O. B. Shatalova
South-West State University
ShatOlg@mail.ru
Z. U. Zeidan
South-West State University
zeinab.zeidan@yandex.ru
A. V. Kiselev
South-West State University
Kiselevalexey1990@gmail.com

THREE-PARAMETER MODEL OF ELECTRICAL RESISTANCE OF BIOMATERIAL CONSTRUCTED ON THE BASIS OF TRANSIENT ANALYSIS

In the article trihalomethane model of electrical resistance of biological material derived from transient analysis, characterized in that its parameters are determined by means of the iterative algorithm based on non-linear regression equation the results of the samples of the voltage transient performance in BATH, and optimized on the basis of variation of Allan. Minimization of the Allan variation allows to find the optimal number of samples of the transition process by which the General nonlinear regression model is constructed, as well as to optimize the number of transients by which the parameters of the bipolar are determined.

***Keywords**: passive bipolar, model, informative features, Allan variation, transition process, biologically active point, optimization.*

УДК 004.42

Сорокин Арсений Николаевич
Вологодский государственный университет
arseny_sorokin@mail.ru

ПРИМЕНЕНИЕ АГЕНТНО-ОРИЕНТИРОВАННЫХ G-СЕТЕЙ ПЕТРИ ДЛЯ АНАЛИЗА БИЗНЕС-ПРОЦЕССОВ ПРОИЗВОДСТВЕННОГО ПРЕДПРИЯТИЯ

В статье рассматриваются вопросы применения агентно-ориентированных G-сетей Петри для моделирования бизнес-процессов производственного предприятия

***Ключевые слова:** агентно-ориентированные G-сети Петри, модель бизнес-процессов.*

Существуют различные инструменты исследования систем, один из них это сети Петри. Теория сетей Петри делает возможным моделирование системы математическим представлением ее в виде сети. Анализ сетей Петри помогает получить важную информацию о структуре и динамическом поведении моде-

лируемой системы и выработки предложений по ее усовершенствованию и изменению. G-сети – это объектно-ориентированное расширение сетей Петри [1]. Система G-сети составлена из некоторого количества G-сетей, каждая из которых представляет автономный модуль или объект, доступ к которым определен через хорошо определенный механизм.

Для моделирования агентно-ориентированной структуры необходимо применить некоторое расширение G-сетей, которое позволит описать класс агента [2]. При создании агента (в виде модели), во-первых, генерируется идентификатор агента и инициализируется ментальное состояние, во-вторых, вводятся три специальных модуля, чтобы сделать агента автономным и внутренне мотивированным, а именно модуль цели, модуль базы знаний и модуль планирования. Абстрактная схема агентно-ориентированной G-сети показана на рис. 1. Модуль цели описывает поставленные перед агентом цели, модуль базы знаний описывает окружающую среду и внутреннее состояние, которое агент данного класса может иметь, и модуль планирования составляет план достижения целей из модуля цели. Например, в модуле планирования агент может решать игнорировать ли входящее сообщение, начинать новую беседу или продолжать беседу, которая была начата другим агентом или им самими.

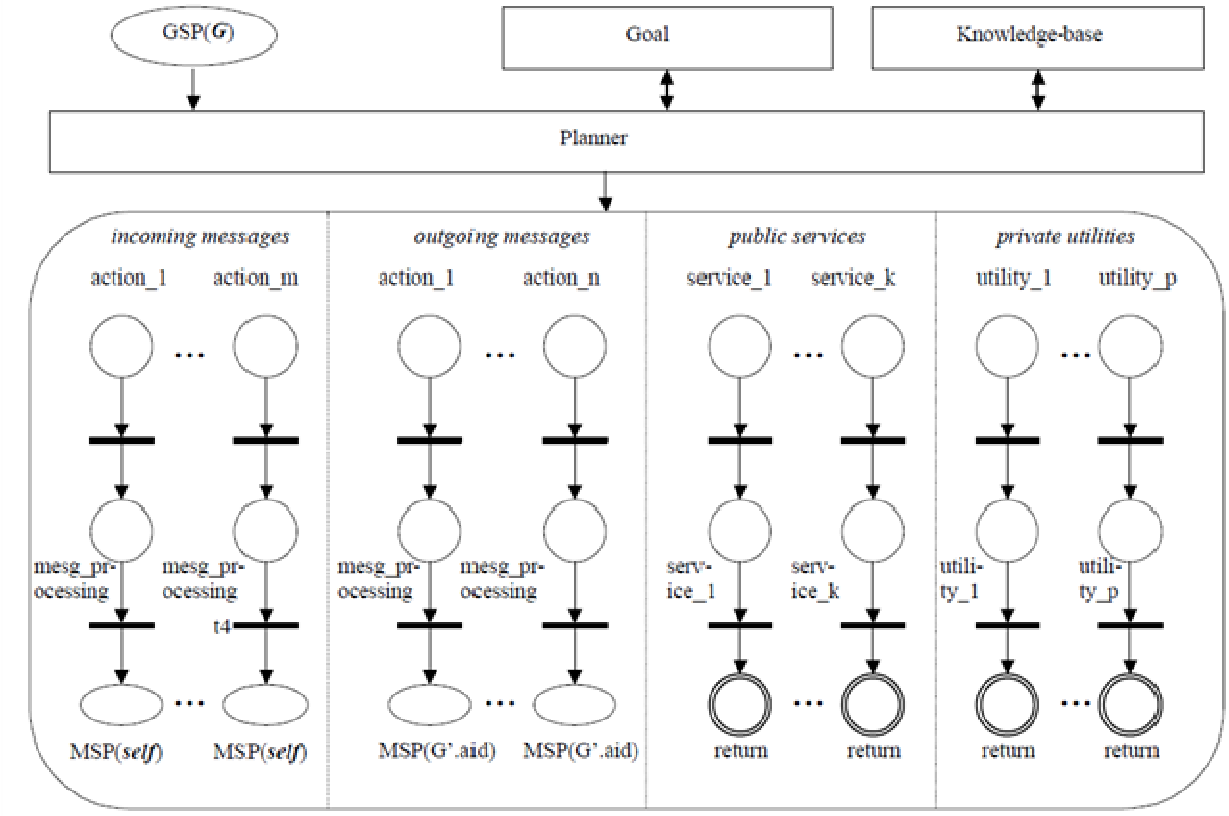

Рис. 1. Абстрактная схема агентно-ориентированной G-сети

Внутренняя структура агентно-ориентированной G-сети состоит из 4 секций: входящие сообщения, выходящие сообщения, публичные сервисы, при-

ватные утилиты. В секциях входящий/выходящих сообщений определенны *единицы обработки информации (MPU – message processing units)*, которые используются для обработки сообщений от других агентов и могут использовать ISP в качестве вызовов методов, определенных в секции приватных утилит. Секция публичных сервисов дает агенту возможность работы в качестве сервера. Другие агенты могут использовать механизм вызова функций ISP для синхронного выполнения этих сервисов. Секция приватных утилит подобна секции публичных сервисов, с отличием в том, что приватные утилиты могут быть вызваны только самим агентом.

Хотя и агенты, и объекты используют посылку сообщений для связи друг с другом, имеются некоторые отличия между ними. Отправка сообщений для объектов это формирование уникального вызова метода, в то время как агенты отличают различные типы сообщений, модели этих сообщений и используют сложные протоколы для переговоров. К тому же большинство агентов связываются посылкой асинхронных сообщений. Асинхронная передача является более фундаментальной, чем синхронная. Для поддержки посылки асинхронных сообщений вводится новых механизм под названием место переключения отправки сообщений (MSP). Когда маркер достигает MSP (на рисунке он представлен эллипсом) маркер удаляется из места вызова и вносится в GSP место вызванного агента. В отличие от ISP механизма, вызывающий агент не ждет, пока маркер вернется, прежде чем он продолжит выполнять следующий шаг. Также данное расширение G-сети позволяет использовать ключевое слово self для ссылки агента на самого себя.

Применение агентно-ориентированных G-сетей для анализа моделей бизнес-процессов

Как и любое расширение, агентно-ориентированное расширение G-сетей поддается декомпозиции в обычные сети Петри. Таким образом, любую агентно-ориентированную систему можно представить в виде модели, тем самым на раннем этапе оценить плюсы и минусы, тупиковые ситуации, эффективность системы и т. д.

Используя данную модель можно смоделировать систему анализа бизнес-процессов на основе агентов, где каждый агент связан с отдельным бизнес-процессом. Таким образом, агенты, общаясь, могут правильно найти зависимости между друг другом и составить граф зависимостей бизнес-процессов между собой. По составленной модели можно проанализировать эффективность системы, промоделировать ее работу и сделать вывод о том, стоит ли ее реализовывать. Также возможен анализ работы отдельных компонентов (отдельных агентов) системы, который укажет на изъяны и поможет исправить их.

Рассмотрим модуль планирования данной системы, показанный на рис. 2.

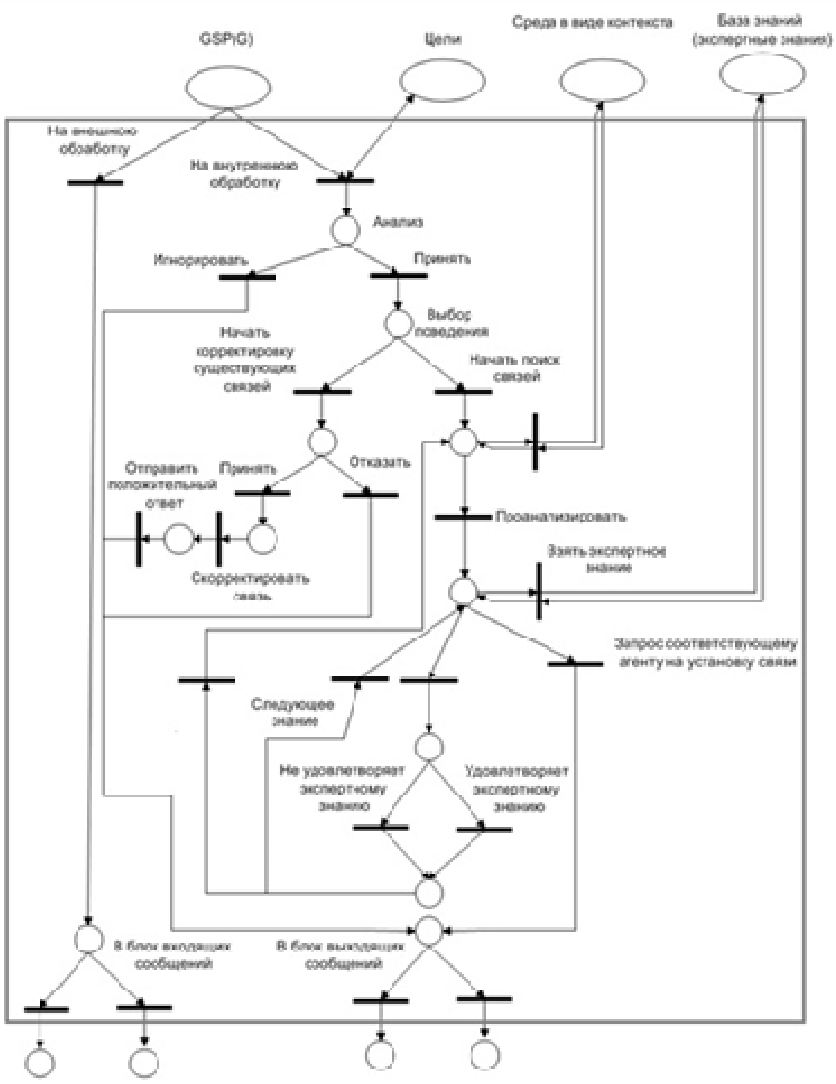

Рис. 2. Модуль планирования агентно-ориентированной системы

Модули цели и базы знаний представлены как два специальных места, каждый из которых содержит токен, который представляет набор целей или набор убеждений. Модуль планирования управляется целями, так как переход может сработать только, когда делается попытка достижения переданной цели. Действия в модуле планирования начинаются, когда на вход подается сообщение (от других агентов или от самого себя, используя слово self). Если сообщение пришло от другого агента, оно отправляется в модуль входящих сообщений, где после обработки возвращается и идет по переходу «На внутреннюю обработку». Далее относительно целей решатся вопрос об игнорировании или принятии данного сообщения. В случае игнорирования оно уходит в блок выходящих сообщений, иначе продолжает обработку. В данном примере для агента

имеется только два поведения: начать поиск новых связей, скорректировать старые. Первый вариант нужен для установки начальных связей, которые впоследствии будут корректироваться, а также поиска новых в случае разрыва. Второй вариант служит для корректировки существующих связей, которые были ранее установлены.

Если сообщение о начале поиска новых связей, то токен идет по ветке «Начать новый поиск». В этом случае модуль планирования берет данные из среды (контекста предложения) и пытается к ней применить экспертные знания, для определения уровня связанности.

В данном примере не указаны публичные сервисы и приватные утилиты. Однако, при более подробной модели они имеют место.

Также следует отметить, токен имеет несколько иную структуру, в зависимости от того, по какой ветке он идет. То есть токен представляет собой кортеж из двух элементов: признака и тела. Признак – это вариант токена (например, для приватных утилит, для публичных сервисов, для внутренней обработки, для внешней обработки), а тело – конкретная реализация. Более конкретно токен можно определить следующим образом:

if (*mTkn.tag* [Уравнение] (**internal**, **external**))
then *mTkn.body* = struct {
int sender; // message sender identifier
int receiver; // message receiver identifier
string protocol_type; // protocol type
string message_name; // message name
string content; // message content
}
else *mTkn.body* = (seq, sc, msg);

Таким образом, применяя существующие G-сети, а также расширяя их для своих нужд, можно смоделировать практически любую агентно-ориентированную систему.

Применение сетей Петри и их расширений позволяет устранить нехватку формальной спецификации и инструментов проектирования агентно-ориентированных и объектно-ориентированных систем. Позволит проектировать крупномасштабные и коммерческие приложения, увеличив гарантии того, что разработанные системы будут надежными и пригодными для выполнения поставленной цели.

Список литературы

1. Perkusich A., de Figueiredo J. G-nets: A Petri Net Based Approach for Logical and Timing Analysis of Complex Software Systems // Journal of Systems and Software. 1997. N 39(1). P. 39–59.

A. N. Sorokin
Vologda state University
arseny_sorokin@mail.ru

APPLICATION OF AGENT-ORIENTED PETRI G-NETWORKS FOR ANALYZING BUSINESS PROCESSES OF A PRODUCTION ENTERPRISE

The article deals with the application of agent-oriented Petri G-networks for modeling business processes of a production enterprise

Keywords: *agent-oriented Petri G-networks, business process model*

УДК 004.432.2

Орлов Александр Валерьевич
Костромской государственный университет
aorlov@list.ru

МЕТОДИЧЕСКИЕ И ТЕХНИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ОБУЧЕНИЮ ВЗАИМОДЕЙСТВИЮ С WINDOWS API ПРИ ИСПОЛЬЗОВАНИИ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ

В данной статье рассматриваются особенности обучения работе с Windows API студентов, не изучавших языки программирования C/C++. Предлагаются программные средства и механизмы, позволяющие проводить обучение с использованием языков высокого уровня, таких как C#, Python и Java. Акцентируется внимание на особенностях работы с разделяемыми ресурсами ЭВМ, представленными типом данных HANDLE, в рассматриваемых языках. Также с целью ознакомления с общим стилем кодирования и объемом требуемого дополнительного кода приводятся минимальные примеры, использующие рассматриваемые программные средства.

Ключевые слова: *winapi, handle, c#, python, java, обучение, библиотека, маршалинг.*

Изучение Windows API [1] является важным подготовительным шагом при изучении архитектуры и функционирования современных операционных систем (в том числе ОС, не принадлежащих семейству Windows). Это позволяет