Автоматика и телемеханика, № 8, 2011

Логическое управление

© 2011 г. А.А. АМБАРЦУМЯН, д-р техн. наук
(Институт проблем управления им. В.А. Трапезникова РАН, Москва)

МОДЕЛИРОВАНИЕ И СИНТЕЗ СУПЕРВИЗОРНОГО УПРАВЛЕНИЯ НА СЕТЯХ ПЕТРИ ДЛЯ РАССРЕДОТОЧЕННЫХ ОБЪЕКТОВ. I. МЕХАНИЗМ ВЗАИМОДЕЙСТВИЯ И БАЗОВЫЙ МЕТОД[1]

Излагается методология дискретно-событийного моделирования класса рассредоточенных объектов и их требуемого поведения (спецификаций) при проектировании систем автоматизации, работающих в реальном времени. В методологии используется модель – структурированная дискретно-событийная система (СДС2) – на первом этапе для анализа функциональности и согласованности объекта и спецификации, а на втором – в рамках СДС2 предлагается базовый метод синтеза модели объекта и супервизора на сетях Петри (моделирующей и управляющей). При этом супервизор предлагается синтезировать как сеть Петри, встроенную в СДС2 по схеме обратной связи с целью ограничения функционирования объекта в соответствии с требованиями спецификации. Предложен механизм взаимодействия моделирующей и управляющей сетей Петри с объектом и внешней средой. Механизм взаимодействия, по сути, является схемой управления объектом на основе сконструированной сети. Этот механизм выполняет анализ текущего состояния объекта и вычисляет управление на актуаторы объекта. Вычисления осуществляются циклической процедурой по матричному представлению сети.

1. Введение

Системы автоматизации управления сложными гибкими производствами интегрируют функции управления процессами, обеспечения безопасности и защиты оборудования, работают в режиме on-line и относятся к классу систем автоматизации, работающих в реальном времени (САРВ). Проектирование САРВ – сложная инжиниринговая задача, требующая значительных человеческих ресурсов и базирующаяся на математическом и компьютерном моделировании. Математической базой моделирования и проектирования подобных систем является теория дискретно-событийных систем (ДСС). Первооткрывателями направления являются В.М. Уонхэм и Дж.Г. Рамадж (W.M. Wonham и J.G. Ramadge) [1]. Обзор состояния и систематическое изложение приведены в [2]. ДСС – это триплет $\langle G, K, S\rangle$, где G – объект, K – требования к поведению объекта, а S – супервизор (управляющий компонент ДСС), обеспечивающий поведение G в соответствии с ограничениями K. При этом S должен быть неблокирующим, т.е. не содержать тупиков и "живых циклов" (ловушек). Функционирование ДСС характеризуется множеством E событий e и генерируемыми последовательностями s этих событий, $e \in E$. Множество генерируемых последовательностей называют языком (например, язык, генерируемый объектом G, обозначается $L(G)$.

[1] Работа выполнена при частичной поддержке Российского фонда фундаментальных исследований (проект № 09-08-00559-а).

В теории получены фундаментальные результаты по основным этапам проектирования ДСС:

- анализу исходного представления поведения объекта G как ничем не ограниченного генератора языка $L(G)$;
- специфицированию ограничений на требуемое поведение объекта G в форме языка $K \subseteq L(G)$ или в терминах недопустимых состояний (линейных ограничений на соотношение активности в позициях для сетей Петри);
- исследованию управляемости и разработке методов синтеза неблокирующего супервизора S.

Полученные результаты открывают новые возможности в проектировании логического управления на основе дискретно-событийного моделирования. Постановка и решение задачи существования и синтеза супервизора дают аналитический ответ на основной вопрос логического управления: ***при заданных объекте и спецификации требований разрешима ли задача управления и, если да, то каким образом***. Более того, если ответ отрицательный, то в [2] исследованы вопросы: при каких изменениях спецификаций (ограничений на поведение объекта) управление разрешимо. Эта теория является прорывом в логическом управлении в части повышения его достоверности, поскольку на фазе проекта аналитически, без отладочных экспериментов, дает ответ проектанту о существовании неблокирующего супервизора, а, следовательно, о согласованности объекта и сформулированных спецификаций. Вместе с тем практика реального проектирования САРВ, опирающаяся на инструментальные средства (например, на SCADA системы), не использует методы теории ДСС.

Имеются две основные трудности в практическом использовании дискретно-событийного моделирования (ДСС-моделирования).

Первая трудность заключается в высокой размерности модели объекта управления G (для реальных задач число состояний конечного автомата достигает порядка нескольких тысяч). Трудности с размерностью исходных данных усугубляются мультипликативной зависимостью сложности синтеза супервизора от сложности объекта G и от сложности спецификации K (как правило, наблюдается "взрыв состояний"). Для преодоления вычислительных трудностей разработаны методы синтеза супервизора в виде совокупности модулей [4–7]. При этом исходным является полное описание G, что практически не уменьшает сложность синтеза. Кроме того, модульность обеспечивается на базе ограничений на языки, представляющие модули, что слабо соответствует структуре исходных спецификаций. Поэтому предложенные методы пока малопривлекательны для практического применения. Сохранение исходной модульности для отдельных классов ДСС достаточно эффективно обеспечивается использованием сетей Петри (СП) как аппарата описания и синтеза супервизорного управления в ДСС ([8–10] и др.).

Вторая трудность – назовем условно "IT-неподготовленность" – проявляется в двух аспектах. Первый аспект: отсутствие структурного подобия синтезированного супервизора и исходных описаний объекта управления. Это вызвано тем, что основные результаты и методы теории ДСС сформулированы в понятийном базисе языков над множеством событий, в то время как в реальном проектировании используются построения, использующие в явном виде понятия "состояние", "условие", "переходы", "структура" и т.п. Эти понятия лежат в основе традиционных способов описания функционирования объекта и его компонент: диаграмм переходов, тактограмм, циклограмм, структурных схем. Сложившийся понятийный базис нашел отражение в международных нормативных документах по проектированию и программированию систем промышленной автоматизации и поддерживается CASE-инструментами (см. например, стандарт *IEC 61131-3* для программируемых промышленных систем управления). Второй аспект: несмотря на более чем двадцатилетний период раз-

вития теории ДСС, методы синтеза супервизорного управления носят скорее теоретический, чем технологический характер. Автору неизвестны работы, в которых проектирование управления в дискретной системе на базе ДСС рассматривалось бы систематически: от методов описания исходных данных (использующих в явном виде понятия "состояние", "условие", "переходы", "структура") до методов построения устройств, программ или вычислительных процедур, реализующих синтезированное супервизорное управление. По мнению автора, указанные причины составляют основное методологическое препятствие на пути создания IT-инструментов, поддерживающих ДСС-моделирование.

Автор делает попытку преодолеть сформулированные выше трудности и ставит задачу: в рамках ДСС разработать методику анализа и проектирования супервизора, ориентированную на сохранение структурного подобия результатов проектирования и исходных описаний поведения объекта управления. Предлагается данные о ДСС последовательно представлять в следующем виде: набор конечных автоматов (этап описания данных о структуре объекта), строки команд (событий) на исполнение операций оборудованием (этап формирования требований технологов на последовательности операций) и структура взаимодействия оборудования при выполнении команд и операций – сети Петри (этап системного аналитика и реализации системы). Алгоритмы, составляющие методику, должны обеспечивать преобразование моделей так, чтобы исходная модульность задания ДСС сохранилась в результирующей управляемой сети Петри и никакие преобразования не приводили бы к эффекту "взрыва состояний". С этой целью исходные структуры данных будут последовательно синтаксически преобразовываться алгоритмами методики в решения по управлению, ограничивающие поведение объекта не более чем того требуют исходные спецификации (ограничения).

Предложения автора оформлены в виде двух статей. В настоящей статье формируется методика моделирования, предлагается механизм взаимодействия (схема управления) и разрабатывается базовый метод синтеза супервизора. Во второй статье будет разработан метод синтеза сети Петри, реализующей супервизор СДС2 по спецификации без ограничения на вид и число строк.

Структура данной статьи. В разделе 2 кратко изложены базовые понятия и результаты, которые используются в предлагаемой методике. В разделе 3 предложен механизм взаимодействия структурированной дискретно-событийной системы (СДС2), представленной сетью Петри, с объектом управления G. В разделе 4 предложены метод преобразования автоматной модели G в S_p(сеть Петри, моделирующую процесс в объекте G), метод синтеза супервизора S_c (управляющей сети Петри) и приведен пример.

2. Базовые понятия и определения

Настоящая работа основывается на модели ДСС[2] $\langle G, K, S\rangle$, где $G = \langle G^1, G^2, \ldots, G^n\rangle$ – набор параллельно функционирующих компонентов (актуаторов), K – требования к их совместному поведению, а S (супервизор – управляющий компонент ДСС, обеспечивающий поведение G в соответствии с ограничениями (спецификацией) K). Ничем не ограниченное поведение G – множество последовательностей (строк) из событий, генерируемых объектом G, – обозначается как язык

[2] Поведение ДСС рассматривается с самых общих позиций как поведение некоторого генератора (источника) последовательностей (строк) *событий* из конечного множества событий E. *Событие* $e \in E$ – это абстракция для множества фактов наблюдения "жизни" ДСС. События мгновенны, их появление спонтанно и происходят они в непредсказуемые моменты времени, поэтому все, что можно наблюдать, это их последовательности, которые и представляются *строками*. Примеры событий: факты изменения положения и состояния отдельных компонент объекта; команды на объект; характеристика исправного или неисправного состояния объекта и т.п. (кратко систему исходных понятий ДСС см. в [11]).

$L(G) = U_1^n L(G^i)$. Ограничения на требуемое поведение G определяются в форме языка $K \subseteq L(G)$. При этом в определении поведения $G = \langle G^1, G^2, \ldots, G^n \rangle$ и K используется разделение множества событий E на E_c и E_{uc} (множества управляемых и неуправляемых событий – традиционное для теории ДСС разбиение) и E_w – множество ожидаемых событий. Эта модель предложена в [11]. События из E_w моделируют состояния (положения) исполнительных механизмов (актуаторов) или компонентов объекта. Эти события нельзя блокировать супервизором как управляемые события из E_c. Однако их появление ожидаемо как отклик на события из E_c, подтверждающий факт выполнения команд на актуаторы.

ДСС с предложенным структурированием назовем *ДСС с углубленным разделением событий*. Поведение каждого компонента $G = \langle G^1, G^2, \ldots, G^n \rangle$ – это язык $L(G^i)$ из событий $e \in E^i = \{E_c^i \cup E_w^i\}$, генерируемый конечным автоматом $G^i = \langle Q^i, E^i, \delta_i, \Gamma^i, Q_m^i, q_0^i \rangle$, компоненты которого определяются традиционно множествами состояний, событий, функциями переходов и допустимых событий, множеством обязательно достижимых состояний и начальным состоянием соответственно. Язык спецификации $K \subseteq L(G)$ задается автоматом $H = (Q^h, E_d, \delta_h, \Gamma^h, Q_m^h, q_0)$, который генерирует строки из событий $e \in E_d = \{E_c \cup E_{uc}\}$, где $E_c = U_1^n E^i$. Ограничение свободного поведения $G = \langle G^1, G^2, \ldots, G^n \rangle$ с целью выполнения K осуществляет супервизор S, встроенный в контур обратной связи ДСС. Поведение G под контролем S обозначается S/G, а соответствующий язык – $L(S/G)$.

Система, состав неуправляемой части которой задан ДСС с углубленным разделением событий, требуемое поведение которой определено языком спецификаций $K \subseteq E_d^*$ $(K \neq \varnothing)$ и которая реализуется под контролем встроенного в систему супервизора S, в [11] названа ***структурированной дискретно-событийной системой (СДС2)***. (E_d^* – множество всевозможных слов из $e \in E_d = \{E_c \cup E_{uc}\}$ любой конечной длины.)

Реализация K предполагает, что проекция $P_{E_d}(L(S/G)) = K$, т.е. K будет совпадать с проекцией по E_d языка $L(S/G)$, генерируемого объектом G под контролем супервизора S. В [11] сформулированы свойства СДС2 и разработана процедура, названная алгоритмом эксперимента $\Re(\mathrm{s})$, которая является основным приемом исследования совместного поведения G и H. При этом предполагается, что G обладает собственным контроллером, обеспечивающим генерацию управляющих событий, а супервизор лишь обеспечивает блокирование тех управляющих событий, появление которых противоречит спецификации. Для моделирования ДСС с пассивными актуаторами (а САРВ именно такими и являются) в теории ДСС используется модель с *форсируемыми управляемыми событиями*, в которой управляемые события генерирует (форсирует) супервизор [12]. Существование супервизора для структурированных ДСС с форсируемыми событиями исследованы в [13].

С целью сохранения исходного структурирования ДСС при проектировании синтез супервизора в предлагаемой методике осуществляется на сетях Петри. Будем использовать общепринятые обозначения для сетей Петри (см. например, [2], хотя эти обозначения согласуются с более ранними из [14]). Сеть Петри определяется через структуру и разметку.

Структура – это двудольный ориентированный граф $S = (P, T, Pre, Post)$, где P – множество из m позиций, T – множество из n переходов (множества конечны, непусты и в графической форме S изображаются двумя типами вершин: кружками и полочками). $Pre : P \times T \to N$ и $Post : P \times T \to N$ – входная и выходная функции инцидентности переходов. Эти функции специфицируют дуги и их вес (кратность) в структуре двудольного графа S. Функции могут определяться в матричной форме, и тогда $W = (Post - Pre)$ является матрицей инцидентности (изменений) размерности $m \times n$. Для всякого перехода t_i (аналогично для позиций) по функциям инцидентности определяются множества его входных $pre(t_i)$ и выходных $post(t_i)$ элементов.

Вектор $\mu : P \to N$ (вектор разметки сети) каждой позиции *ставит в соответствие* целое положительное число (при графическом представлении сети разметка позиции представляется точками в кружке, которые называют метками или фишками). Количество меток в позиции p_j обозначает $\mu(p_j)$. Если $\mu(p_i) = k \in N$, то говорят, что в позиции $p_i \in P$ имеется k фишек. Переход t_i допусти́м для S на разметке μ_j, если в каждой его входной позиции число меток больше или равно требованиям функции *Pre* на паре (p, t). Формально: $(S, \mu_j) \Leftrightarrow \forall p \in pre(t_i) : \mu(p) \geqslant Pre(p, t)$. Разметка является фундаментальной характеристикой сети S (состоянием), при этом μ_0 – начальная разметка сети S. Сеть Петри[3] – это пара (S, μ_0). Динамика состояний (достижимых разметок) СП определяется срабатыванием (запуском) переходов. Переход t_j срабатывает, если он допустим в μ_i текущей разметке S, и при срабатывании число меток во всех его входных позициях в соответствии с функцией *Pre* уменьшается, а в выходных позициях в соответствии с *Post* – увеличивается. Заметим, что w_{ij} – элемент матрицы W – показывает, какие изменения в позицию p_i вносит переход t_j при срабатывании. Пусть $y(t_j^1)$ – вектор размерности m, у которого в позиции j $y(j) = 1$, а в остальных позициях он равен нулю. Тогда выражение $\mu_k = \mu_i + Wy^{\mathrm{T}}(t_j^1)$ в матричной форме представляет изменения маркировки сети, вызванные срабатыванием перехода t_j. По последовательности срабатывания переходов $s := t_1, t_2, \ldots, t_k$ построим вектор размерности m $y(m)$, значение которого в позиции $y(i)$ равно числу срабатываний перехода t_i в последовательности s. Такой вектор y называют вектором срабатывания или вектором Париха. Тогда в векторной форме маркировка μ_k, достижимая из μ_0 после последовательности s, определяется следующим образом: $\mu_k = \mu_0 + Wy^{\mathrm{T}}$. Часто используют способ задания сети Петри с матрицей инцидентности (различий) $S = (P, T, W, \mu_0)$. При моделировании ДСС переходы взвешиваются именами *событий*.

Следующие два понятия относятся к свойствам структуры сети Петри.

P-инвариантом (*инвариантом позиций*) называют целочисленный вектор $X = \{x_i\}$, $i = 1, 2, \ldots, n$, такой что $x_i \in \{0, 1\}$, и этот вектор является решением линейной системы $XW = 0$. Вектор X единичными значениями характеризует подмножество позиций, в которых при любой достижимой маркировке число фишек постоянно.

T-инвариантом (*инвариантом переходов*) называется целочисленный неотрицательный вектор $Y = \{y_i\}$, $i = 1, 2, \ldots, m$, если он является решением линейной системы $WY^{\mathrm{T}} = 0$. Если для сети S существует T-инвариант Y, то для сети допустима хотя бы одна циклическая последовательность срабатывания s, для которой Y является вектором Париха. Иными словами, T-инвариант определяет наличие циклов и подциклов в сети Петри.

Распространим на непустые подмножества P определения функций *pre* и *post*. Непустое подмножество позиций $S \subseteq P$ называется ***сифоном* (*или тупиком*)**, если $pre(S) \subseteq post(S)$ и – ***ловушкой***, если $post(S) \subseteq pre(S)$. В частности, при $pre(S) = post(S)$ относим S к сифонам. Пустой сифон S – это сифон, для которого $\sum_{p \in S} \mu(p) = 0$. Сифон сохраняет ноль меток. Это значит, что если при какой-то маркировке сифон становится пустым, то это сохраняется при любой другой маркировке, достижимой из данной. Ловушка, напротив, сохраняет хотя бы одну метку, если метка попала в ловушку.

В качестве аппарата анализа и синтеза управления для систем промышленной автоматизации се́ти Петри использовались и раньше. В работах 70-х гг. прошлого века М. Хэка и других авторов (обзор можно найти в [14]) исследовались методы анализа поведения сложных производственных систем на сетях Петри. Структурированные (в том числе иерархические) сети Петри исследовались С. Гушем [15]. В русско-

[3] Далее, там где это не приводит к перенасыщению аббревиатурами, вместо словосочетания "сеть Петри" будем использовать сокращение СП.

язычной литературе известны исследования А.А. Таля и С.А. Юдицкого [16] по иерархическим СП для задачи описания и анализа производственных систем. Для управления промышленными системами в 70–80 гг. XX в. по СП была характерна следующая методология: задана СП S, предполагается, что она определяет алгоритм управления. Задача проектирования состоит в исследовании соответствия S требованиям корректности (безопасность, живучесть и т.п.). При наличии этого соответствия решается задача, как преобразовать алгоритм, заданный S, в проектное решение (программу или логическое устройство).

В ДСС постановка задачи управления кардинально отличается: дана СП S_1, моделирующая ничем не ограниченное поведение объекта управления G. Определены ограничения на поведение G – спецификация K. Задача: существует ли СП S_2, дополняющая S_1 (сеть, встроенная в контур обратной связи) так, что совместно эти сети (S_1 и S_2) удовлетворяют K. Если существует S_2, то каким образом эту сеть синтезировать.

При использовании СП моделирование ДСС и синтез супервизора (см. [8–10] и др.) ориентированы прежде всего на модульный синтез для отдельных классов ДСС. Метод, предложенный в [8], основан на использовании анализа запрещенных состояний, и в общем случае синтез сети супервизора (как дополнение сети процесса) решается методами целочисленного программирования. Методы в [9, 10] основываются на матричном представлении СП и используют технику инвариантов позиций. Эти методы дают хорошие решения для управления ресурсным взаимодействием компонентов ДСС (соблюдение ограничений на число одновременно активных компонентов или задействованных ресурсов), но эта техника неэффективна при задании спецификации в виде порядка срабатывания всех переходов. При разработке САРВ характерно задание спецификации именно в виде требуемой последовательности операций актуаторов, при этом операции циклически повторяются, что соответствует T-инварианту в результирующей сети Петри.

3. Механизм взаимодействия модели СДС2 на сети Петри и объекта управления

Техника, используемая в методе проектирования СП, существенно зависит от принятого механизма взаимодействия модели СДС2 с объектом и внешней средой. В [16] используется механизм, подобный конечно-автоматному механизму, когда выходы соотносятся с позициями, а входные условия – с переходами. Такой механизм неприемлем для СДС2, поскольку неуправляемые события (входы) и управляемые события (выходы) соотносятся с переходами, при этом наличие первых определяется внешней средой, а появление управляемых событий соответствует требуемому управлению. В [14] рассмотрен механизм, предполагающий введение специальных дополнительных входных и выходных позиций, связанных с соответствующими переходами. Однако ДСС требует согласования динамики модели и объекта в определенном порядке, что в [14] отсутствует. Автор предлагает такой механизм взаимодействия, когда структура и функционирование учитывают специфику СДС2 и обеспечивают требуемый порядок взаимодействия. Предлагаемый механизм, по сути, является развитием схемы взаимодействия сети Петри, моделирующей конечный автомат, с внешним миром (см. [14, рис. 3.11]).

Пусть поведение дискретного объекта моделируется ДСС $G = \langle G^1, G^2, \ldots, G^n \rangle$, спецификацией H и выполнен синтез супервизора S с форсируемыми управляемыми событиями, обеспечивающего ограничение поведения дискретного объекта в соответствии со спецификацией. На фазе реализации естественно возникает вопрос о введении механизма взаимодействия модели ДСС и реального дискретного объекта с целью преобразования ограничений на управляемые события в команды (сигналы управления) в соответствии с синтезированной моделью супервизора. Если G^i и H

определены конечными автоматами, то решение довольно естественно: преобразуем G^i (компонентные конечные автоматы (ККА)) и H в управляющие конечные автоматы путем определения для каждого из них функции выходов $\lambda : s(E) \times Q \to E_c$, которая определяет по состоянию и подстроке соответствующий управляющий символ (как это сделано, например, в [13]).

При моделировании ДСС сетью Петри события представляются переходами, а строки событий – последовательностями срабатывания переходов. Таким образом, язык сети Петри S – это множество строк $L(S)$, соответствующих вариантам срабатывания переходов в S (см. например, [2, 5–10]). Если СП моделирует СДС2, то типы событий (управляемое – c, неуправляемое – uc, ожидаемое – w) распространим и на соответствующие этим событиям переходы. На этом основании функцию типа события $\theta(e_i) = \{c|uc|w\}$ определим и на переходах $\theta(t_i) = \{c|uc|w\}$.

В определении сети Петри нет понятия входов, выходов и соответственно функции выходов (как в конечных автоматах). Поэтому в моделирующей системе должен быть определен механизм, отвечающий следующим *требованиям*:

- с одной стороны, при появлении во внешней среде (дискретном объекте) неуправляемых – uc и ожидаемых – w событий механизм должен допускать срабатывание соответствующих переходов в СП,
- с другой – при срабатывании в СП перехода типа c этот механизм должен обеспечивать передачу на объект сигнала, соответствующего управляемому событию.

В ДСС-моделировании сеть Петри, моделирующую G, называют сетью процесса S_p, а компоненту сети, моделирующую супервизор, называют управляющей сетью S_c (сетью контроллера). Допустим, сеть Петри $S = \{S_p \cup S_c\}$ уже сконструирована $S = (P, T, W, \mu_0)$ и S по классификации [14] относится к классу простых (ординарная, без петель), безопасных, живых, не содержит пустых сифонов, но допускающих ловушки с одной меткой. Рассмотрим, как выполнять по этой модели анализ текущего состояния объекта и вычислять управление актуаторами объекта в соответствии с сформулированными выше требованиями. Вычисление будем выполнять циклической процедурой по матричному уравнению $\mu_k = \mu_i + Wy^{\mathrm{T}}(t_j^1)$, где W – матрица инцидентности сети S, μ_i – текущая разметка, $y(t_j^1)$ – единичный вектор для перехода t_j, допустимый в текущей разметке, μ_k – следующая разметка. Очередной цикл вычислений завершается, когда на достигнутой разметке μ_k нет допустимых переходов. В каждом цикле формируется последовательность срабатывания. Система моделирования по множеству управляемых переходов, содержащихся в последовательности срабатывания, будет формировать управляющие воздействия на объект.

Пусть моделирующая ДСС сеть Петри $S = \{S_p \cup S_c\}$ включает $q = |T_p| + |T_c|$ переходов. Расширим СП q дополнительными позициями и установим между переходами и дополнительными позициями взаимно однозначное (1-1) соответствие. Введем ребра между дополнительными позициями и переходами сети $\{S_p \cup S_c\}$ по следующему правилу.

Правило ребер: 1. Все дополнительные позиции p_i, соответствующие переходам t_i типа uc и w, включаются во входные множества $pre(t_i)$ соответствующих переходов;

2. Позиции p_j, соответствующие управляемым переходам t_j, включаются в выходные множества $post(t_i)$ этих переходов. Схематично дополнения представлены на рис. 1.

Упорядочим все дополнительные позиции в векторе V_{iop}: слева (в младших разрядах) сгруппированы все позиции, соответствующие неуправляемым и ожидаемым переходам, а за ними справа сгруппируем позиции, соответствующие управляемым переходам. Введем регистр сигналов R_{s-e}, устанавливающий соответствие факта наличия события и двоичного сигнала $\{0, 1\}$ (нет события e_k – ноль, свершилось событие e_k – единица). Вектор V_{iop} и регистр сигналов R_{s-e} имеют одинаковую

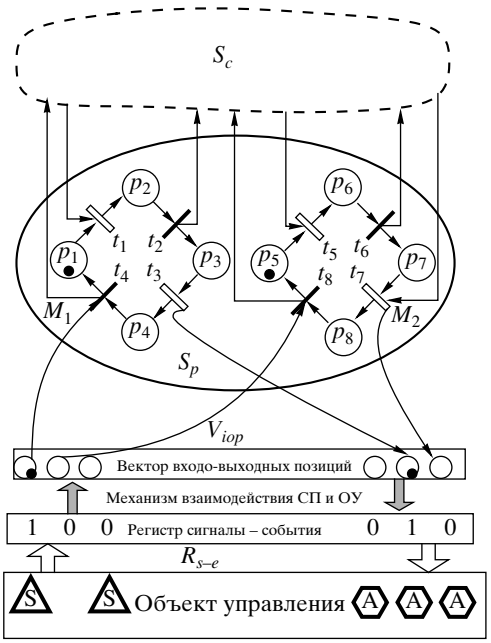

Рис. 1. Механизм взаимодействия СП, ОУ с внешней средой. Обозначения в объекте управления: S – датчики, A – актуаторы.

размерность. Каждый разряд регистра сигналов R_{s-e} (например, k) соотнесен с событием e_k, причем с таким, которое соответствует переходу t_i, имеющему отношение 1-1 с дополнительной позицией, размещенной в разряде k вектора V_{iop}. Установленные соответствия связывают переходы в СП и события в объекте и внешней среде.

Механизм взаимодействия (схематично представлен на рис. 1) содержит:

- сеть $\{S_p \cup S_c\}$
- V_{iop} – вектор входо-выходных позиций, соотнесенных со всеми переходами S_p и переходами t_i из S_c, для которых $\theta(t_i) = uc$;
- R_{s-e} – регистр соответствия факта наличия события и двоичного сигнала $\{0, 1\}$.

Поскольку в дальнейшем исследовании механизм взаимодействия не участвует, ограничимся содержательным изложением его структуры связей с $\{S_p \cup S_c\}$ и правил функционирования. На рис. 1 S_p показан состоящим из двух подсетей, управляемые переходы изображены прямоугольными полочками. Заметим, что подсети S_p соответствуют ККА. По структуре связей можно видеть, что условия срабатывания управляемых переходов формируются в S_p и S_c, затем при срабатывании каждого управляемого перехода в соответствующей позиции V_{iop} появляется метка.

Правила функционирования механизма взаимодействия (МВ).

1. На первом шаге по факту наличия событий на объекте и во внешней среде (события типа uc и w) заполняется левая часть регистра R_{s-e} (записывается единица в разряд k, если соответствующее событие e_k имеет место, и ноль – в противном случае). Затем единичные значения регистра R_{s-e} отображаются в метки в позициях V_{iop}.

2. На втором шаге по матричному уравнению $\mu_k = \mu_i + Wy^{\mathrm{T}}(t_j^1)$ циклической процедурой просчитываются всевозможные срабатывания переходов до достижения устойчивой маркировки (нет допустимых к срабатыванию переходов). Заметим, что

при срабатывании управляемых переходов появляются метки в правой части вектора V_{iop} – вектора входо-выходных позиций (в соответствии с п. 2 правила ребер).

3. На третьем шаге в правой части регистра проставляются единицы и нули в соответствии с наличием меток в векторе входо-выходных позиций и эти метки из правой части вектора убираются. Содержимое регистра R_{s-e} посылается в качестве управляющих сигналов на объект. Далее, при изменении факта наличия события типа uc и w шаги повторяются в бесконечном цикле.

Расширение механизма взаимодействия по сравнению со схемой, рассмотренной в [14], состоит во введении вектора V_{iop}, регистра сигналов R_{s-e} и определении Правил функционирования. Заметим, что структура механизма взаимодействия и его функционирование не зависят от S и МВ необходим только на фазе реализации, поэтому на фазе анализа и синтеза супервизора достаточно ограничиться конструированием S_p и S_c.

4. Синтез сети Петри, моделирующей СДС2

СДС2-моделированием будем называть структурирование объекта управления в виде набора ККА и определение требований к их совместному поведению в виде языка спецификации, определяющего последовательность команд на эти автоматы [11]. Результатом СДС2-моделирования являются:

- набор ККА $G = \langle G^1, G^2, \ldots, G^n \rangle$;
- конечный автомат $H = (Q^h, E_d, \delta_h, \Gamma^h, Q^h_m, q_0)$, определяющий спецификацию K;
- результат проверки условий существования неблокирующего супервизора S [11] и определение $Q = \{Q_1, Q_2, \ldots, Q_r\}$ – множества достижимых векторов-состояний компонентов $G = \langle G^1, G^2, \ldots, G^n \rangle$.

Поскольку все компоненты СДС2-модели – конечные автоматы, можно применить известные подходы к переходу от КА к СП (см. например, [14]). Однако модель ДСС2 определяет иерархию автоматов с пересекающимися алфавитами событий. С позиции управления выполнение команд актуаторами САРВ моделируется множеством ККА, представленных $G = \langle G^1, G^2, \ldots, G^n \rangle$, а порядок следования этих команд определяется в спецификации $H = (Q^h, E_d, \delta_h, \Gamma^h, Q^h_m, q_0)$. Появление события в H требует одновременного его выполнения в соответствующем ККА.

В настоящей работе предлагается задачу синтеза СП, моделирующей СДС2, разделить на задачу синтеза сети процесса S_p по $G = \langle G^1, G^2, \ldots, G^n \rangle$ (моделирование на СП поведения технологического объекта) и задачу синтеза сети контроллера S_c. При этом синтез сети процесса S_p выполняется, как и в [14], путем отображения состояний в позиции, а ребер – в переходы. При синтезе сети контроллера S_c предлагается использовать спецификацию требуемого поведения $H = (Q^h, E_d, \delta_h, \Gamma^h, Q^h_m, q_0)$ как описание T-инвариантов будущей управляемой сети. Исходя из этих предпосылок разрабатывается метод конструирования супервизора.

Сети Петри характеризуются очень бедным набором базовых средств (позиция и переход), поэтому при их использовании для моделирования СДС2 возникают объективные методологические трудности. Базовым понятием технологических данных является технологическая операция. Технологическая операция характеризуется событием (командой на актуатор), изменением состояния объекта и соответствующими откликами (реагированием). Один из вопросов – как технологическую операцию моделировать позициями и переходами. Моделирование объекта как СДС2 включает все события, детально представляющие его поведение. Каждая операция механизма представляется строкой, которая начинается с управляемого события (команды e_i) и далее продолжается ожидаемыми событиями $e_{i1}e_{i2}\ldots e_{in}$. Такая детализация оправдана на фазе анализа на согласованность, но при синтезе управления

достаточно в последовательности отличать каждую операцию от последующих. Следовательно, в СП операцию достаточно представить фрагментом $t_i p_i t_{ir}$, в котором переходы представляют событие-команду и событие–конец операции, а позиция p_i моделирует процесс выполнения технологической операции.

Синтез супервизора основан на анализе допустимых строк в языках $L(G^i)$ и $K \subseteq E_d^*$, заданных компонентными конечными автоматами $G = \langle G^1, G^2, \ldots, G^n \rangle$ и конечным автоматом $H = (Q^h, E_d, \delta_h, \Gamma^h, Q_m^h, q_0)$, определяющим спецификацию. Поиск фрагментов типа $e_i e_{i1} e_{i2} \ldots e_{ir}$ с целью их последующей редукции в $t_i p_i t_{ir}$ естественно выполнять на языке $L(G^i)$, представленном множеством строк, с последующей редукцией строк.

Специфика промышленных объектов такова, что логика отдельных компонентов объекта довольно прозрачна. Графы переходов, определяющие автоматы G_i и H, слабоветвящиеся и, как правило, задают ограниченное число циклически выполняемых последовательностей. Поэтому сами строки и их последовательности достаточно просто извлекаются по результатам анализа последовательности ребер графа переходов. Такие последовательности довольно просто описываются на языке логических схем алгоритмов (ЛСА), хорошо изученном В.Г. Лазаревым и С.И. Барановым [17–19]. Далее излагается модификация ЛСА, ориентированная на поставленную задачу.

4.1. Моделирование СДС2 (компонентных конечных автоматов и спецификации) логическими схемами последовательности строк

В данной работе используется упрощенный вариант ЛСА для описания последовательности строк событий только из методических соображений *упрощения изложения метода синтеза* сети Петри и простоты *приведения иллюстративного материала*. Напомним, что в [17–19] ЛСА – это последовательность символов и исходящих $\uparrow^n$ и входящих $\downarrow^n$ стрелок с индексами, организующих различные последовательности выполнения строк. В СДС2-моделировании под логической структурой последовательности строк (ЛСПС) будем понимать представление графа переходов конечного автомата в виде строк символов состояний и событий этого графа.

Правила преобразования графа переходов в ЛСПС и обратно довольно просты и аналогичны описанным в [17, 18] правилам перехода от граф-схем алгоритмов к ЛСА. Поэтому ограничимся их изложением на концептуальном уровне. Каждая строка ЛСПС соответствует простому пути графа переходов и составляется из имен состояний и событий, взвешивающих ребра в порядке появления этих имен в этом пути. Следование строк соответствует порядку обхода простых путей в графе переходов и представляется стрелками с индексами. Стрелка вверх с индексом $\uparrow^n$ – исходящее ребро, заканчивающее простой путь (после события, взвешивающего это ребро), стрелка вниз с индексом $\downarrow^n$ – входящее ребро (перед состоянием, в которое входит помеченное ранее стрелкой исходящее ребро). Индексы стрелок однозначно связывают начало и конец переходов между строками. Если после некоторого состояния имеет место ветвление (исходящих ребер более одного), то ветвление заканчивает строку, а события, соответствующие этим ребрам, со стрелками вверх заключаются в *фигурные* скобки. Список таких событий будем называть ветвителем. На рис. 2 представлен граф переходов автомата, моделирующего поведение некоторого механизма G^3, стрелками-молниями помечены ребра графа, разрыв которых выделяет простые пути для построения строк ЛСПС. Для графа переходов G^3 ЛСПС: $\downarrow^1 q_1\, e_{3-1}\, q_2\, e_{3-2}\, q_3\, e_{3-3} \downarrow^4 q_4 \{e_{3-8} \uparrow^2, e_{3-4} \uparrow^3\}$; $\downarrow^2 q_{10}\, e_{3-2}\, q_{11}\, e_{3-9} \uparrow^1$; $\downarrow^3 q_5\, e_{3-5}\, q_6\, e_{3-6}\, q_7\, e_{3-7}\, q_8\, e_{3-5}\, q_9\, e_{3-3} \uparrow^4$.

Легко прослеживается связь строки ЛСПС и пути на графе: путь $q_1 \to q_4$ – это первая строка, заканчивающаяся событиями в фигурных скобках. Остальные

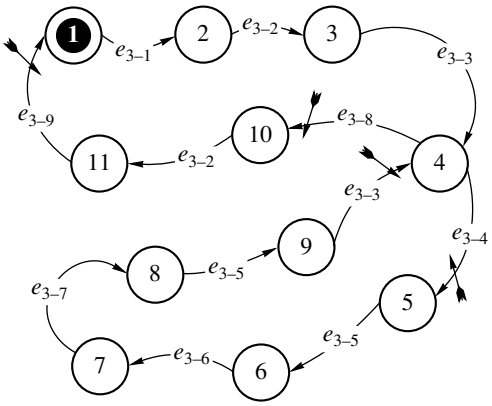

Рис. 2. Граф переходов автомата G^3.

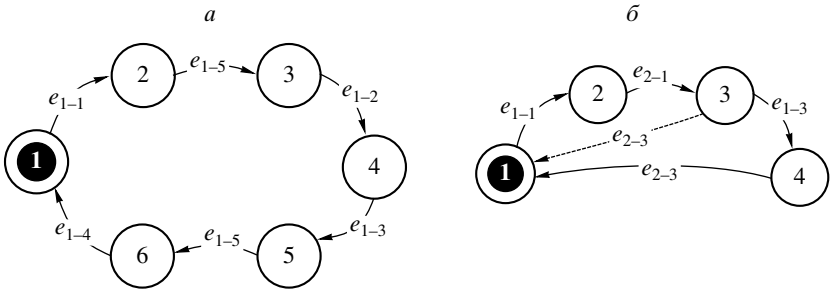

Рис. 3. ККА – модель механизма G^1 ($а$); автомат спецификации H ($б$).

фрагменты графа представлены строками, стрелки которых указывают на связь с первой строкой.

Поясним сказанное на примере устройства перекладчика, состоящего из двух простых однотипных механизмов G^1 и G^2. Граф переходов ККА модели одного механизма представлен на рис. 3,$а$. События e_{1-1}, e_{1-3} соответствуют командам на механизм, а события e_{14}, e_{15}, e_{12} – положениям механизма (левому, промежуточному и правому соответственно). Механизм выполняет циклическую последовательность операций. Второй ККА, совпадающий с первым в точности до обозначений событий (индексы начинаются с двойки), опущен.

Востребованное поведение набора G^1 и G^2 задается в виде последовательности команд из $E_c = U_1^n E_c^i$, представленной строками в языке $K \in E_d^*$. При этом все последовательности команд, которые должны быть отработаны ККА, представляются всевозможными простыми путями графа переходов конечного автомата $H = (Q^h, E_d, \delta_h, \Gamma^h, Q_m^h, q_0)$. Пусть в рассматриваемом примере требуется, чтобы механизмы G^1 и G^2 последовательно выполняли операции вначале первые – e_{1-1}, e_{2-1}, а затем вторые – e_{1-3}, e_{2-3} и эта последовательность циклически повторялась. Спецификация сформулированного совместного поведения двух механизмов представлена на рис. 3,$б$ графом переходов (пунктирную стрелку не учитывать). Перейдем от представления поведения ККА графами переходов к ЛСПС. Для G^1 (см. рис. 3,$а$) ЛСПС имеет вид: $\downarrow^1 q_1 e_{1-1} q_2 e_{1-5} q_3 e_{1-2} q_4 e_{1-3} q_5 e_{1-5} q_6 e_{1-4} \uparrow^1$. ЛСПС для G^2 имеет вид: $\downarrow^1 q_1\, e_{2-1}\, q_2\, e_{2-5}\, q_3\, e_{2-2}\, q_4\, e_{2-3}\, q_5\, e_{2-5}\, q_6\, e_{2-4} \uparrow^1$. Построение ЛСПС по графу спецификации выполняется аналогично. ЛСПС для автомата H будем обозна-

чать LS^H. Напомним, что в автомате H не используются ожидаемые события. Тогда ЛСПС LS^H автомата H, представленного на рис. 3,*б*, имеет следующий вид: $\downarrow^1 q_1\, e_{1-1}\, q_2\, e_{2-1}\, q_3\, e_{1-3}\, q_4\, e_{2-3} \uparrow^1$.

4.2. Редукция логической структуры последовательности строк

Как уже отмечалось, каждая операция механизма представляется строкой, которая начинается с управляемого события (команды e_i) и далее продолжается ожидаемыми событиями $e_{i1}e_{i2}\ldots e_{ir}$. Так, например, для G^1 (см. рис. 3) операция "сдвиг детали" представляется строкой: $e_{1-1}\, q_2\, e_{1-5}\, q_3\, e_{1-2}\, q_4$. Поэтому при моделировании операции сетью Петри достаточно после события-команды e_{1-1} перейти в позицию выполнения операции, а по событию e_{1-2} (последнему из строки ожидаемых событий) перейти в позицию, соответствующую завершению операции. Таким образом, все состояния ККА, соответствующие выполнению операции (в нашем примере это q_2 и q_3), моделируются одной позицией сети, и в строке их достаточно представить последним (в соответствующей подстроке) состоянием ККА и событием, соответствующим завершению операции. Это преобразование определим в виде правила редукции подстрок ЛСПС.

Правило редукции строк ЛСПС. Подстрока из четырех символов типа:

$$< \text{состояние} > i_1, \quad < \text{ожидаемое событие} > j_1,$$
$$< \text{состояние} > i_2, \quad < \text{ожидаемое событие} > j_2,$$

в которой все события относятся к типу ***ожидаемых***, редуцируется в подстроку:

$$< \text{состояние} > i_2, \quad < \text{ожидаемое событие} > j_2.$$

Применим правило редукции к ЛСПС G^1:

$$\downarrow^1 q_1\, e_{1-1}\, q_2\, e_{1-5}\, q_3\, e_{1-2}\, q_4\, e_{1-3}\, q_5\, e_{1-5}\, q_6\, e_{1-4} \uparrow^1 .$$

Для подстроки из четырех символов $q_2\, e_{1-5}\, q_3\, e_{1-2}$ будет иметь место следующее преобразование: $q_2\, e_{1-5}\, q_3\, e_{1-2} \Rightarrow q_2\, e_{1-5}\, q_3\, e_{1-2} \Rightarrow q_3\, e_{1-2}$. В соответствии с правилом редукции строка ЛСПС G^1 последовательно преобразуется:

$$\downarrow^1 q_1\, e_{1-1}\, q_2\, e_{1-5}\, q_3\, e_{1-2}\, q_4\, e_{1-3}\, q_5\, e_{1-5}\, q_6\, e_{1-4} \uparrow^1 \Rightarrow$$
$$\Rightarrow \downarrow^1 q_1\, e_{1-1}\, q_3\, e_{1-2}\, q_4\, e_{1-3}\, q_6\, e_{1-4} \uparrow^1 .$$

С учетом введенных обозначений выражение $G = \langle G^1, G^2, \ldots, G^n \rangle$ соответствует выражению $LS = \{LS^i\}$, а ЛСПС автомата H будем обозначать LS^H.

4.3. Синтез сети Петри, моделирующей технологический объект управления

После редукции ЛСПС задача синтеза сети процесса S_p по $LS = \{LS^i\}$ (моделирование на СП технологического объекта) сводится к замене символов состояний символами позиций, к переобозначению событий переходами и к построению по строкам ЛСПС сети Петри. Если подстрока h встречается в строках ЛСПС LS^i, то обозначим это отношение знаком $\hookleftarrow$, а соответствующую фразу – выражением $h \hookleftarrow LS^i$.

Задача этапа конструирования СП для механизмов решается преобразованием LS^i по следующим правилам.

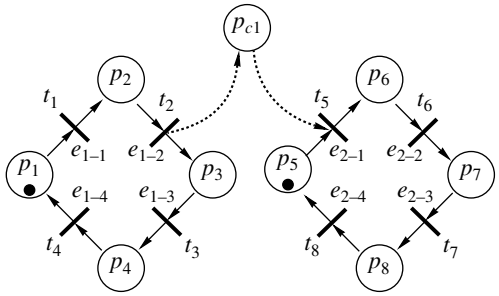

Рис. 4. Сеть Петри процесса для G^1 и G^2.

1. Множество G преобразуется в объединение множества подсетей $G = \langle G^1, G^2, \ldots, G^n \rangle \Rightarrow S_p = \cup_1^n S^i$, при этом каждый $G^i = \langle Q^i, E^i, \delta_i \Gamma^i, Q_m^i, q_0^i \rangle$ преобразуется в $S^i = \{P^i, T^i, Pre^i, Post^i, \mu_0\}$ следующим образом:

$$P^i = \left\{ p_k \left| q_k \leftarrow\!\!\lrcorner\, LS^i \right. \right\}; \quad T^i = \left\{ t_{k-l} \left| e_{k-l} \leftarrow\!\!\lrcorner\, LS^i \right. \right\}.$$

2. Структура переходов (матрицы Pre^i, $Post^i$) определяется через входные и выходные функции переходов и соответствующие подстроки LS^i:

$$pre(t_{k-l}) = \left\{ p_j \left| q_j e_{k-l} \leftarrow\!\!\lrcorner\, LS^i \right. \right\}; \quad post(t_{k-l}) = \left\{ p_j \left| e_{k-l} q_j \leftarrow\!\!\lrcorner\, LS^i \right. \right\}.$$

3. Вектор начальной маркировки сети S_p $\mu_{0p} = [\mu_0^1\ \mu_0^2 \ldots \mu_0^n]$ составляется из блоков μ_0^i, соответствующих позициям компонентных подсетей S_p^i. В каждом блоке μ_0^i в разряде, соотнесенном с q_0^i, содержится единица, а в остальных – ноль.

Для механизмов G^1 и G^2 сеть имеет вид, представленный на рис. 4. Построим по автомату H соответствующую ЛСПС LS^H и осуществим ее расширение: каждое управляемое событие e_{k-i} заменим парой $e_{k-i}e_{k-j}$, где e_{k-j} – событие, завершающее операцию e_{k-i} в соответствующем LS^i. ЛСПС автомата H – LS^H приобретает следующий вид: $\downarrow^1 q_1\ e_{1-1}\ e_{1-2}\ q_2\ e_{2-1}\ e_{2-2}\ q_3\ e_{1-3}\ e_{1-4}\ q_4\ e_{2-3}\ e_{2-4} \uparrow^1$.

4.4. Исследование свойств сетей Петри, моделирующих СДС2

Введем необходимые определения. Далее в тексте все события e_{i-j} заменим именами переходов t_k и пронумеруем по мере их следования в S_p, например, как это показано на рис. 4.

Замечание 1. В соответствии с определением правил срабатывания переходов в СП всякий переход срабатывает, если он допустим в текущей разметке, т.е. во всех его входных позициях достаточно меток. Пусть подстрока $t_j t_k$ входит в LS^H и $\theta(t_j) = \{uc|w\}$, а $\theta(t_k) = \{c\}$. Задача сети S_c: отреагировать на срабатывание в соответствующих фрагментах сети S_p переходов типа uc и w (в принятом допущении – это $t_{j,}$) подготовкой меток во входном множестве t_k так, чтобы после срабатывания t_j переход t_k стал допустимым для сформировавшейся разметки. Это необходимо, поскольку подстрока $t_j t_k$ востребована в LS^H.

Определение 1. Пусть сеть S_p моделирует объект (процесс) в СДС2 с форсируемыми управляющими событиями и в моделируемой СДС2 должна выполняться LS^H. Тогда переходы t_j, t_i сети S_p называются несвязанной парой (н-парой), если подстрока $t_j t_i$ входит в LS^H ($t_j t_i \leftarrow\!\!\lrcorner\, LS^H$), и в сети S_p не существует позиции p_k такой, что $t_j \in pre(p_k)$, а $t_i \in post(p_k)$.

Следствие 1. Если t_j, t_i являются н-парой, и $t_j \in T^k$, а $t_i \in T^l$, то $k \neq l$. Иными словами, переходы t_j и t_i принадлежат различным подсетям S^k и S^l сети S_p.

Определение 2. Пусть фрагмент F сети S содержит единственный простой путь $t_i, \ldots, t_j$ такой, что только для его начала t_i и конца t_j выполняется: тип $\theta(t_i), \theta(t_j) \in \{c, u, uc\}$, тогда F называется структурой прямого взаимодействия (СПВ) переходов t_i и t_j.

Лемма 1. Пусть сети процесса S_p и S_c моделируют СДС2 с форсируемыми управляемыми событиями так, что выполняется LS^H, и переходы t_j, t_i являются н-парой в S_p. Тогда в управляющей сети S_c (в супервизоре) существует СПВ из непосредственно связанных компонент $t_j \to p_c \to t_i$.

Доказательство необходимости (от противного). Допустим, выполняются условия леммы 1 и подстрока $h := t_j t_i$ входит в LS^H ($h \leftarrow\!\!\!\lrcorner\, LS^H$), но в S_c (управляющей сети СДС2) нет СПВ из непосредственно связанных компонент $t_j \to p_c \to t_i$ (далее просто СПВ $t_j p_c t_i$). Поскольку t_j, t_i являются н-парой, то такой СПВ нет и в S_p. Поскольку в объединенной сети должно выполняться основное свойство сетевой структуры (СП – двудольный граф), то последовательность срабатывания t_j, t_i не выполняется в объединенной сети и не может входить в LS^H. Приходим к противоречию.

Доказательство достаточности. Допустим $u := r t_j t_i v$, $u \in LS^H$, и u есть начальная подстрока LS^H, переходы t_j, t_i образуют н-пару в сети процесса S_p, а сеть S_c конструируется последовательно и в ней уже выполняется подстрока r. Расширим матрицу инцидентности W_p дополнительной строкой p_c. Определим элементы дополнительной строки матрицы следующим образом: $W(p_c, t_j) = +1, W(p_c, t_i) = = -1$, все остальные элементы строки равны нулю. Тем самым выполнено преобразование сети S_p в S'_p путем расширения S_p *структурой прямого взаимодействия* $t_j p_c t_i$ так как это показано на рис. 4, и в этой сети становится немедленно допустимой строка срабатывания $r t_j t_i$, содержащая подстроку $h := t_j t_i$. Поскольку LS^H содержит все только допустимые строки срабатывания, то введение *структуры прямого взаимодействия* $t_j p_c t_i$ является достаточным преобразованием S_p. Это и требовалось доказать.

Следствие 2. Поскольку LS^H и сети процесса S_p моделируют СДС2 с форсируемыми управляющими событиями, которые по определению инициирует (разрешает) только супервизор, то из этого следует важное свойство СПВ $t_j p_c t_i$: первый переход в $t_j p_c t_i$ всегда соответствует ожидаемому или неуправляемому событию (не может быть взвешен управляемым событием), а второй переход в $t_j p_c t_i$ соответствует управляемому или неуправляемому событию (не может быть взвешен ожидаемым событием).

Сформулированное следствие иллюстрирует табл. 1.

Таблица 1. Допустимые сочетания типов переходов в н-парах и СПВ

1-е событие	2-е событие
w	c
w	uc
uc	c
uc	uc

Следствие 3. При дополнении сети процесса S_p цепочкой $t_j p_c t_i$ (структурой передачи управления) получается сеть S'_p. При этом P-инварианты сети S_p со-

храняются и в сети S'_p (преобразование иллюстрирует рис. 3, цепочка СПВ $t_j p_c t_i$ выделена пунктиром). Однако множество достижимых состояний, несомненно, изменится. Более того, сеть из ограниченной превращается в неограниченную, поскольку в позиции p_c возможен неограниченный рост числа меток при бесконечном цикле в первой сети.

Лемма 2. Для всякой н-пары t_j, t_i скалярное произведение векторов–столбцов $W(t_i) \bullet W(t_j) = 0$.

Доказательство. Все подсети S_P^k сети S_p автономны и не пересекаются ни по позициям, ни по переходам. Сгруппируем позиции и переходы каждой S_P^k компактно в W_p (друг за другом в столбцах и строках), тогда подматрицы W_P^k расположатся по диагонали матрицы инцидентности процесса W_p. Отсутствие в S_p позиции, связывающей переходы t_j, t_i, свидетельствует о принадлежности t_j и t_i к различным S_P^i, поэтому в пересечениях столбцов, соответствующих t_j и t_i, со всеми строками W нет отличных от нуля значений. Из этого следует, что скалярное произведение векторов–столбцов $W(t_i) \bullet W(t_j) = 0$. Что и требовалось доказать.

Лемма 3. Пусть расширение сети S_p осуществляется множеством СПВ, сконструированных для всех н-пар последовательности u, входящей в LS^H. Тогда существует изоморфное отображение $M' \Rightarrow Q$, где M' – множество достижимых маркировок сети S'_p, а $Q = \{Q_1, Q_2, \ldots, Q_r\}$ – множество достижимых векторов-состояний компонентов $G = \langle G^1, G^2, \ldots, G^n \rangle$.

Доказательство. Существование изоморфного отображения $M' \Rightarrow Q$ следует из того, что, во-первых, $G = \langle G^1, G^2, \ldots, G^n \rangle$ и $H = (Q^h, E_d, \delta_h, \Gamma^h, Q_m^h, q_0)$ согласованы; во-вторых, $G = \langle G^1, G^2, \ldots, G^n \rangle \Rightarrow S_p = \bigcup_{i=1}^{n} S^i$; множество позиций S_p конструируется как отображение множества состояний $G = \langle G^1, G^2, \ldots, G^n \rangle$ на множество позиций сети (переобозначение). Наконец, в-третьих, правила определения структуры переходов S_p сформулированы через входные и выходные функции переходов и соответствующие подстроки LS^i, поэтому они соответствуют функциям переходов автоматов G^i. Из этого следует, что срабатывание переходов в сети $\langle S_p \cup S_c \rangle$ соответствует допустимой последовательности $Q_i, Q_j, \ldots, Q_r$ векторов-состояний компонентов $G = \langle G^1, G^2, \ldots, G^n \rangle$. Таким образом, сохранение порядка следования при отображении M' на Q определяет его изоморфизм.

Теорема 1. Пусть $S = S_p \cup S_c$, а S_c определяется как расширение сети S_p множеством структур передачи взаимодействия, сконструированных для всех н-пар из последовательностей срабатывания u, входящих в LS^H. Тогда вектор Парика $Y(LS^H)$, соответствующий u, является T-инвариантом сети S и, кроме того, P-инварианты сети S_p сохраняются в S.

Доказательство. T-инвариантность вектора Парика $Y(LS^H)$, соответствующего u, является следствием того, что $H = (Q^h, E_d, \delta_h, \Gamma^h, Q_m^h, q_0)$ задает связный автомат, поэтому LS^H, сконструированная по конечному автомату H, описывает циклический процесс. Справедливость того, что P-инварианты сети S_p сохраняются в S'_p, следует из справедливости утверждений леммы 1, леммы 3 и следствия 1. Что и требовалось доказать.

Замечание 2. Из утверждений лемм, теоремы и сформулированных свойств следует, что если для всех возможных последовательностей строк, задаваемых, например, ЛСПС, выявить все н-пары t_j, t_i и каждую пару в сети процесса обеспечить структурой передачи взаимодействия $t_j p_c t_i$, то:

- во-первых, все структурные свойства исходной сети процесса сохраняются;

- во-вторых, все требуемые последовательности операций, определяемые спецификацией, будут выполняться.

4.5. Синтез управляющей сети по S_p и LS^H

Идея предлагаемого метода основана на следующем свойстве циклических реактивных систем с форсируемыми событиями: каждая операция объекта выполняется по событию, завершающему предшествующую операцию (подобно падению костяшек домино), или по внешнему событию (например, при пуске или выборе). В этом случае задача супервизора – организовать передачу управления от механизма, выполнившего операцию, к механизму, операция которого по LS^H должна выполняться следующей. Метод синтеза управляющей сети Петри назван методом домино. Опишем базовый метод, ограничивающий LS^H бесповторной[4] ЛСПС, содержащей одну последовательность.

Процедура построения управляющей сети S_c (базовый алгоритм). Обозначения, используемые в тексте процедуры: Sintez – имя процедуры; W, n, u, r – формальные параметры; $u := t_1t_2t_3\dots t_{r-1}t_r$ – строка ЛСПС, обрабатываемая процедурой (в u стрелки заменены именами переходов, на которые эти стрелки указывают в ЛСПС); r – длина строки u; W – $(n \times m)$-матрица инцидентности сети процесса S_p; $W(k, \sim) = 0$ означает присвоение всем элементам строки k матрицы W нулей; $u(i)$ – i-й символ строки u; $W(u(i))$, $W(u(i+1))$ – векторы-столбцы матрицы W; $\mathrm{SProdct}(W(u(i)), W(u(i+1)))$ – скалярное произведение векторов.

Представим процедуру на языке Паскаль:

```
Sintez (W, Pc, n, u, r)
i=1
While i≤(r-1) do
Begin
If SProdct ( W(u(i)), W(u(i+1)))≠ 0 go to M1;
n=n+1;
W(n, ∼)=0;
W(n, u(i))=+1;
W(n, u(i+1))=-1;
Pc= {Pc}∪{p_n};
M1: i=i+1;
End.
```

Применим процедуру Sintez к сети (см. рис. 3,*а*) для двух механизмов G^1 и G^2. Матрица инцидентности W этой сети представлена в верхней части табл. 2 (строки 1–8). В матрице W имена событий заменены именами соответствующих переходов следующим образом: $(e_{1-1} \to t_1)\dots(e_{2-4} \to t_8)$. Обрабатываемая строка $u := t_1t_2t_5t_6t_3t_4t_7t_8t_1$; $r = 9$.

Результатом работы процедуры Sintez является построение сети из четырех управляющих позиций $(p_{c1}, \dots, p_{c4})$ и соответствующих связей, что отражено в дополнительных строках $p_{c1}, \dots, p_{c4}$ матрицы инцидентности W. Построенная сеть представлена на рис. 5.

Вычисление μ_0. Вектор начальной маркировки состоит из начальной маркировки сети процесса и начальной маркировки сети контроллера $\mu_0^{\mathrm{T}} = [\mu_{0P}^{\mathrm{T}}\mu_{0C}^{\mathrm{T}}]$. Начальная маркировка сети процесса μ_{0p}^{T} определяется при построении и соответствует на-

[4] В бесповторной последовательности срабатывания каждый переход упоминается не более одного раза. Далее это ограничение будет снято.

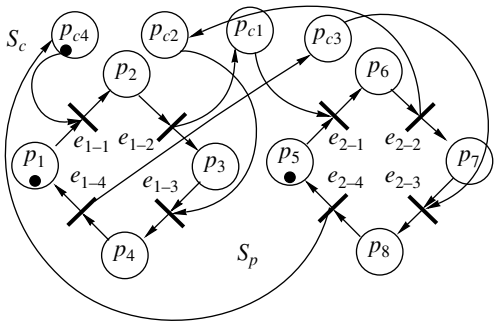

Рис. 5. Сеть Петри с управляющими позициями $p_{c1}, p_{c2}, p_{c3}, p_{c4}$, реализующими управление, так что выполняется последовательность $u := t_1t_2t_5t_6t_3t_4t_7t_8t_1$.

чальным состояниям ЛСПС (в позициях вектора μ_{0p}^{T}, соответствующих начальным состояниям в определении ЛСПС, присваивается единица, а в остальных позициях – ноль). Для рассматриваемого примера $\mu_{0p}^{\mathrm{T}} = 10001000$.

Правило для μ_{0p}^{T} – начальной разметки управляющей сети S_c. Конструируем по u вектор Париха Y. Поскольку вектор Y является T-инвариантом (задает цикл), то если из него исключить срабатывание последних в цикле переходов, матричная операция W с модифицированным Y' определит начальную разметку для S_c. Определим модифицированный вектор как $Y' = Y(-1)$. В Y' уменьшены на единицу значения в позициях вектора Y, соответствующих переходам из н-пар, вторыми компонентами в которых является t_1 из ЛСПС (первый переход в ЛСПС). В приведенном примере – это t_8. Тогда $\mu_{0C}^{\mathrm{T}} = -WY'$ для позиций, относящихся к сети контроллера. Для рассмотренного примера $\mu_{0C}^{\mathrm{T}} = 0001$. В итоге, $\mu_{0p}^{\mathrm{T}} = 100010000001$.

Таблица 2. Матрица инцидентности W для СП механизмов G^1 и G^2

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
1	−1	0	0	1	0	0	0	0
2	1	−1	0	0	0	0	0	0
3	0	1	−1	0	0	0	0	0
4	0	0	1	−1	0	0	0	0
5	0	0	0	0	−1	0	0	1
6	0	0	0	0	1	−1	0	0
7	0	0	0	0	0	1	−1	0
8	0	0	0	0	0	0	1	−1
p_{c1}	0	+1	0	0	−1	0	0	0
p_{c2}	0	0	−1	0	0	+1	0	0
p_{c3}	0	0	0	+1	0	0	−1	0
p_{c4}	−1	0	0	0	0	0	0	+1

4.6. Улучшение S_c путем минимизация каналов управления

Процедура синтеза вводит позицию, передающую управление, для каждой н-пары подобно каналу связи. Эти каналы "работают" только в момент активности соответствующей н-пары, а после передачи управления в следующие переходы не участвуют в процессе активизации других переходов, пока эти каналы не будут вновь востребованы в следующем цикле. Из этого следует, что формирование для каждой н-пары

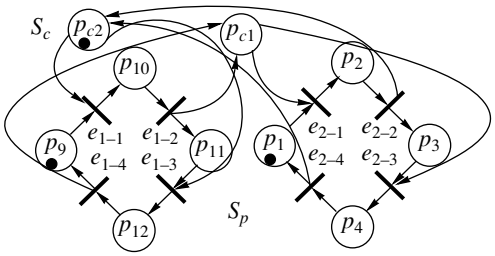

Рис. 6. Преобразованная сеть Петри.

нового канала в сети во многих случаях избыточно. Прозрачно определяются отношения совместимости каналов (по отсутствию пересечения предшественников и последователей) и соответствующие правила, подобные правилам совместимости состояний для конечного автомата. Применив эти правила и проделав традиционные преобразования, в рассмотренном примере удалось сократить число позиций в два раза. Результат представлен на рис. 6.

5. Заключение

В статье разработан базовый метод синтеза супервизорного управления на сетях Петри по СДС2 – модели распределенного объекта. Предложен механизм взаимодействия моделирующей и управляющей сетей Петри с объектом и внешней средой. Механизм взаимодействия, по сути, является схемой управления объектом по сконструированной сети $S = (P, T, W, \mu_0)$. Схема управления выполняет анализ текущего состояния объекта и вычисляет управление актуаторами объекта. Вычисления осуществляются циклической процедурой по матричному уравнению $\mu_k = \mu_i + Wy^{\mathrm{T}}(t_j^1)$ для сети S. В каждом цикле формируется последовательность срабатывания. Процедуры механизма взаимодействия по множеству управляемых переходов, содержащихся в последовательности срабатывания, формируют управляющие воздействия на объект. Схема управления поддержана методом синтеза по СДС2-модели сети Петри S_p, моделирующей процессы в объекте, техникой анализа S_p и базовым алгоритмом синтеза управляющей сети супервизора S_c, обеспечивающей совместно с сетью S_p выполнение исходных спецификаций. Базовый алгоритм синтеза управляющей сети S_c основан на анализе ЛСПС – компактной модели последовательности срабатывания переходов в сети процесса.

Разработанные алгоритмы синтеза S_p и S_c работают с одной бесповторной последовательностью команд, заданных ЛСПС, и составляют базовый метод синтеза СП-модели рассредоточенного объекта по его СДС2-модели. В представленной к публикации второй статье будет предложено развитие базового метода для множества последовательностей срабатывания в LS^H без ограничения на повторность событий в последовательностях.

СПИСОК ЛИТЕРАТУРЫ

1. *Ramadge J.G., Wonham W.M.* Supervisory control of a class of discrete event processes // SIAM J. Control Optim. 1987. № 25. P. 206–230.
2. *Cassandras C.G., Lafortune S.* Introduction to discrete event systems. Dordrecht: Kluwer Academic Publishers, 2008.
3. *Wonham W.M., Ramadge J.G.* Modular supervisory control of discrete event systems // Math. Control Signal. Syst. 1988. No. 1. P. 13–30.
4. *Yoo T.-S., Lafortune S.* A general architecture for decentralized supervisory control of discrete event systems // Discrete Event Dynam. Syst.: Theory& Applications. 2002. No. 12(3). P. 335–347.
5. *De Queiroz M.H., Cury J.E.R.* Modular supervisory control of large scale discrete event systems // Discret. Event Syst.: Anal. Control. Proc. WODES. 2000. P. 103–110.
6. *Akesson K., Flordal H., Fabian M.* Exploiting modularity for synthesis and verification of supervisors // Proc. IFAC World Congr. Barcelona, Spain. 2002. P. 1452–1457.
7. *Gaudin B., Marchand H.* Modular supervisory control of asynchronous and hierarchical finite state machines // Europ. Control Conf. Cambridge. 2003. P. 542–547.
8. *Holloway L.E., Krogh B.H.* Synthesis of feedback logic for a class of controlled Petri nets // IEEE Trans. Autom. Control. 1990. AC-35. No. 5. P. 514–523.
9. *Yamalidou K., Moody J.O., Lemmon M.D., Antsaklis P.J.* Feedback control of Petri nets based on place invariants // IEEE Trans. Robot. Automat. 1996. No. 32(1). P. 15–28.
10. *Dideban A., Alla H.* Reduction of Constraints for Controller Synthesis based on Safe Petri Nets // Automatica. 2008. No. 44(7). P. 1697–1706.
11. *Амбарцумян А.А.* Супервизорное управление структурированными динамическими дискретно-событийными системами // АиТ. 2009. № 8. С. 156–176.
12. *Golaszewski C.H., Ramadge P.J.* Control of discrete event processes with forced events // Proc. 28 Conf. on Decision and Control. Los Angeles. 1987. P. 247–251.
13. *Амбарцумян А.А., Томилин Е.Е.* Метод прямого синтеза супервизора для структурированных дискретно-событийных систем // АиТ. 2010. № 8. С. 168–188.
14. *Petrson J.L.* Petri Net theory and the modeling of systems. Prentice-Hall, Inc. Englewood Cliffs, N.J., 1981.
15. *Ghosh S.* Structured Petri Nets. Report 49/77, Lehrstuhl Informatik, Universitat Dortmund, West Germany. August 1977. P. 27.
16. Таль А.А., Юдицкий С.А. Иерархия и параллелизм в сетях Петри //АиТ. 1982. № 7. С. 121–139; 1982. № 8. С. 111–128.
17. *Лазарев В.Г., Пийль Е.И.* Синтез управляющих автоматов. М.: Энергоатомиздат, 1989.
18. *Baranov S.* Logic and System Design of Digital Systems // Published Jointly Tallinn Univ. Technol. Press and SIB Publishers (Toronto). 2008.
19. *Закревский А.Д., Поттосин Ю.В., Черемисинова Л.Д.* Логические основы проектирования дискретных устройств. М.: Физматлит, 2007.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Поступила в редакцию 02.08.2010