. Будем иметь в виду, каждый результат на выходе бизнес-процесса д.б. индивидуально идентифицируем, так что бы было возможно посчитать результативность процесса за определенный интервал времени [11]. Если предположить, что одно входное воздействие может сгенерировать несколько выходных, то следует допустить, что их число может оказаться неисчислимо большим. Но это противоречит требованию счетности результата. Конечно, можно представить модель, которая на одно входное воздействие будет генерировать выходной сигнал с определенной периодичностью, однако вряд ли это задача моделирования бизнес-процесса, скорее вопрос программирования конечного автомата. Таким образом, СП должна быть сохраняющей относительно некоторого вектора взвешивания.

Подведем итог, рассмотрим набор свойств бизнес-процесса, которые мы положим в основу уточненных критериев доказательства бездефектности. Во-первых, модель должна быть вполне завершаемой, так что маркер, поступивший на вход, за конечное число шагов достигнет выхода. Утверждение справедливо, если СП не содержит мертвых точек и ловушек. А когда маркер достигнет выхода, в других позициях не должно быть других маркеров. Таким образом мы сформулируем второе требование – модель должна быть сохраняющей относительно вектора взвешивания. В-третьих, в сети не может быть мертвых зон и сифонов. Для этого граф СП должен быть сильно связным. В-четвертых, модель может включать несколько точек старта и финиша, при условии, что они являются альтернативными. В-пятых, недопустимо помещать в точку старта сразу несколько маркеров. Первый из них инициирует старт процесса, а остальные не должны оказать влияние на стартовавший ранее экземпляр процесса.

Учитывая, что модель процесса ответствует СП свободного выбора, можно использовать многочисленные известные способы доказательства перечисленных выше свойств [10].

7. Обсуждение и выводы

В основополагающих работах, посвященных анализу бизнес-процессов с использованием сетей Петри [2], [3], [4], [5], [6], присутствует некоторый

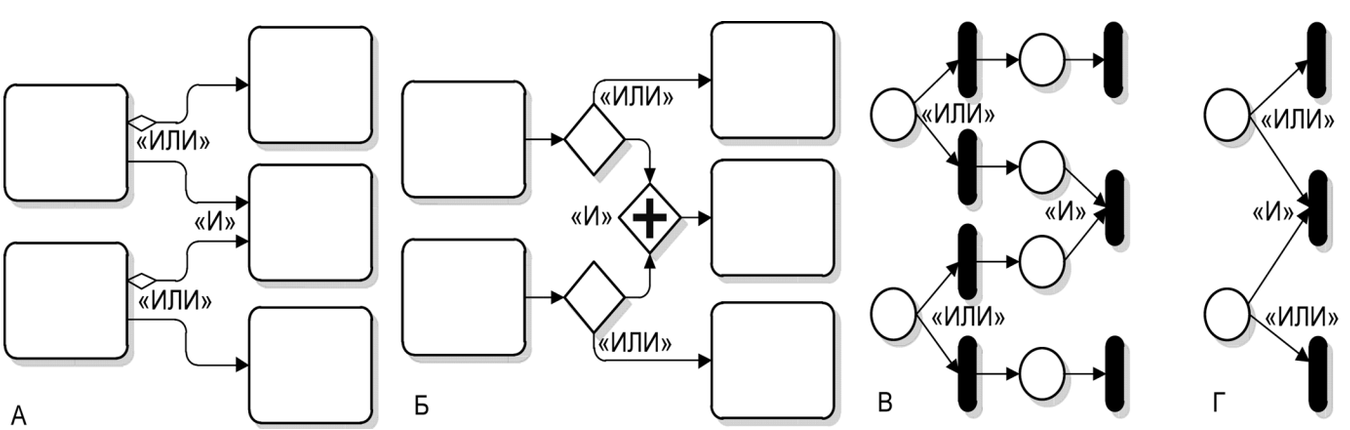

Рис. 6. Отображение BPMN в сеть свободного выбора

дуализм. С одной стороны, справедливо утверждается, что не надо моделировать данные процесса и правила их трансформации операциями процесса, поскольку это привязывает модель к конкретным условиям эксплуатации, делает ее излишне сложной для понимания и анализа. Из этой установки делается правильный вывод, что свойство, доказанное безотносительно данных, будет справедливо для любого сочетания переменных процесса. Однако в рассуждениях постоянно проглядываются попытки поведенческого моделирования всех возможных сценариев исполнения. При этом структурный анализ модели остается без внимания, хотя именно он доказывает соответствующие свойства, справедливые для произвольной маркировки и любого сочетания данных процесса.

Главный вывод данной работы заключается в отказе от анализа поведенческих свойств СП, эквивалентной модели процесса, и проведении исследования исключительно структурными методами. Такой подход обеспечит выполнение главной задачи поиска дефектов модели процесса и позволит существенно упростить способ отображения. Изменение способа отображения заставляет обратить внимание на правила редукции СП. Поскольку в выполненных ранее работах предполагалось проверять живость и безопасность сети, применялись правила редукции, обеспечивающие сохранение этих свойств [8]. Однако такая редукция изменяет структурные свойства сети [10]. В проводимых рассуждениях мы не объединяли переходы, отображающие работу процесса и используемые в логических операторах и, таким образом сохранили структуру неизменной. С учетом замечаний о правилах редукции, удалось обосновать важные свойства сети Петри. Во-первых, она является сетью свободного выбора. Этот вывод позволяет упростить анализ, поскольку свойства сети этого класса могут быть обоснованы аналитически. Во-вторых, модель процесса не содержит зацикливаний, которые могли бы затруднить структурный анализ.

Литература

1. Silver B. // BPMS Watch. 2006. URL: http://brsilver.com/whats-wrong-with-this-picture-part-3/ 2. van der Aalst W. van Hee K. ter Hofstede A.Sidorova N. Verbeek H. Voorhoeve M. W.M., "Soundness of Workflow Nets: Classification, Decidability, and Analysis," Formal Aspects of Computing, Vol. Volume 23, No. Issue 3, May 2011. pp. 333-363.

3. Kiepuszewski B., ter Hofstede A., and van der Aalst W., "Fundamentals of Control Flow in Workflows," Vol. 39, No. 3, 2002.

4. van der Aalst W., "A class of Petri nets for modeling and analyzing business processes," Computing Science Report, Eindhoven Univ. of Technology , Vol. 95, No. 26, 1995.

5. van der Aalst W., "The application of Petri nets to workflow management," Journal of Circuits, Systems and Computers, Vol. 8, No. 1, 1998. pp. 21-66.

6. Dijkman R. Dumas M. Ouyang C., "Formal Semantics and Analysis of BPMN Process Models using Petri Nets," No. http://eprints.qut.edu.au/7115/01/7115.pdf, 2007.

7. Питерсон Д. Теория сетей Петри и моделирование систем. Москва: Мир, 1984.

8. Murata T., "Petri Nets: Properties, Analysis and Applikations," Proceedings of the IEEE, , Vol. 77, No. 4, Apri 1989.. pp. 541-80.

9. Фёдоров И.Г. Моделирование бизнес-процесов в нотации BPMN 2.0. МЭСИ ed. Москва. 2013.

10. Desel J. E.J. Free Choice Petri Nets. Cambridge University Press, 1997.

11. Sharp A. M.P. Workflow Modeling, Artech House Publishers. Artech House Publishers.

12. Schmidt K. Computation of Invariants for Algebraic Petri Nets // Workshop on Concurrency, Specification and Programming (CS&P'93), Proceedings. Warsaw, Poland,. 1993. pp. 196–218.

References

1. Silver B. // BPMS Watch. 2006. URL: http://brsilver.com/whats-wrong-with-this-picture-part-3/ 2. van der Aalst W. van Hee K. ter Hofstede A.Sidorova N. Verbeek H. Voorhoeve M. W.M., "Soundness of Workflow Nets: Classification, Decidability, and Analysis," Formal Aspects of Computing, Vol. Volume 23, No. Issue 3, May 2011. pp. 333-363.

3. Kiepuszewski B., ter Hofstede A., and van der Aalst W., "Fundamentals of Control Flow in Workflows," Vol. 39, No. 3, 2002.

4. van der Aalst W., "A class of Petri nets for modeling and analyzing business processes," Computing Science Report, Eindhoven Univ. of Technology , Vol. 95, No. 26, 1995.

5. van der Aalst W., "The application of Petri nets to workflow management," Journal of Circuits, Systems and Computers, Vol. 8, No. 1, 1998. pp. 21-66.

6. Dijkman R. Dumas M. Ouyang C., "Formal Semantics and Analysis of BPMN Process Models using Petri Nets," No. http://eprints.qut.edu.au/7115/01/7115.pdf, 2007.

7. Piterson D. The theory of Petri nets and simulation systems. Moskva: Mir, 1984.

8. Murata T., "Petri Nets: Properties, Analysis and Applikations," Proceedings of the IEEE, , Vol. 77, No. 4, Apri 1989.. pp. 541-80.

9. Fedorov I.G. Business process modeling notation in BPMN 2.0. MESI ed. Moskva. 2013.

10. Desel J. E.J. Free Choice Petri Nets. Cambridge University Press, 1997.

11. Sharp A. M.P. Workflow Modeling, Artech House Publishers. Artech House Publishers.

12. Schmidt K. Computation of Invariants for Algebraic Petri Nets // Workshop on Concurrency, Specification and Programming (CS&P'93), Proceedings. Warsaw, Poland,. 1993. pp. 196–218.