M. A. Navrotsky, D. I. Muromtsev
University ITMO

N. A. Zhukova, A. I. Vodyaho
Saint Petersburg Electrotechnical University «LETI»

ARCHITECTURAL SOLUTIONS FOR OF SCIENTIFIC INFORMATION PORTALS

A new class of information systems – educational scientific information portals is described. The set of reference architectural solutions, which can be used for different purposes, depending on the requirements and the subject domain, is presented. Specific feature of the proposed architectural solutions is the use of a service oriented approach for building the subsystem and modules. This approach allows to tune the system according to the requirements of each organization and the subject domain. An essential advantage of the suggested approach as compared to the existing one is the possibility of using standard interfaces for services interaction, making it possible to replace any module when necessary. Each service can be realized in such a way that they can be packed into separate containers, which enables services to be deployed in different clouds or hosts in order to optimize the resource usage. The questions of developing and building a software platform for domain-specific research and educational portals implementation are also discussed in the paper.

Scientific informational portals, ontology, open linked data, search in open data

УДК 004.451.44:004.942

С. А. Даденков, А. Н. Кокоулин, Е. Л. Кон
Пермский национальный исследовательский политехнический университет

Разработка имитационной модели циклического алгоритма планирования задач реального времени

Разрабатывается имитационная модель циклического алгоритма планирования задач на основе приоритетов (round robin priority driven preemptive scheduler), используемого на узлах-датчиках, -контроллерах, -исполнительных механизмах распределенных fieldbus-сетей мягкого реального времени, в том числе LonWorks и BacNet. Предлагаемая модель отличается от известных аналитических и имитационных аналогов учетом ранее не анализируемых значимых факторов функционирования алгоритма: индивидуальных номеров, приоритетов, конфигурационных параметров задач. Модель создается в системе имитационного моделирования AnyLogic и дополняется элементами визуализации результатов оценки вероятностных и временных характеристик обработки задач. Для разработки использован графический инструментарий диаграмм состояний и переходов совместно с программным расширением модели. Указанное позволило создать легко масштабируемую и корректную модель, пригодную для оценки характеристик и проектирования структуры планировщика задач реального времени.

Циклический алгоритм планирования задач, реальное время, промышленные fieldbus-сети, имитационное моделирование, вероятностные и временные характеристики

Циклический алгоритм (Round-robin) широко используется программными планировщиками задач в вычислениях реального времени. Распространение алгоритм получил за счет детерминированности интервалов времени решения задач в системах мягкого и жесткого реального времени. Различные модификации алгоритма применяются при пакетном планировании задач передачи данных в компьютерных и сенсорных сетях, для сбора, обработки и передачи технологической (измерительной и управляющей) информации в распределенных системах управления (Distributed control system), в том числе построенных на основе сетевых fieldbus-технологий. Соответственно применяются алгоритмы: Round Robin, Weighted Round Robin, Deficit Round Robin and

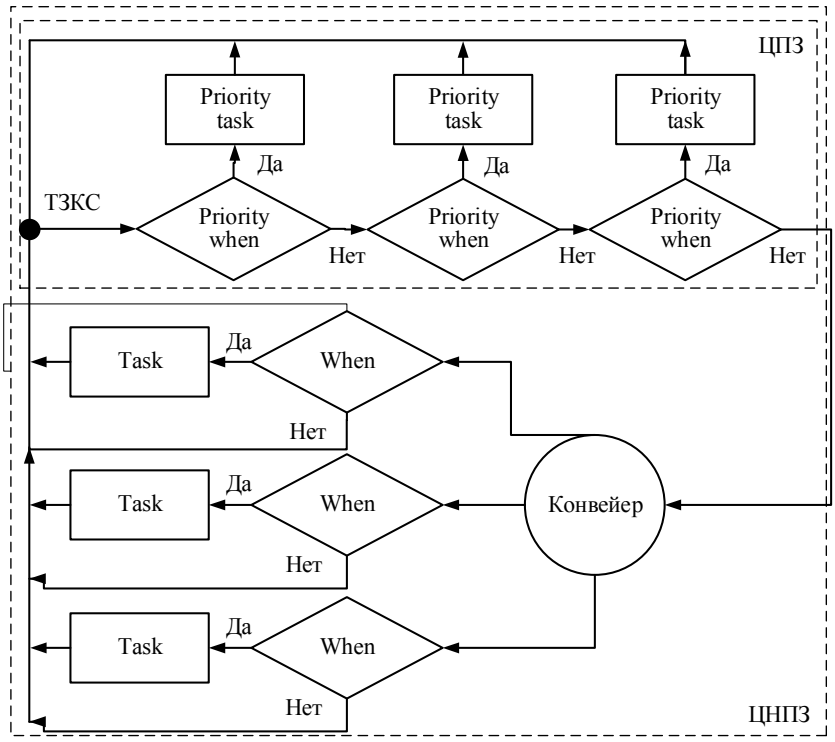

Рис. 1

Modified Deficit Round-Robin [1], [2]; Priority-based, Rate Monotonic Analysis, Earliest Deadline First [3]–[6]. Область применения алгоритмов определена эффективностью обработки планируемых задач в зависимости от их количества, приоритетности, периодичности/спорадичности [5], [6], вычислительной сложности, продолжительности и других параметров. Эффективность алгоритма оценивается сверткой характеристик: детерминированности временных интервалов, времени реакции планировщика на контролируемые события и задержки обработки задач, эффективности использования времени.

В статье анализируется одна из модификаций классического циклического алгоритма планирования задач на основе приоритетов (priority-based), применяемого в распределенных системах с событийно-ориентированным (event-driven) принципом обработки задач и передачи информации узлами (контроллерами) fieldbus-сети (LonWorks, BacNet, CAN, EIB/KNX и др.). Цель – создание имитационной модели функционирования алгоритма (round robin priority driven preemptive scheduler) и способа количественной оценки временных и вероятностных характеристик обработки задач, необходимых для проектирования структур планировщиков задач реального времени. Необходимость создания новой модели заключается в повышении точности оценки характеристик алгоритма планирования задач. Поставленная цель достигается созданием модели, учитывающей значимые факторы функционирования алгоритма [7]–[12], ранее не анализируемые в научной литературе [8]–[10]: индивидуальные номера, приоритеты, конфигурационные параметры задач.

Краткая характеристика циклического алгоритма планирования задач на основе приоритетов. Циклический алгоритм планирования задач round robin priority driven preemptive scheduler [7], [8] определяет последовательность обработки задач, связанных с преобразованием собранной и полученной узлом из сети измерительной и управляющей информации о контролируемом технологическом процессе. На каждом такте цикла выполняется обработка программного модуля планировщика задач, состоящего из проверки «When» условия наступления события (изменения контролируемой технологической информации) и задачи «Task» обработки события (информации) при его наступлении (рис. 1).

Процесс планирования осуществляется на основе приоритета и номера задачи. Выделяют 2 уровня приоритета: приоритетный и неприоритетный. Цикл решения задач состоит из цикла приоритетных задач (ЦПЗ), в рамках которого происходит последовательная обработка приори-

тетных модулей, и цикла неприоритетных задач (ЦНПЗ), который наступает только после завершения ЦПЗ и характеризуется обработкой только одного модуля планировщика (в порядке следования) (рис. 1). При успешной проверке «When» условия наступления события модуля выполняется решение соответствующей задачи «Task» и переход в точку завершения критической секции (ТЗКС, точку входа), после чего начинается новый ЦПЗ. В точке завершения критической секции выполняется обработка исходящей и поступающей к планировщику информации от других узлов распределенной сети. При планировании задач с одним уровнем приоритета схема алгоритма обслуживания задач данного приоритета не изменяется.

Условия проверки наступления событий «When» модулей планировщика представляют собой элементы контроля изменчивости обрабатываемой узлом технологической информации. Контроль производится по так называемым конфигурационным свойствам узла [7], [11]: минимальное и максимальное время обновления/передачи значения параметра контролируемого технологического процесса (тактирование); граничное значение уровня контролируемого параметра; минимальное изменение значения (дельта) параметра контролируемого процесса. Таким образом, тип и значение используемого конфигурационного свойства, а также характер изменения контролируемых параметров определяют частоту наступления потенциальных событий и обработки связанных с ними задач модулей планировщика. В разрабатываемой модели частота наступления различных событий может быть задана индивидуальными значениями интервалов времени их наступления. В зависимости от типа конфигурационного свойства время между событиями может быть распределено по равномерному (при тактировании) или экспоненциальному закону. Частота наступления событий может быть адаптивна к быстродействию их обработки. Тактирование событий через определенные интервалы времени чаще всего производится адаптивно к частоте их обработки, т. е. новое событие может возникнуть только после окончания обработки предыдущего. Для данного вида события характерно запаздывание обработки задач планировщика. При внешнем тактировании (от узла сети), за счет задержки прохождения обработки циклов задач, характерно появление пропусков (потерь) тактов обработки условий и задач. В модели для оценки такой величины вводится переменная вероятности потери информации. При обработке событий контроля технологических параметров условие наступления события определяется характером изменения величины и не зависит от процесса обработки информации. В таком случае при обработке возможны задержки до начала обработки задач и пропуски (потери) обработки важной информации до момента наступления и обработки последующего события (изменения контролируемой величины). В разрабатываемой модели сценарии с адаптивной и неадаптивной частотами наступления событий называются сценариями 1 и 2 соответственно.

Разработка имитационной модели циклического алгоритма планирования задач. Разработка модели алгоритма планирования задач выполняется в системе имитационного моделирования AnyLogic. Выбор обусловлен эффективным сочетанием гибкости программной разработки модели и простоты графического построения (описания) анализируемого алгоритма с помощью аппарата диаграмм состояний и переходов (stateChart) [13].

Разработка модели начинается с создания структуры диаграммы состояния и переходов с помощью элементов одноименной библиотеки AnyLogic. Последовательно создается начало диаграммы состояний, состояния Start, Task, When, моделирующие соответственно состояния алгоритма: точка входа (завершения критической секции), задача, проверка (рис. 2). Используются 2 типа переходов между состояниями модели: с временной задержкой и по условию.

Настройка переходов, моделирующих временную задержку: переход из состояния Start характеризуется задержкой реализации системных приоритетных задач; переход из состояния When характеризуется задержкой проверки условия модуля; переход из состояния Task характеризуется вычислительной задержкой обработки задачи. Средние значения задержек переходов в модели задаются параметрами Tstart, Twhen, Ttask. Для получения оценки верхней границы задержки обработки задач времена задержек в состояниях модели считаются случайными величинами, распределенными по экспоненциальному закону. Для моделирования случайной задержки в AnyLogic используется функция exponential (1/time), где time – среднее время обработки. В таблице представлено описание используемых в модели параметров и их значений.

Параметр (тип: значение)	Описание
Исходные данные	
количествоМодулей (int: 4)	Количество модулей планировщика задач
количествоПрМодулей (int: 2)	Количество приоритетных модулей планировщика
сценарий1 (boolean: false)	Сценарий функционирования (true – 1, false – 2)
Tstart (double: 1.0)	Задержка обработки точки завершения критической секции
Twhen (double: 0.3)	Задержка проверки условия модуля планировщика
Ttask (double: 1.0)	Задержка обработки задачи
интервалыВремени (ArrayList<double>: {0.0, 50.0, 50.0, 50.0, 50.0})	Коллекция индивидуальных интервалов времени между контролируемыми событиями модулей планировщика задач
Динамически изменяемые переменные и коллекции	
номерМодуля (int: 0)	Текущий номер обрабатываемого модуля планировщика задач
номерНпрМодуля (int: 0)	Номер текущего неприоритетного модуля
состояния (ArrayList<boolean>: {false, false, false, false, false})	Коллекция состояний наступления событий модулей планировщика задач (true – событие наступило, false – событие не наступило)
времяСобытий (ArrayList<double>: {0.0, 0.0, 0.0, 0.0, 0.0})	Коллекция моментов времени наступления последних событий модулей планировщика
события	Динамическое событие, моделирующее изменение состояний модулей
Элементы сбора результатов моделирования	
задержкаОбработкиЗадачи (ArrayList<StatisticsDiscrete>)	Коллекция элементов статистики по времени задержки обработки задач модулями планировщика
обработаноПроверок (ArrayList<int>)	Коллекция числа обработанных проверок модулей планировщика
обработаноЗадач (ArrayList<int>)	Коллекция числа обработанных задач по модулям планировщика
пропущеноЗадач (ArrayList<int>)	Коллекция числа пропущенных задач модулей планировщика

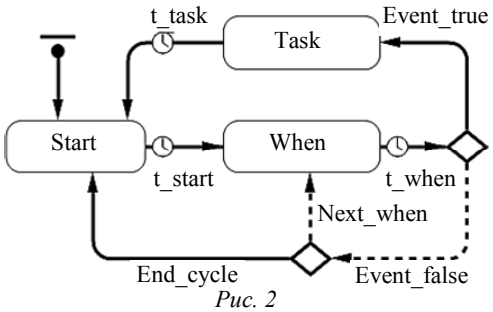

Рис. 2

Настройка условных переходов модели: переход «When→Task» к обработке задачи выполняется при успешном прохождении проверки условия модуля (условие перехода: *состояния.get*(*номерМодуля*)); переход «When→Start» к точке завершения критической секции осуществляется при условии завершения обработки последнего модуля или очередного неприоритетного модуля планировщика (условие: (*номерМодуля==количествоМодулей*) || (*номерМодуля>количествоПрМодулей*)); переход «When→When» к проверке условия следующего модуля планировщика задач осуществляется, когда не выполняется условие перехода к точке входа.

Созданная структура модели в виде диаграммы состояний наглядно показывает работу циклического алгоритма планирования задач реального времени. Инвариантность структуры модели к числу задач планировщика, а также к их приоритетам позволяет создавать модели любого масштаба и соответственно выполнять необходимые для проектирования таких подсистем исследования. Моделирование поведения модели в зависимости от индивидуальных свойств каждого модуля планировщика задач требует применения программных способов управления параметрами модели: номер модуля, текущий номер неприоритетной задачи и др. (таблица).

Следующий этап разработки связан с созданием программ управления параметрами, событиями и сбора результатов моделирования в состояниях и переходах модели.

В начале диаграммы состояний модели выполняется подпрограмма планирования моментов времени наступления первых событий модулей планировщика в соответствии со свойственными им индивидуальными интервалами времени и инициализация коллекций (таблица) сбора результатов функционирования алгоритма планирования задач (рис. 3). Для удобства работы с номерами модулей в программах нулевой элемент (индекс) коллекций модели не задействован. Моменты времени между наступлением событий в модели распределены экспоненциально. Для сбора статистики создаются элементы числа обработанных проверок и задач, а также числа пропущенных из-за задержек обработки модулей задач планировщика. В конце подпрограммы производится предустановка начальных значений номера модуля и номера неприоритетного модуля планировщика.

```
for( int i=0; i<=количествоМодулей; i++ ) {
    // Планирование времени событий для модулей планировщика
    if (i>0){
        double время = exponential(1/интервалыВремени.get(i));
        create_события(время, i);
    }
    // Инициализация элементов сбора статистики по модулям
    обработаноПроверок.add(0);
    обработаноЗадач.add(0);
    пропущеноЗадач.add(0);
    задержкаОбработкиЗадачи.add(new StatisticsDiscrete());
}

// Установка начальных значений номеров модулей
номерМодуля = 1;
if (номерМодуля>количествоПрМодулей)
    номерНпрМодуля = 0;
else
    номерНпрМодуля = количествоПрМодулей+1;
```

Рис. 3

Планирование моментов времени наступления событий в листинге подпрограммы (рис. 3) выполняется с помощью элемента «события». Вызов функции *create_события(time, number)* планирует момент времени наступления события с задержкой *time* для модуля с номером *number*. При наступлении события выполняется подпрограмма (рис. 4). Особенность подпрограммы заключается в обработке сценария 2 функционирования алгоритма. Планирование наступления следующего события модуля при этом осуществляется сразу после наступления текущего события. Также ведется сбор статистики по числу пропущенных из-за задержек задач.

```
// Сценарий 2: планирование события независимо от обработки
if (!сценарий1){
    // События наступают через случайные промежутки времени:
    double время = exponential(1/интервалыВремени.get(номерМодуля));
    // События наступают через равные промежутки времени:
    // double время = интервалыВремени.get(номерМодуля);
    create_события(время, номерМодуля);
    // Сбор статистики по пропущенным задачам.
    //      Условие: предыдущее состояние - наступило
    if (состояния.get(номерМодуля))
        пропущеноЗадач.set(номерМодуля,
            пропущеноЗадач.get(номерМодуля)+1);
}

состояния.set(номерМодуля,true);       // Событие модуля наступило
времяСобытий.set(номерМодуля, time()); // Сохранение момента времени
```

Рис. 4

В состоянии модели Start выполняется подпрограмма расчета значений предполагаемых к обработке абсолютного номера модуля и номера неприоритетного модуля планировщика задач (рис. 5).

```
// Нормировка номера неприоритетного модуля
if ((номерНпрМодуля == количествоМодулей)&&
      (номерМодуля>количествоПрМодулей))
    // Начало обработки с первого Нпр модуля
    номерНпрМодуля = количествоПрМодулей+1;
else
    if (номерМодуля>количествоПрМодулей)
    // Изменение номера только после обработки Нпр модуля
        номерНпрМодуля += 1;

// Нормировка абсолютного номера модуля
if (количествоПрМодулей>0)// Для случая наличия Пр модулей
    номерМодуля = 1;
else                      // Для случая отсутствия Пр модулей
    номерМодуля = номерНпрМодуля;
```

Рис. 5

В состоянии модели When собирается статистика по числу обработанных проверок модулей планировщика: *обработаноПроверок.set(номерМодуля, обработаноПроверок.get(номерМодуля)+1)*.

В состоянии решения задачи Task выполняется подпрограмма сбора статистики для обрабатываемого модуля: количество обработанных задач и задержка обработки задачи, равная разнице моментов времени наступления события и завершения обработки задач. При моделировании сценария 1 планируется момент времени наступления следующего события модуля (рис. 6).

```
обработаноЗадач.set(номерМодуля,
    обработаноЗадач.get(номерМодуля)+1);
состояния.set(номерМодуля, false); // Событие задачи обработано
(задержкаОбработкиЗадачи.get(номерМодуля)).add(time() +
    exponential(1.0/Tstart) - времяСобытий.get(номерМодуля));

// Сценарий 1: планирование нового события после обработки задачи
if (сценарий1)
    create_события(exponential(1.0/Tstart) +
        exponential(1/интервалыВремени.get(номерМодуля)),
            номерМодуля);
```

Рис. 6

При переходе next_when к проверке «When» последующего модуля выполняется перерасчет номера обрабатываемого модуля (рис. 7).

```
// Абсолютный номер модуля
номерМодуля += 1;
// Номер неприоритетного модуля
if (номерМодуля > количествоПрМодулей)
    номерМодуля = номерНпрМодуля;
```

Рис. 7

В настоящей статье описана легко масштабируемая модель циклического алгоритма планирования задач реального времени на основе приоритетов и предложен способ оценки характеристик времени и вероятности успешной обработки задачи. Дальнейшее исследование модели направлено на поиск способов обеспечения заданных вероятностных и временных характеристик обработки задач планировщиком. Для этого планируется постановка экспериментов и исследование характеристик обработки задач планировщика с изменяющимся в широком диапазоне количеством приоритетных задач планировщика. Это позволит оценить целесообразность и аргументированно использовать на практике механизм приоритезации узлов сети. Конечной целью работы является синтез алгоритма построения структуры планировщика задач реального времени с заданными характеристиками.

СПИСОК ЛИТЕРАТУРЫ

1. An Enhanced Scheduling Scheme for QoS in Mobile WiMAX Networks / D. David Neels Pon Kumar, J. Raj, K. Arun Kumar, K. Murugesan // Intern. J. of Engineering and Innovative Technology (IJEIT). 2012. Vol. 2, № 5. P. 163–169.

2. Park W., Cho S., Bahk S. Scheduler Design for Multiple Traffic Classes in OFDMA Networks // Proc. IEEE Int. Conf. Communications. 2006. Vol. 2. P. 790–795.

3. Кавалеров М. В., Матушкин Н. Н. Повышение эффективности планирования с фиксированными приоритетами задач жесткого реального времени на основе применения r-алгоритма для формирования запросов этих задач // Нейрокомпьютеры: разработка, применение. 2014. № 12. С. 14–19.

4. Кавалеров М. В., Матушкин Н. Н. Новый алгоритм назначения параметров задач реального времени с линейными интервальными ограничениями в условиях планирования с фиксированными приоритетами, основанный на сокращенном переборе приоритетов // Нейрокомпьютеры: разработка, применение. 2013. № 11. С. 12–17.

5. Embedded System Software. Module 6. Version 2. URL: http://www.nptel.ac.in/courses/108105057/Pdf/Lesson-30.pdf.

6. Völp M., Härtig H. Real-Time Systems: Event-Driven Scheduling. URL: https://os.inf.tu-dresden.de/Studium/RTS/WS2013/05-EventDriven.pdf.

7. LonTalk protocol specification: ANSI/CEA-709.1-B. URL: http://www.enerlon.com/JobAids/Lontalk%20Protocol%20Spec.pdf.

8. Дитрих Д., Лой Д., Швайнцер Г. Ю. LON-технология, построение распределенных приложений / пер. с нем.; под ред. О. Б. Низамутдинова. Пермь: Звезда, 1999. 242 с.

9. Miśkowicz M. Latency characteristics of event-driven task scheduler embedded in neuron chip // IJCSNS Intern. J. of Computer Science and Network Security. 2007. Vol. 7, № 12. P. 132–149.

10. Даденков С. А., Чмыков В. В. Разработка имитационной модели уровня приложений узла LonWorks-сети // Автоматизированные системы управления и информационные технологии: материалы краев. науч.-техн. конф. Пермь: Изд-во Перм. нац. исслед. политехн. ун-та, 2014. С. 278–282.

11. Даденков С. А., Чмыков В. В. К вопросу анализа конфигурационных свойств узлов как фактора, влияющего на производительность сети LonWorks // Науч. тр. Sworld. 2014. Т. 5, № 1. С. 84–90.

12. Даденков С. А. Анализ значимости факторов коммуникационного стека протоколов LonTalk для разработки адекватных моделей LonWorks-сетей // Тр. XII Всерос. совещ. по проблемам управления. М.: ИПУ РАН, 2014. С. 8527–8535.

13. Даденков С. А., Кон Е. Л. Анализ моделей и методов агентного и дискретно-событийного имитационного моделирования // Изв. СПбГЭТУ «ЛЭТИ». 2015. № 5. С. 35–41.

S. A. Dadenkov, A. N. Kokoulin, E. L. Kon
Perm national research polytechnical university

CREATION OF A MODEL CYCLIC ALGORITHM FOR PLANNING REAL TIME TASKS

The work consists in the development of a simulation model cyclic algorithm for scheduling tasks based on priorities (round robin priority driven preemptive scheduler). The algorithm is widely used on the nodes-sensors, controllers, -multipliers of distributed fieldbus networks soft real time, including LonWorks and BacNet. Model differs from the known analytical and simulation analogs by taking into account the parameters of the algorithm: individual numbers, priorities, configuration parameters tasks. Model is created in the AnyLogic simulation system and is supplemented with elements for collecting and analyzing simulation results – probabilistic and temporal characteristics of task processing. The structure of the model is constructed using the graphical toolkit of state diagrams and transitions in conjunction with program extension of model. This ensured the construction of an easily scalable and correct model. The model will be used to design the scheduler structures of real-time tasks.

Cyclic task scheduling algorithm, real-time, industrial fieldbus networks, simulation, probabilistic and time characteristics