УДК 519.711.3

ИЕРАРХИЧЕСКАЯ ДЕКОМПОЗИЦИЯ СОБЫТИЙНЫХ ГРАФОВ

© 2010 Е. А. Бабкин

проф. каф. программного обеспечения
и администрирования информационных систем,
канд. тех. наук, доц.
e-mail: eababkin@gmail.com

Курский государственный университет

Рассматривается расширение событийных графов введением дуг, помеченных метками спецификаций. Эти дуги отображают транзитивные зависимости следования событий и позволяют реализовать многоуровневые иерархические спецификации событийных моделей дискретных систем. Иерархическая декомпозиция событийного графа, являющаяся разновидностью объектной декомпозиции, основана на выделении описаний событийных графов иерархически вложенных процессов. Такая декомпозиция позволяет строить иерархические спецификации событийных моделей. Предлагается для наглядного представления иерархических моделей форма макрособытийного графа с дорожками, в которой каждая дорожка представляет процесс. Рассматривается реализация иерархической программной модели.

Ключевые слова: дискретная система, событие, макрособытие, событийная модель, событийный граф, макрособытийный граф, иерархическое представление.

1. Введение

Событийные графы являются удобным средством описания поведения дискретной системы (ДС) в терминах событий [Schruben 1983; Бабкин 1988; Бабкин 2005]. Однако недостатком событийно-ориентированного подхода считается невозможность многоуровневого иерархического представления модели и ее программной реализации [Брейер 1979]. Действительно, поскольку событие определяется как мгновенное изменение состояния системы или внешней среды, можно выполнять спецификации либо на уровне элементарных событий, либо на уровне макрособытий [Бабкин 2005]. Объединение наборов событий в макрособытия имеет следующие ограничения:

- события должны быть связаны,
- не должно быть задержек между событиями, то есть следование событий должно быть мгновенным.

В связи с этим в работах по событийным графам рассматривается либо только одноуровневое [Schruben 1983], либо двухуровневое [Бабкин 2005] событийное представление модели и одноуровневая ее программная реализация. Для процессо-ориентированного представления таких ограничений нет [Брейер 1979]. В работе [Бабкин 2009] рассмотрены возможности неограниченного по числу уровней иерархического представления событийных моделей за счет расширения событийных графов введением дуг с метками спецификаций и использования иерархического представления процессов и активностей. В настоящей работе рассматривается развитие подхода, предложенного автором ранее.

2. Основные элементы иерархического представления событийных графов

Основным видом вершин в событийном графе являются вершины-события [Schruben 1983]. В форме графа, рассматриваемой в работах [Бабкин 1988; Бабкин

2005], используются также вспомогательные вершины, отображающие следование событий: условные, распараллеливания и окончания распараллеливания процесса. Дуги в этой форме отображают причинно-следственные зависимости между событиями и вспомогательными вершинами. Возможны дуги трех типов: первого типа – мгновенного следования событий (рис. 1а), второго типа – следования событий с задержкой T_{ij} (рис. 1б), третьего типа – отмены событий (рис. 1в). Если выполнение события e_i приводит непосредственно к выполнению события e_j, то это представляется либо дугой первого типа при мгновенном следовании событий, либо дугой второго типа при следовании событий с задержкой [Бабкин 1988]. В первом случае время следования равно нулю ($T_{ij} = 0$). Во втором случае время следования T_{ij} может быть определено как детерминированная или стохастическая функция. Если выполнение события e_i приводит непосредственно к отмене ранее запланированного события e_j, то это представляется дугой третьего типа. В этом случае отмена события e_j происходит мгновенно после выполнения события e_i.

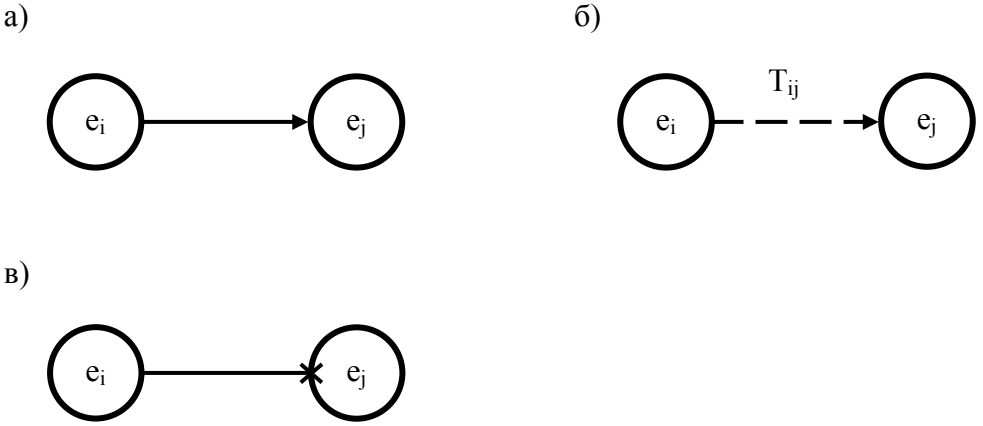

Рис. 1. Непосредственные причинно-следственные зависимости между событиями e_i и e_j:
а) мгновенное следование события e_j после e_i,
б) следование с задержкой T_{ij} события e_j после e_i,
в) отмена события e_j после выполнения e_i

На начальных этапах формализации существует необходимость отображать в графе наличие причинно-следственной связи между событиями с неизвестным на данном этапе временем следования или/и с неизвестным условием следования событий. В дальнейшем в ходе формализации определяются эти характеристики причинно-следственных зависимостей.

Введем для представления неизвестной зависимости новую разновидность дуги – помеченную дугу. На рисунке 2 представлены два случая следования событий с неизвестным условием следования и с неизвестным условием и временем следования.

Рис. 2. Следование событий e_i и e_j:
а) мгновенное следование с неизвестным условием следования,
б) следование с задержкой с неизвестным условием и временем следования

Будем использовать помеченную дугу также для отображения транзитивной зависимости следования событий. При иерархическом подходе к разработке и представлению модели такая дуга позволяет детализировать зависимости следования на следующем, более низком уровне представления модели. На рисунке 3 представлены два случая следования событий. В первом случае (рис. 3а) метка ***{Mij}*** указывает на дополнительную спецификацию условия следования. Во втором случае (рис. 3б) метка ***{Mij}*** указывает на дополнительную спецификацию и условия и времени следования.

а) б)

Рис. 3. Следование событий e_i и e_j:
а) мгновенное следование с условием специфицированным в M_{ij},
б) следование с задержкой со спецификацией в M_{ij}

Дуга второго типа, помеченная меткой спецификации ***{Mij}***, представляет на данном уровне сложную активность с событиями начала e_i и конца e_j активности, которая на следующем более низком уровне представляется процессом и описывается событийным графом. Причем процесс, детализирующий активность, имеет те же события начала e_i и конца e_j процесса, что и детализируемая активность. Таким образом, помеченная дуга отображает сложные активности-процессы, специфицируемые отдельными событийными графами.

В работе [Бабкин, Бобрышев 2008] рассматриваются два способа декомпозиции событийного графа: структурный и объектный. Разновидностью объектной декомпозиции является иерархическая декомпозиция событийного графа [Бабкин 2009]. Иерархическая декомпозиция событийного графа основана на выделении событийных подграфов иерархически вложенных процессов. Такая декомпозиция позволяет строить иерархические спецификации событийных моделей.

Иерархия моделей функционирования определяется иерархией статических объектов и, соответственно, процессов их функционирования, существующей в рассматриваемой ДС. Процесс представляет собой связанную совокупность активностей, то есть элементарных действий, неделимых на данном уровне представления модели. С другой стороны, процессы можно рассматривать как взаимосвязанные совокупности событий. В связи с этим в соответствии с иерархией процессов множество событий модели разбивается на группы событий, специфицирующих соответствующие процессы. Спецификация имитационной модели в целом состоит из следующих основных компонентов, имеющих ссылки друг на друга: событийных графов и фрагментов текста, описывающих события и макрособытия.

3. Пример иерархической декомпозиции событийной модели

Рассмотрим использование помеченных дуг для иерархического представления событийных графов на примере модели двухфазной сети массового обслуживания (СтМО) (рис. 4), состоящей из трех систем массового обслуживания $СМО_1$, $СМО_2$ и $СМО_3$. Транзакты поступают из источника $И$ – входной поток сети W. Они имеют различные значения атрибутов приоритета и номера СМО второй фазы, в которой они обслуживаются. $СМО_u$ состоит из канала $К_u$ и накопителя $Н_u$. Емкости накопителей ограничены, поэтому с накопителями связаны выходные потоки транзактов W_1, W_2 и

W_3, получивших отказ в обслуживании в связи с конечным размером соответствующего накопителя. W_4 и W_5 – выходные потоки транзактов, обслуженных в СтМО.

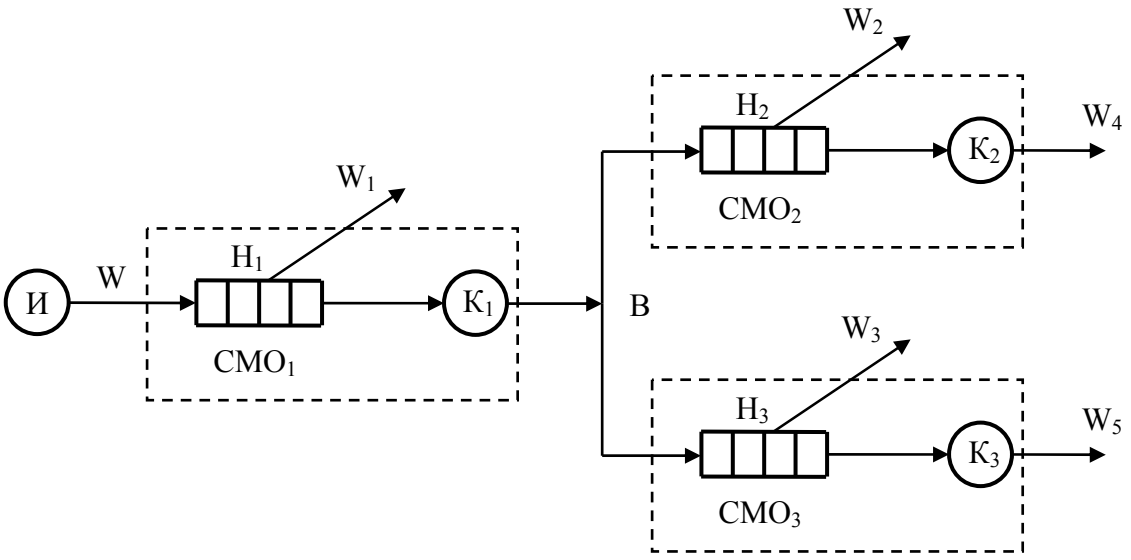

Рис. 4. Сеть массового обслуживания

Процесс имитационного моделирования состоит из ожиданий поступления транзактов и из активностей ***a*** обслуживания этих транзактов. Временная диаграмма процесса имитационного моделирования приведена на рисунке 5.

Рис. 5. Временная диаграмма процесса имитационного моделирования

Сначала выполним описание процесса имитационного моделирования на высшем уровне представления. На этом уровне процесс имитационного моделирования представим в виде простейшего компонента-процесса, состоящего из одной активности ***PM*** (рис. 6).

На этом графе:

$e_{нPM}$ – событие начала активности (процесса) имитационного моделирования ***PM***,

$e_{кPM}$ – событие конца активности (процесса) имитационного моделирования ***PM***,

{PM} – метка спецификации процесса ***PM***.

Будем различать простые и сложные активности. Простые активности не разделяются на следующих уровнях на составляющие активности, то есть они не представляются процессами на других низших уровнях представления и полностью специфицируются на данном уровне представления. Сложные или составные

активности представляются на следующем нижнем уровне в виде процесса и специфицируются отдельным событийным графом.

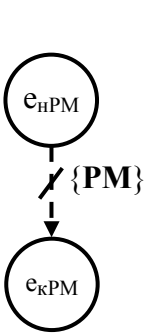

Рис. 6. Спецификация верхнего уровня представления имитационной модели

Процесс имитационного моделирования, содержащий одну сложную активность ***PM***, детализируем на следующем событийном графе (рис. 7), представляющем ожидания транзактов и активности обслуживания транзактов в СтМО в соответствии с временной диаграммой (см. рис. 5).

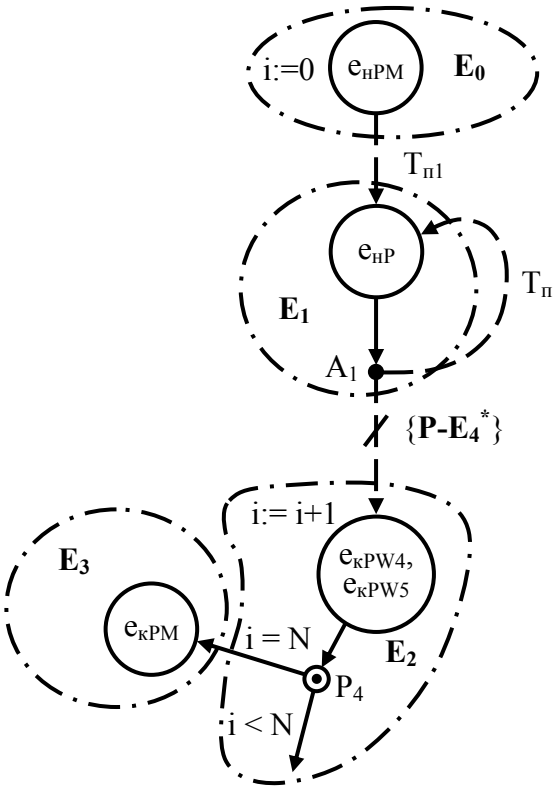

Рис. 7. Спецификация процесса моделирования ***PM***

На этом графе детализируется активность ***PM*** и, соответственно, причинно-следственная зависимость между событиями $e_{нРМ}$ и $e_{кРМ}$. Здесь:

$T_{п1}$ – время ожидания поступления первого транзакта;

$T_{п}$ – время ожидания поступления следующего транзакта;

i – число транзактов, обслуженных в СтМО;

N – число транзактов, которые должны быть обслужены в СтМО, определяет длительность имитационного эксперимента;

P – активность (процесс) обслуживания транзакта в СтМО;

A_1 – начало параллельного выполнения процесса;

$e_{нP}$ – событие начала активности (процесса) обслуживания транзакта в СтМО;

$e_{кPN4}$ – событие конца активности (процесса) обслуживания транзакта в СтМО (выходной поток N_4);

$e_{кPN5}$ – событие конца активности (процесса) обслуживания транзакта в СтМО (выходной поток N_5);

P_4 – условие окончания процесса моделирования (по числу заявок $i = K$).

В событийном графе на основе подхода, изложенного в [Бабкин 2005], выделяются следующие макрособытия:

E_0 – начало процесса моделирования;

E_1 – начало активности (процесса) обслуживания транзакта в СтМО;

E_2 – конец активности (процесса) обслуживания транзакта в СтМО;

E_3 – конец процесса моделирования.

Кроме того:

E_4* – группа событий процесса ***P***;

$\{P\text{-}E_4^*\}$ – метка спецификации активности (процесса) обслуживания ***P*** с указанием номера соответствующей группы событий.

Активность обслуживания ***P*** является сложной активностью и может быть представлена в виде процесса, состоящего из последовательности активностей обслуживания транзакта в CMO_1 и в CMO_2 или CMO_3 (рис. 8).

Рис. 8. Временная диаграмма обслуживания транзакта в сети массового обслуживания

Этой временной диаграмме соответствует спецификация процесса ***P*** обслуживания транзакта в СтМО в виде событийного графа (рис. 9), которая детализирует причинно-следственную зависимость между событиями $e_{нP}$ и $e_{кP}$. В этой спецификации параллельные однотипные участки СтМО, а именно CMO_2 и CMO_3, объединяются в одно описание с параметром номера участка ***U***. В рассматриваемом событийном графе:

J – номер транзакта;

J:= – изменение номера транзакта;

M(J) – номер СМО второй фазы, в которой обслуживается транзакт ***J***;

$e_{нPS1}$ – событие начала активности (процесса) обслуживания в CMO_1;

U – номер СМО, в которой выполняется обслуживание транзакта; номер ***U*** СМО и его объектов канала и накопителя может состоять из номера фазы сети массового обслуживания и из номера СМО в фазе;

$e_{кPS1}$ – событие конца активности (процесса) обслуживания в CMO_1;

$e_{нPSU}$ – событие начала активности (процесса) обслуживания в CMO_U;

$e_{кPSU}$ – событие конца активности (процесса) обслуживания в CMO_U;

PS(U) – активность (процесс) обслуживания транзакта в СМО с номером ***U***;

E₃* – группа событий активности (процесса) ***PS(U)***;

{PS(U)-E₃*} – метка спецификации активности (процесса) обслуживания ***PS(U)*** с указанием номера соответствующей группы событий.

В событийном графе выделены следующие макрособытия:

P-E₀ – начало процесса ***P*** обслуживания в СтМО и начало активности (процесса) ***PS*** обслуживания транзакта в ***CMO₁***;

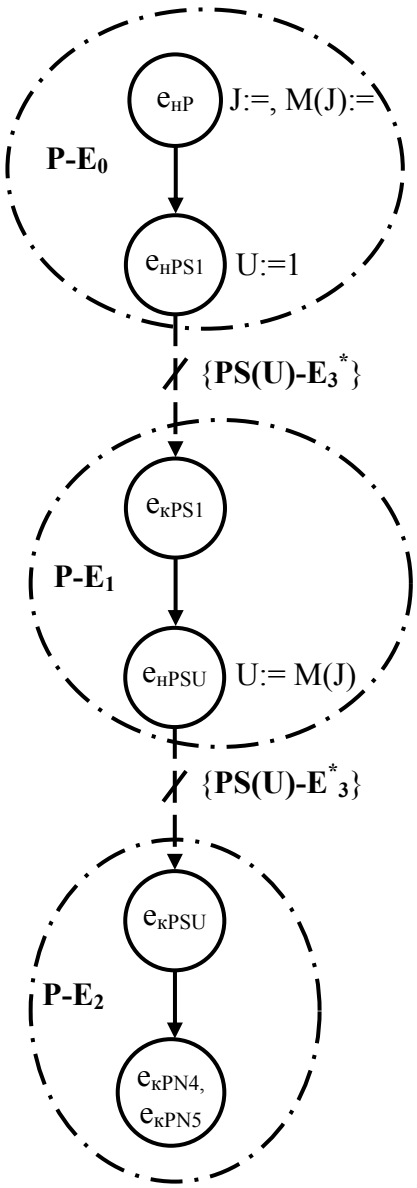

Рис. 9. Спецификация процесса ***P*** обслуживания транзакта в СтМО

P-E₁ – конец активности (процесса) ***PS*** обслуживания транзакта в ***CMO₁*** и начало активности (процесса) ***PS*** обслуживания транзакта в ***CMO$_U$***;

P-E₂ – конец активности (процесса) ***PS*** обслуживания транзакта в ***CMO$_U$***.

Активность обслуживания ***PS(U)*** является сложной активностью и может быть детализирована в виде процесса ***PS*** обслуживания транзакта в СМО с номером ***U***. Событийный граф этого процесса, специфицирующий причинно-следственную зависимость между событиями ***e$_{нPSU}$*** и ***e$_{кPSU}$***, приведен на рисунке 10.

На этом графе:

P1(U) – анализ состояния $z_{K(U)}$ канала ***K*** системы CMO_U: Св – свободен ($z_{K(U)} = 0$), Зан – занят ($z_{K(U)} = J$);

P2(U) – анализ состояния $z_{H(U)}$ накопителя ***H*** системы CMO_U: Нплн – неполон ($z_{H(U)} < L_U$, где – L_U емкость накопителя системы CMO_U), Плн – полон ($z_{H(U)} = L_U$);

P3(U) – анализ состояния $z_{H(U)}$ накопителя ***H*** системы CMO_U: Нпст – непуст ($z_{H(U)} > 0$), Пст – пуст ($z_{H(U)} = 0$);

A_2 – начало параллельного выполнения участков процесса;

$e_{зK(U)}$ – событие занятия канала ***K*** с номером ***U***;

$e_{оK(U)}$ – событие освобождения канала ***K*** с номером ***U***;

$e_{на(U)}$ – событие начала выполнения активности обслуживания ***a*** в канале ***K*** с номером ***U***;

$e_{ка(U)}$ – событие конца выполнения активности обслуживания ***a*** в канале ***K*** с номером ***U***;

$T_{a(U)}$ – время выполнения активности обслуживания ***a*** в канале ***K*** с номером ***U***;

$e_{кPW(U)}$ – событие конца ***W(U)*** процесса ***P*** в связи с переполнением накопителя;

$e_{зH(U)}$ – событие занятия накопителя ***H*** с номером ***U***;

$e_{о1H(U)}$ – событие освобождения из очереди накопителя ***H*** с номером ***U*** транзакта, стоящего первым.

В событийном графе выделены следующие макрособытия:

PS-E0 – анализ состояния канала и накопителя CMO_U;

PS-E1 – занятие канала ***K*** с номером ***U***;

PS-E2 – конец обслуживания в канале ***K*** с номером ***U*** транзакта ***J***.

Поскольку в спецификации процесса ***PS*** нет дуг с метками других спецификаций, то это терминальная спецификация, являющаяся самым нижним уровнем в представлении модели.

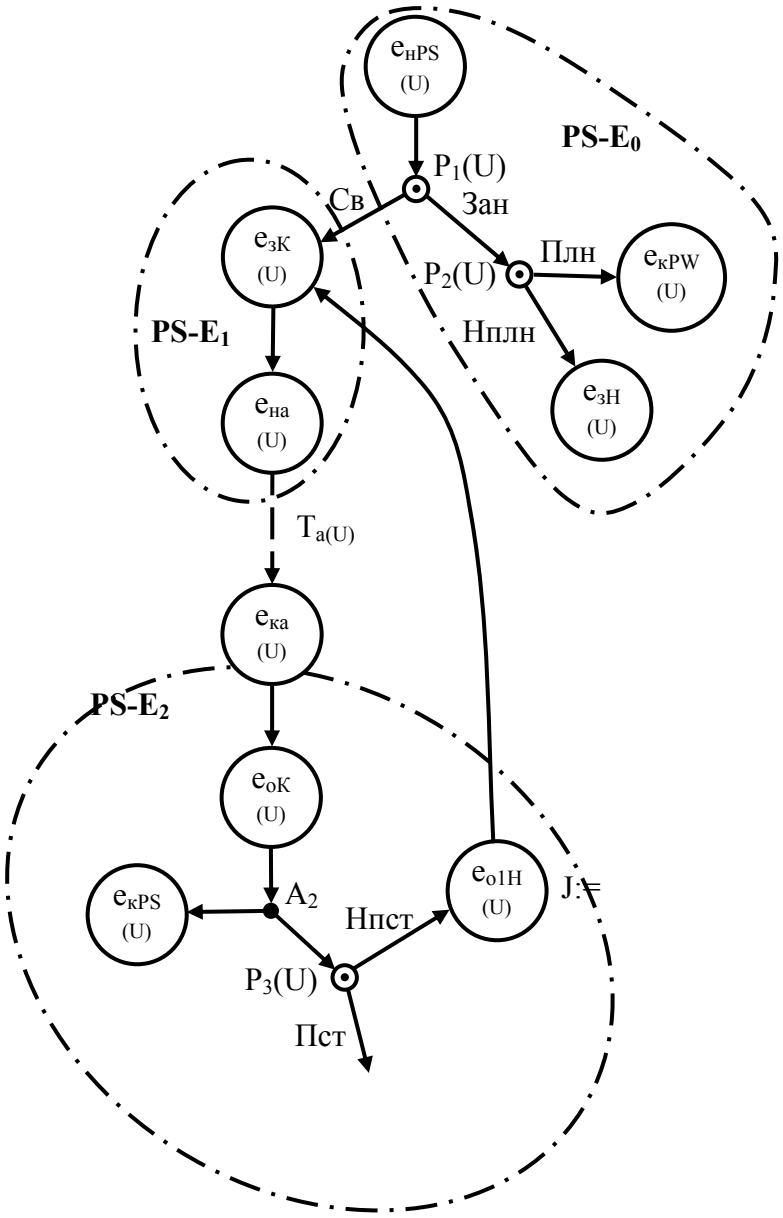

Рис. 10. Спецификация процесса ***PS*** обслуживания транзакта в СМО

Спецификации процессов составляют иерархию спецификаций событийной модели. Таким образом для рассматриваемой модели выделены три уровня представления (рис. 11):

– уровень имитационной модели – спецификация процесса имитационного моделирования ***PM*** (см. рис. 7); элементами модели являются события процесса моделирования, а также активность и события обслуживания в СтМО;

– уровень модели СтМО – спецификация процесса обслуживания ***P*** (см. рис. 9) в сети массового обслуживания, представляет порядок и маршрут обслуживания в сети массового обслуживания; элементами модели являются активности и события обслуживания в СМО;

– уровень модели СМО – спецификация процесса обслуживания ***PS*** (см. рис. 10) в системе массового обслуживания, представляет порядок обслуживания в системе массового обслуживания; элементами модели являются активность обслуживания в канале и события обслуживания в канале и накопителе СМО.

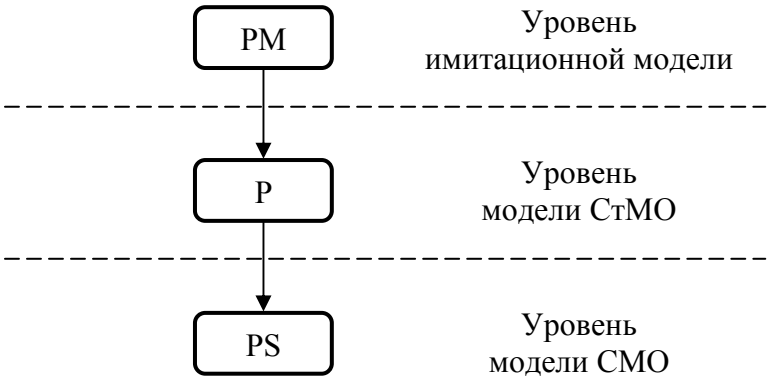

Рис. 11. Иерархия спецификаций для имитационной модели СМО

Для сложных систем уровень СтМО может разделяться на несколько подуровней. Событийная модель уровня СМО также может быть представлена в виде композиции моделей следующего уровня – уровня статических объектов СМО: канала и накопителя [Бабкин, Бобрышев 2008].

4. Иерархическое представление макрособытийных графов

Программно-реализуемой формой событийного графа является макрособытийный граф [Бабкин, Бобрышев 2007]. Макрособытийный граф представляет модель на уровне макрособытий, которые реализуются в виде программных секций или процедур. Для каждого выделенного процесса событийный граф преобразуется в макрособытийный на основе метода структурной декомпозиции. В результате этого спецификацию модели можно представить также многоуровневой совокупностью спецификаций макрособытийных графов (рис. 12). Макрособытийные графы состоят из одного типа вершин – макрособытий и четырех типов дуг: дуги мгновенного следования макрособытий, дуги следования макрособытий с задержкой и дуги отмены макрособытий (см. рис. 1). Вершины-макрособытия могут быть помечены основными операторами, которые выполняются в модели в этих вершинах. Полная спецификация вершин-макрособытий выполняется отдельно либо в виде текста, либо в виде событийного графа макрособытия и текста. Дуги помечаются условиями и временем следования макрособытий:

[<условие следования макрособытия>] / <время задержки следования макрособытия>

При безусловном следовании макрособытий E_i и E_j дуга помечается символом "–". Дуги первого типа, отображающие в модели безусловное следование макрособытий без задержки ($T_{ij} = 0$), не помечаются.

Введем, так же как и для событийных графов еще один тип дуг – дуги, помеченные меткой спецификации, которая детализирует причинно-следственную связь между макрособытиями (см. рис. 2–3). Для рассматриваемого примера иерархическая декомпозиция модели на уровне макрособытий приведена на рисунке 12.

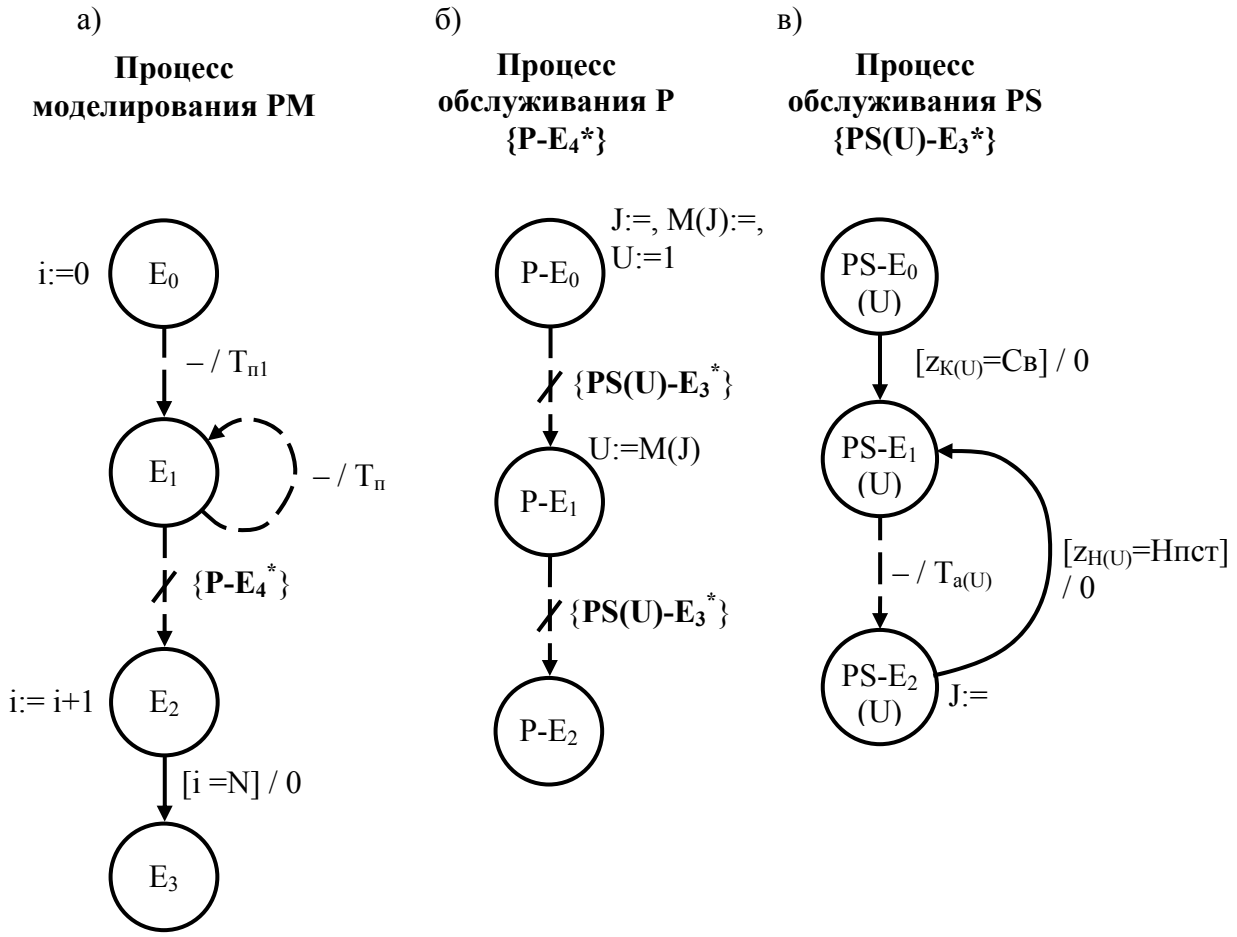

Рис. 12. Спецификации процессов модели на уровне макрособытий

Для большей наглядности объединим эти спецификации в один общий макрособытийный граф, описывающий функционирование моделируемой системы в целом. Объединение выполняется заменой дуг с пометками спецификаций в основном процессе на соответствующие спецификации вложенных процессов. При этом макрособытие, из которого исходит дуга с меткой спецификации, соединяется дугой первого типа с начальным макрособытием событийного графа вложенного процесса. Условие следования этой дуги совпадает с условием следования помеченной дуги. Конечное событие событийного графа вложенного процесса соединяется дугой первого типа с макрособытием основного процесса, в которое входит дуга с меткой спецификации. Условие следования этой дуги определяется условием следования конечного события во вложенной спецификации и повторяемостью вложенной спецификации в основном процессе. Чтобы отразить декомпозицию макрособытийной модели на иерархически вложенные процессы, разделим представление общего макрособытийного графа на дорожки [Бабкин, Бобрышев 2008]. Будем использовать дорожки для группирования макрособытий процессов одного типа или одного уровня. В макрособытийном графе (рис. 13) выделены дорожки иерархически-вложенных процессов ***PM***, ***P*** и ***PS***.

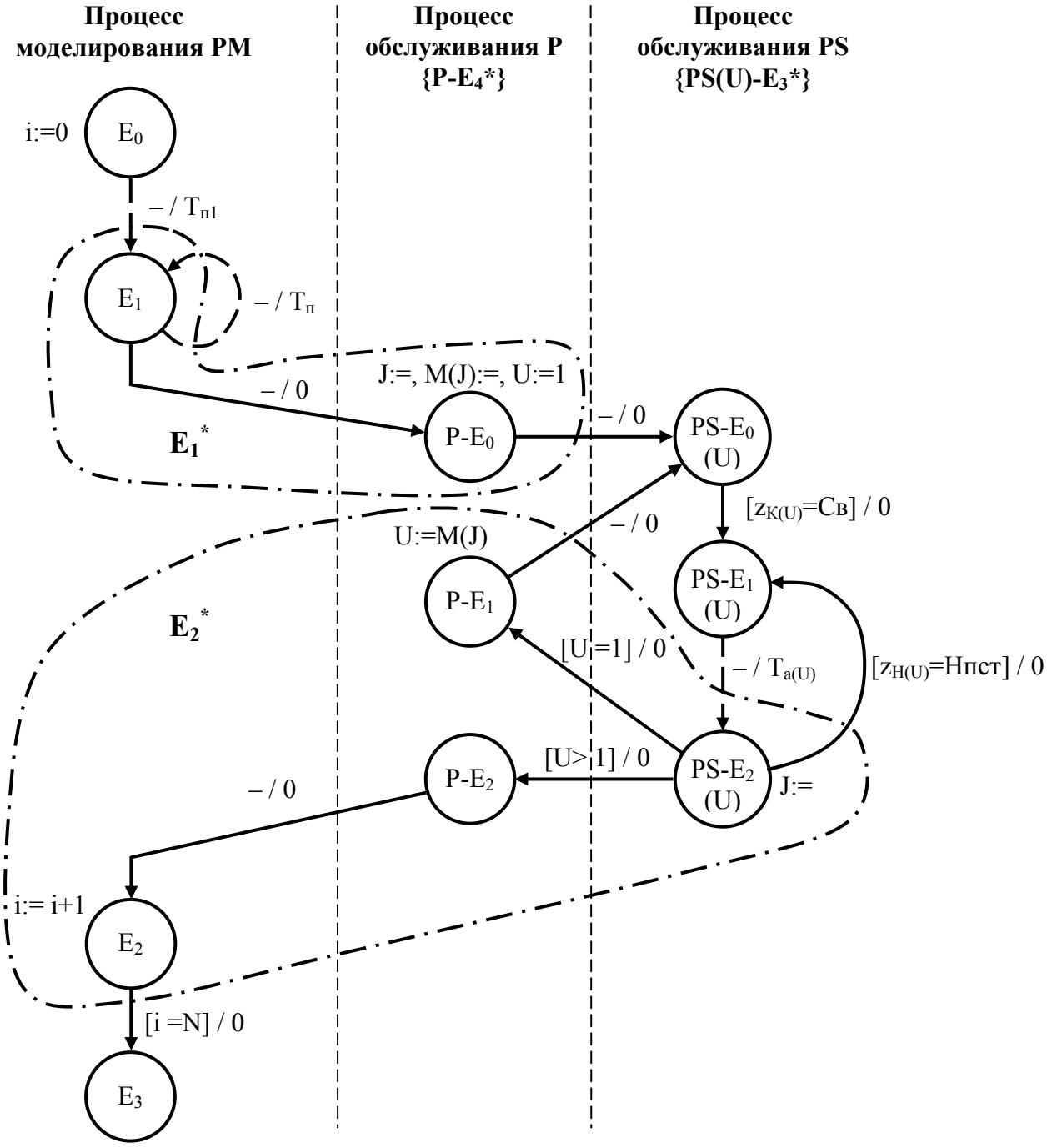

Рис. 13. Имитационная модель СтМО на уровне макрособытий

В результате иерархической декомпозиции число макрособытий будет больше, чем при обычной структурной декомпозиции событийных графов. Увеличение числа макрособытий происходит за счет разделения (декомпозиции) макрособытий начала и конца вложенного процесса. При структурной декомпозиции макрособытия E_1 процесса верхнего уровня ***PM*** и $P\text{-}E_0$ вложенного процесса ***P*** объединяются в макрособытие E_1^*, а макрособытия $PS\text{-}E_2$ процесса ***PS***, $P\text{-}E_1$ и $P\text{-}E_2$ процесса ***P*** и E_2 процесса ***PM*** объединяются в макрособытие E_2^*. Однако при этом в одно макрособытие включаются события различных процессов, то есть спецификации процессов

смешиваются. Так, в рассматриваемом примере в макрособытие E_2^* включаются события трех процессов ***PM***, ***P*** и ***PS***.

Таким образом, при иерархическом представлении модели число макрособытий увеличивается, а следовательно, увеличивается и число планирований или вызовов макрособытий. Но при этом в значительной мере структурируется спецификация модели. Полученная спецификация состоит из частей модели, описывающих группы событий выделенных процессов. Это позволяет упростить внесение изменений в спецификацию, что приводит к уменьшению трудоемкости отладки и коррекции модели.

5. Об организации событийных программных моделей

Рассмотренная иерархическая декомпозиция событийных моделей позволяет структурировать модели на уровне событийных графов. Программные модели, построенные на основе иерархических событийных моделей, могут быть ориентированными как на процессы, так и на события. Однако существует противоречие между многоуровневым иерархическим представлением событийной математической модели и одноуровневой ее реализацией в программной модели.

Простым способом программной реализации иерархической событийной модели является группирование процедур событий в спецификации программной модели в соответствии с принадлежностью к выделенным процессам. Это позволяет в значительной степени структурировать спецификацию программной модели. Программная модель в этом случае состоит из частей – процессных секций, представляющих выделенные процессы, то есть является событийно-процессной моделью. Каждая процессная секция представляет собой набор процедур макрособытий одного процесса.

Для рассматриваемого примера программная событийно-процессная модель будет состоять из следующих секций:

– секция процесса ***PS*** обслуживания в СМО, включающая процедуры макрособытий $PS\text{-}E_0$, $PS\text{-}E_1$ и $PS\text{-}E_2$;

– секция процесса ***P*** обслуживания в СтМО, включающая процедуры макрособытий $P\text{-}E_0$, $P\text{-}E_1$ и $P\text{-}E_2$;

– секция процесса моделирования ***PM***, включающая процедуры макрособытий E_0, E_1, E_2 и E_3;

– диспетчер событий – процедура, осуществляющая выборку текущего макрособытия из списка событий и вызов соответствующей процедуры макрособытия.

Разделение спецификации модели на процессные секции в значительной мере позволяет разделить описания процессов различных уровней. Однако остается некоторое смешивание описаний процессов. Например, при возврате из нижнего процесса необходимо явно указывать макрособытие верхнего процесса, к которому осуществляется возврат. Кроме того, макрособытия процессов связаны дугами первого типа (мгновенного следования) и их удобно реализовать не через механизм планирования событий, а непосредственным вызовом процедур событий. Однако поскольку процесс верхнего уровня вызывает начальное событие нижнего уровня, а процесс нижнего уровня при возврате вызывает событие верхнего процесса, возникает противоречие в требованиях к порядку описания процедур и их группировки по процессам, поэтому одна из этих дуг-связей должна быть реализована через механизм планирования. Структура списка событий и, соответственно, механизм планирования в этом случае являются простыми. Каждая запись о планируемом макрособытии представляет набор элементов ***(J, E, T, Par)***, где ***J*** – номер транзакта, ***E*** – номер

макрособытия, ***T*** – время, когда должно произойти планируемое макрособытие, ***Par*** – остальные параметры макрособытия (приоритет, номер входа и т.д.).

Другим, более сложным способом является реализация процессных секций в виде программных процедур. При этом значительно усложняется структура списка событий: указатель на событие должен включать указатели (номера) всех процессов, в которые вложен данный процесс, и событий возврата из каждого процесса. Каждая запись о планируемом макрособытии представляет набор элементов ***(J, S, T)***, где ***S*** – список записей ***(P, E)***, причем ***P*** – номер типа процесса, ***E*** – номер макрособытия возврата (транзитивно-планируемое макрособытие). Для последней записи в списке ***E*** – номер планируемого макрособытия.

В язык моделирования должны быть добавлены операторы начала (вызова) процесса и конца (возврата) процесса. При вызове (оператор начала) процесса к составному указателю на событие добавляется указатель на вызываемый процесс и указатель на событие вызывающего процесса, которое должно выполниться при завершении вызванного процесса – процесса нижнего уровня. При возврате (оператор конца) процесса из указателя события удаляется указатель текущего процесса и выполняется переход к событию возврата. Кроме того, каждая процедура процесса должна содержать диспетчер событий – управляющую часть, осуществляющую выбор событий данного процесса.

Такое усложнение механизма планирования позволяет реализовать многоуровневую иерархическую событийно-процессную программную модель, структурно соответствующую математической модели в виде событийного графа.

6. Заключение

В работе [Бабкин, Бобрышев 2008] рассматриваются два способа декомпозиции событийного графа: структурный и объектный. Разновидностью объектной декомпозиции является иерархическая декомпозиция событийного графа, рассматриваемая в настоящей работе. Иерархической декомпозиция событийного графа и событийной модели основана на выделении (описаний) событийных подграфов иерархически вложенных процессов. Такая декомпозиция позволяет строить иерархические спецификации событийных моделей. Для целей иерархического представления и формализации событийных моделей используется в событийных графах новая разновидность помеченной дуги, отображающая причинно-следственные связи с неизвестным временем и условием следования и связи, представляющие сложные активности-процессы, специфицируемые вложенными событийными графами. Это позволяет описывать многоуровневые событийные модели ДС.

Для наглядного представления иерархических моделей предлагается форма макрособытийного графа с дорожками, в которой каждая дорожка представляет процесс.

Событийные модели, построенные на основе предлагаемого подхода, могут быть реализованы как процессо-ориентированными программными моделями, так и программными событийно-ориентированными моделями. Группирование процедур событий в спецификации программной модели в соответствии с принадлежностью выделенным процессам позволяет в значительной степени структурировать спецификацию программной модели и упростить внесение изменений в спецификацию. Программная модель в этом случае состоит из процессных секций, то есть является событийно-процессной моделью. Каждая процессная секция представляет собой набор процедур событий данного процесса. При реализации процессных секций в виде процедур процессов усложняется механизм планирования (структура списка событий).

Кроме того, в язык моделирования должны быть добавлены операторы начала и конца процесса.

Библиографический список

Автоматизация проектирования вычислительных систем. Языки, моделирование и базы данных / под ред. М. Брейера. М.: Мир, 1979. С. 12–29, 35–44.

Бабкин Е.А. Методические указания по моделированию вычислительных систем на событийно-ориентированном языке. Курск : Курск. гос. пед. ин-т, 1988.

Бабкин Е.А. Событийные модели дискретных систем / Курск. гос. ун-т. Курск, 2005. 18 с. Деп. в ВИНИТИ 14.01.05, № 30–В2005.

Бабкин Е.А. О синтезе событийных моделей дискретных систем // Ученые записки : электронный научный журнал Курского государственного университета. 2006. № 1. URL: http://www.scientific-notes.ru/pdf/s15.pdf (дата обращения: 11.02.2010). Эл № 77-26463.

Бабкин, Е.А., Бобрышев Е.А. О методах декомпозиции событийных графов // Ученые записки : электронный научный журнал Курского государственного университета. 2008. № 2(6). URL: http://www.scientific-notes.ru/pdf/006-02.pdf (дата обращения: 15.03.2010). № гос. регистрации №0420800068\0024

Бабкин, Е.А. Бобрышев Е.А. Иерархическое событийно-автоматное моделирование // Информационные технологии моделирования и управления: научно-технический журнал. Воронеж: Научная книга, 2007. № 1 (35). С. 39–48.

Бабкин Е.А. Иерархическое представление событийных графов // Четвертая всероссийская научно-практическая конференция по имитационному моделированию и его применению в науке и промышленности «Имитационное моделирование. Теория и практика» ИММОД-2009 : сб. докладов. Т. I. СПб.: ОАО "ЦТСС", 2009. С. 78–82.

Schruben L.W. Simulation Modeling with Event Graphs // Communications of the ACM. 1983. Vol. 26. Num. 11. P. 957–963.