УДК 004.4'2
DOI: 10.15827/0236-235X.120.574-582
Дата подачи статьи: 21.07.17
2017. Т. 30. № 4. С. 574–582

ПРОГРАММНОЕ СРЕДСТВО МОДЕЛИРОВАНИЯ МОДУЛЬНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ ДЛЯ ПРОВЕРКИ ДОПУСТИМОСТИ ИХ КОНФИГУРАЦИЙ

А.Б. Глонина, программист, alevtina@lvk.cs.msu.su
(Московский государственный университет им. М.В. Ломоносова, Ленинские горы, г. Москва, 119991, Россия)

В работе представлено программное средство для проверки критерия допустимости конфигураций *модульных вычислительных систем* (МВС) реального времени. В качестве примера таких систем рассматриваются системы интегрированной модульной авионики. Конфигурация МВС реального времени считается допустимой, если для нее все работы всех вычислительных задач выполняются в рамках директивных сроков. На основе анализа задач, возникающих при проектировании систем, были сформулированы требования к средству проверки критерия допустимости конфигурации. Обзор существующих программных средств показал, что ни одно из них не удовлетворяет всем требованиям, поэтому было принято решение о разработке собственного средства.

Разработанное программное средство позволяет моделировать МВС реального времени и получать временные диаграммы их функционирования, необходимые для проверки критерия допустимости. Благодаря выбранному для моделирования математическому аппарату (сетям временных автоматов с остановкой таймеров) корректность построенных моделей формально доказана. Процесс построения и прогона модели для конкретной конфигурации полностью автоматизирован, поэтому разработанное средство может быть использовано в цикле работы алгоритмов поиска оптимальных конфигураций.

Программное средство было интегрировано с САПР планирования вычислений в МВС реального времени, используемой в промышленности, и апробировано на данных, приближенных к реальным. Эксперименты показали, что разработанное средство применимо на практике и для конфигураций с длительным интервалом планирования превосходит по быстродействию встроенную в САПР модель МВС. Кроме того, предложенное средство удовлетворяет всем сформулированным ранее требованиям в отличие от встроенной в САПР модели, удовлетворяющей им лишь частично.

Ключевые слова*: сети временных автоматов, интегрированная модульная авионика, имитационное моделирование, планирование вычислений, верификация, корректность.*

На сегодняшний день модульная архитектура является наиболее перспективным типом архитектур встроенных вычислительных систем. В данной работе *модульные вычислительные системы* (МВС) *реального времени* (РВ) рассматриваются на примере систем *интегрированной модульной авионики* (ИМА) [1], однако предложенный подход применим и к другим модульным архитектурам, например к AUTOSAR [2] и DECOS [3].

МВС РВ состоит из нескольких стандартизированных вычислительных модулей, каждый из которых содержит набор многоядерных процессоров. Процессоры могут иметь различную производительность. Вычислительные модули связаны коммутируемой сетью с виртуальными каналами.

Рабочая нагрузка МВС РВ представлена набором разделов. Разделом называется группа логически связанных задач, как правило, соответствующих одному приложению и взаимодействующих посредством общей памяти. Все задачи являются периодическими. За период должен выполниться экземпляр задачи, называемый работой. Между задачами с одинаковым периодом могут существовать зависимости по данным: очередная работа задачи-получателя не может начать выполнение до тех пор, пока не получит данные от всех соответствующих работ задач-отправителей. Каждая работа должна завершить выполнение не позднее определенного для нее директивного срока, который не превышает начала следующего периода. Если директивный срок наступает до завершения выполнения работы, она считается опоздавшей и далее выполняться не может.

Каждый раздел привязан к вычислительному ядру некоторого процессора. При этом несколько разделов могут быть привязаны к одному и тому же ядру. На интервале планирования для ядра задается статическое расписание временных интервалов, называемых окнами, в течение каждого из которых выполняются работы определенного раздела. Каждый раздел имеет собственный планировщик, управляющий выполнением работ внутри окон и работающий согласно одному из алгоритмов планирования. Допускается использование разных алгоритмов планирования для разных разделов. Одним из наиболее часто применяемых алгоритмов планирования является планирование с вытеснением на основе статических приоритетов.

Конфигурация МВС РВ определяет набор ее модулей, характеристики рабочей нагрузки, привязку разделов к ядрам и расписания окон для каждого ядра. Конфигурация является *допустимой*, если для нее все работы полностью выполняются в рамках своих директивных сроков.

В процессе проектирования МВС РВ некоторые составляющие конфигурации варьируются и ана-

лизируется множество потенциальных конфигураций. При этом возникает задача проверки допустимости конкретных конфигураций.

Существующие аналитические методы решения подобных задач (например [4]) не учитывают всех особенностей организации МВС РВ, а метод формальной верификации модели (например [5]) системы имеет вычислительную сложность, экспоненциально зависящую от количества задач, и поэтому не применим на практике.

Еще одним методом проверки допустимости конфигураций МВС РВ является построение и анализ *временной диаграммы* (ВД) ее функционирования. В данной работе представлено инструментальное средство, реализующее этот метод.

Требования к средству моделирования

Автором проанализирован ряд задач, возникающих при проектировании МВС РВ и требующих проверки допустимости конфигураций, например, задача поиска конфигурации со сбалансированным распределением разделов по ядрам и минимальной нагрузкой на вычислительную среду [6], задача поиска конфигурации с минимальным числом переключений контекста [7] и др. По результатам данного анализа сформулированы следующие требования к средствам моделирования таких систем.

1. *Возможность получения ВД, включающих события постановки на выполнение, вытеснения и завершения работ.* Такой ВД достаточно для проверки критерия допустимости конфигурации.

2. *Применимость системы для широкого класса МВС РВ.* В частности, существуют операционные системы для МВС РВ (например VxWorks653, см. [8]), поддерживающие использование различных стратегий планирования в разных разделах. Средство моделирования должно позволять анализировать такие системы.

3. *Возможность формальной проверки корректности моделей.* Такая проверка необходима, так как на этапе проектирования МВС РВ невозможно экспериментально сравнить поведение модели с поведением реальной МВС РВ.

4. *Автоматизация построения и прогона модели.* На ранних стадиях проектирования могут варьироваться многие составляющие конфигурации (число и тип модулей, приоритеты задач, привязка разделов к ядрам и т.д.), поэтому число потенциальных конфигураций очень велико. Чтобы выбрать оптимальную по некоторому критерию конфигурацию, используются алгоритмы, перебирающие множество конфигураций, для каждой из которых должен быть проверен критерий допустимости. Для использования системы моделирования в цикле работы таких алгоритмов процессы построения модели по описанию конфигурации, прогона модели и анализа результатов моделирования должны быть полностью автоматизированы.

5. *Возможность включения в модель МВС РВ пользовательских моделей ее компонентов.* В процессе проектирования может потребоваться проанализировать конфигурации, включающие планировщики, функционирующие согласно некоторым новым алгоритмам планирования, новые среды передачи данных и т.п. Для этого у проектировщика должна быть возможность самостоятельно создавать модели новых компонентов системы.

6. *Скорость* построения и прогона модели, а также анализа результатов моделирования должна позволять работать с конфигурациями размерности, соответствующей реальным системам.

Автором проведен обзор ряда средств моделирования МВС РВ, в результате которого не было выявлено ни одного, в полной мере удовлетворяющего перечисленным требованиям. Так, некоторые средства поддерживают наличие лишь одного окна на интервале планирования [9], другие работают с ограниченным набором алгоритмов планирования и не позволяют задавать статическое расписание окон [10], моделировать вытеснение [11, 12] или поддерживают лишь один алгоритм планирования [13, 14]. Анализ средств, исходный код которых находится в открытом доступе [10, 12, 16], показал, что их доработка для поддержки всех перечисленных требований значительно более трудоемка, чем создание нового средства, поэтому было принято решение о разработке собственного средства моделирования МВС РВ.

Математическая модель функционирования МВС РВ

Разрабатываемое средство должно предоставлять возможность формальной проверки корректности моделей, поэтому прежде всего необходимо было выбрать соответствующий математический аппарат. Также математический аппарат должен позволять моделировать все описанные выше аспекты функционирования МВС РВ (в том числе вытеснение и одновременную работу нескольких планировщиков). Автором был проанализирован ряд математических аппаратов и выбран аппарат сетей временных автоматов с остановкой таймеров как позволяющий описывать и формально верифицировать модели МВС РВ, что реализовано в существующих программных средствах [17].

Опишем кратко выбранный математический аппарат. Подробное описание приведено в [17].

Автомат с остановкой таймеров – это конечный автомат с целочисленными переменными и таймерами. Графически такой автомат может быть представлен размеченным ориентированным графом, вершины которого называются локациями, а дуги – переходами. Состоянием автомата называется совокупность его текущей локации, значений переменных и таймеров. Таймер – это специальная переменная, принимающая вещественные значения.

Таймер считается активным, если его условие активности (булево выражение над множеством переменных) в текущей локации автомата истинно. В противном случае таймер считается остановленным.

Для автомата указывается начальная локация. Каждой локации сопоставлен инвариант – булево выражение над таймерами и переменными. Автомат может оставаться в данной локации до тех пор, пока инвариант этой локации имеет истинное значение.

Каждый переход характеризуется условием перехода, действиями и, возможно, синхронизацией. Переход активен, если автомат находится в локации, предшествующей этому переходу, значения переменных и таймеров удовлетворяют условию перехода, и инвариант локации-назначения перехода имеет истинное значение. Переход может (но не обязан) быть совершен, если он активен и может быть выполнена синхронизация. При совершении перехода меняется текущая локация автомата, происходит синхронизация и выполняются действия по изменению переменных и таймеров.

Состояние автомата может измениться в результате не только выполнения перехода, но и синхронного увеличения значений всех таймеров.

Сеть автоматов представляет собой набор автоматов, функционирующих синхронно и взаимодействующих посредством общих переменных и каналов.

Канал – средство синхронизации автоматов. Синхронизация характеризуется именем канала и действием. Существуют два парных действия: отправка и прием сигнала по каналу. Если в некоторый момент времени в двух автоматах сети активны переходы с парными действиями синхронизации по одному и тому же каналу, то эти переходы выполняются одновременно.

Конечная или бесконечная последовательность состояний, получаемая при некотором сценарии функционирования сети автоматов, называется вычислением этой сети.

На рисунке 1 приведен пример автомата, моделирующего канал передачи данных с фиксированной задержкой на передачу сообщения заданного размера. Начальной локацией автомата является локация Idle, соответствующая отсутствию передачи данных по каналу. Началу передачи данных соответствуют синхронизация с автоматом, моделирующим отравителя данных, по каналу send и переход в локацию Transmitting. При этом происходит обнуление таймера time. Как только значение таймера достигнет значения, равного задержке на передачу данного сообщения (data.delay), происходит переход в локацию TransmittingFinished с присваиванием переменной is_data_ready (эта переменная служит для взаимодействия с автоматом-моделью получателя данных) значения 1. После этого без продвижения времени (локация TransmittingFinished помечена символом «C») происходят синхронизация с моделью получателя по каналу receive и возвращение в начальную локацию. Таймер, служащий для моделирования задержки на передачу данных, неактивен (обозначается как «time'==0») в локации Idle и активен («time'==1») в локации Transmitting. Данный автомат построен в редакторе средства UPPAAL, используемого для построения и верификации автоматов.

Изначально формализм сетей временных автоматов с остановкой таймеров применялся для формальной верификации моделей, не требующей построения временных диаграмм. Чтобы описать метод построения ВД МВС РВ по вычислению сети автоматов (требование 1), введем дополнительные определения. *Модельное время сети автоматов* – значение некоторого служебного таймера, условие активности которого всегда истинно и который никогда не обнуляется. *Событие синхронизации* – тройка <*CH*, *A*, *t*> , где *CH* – канал, *A* – набор автоматов, участвующих в синхронизации, *t* – модельное время. *ВД сети автоматов* – набор событий синхронизации для некоторого вычисления сети.

Для описания предлагаемой модели потребуется еще несколько определений. *Конкретный тип автомата (или параметризованный автомат)* – это автомат, в арифметических и логических выражениях которого вместо чисел используются параметры с незаданными значениями. *Интерфейс автомата* – набор переменных и синхронизаций, посредством которых автомат взаимодействует с другими автоматами. *Базовый тип автоматов* определяется только интерфейсом. Конкретный тип автоматов *реализует* базовый тип, если интерфейс конкретного типа расширяет интерфейс базового типа.

Набор базовых типов автоматов с заданными логическими связями назовем *обобщенной сетью автоматов.* Логические связи определяют, какие базовые типы автоматов могут взаимодействовать

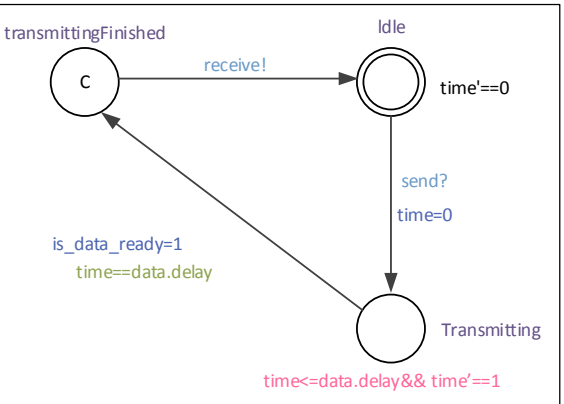

Рис. 1. Пример временного автомата (модель виртуального канала)

Fig. 1. An example of a time automaton (virtual channel model)

и с помощью каких средств (переменных и синхронизаций). *Конкретная сеть автоматов* — набор конкретных типов автоматов с заданными логическими связями. Конкретная сеть автоматов *реализует* обобщенную сеть автоматов, если каждый базовый тип автоматов обобщенной сети реализуется одним или несколькими конкретными типами автоматов конкретной сети, и логические связи в базовой сети соответствуют логическим связям в конкретной сети.

В данной работе предлагается обобщенная модель МВС РВ, представляющая собой обобщенную сеть временных автоматов следующих базовых типов:

– CS, моделирующий планировщик ядра; планировщик ядра переключает окна разделов согласно статическому расписанию;

– TS, моделирующий планировщик работ раздела;

– T, моделирующий задачу; директивный срок не превышает периода задачи, поэтому в каждый момент времени задаче соответствует только одна работа и для моделирования всех работ некоторой задачи достаточно одного автомата;

– L, моделирующий виртуальный канал.

Логические связи между перечисленными базовыми типами автоматов показаны на рисунке 2. Интерфейс базового типа автоматов T содержит в том числе синхронизации по каналам exec, preempt и finished, соответствующие постановке на выполнение, вытеснению и завершению (вследствие окончания выполнения или наступления директивного срока) текущей работы задачи.

Отметим, что представленная обобщенная модель специфицирует лишь интерфейсы базовых типов автоматов и логические связи между ними. Для создания конкретной сети автоматов, реализующей данную базовую сеть, необходимо создать как минимум по одному конкретному типу автоматов, реализующему каждый базовый тип автоматов. Автором были разработаны конкретные типы автоматов, реализующие базовый тип TS и соответствующие наиболее распространенным на практике планировщикам разделов (со статическими и динамическими приоритетами, с вытеснением и без), а также конкретные типы автоматов, реализующие базовые типы CS, T и L. Необходимо отметить, что в конкретную сеть автоматов могут быть добавлены и другие (пользовательские) конкретные типы автоматов, реализующие перечисленные базовые типы, то есть имеющие соответствующий интерфейс. Таким образом, предложенная структура модели удовлетворяет требованию 5.

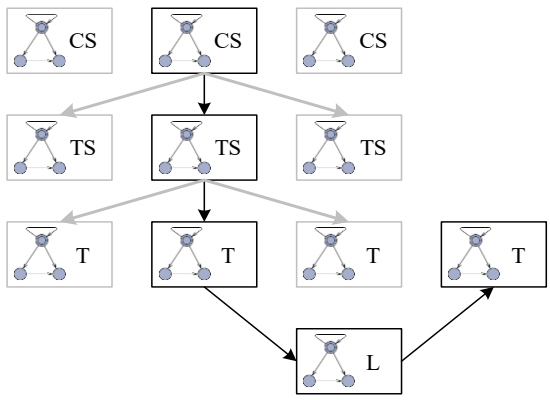

Рис. 2. Структура предложенной обобщенной сети автоматов

Fig. 2. The proposed general automata network structure

Конкретная сеть автоматов, реализующая описанную обобщенную сеть, является параметризованной моделью МВС РВ. На основе этой модели по описанию конфигурации конкретной МВС РВ однозначным образом строится экземпляр модели МВС РВ, соответствующий этой конфигурации: для каждого компонента МВС РВ создается автомат конкретного типа, в соответствии со связями компонентов МВС РВ создаются средства синхронизации автоматов, параметры каждого автомата инициализируются соответствующими числовыми значениями, а элементы интерфейса – созданными средствами синхронизации. Данный процесс может быть полностью автоматизирован, что позволило разработать средство, удовлетворяющее требованию 4.

По ВД сети автоматов, моделирующей функционирование МВС РВ заданной конфигурации, можно однозначно построить ВД функционирования такой МВС РВ, включающую постановку на выполнение, вытеснение и завершение работ (что необходимо согласно требованию 1). ВД функционирования МВС РВ строится следующим образом. Изначально искомая ВД не содержит никаких событий. Далее для каждого события синхронизации по каналам exec, preempt и finished из ВД сети автоматов в создаваемую ВД добавляется событие постановки на выполнение, вытеснение или завершение работы. Время данного события равно модельному времени синхронизации, работа-источник события однозначно определяется идентификатором задачи, которую моделирует автомат-участник синхронизации, и модельным временем синхронизации.

Корректность построенных моделей

При проектировании МВС РВ отсутствует возможность экспериментального сравнения поведения построенных моделей с поведением целевых систем и их компонентов. Поэтому корректность моделей должна быть доказана формально (требование 3).

Спецификации МВС РВ содержат ряд требований корректности к функционированию системы в целом и ее компонентов. Эти требования представляют собой ограничения на последовательности

событий, происходящих в системе, и на длительности интервалов между определенными типами событий. Построенные модели также должны удовлетворять тем из требований, которые применимы на рассматриваемом уровне абстракции.

Выполнение требований к моделям компонентов (то есть к параметризованным автоматам) предлагается проверять автоматически с помощью верификатора UPPAAL и подхода «автоматов-наблюдателей» [18]. Для проверки корректности компонента параметризованной модели необходимо выделить из спецификаций МВС РВ применимые к нему на выбранном уровне абстракции требования, для каждого из них построить автомат-наблюдатель и проверить недостижимость его специальной плохой локации. Такая проверка была выполнена автором для всех разработанных конкретных типов автоматов и требований, выделенных из стандарта ARINC653 [19] на системы ИМА.

Далее приведен пример требования к базовому типу автоматов TS, моделирующему планировщик работ, которое должно быть выполнено для всех конкретных типов автоматов, реализующих TS:

– *для любого раздела системы верно, что в каждый момент времени в нем выполняется не более одной работы.*

Проверка требований к параметризованной модели МВС РВ в целом не может быть выполнена автоматически, так как количество автоматов, входящих в модель, является ее параметром и в общем случае неизвестно. Поэтому доказательство выполнения требований к модели в целом было выполнено вручную посредством строгих логических рассуждений и основано на доказанном ранее выполнении требований корректности к отдельным компонентам модели.

Примером требований к модели в целом является следующее условие:

– *для любого процессорного ядра системы верно, что в каждый момент времени на нем выполняется не более одной работы.*

Выполнение данного требования напрямую следует из выполнения следующих требований к моделям компонентов МВС, которые были доказаны с использованием соответствующих автоматов-наблюдателей:

– *для любого раздела системы верно, что в каждый момент времени в нем выполняется не более одной работы;*

– *для любого процессорного ядра системы верно, что в каждый момент времени на нем выполняется не более одного раздела*.

Аналогичным образом доказывается и детерминированность модели, то есть однозначность построения ВД сети автоматов для фиксированной конфигурации.

Таким образом, между конфигурацией МВС и ее моделью существует однозначное соответствие, модель детерминирована и корректна, между ВД модели и ВД МВС также существует однозначное соответствие. Следовательно, проверка критерия допустимости конфигураций с помощью предложенной модели корректна.

Программная реализация

Предложенные методы и средства были реализованы в виде программного средства, схема работы которого представлена на рисунке 3.

Все разработанные и верифицированные модели компонентов МВС РВ образуют библиотеку автоматных моделей. Каждая модель компонента представляет собой параметризованный автомат, описанный в формате, совместимом со средством UPPAAL [17].

В открытых источниках не было найдено средство моделирования сетей временных автоматов с остановкой таймеров, позволяющее автоматически

Рис. 3. Схема работы реализованного программного средства

Fig. 3. An operation scheme of the implemented software tool

получать вычисления таких сетей. UPPAAL предоставляет для этого лишь графический интерфейс, требующий ручных действий пользователей, что не соответствует требованию 4. Поэтому автором была реализована на языке C++ собственная библиотека моделирования сетей автоматов. С использованием средства генерации кода по шаблонам Cheetah [20] был реализован транслятор, преобразующий автоматные модели в их представления на C++. Полученные с помощью этого транслятора модели формируют библиотеку программных моделей компонентов МВС РВ.

Параметризованная программная модель МВС РВ представляет собой программу, выполнение которой состоит из трех этапов:

– построение с использованием библиотеки моделей компонентов МВС РВ экземпляра модели МВС РВ по описанию конфигурации системы (описание конфигурации представляет собой файл формата XML, содержащий информацию о количестве, типах, числовых характеристиках и связях компонентов системы);

– прогон построенной модели, в процессе которого формируется ВД сети автоматов;

– построение по ВД сети автоматов искомой ВД МВС РВ и ее сохранение в виде XML-файла.

Библиотека автоматных моделей компонентов МВС РВ может быть пополнена пользовательскими моделями. Добавляемая модель должна представлять собой конкретный тип автомата, реализующий один из описанных выше базовых типов автоматов. Для добавления новой модели в библиотеку необходимо описать соответствующий параметризованный автомат в графическом редакторе UPPAAL, а затем проверить для этого автомата с использованием верификатора выполнение всех требований, определенных для соответствующего базового типа. Набор требований для каждого базового типа автоматов составлен автором работы и представляет собой файл формата, совместимого с UPPAAL, в котором заданы соответствующие автоматы-наблюдатели. Кроме того, пользователь, построив собственные автоматы-наблюдатели, может проверить выполнение ряда требований, специфичных для созданного им конкретного типа автомата. Если все необходимые требования выполнены, новая модель добавляется в библиотеку автоматных моделей, после чего автоматически с помощью транслятора создается соответствующая программная модель.

Апробация

Для апробации разработанного средства была выполнена его интеграция с используемой в отечественной промышленности САПР планирования вычислений в МВС [13].

Данная САПР создана для решения следующей задачи. Задается описание вычислительной системы: набор модулей; набор типов используемых процессоров; для каждого модуля – число и тип процессоров; для каждого процессора – число ядер, время инициализации окна и переключения контекста между разделами; для каждого ядра – наибольшее допустимое значение загрузки. Также задается описание рабочей нагрузки: набор задач; для каждой задачи – период, начальное значение приоритета (начальные значения приоритетов нескольких задач одного раздела могут совпадать), время выполнения на каждом из типов процессоров; набор разделов; для каждого раздела – набор принадлежащих ему задач и набор ядер, к которым может быть привязан данный раздел; набор сообщений; для каждого сообщения – отправитель, получатель, размер и максимальные задержки на передачу через бортовую сеть и через память модуля. Необходимо найти привязку разделов к ядрам, расписание окон разделов для каждого ядра и уточненные значения приоритетов задач, уникальные в рамках раздела. При этом объем сообщений, передаваемых через бортовую сеть в течение интервала планирования, должен быть минимальным. Иными словами, среди множества конфигураций с перечисленными во входных данных фиксированными составляющими необходимо найти конфигурацию, минимизирующую нагрузку на бортовую сеть. При этом найденная конфигурация должна удовлетворять критерию допустимости.

Для решения описанной задачи в САПР используются различные оптимизационные алгоритмы (жадный, генетический, метод ветвей и границ). В процессе работы алгоритма оптимизации формируется некоторая конфигурация, для которой требуется проверить критерий допустимости. Если критерий выполнен, конфигурация может быть принята в качестве решения задачи. В противном случае конфигурация отбрасывается и поиск продолжается.

В САПР существует встроенная имитационная модель для проверки критерия допустимости. Однако для нее не имеется формального обоснования корректности, и в ее основе нет математического аппарата, который позволил бы такое обоснование выполнить. Кроме того, модель не является расширяемой и рассчитана на решение лишь конкретной задачи. Поэтому целесообразно заменить ее на разработанное средство, не обладающее указанными недостатками.

Схема интеграции разработанной модели с САПР изображена на рисунке 4. На этапе работы поискового алгоритма, требующем проверки критерия допустимости некоторой конфигурации, генерируется XML-файл с описанием этой конфигурации. После этого запускается параметризованная программная модель, принимающая на вход сгенерированный файл. Результатом работы модели является XML-файл с ВД функционирования системы. Этот файл разбирается в САПР, и на основе

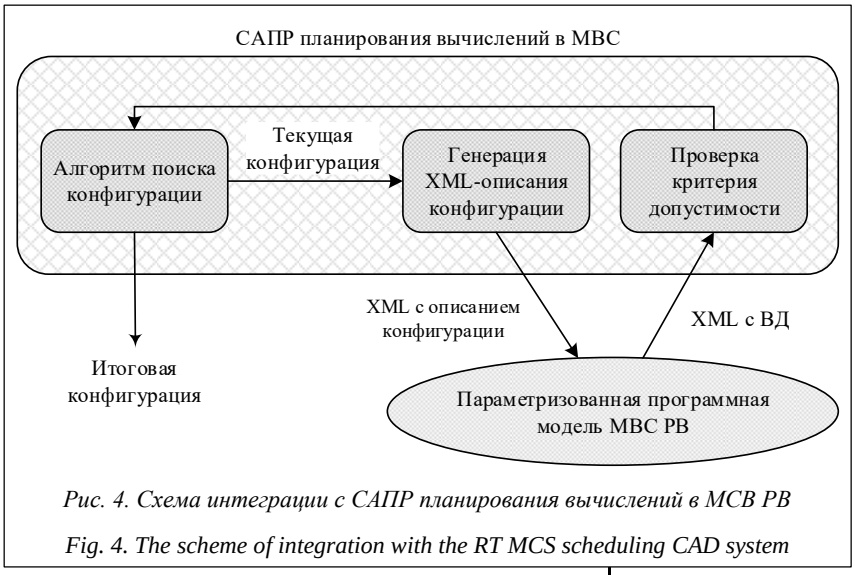

Рис. 4. Схема интеграции с САПР планирования вычислений в МСВ РВ

Fig. 4. The scheme of integration with the RT MCS scheduling CAD system

его данных принимается решение о допустимости конфигурации.

Фрагмент ВД, построенной с использованием разработанного средства моделирования и визуализированной средствами САПР, приведен на рисунке (см. http://www.swsys.ru/uploaded/image/2017_4/2017-4-dop/7.jpg).

В процессе апробации было проведено сравнение ВД, построенных с помощью разработанного средства, с ВД, построенными с помощью встроенной модели САПР, на различных наборах данных. ВД, построенные обоими средствами, совпадают.

В таблице приведено сравнение времени работы двух моделей на трех наборах данных. Набор 2 соответствует данным по одной из реальных МВС РВ авиационного назначения (3 модуля, всего 6 процессорных ядер, 10 разделов, 160 задач, 100 виртуальных каналов, около 12 500 работ на интервале планирования). Набор 1 получен из набора 2 исключением половины разделов (общее количество задач и сообщений при этом сократилось приблизительно в 2 раза). Набор 3 получен из набора 2 удвоением количества процессорных ядер и разделов (общее количество задач и сообщений при этом также удвоилось).

Сравнение времени работы моделей на различных наборах данных
Comparison of model operation times at different data sets

Модель	Набор 1	Набор 2	Набор 3
Модель САПР	0,55	4,6	18,3
Предложенная модель	1,22	7,1	27

Эксперименты показали, что время работы предложенной модели превышает время работы встроенной в САПР модели. Однако на реальных данных время работы каждой из моделей не превышает 10 секунд, что приемлемо на практике. Кроме того, с увеличением рабочей нагрузки отношение длительностей работы моделей уменьшается: так, на наборе 1 предложенная модель работает в 2,2 раза дольше, а на наборе 3 – только в 1,5 раза.

На рисунке 5 приведено сравнение времени работы моделей в зависимости от длительности интервала планирования для фиксированного набора данных, соответствующего реальной МВС РВ. На практике для данного набора используется интервал планирования, равный двум секундам. Эксперименты показали, что на коротких интервалах планирования встроенная модель САПР работает незначительно быстрее, однако, начиная с интервала, равного четырем секундам, предложенная модель начинает превосходить по скорости модель САПР. При этом с увеличением длительности интервала планирования превосходство растет.

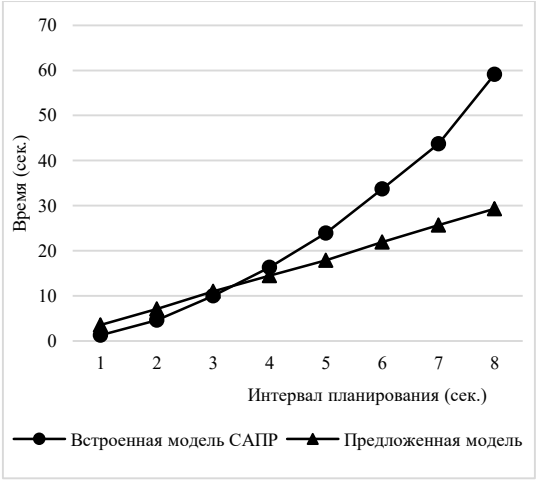

Рис. 5. Сравнение времени работы моделей в зависимости от длительности интервала планирования

Fig. 5. Comparison of model operation times for different scheduling intervals

Таким образом, эксперименты показали, что разработанные методы и средства применимы на практике. Особенный интерес представляет их использование на конфигурациях большой размерности, включающих большое количество работ и длинные интервалы планирования.

Заключение

В работе была рассмотрена задача проверки допустимости конфигураций МВС РВ и сформулированы требования к инструментальным средствам решения этой задачи. Автором разработано средство моделирования МВС РВ, которое, в отличие

от описанных в открытых источниках других средств, удовлетворяет всем сформулированным требованиям. Разработанное средство позволяет получать временные диаграммы функционирования МВС РВ, необходимые для проверки допустимости конфигураций. При этом процесс построения и прогона модели для заданной конфигурации полностью автоматизирован. Разработана библиотека моделей компонентов МВС РВ, которая при необходимости может быть пополнена пользовательскими моделями. Выбранный для моделирования математический аппарат позволил обосновать корректность класса моделей, синтезируемых согласно предложенному методу. Представленное средство интегрировано с используемой в промышленности САПР планирования вычислений в МВС и успешно апробировано.

Возможными направлениями дальнейших исследований являются расширение библиотеки моделей компонентов МВС РВ, исследование моделей на данных большой размерности и интеграция с другими средствами планирования вычислений.

Работа выполнена при финансовой поддержке РФФИ, грант № 17-07-01566.

Литература

1. Федосов Е.А., Косьянчук В.В., Сельвесюк Н.И. Интегрированная модульная авионика // Радиоэлектронные технологии. 2015. Вып. 1. С. 66–71.

2. AUTOSAR. Enabling Innovation. URL: http://www.autosar.org/ (дата обращения: 13.07.2017).

3. Obermaisser R., Peti P., Huber B., El Salloum C. DECOS: an integrated time-triggered architecture. Elektrotechnik und Informationstechnik, 2006, vol. 123, no. 3, pp. 83–95.

4. Marinescu S., Tamas-Selicean D., Acretoaie V., et al. Timing analysis of mixed-criticality hard real-time applications implemented on distributed partitioned architectures. Proc. ETFA, Krakow, Poland, 2012, pp. 1–4.

5. Macariu G., Cretu V. Timed automata model for component-based real-time systems. Proc. IEEE Intern. Conf. and Workshops on Eng. of Comp. Based Syst., Oxford, UK, 2010, pp. 121–130.

6. Zhou T., Xiong H., Zhang Z. Hierarchical resource allocation for integrated modular avionics systems. Jour. of Syst. Eng. and Electronics, 2011, vol. 22, no. 5, pp. 780–787.

7. Третьяков А. Автоматизация построения расписаний для периодических систем реального времени // Тр. ИСП РАН. 2012. Т. 22. С. 375–400.

8. Wind River VxWorks 653 platform 2.4 and 2.5. URL: http://www.windriver.com/products/productnotes/vxworks-653-product-note.pdf (дата обращения: 13.07.2017).

9. Singhoff F., Plantec A., Dissaux P., et al. Investigating the usability of real-time scheduling theory with the Cheddar project. Real-Time Systems, 2009, vol. 43, no. 3, pp. 259–295.

10. Craveiro J.P., Silveira R.O., Rufino J. HsSim: an extensible interoperable object-oriented n-level hierarchical scheduling simulator. Proc. WATERS, Pisa, Italy, 2012, pp. 9–14.

11. Смелянский Р.Л. Модель функционирования распределенной вычислительной системы с временем // Программирование. 2013. Т. 39. №. 5. С. 22–34.

12. Smelyansky R.L., Bakhmurov A.G., Volkanov D.Yu., Chemeritskii E.V. Integrated environment for the analysis and design of distributed real-time embedded computing systems. Programming and Computer Software, 2013, vol. 39, no. 5, pp. 242–254.

13. Balashov V.V., Balakhanov V.A., Kostenko V.A. Scheduling of computational tasks in switched network-based IMA systems. Proc. Intern. Conf. on Eng. and Applied Sc. Optimization, NTUA, Athens, Greece, 2014, pp. 1001–1014.

14. Wang D., Han J., Ma D., Zhao X. Studying on ARINC653 Partition run-time scheduling and simulation. Proc. WASET, 2012, no. 71, pp. 1583–1587.

15. Dissaux P., Marc O., Fotsing C., et al. The SMART project: multi-agent scheduling simulation of real-time architectures. Embedded Real Time Software and Systems, 2014, pp. 1–9.

16. Khoroshilov A., Albitskiy D., Koverninskiy I., et al. AADL-based toolset for IMA system design and integration. SAE Intern. Jour. of Aerospace, 2012, vol. 5, no. 2, pp. 294–299.

17. Bengtsson J., Yi W. Timed automata: semantics, algorithms and tools. Lectures on Concurrency and Petri Nets, LNCS, Springer, 2004, vol. 3098, pp. 87–124.

18. Andre E. Observer patterns for real-time systems. Proc. ICECCS, 2013, pp. 125–134.

19. Avionics application software standard interface. ARINC specification 653. Aeronautical Radio. Annapolis, 1997.

20. Cheetah. The Python-Powered Template Engine. URL: http://cheetahtemplate.sourceforge.net/ (дата обращения: 13.07.2017).

Software & Systems
DOI: 10.15827/0236-235X.120.574-582

Received 21.07.17
2017, vol. 30, no. 4, pp. 574–582

MODULAR COMPUTER SYSTEMS SIMULATION SOFTWARE TOOL FOR CHECKING FEASIBILITY OF THEIR CONFIGURATIONS

A.B. Glonina [1], *Programmer, alevtina@lvk.cs.msu.su*

[1] *Lomonosov Moscow State University, Leninskie Gory, Moscow, 119991, Russian Federation*

Abstract. The paper presents a software tool for checking feasibility of real-time modular computer systems (RT MCS) configuration. Integrated modular avionics (IMA) systems are considered as an example of RT MCS. An RT MCS configuration is feasible if all works for all computational tasks complete within their deadlines. The author formulates a set of requirements to configurations feasibility checking tool based on analysis of RT MCS design problems. The review of the existing

software tools has shown that none of them satisfies all the requirements. Thus, it has become necessary to develop our own tool that would satisfy all the requirements.

The proposed software tool allows simulating RT MCS and obtaining timing diagrams of their operation, which are necessary for feasibility checking. The correctness of the developed models was formally proven due to the chosen mathematical formalism (stopwatch time automata networks). As a model for a given configuration can be built and run automatically, our tool can be used together with the optimal configurations search algorithm.

The tool was integrated with the scheduling CAD system into the industrial RT MCS and tested on realistic datasets. The experiments showed that the tool is suitable for practical use. Moreover, it is more efficient then a MCS model that is built into a CAD system for the configurations with long scheduling interval. Furthermore, in contrast with the proposed tool, the CAD system model does not satisfy all the requirements.

Keywords: time automata networks, integrated modular avionics, simulation, scheduling, model checking, feasibility, correctness.

Acknowledgements. *This work has been financially supported by RFBR grant no. 17-07-01566.*

References

1. Fedosov E.A., Kosyanchuk V.V., Selvesyuk N.I. Integrated Modular Avionics. *Radioelektronnye tekhnologii* [Radioelectronic Technologies]. 2015, no. 1, pp. 66–71 (in Russ.).

2. *AUTOSAR. Enabling Innovation*. Available at: http://www.autosar.org/ (accessed July 13, 2017).

3. Obermaisser R., Peti P., Huber B., El Salloum C. DECOS: an integrated time-triggered architecture. *Elektrotechnik und Informationstechnik*. 2006, vol. 123, no. 3, pp. 83–95.

4. Marinescu S., Tamas-Selicean D., Acretoaie V., Pop P. Timing analysis of mixed-criticality hard real-time applications implemented on distributed partitioned architectures. *Proc. ETFA*. 2012, Krakow, Poland, pp. 1–4.

5. Macariu G., Cretu V. Timed automata model for component-based real-time systems. *Proc. IEEE Int. Conf. and Workshops on Engineering of Computer Based Systems*. 2010, Oxford, UK, pp. 121–130.

6. Zhou T., Xiong H., Zhang Z. Hierarchical resource allocation for integrated modular avionics systems. *Jour. of Systems Engineering and Electronics*. 2011, vol. 22, no. 5, pp. 780–787.

7. Tretyakov A. Automation of scheduling for periodic real-time systems. *Tr. Instituta sistemnogo programmirovaniya RAN* [Proc. of ISP RAS]. 2012, vol. 22, pp. 375–400 (in Russ.).

8. *Wind River VxWorks 653 platform 2.4 and 2.5*. Available at: http://www.windriver.com/products/product-notes/vxworks-653-product-note.pdf (accessed July 13, 2017).

9. Singhoff F., Plantec A., Dissaux P., Legrand J. Investigating the usability of real-time scheduling theory with the Cheddar project. *Real-Time Systems*. 2009, vol. 43, no. 3, pp. 259–295.

10. Craveiro J.P., Silveira R.O., Rufino J. HsSim: an extensible interoperable object-oriented n-level hierarchical scheduling simulator. *Proc. WATERS*. 2012, Pisa, Italy, pp. 9–14.

11. Smelyansky R.L. Model of Distributed Computer System Operation with Time. *Programming and Computer Software*. 2013, vol. 39, iss. 5, pp. 233–241.

12. Smelyansky R.L., Bakhmurov A.G., Volkanov D.Yu., Chemeritskii E.V. Integrated environment for the analysis and design of distributed real-time embedded computing systems. *Programming and Computer Software*. 2013, vol. 39, no. 5, pp. 242–254.

13. Balashov V.V., Balakhanov V.A., Kostenko V.A. Scheduling of computational tasks in switched network-based IMA systems. *Proc. Int. Conf. on Engineering and Applied Sciences Optimization*. NTUA, Athens, Greece, 2014, pp. 1001–1014.

14. Wang D., Han J., Ma D., Zhao X. Studying on ARINC653 partition run-time scheduling and simulation. *Proc. WASET*. 2012, no. 71, pp. 1583–1587.

15. Dissaux P., Marc O., Fotsing C., Rubini S., Gaudel V., Singhoff F., Plantec A., Nguyen-Hong V., Tran Hai-Nam. The SMART project: Multi-agent scheduling simulation of real-time architectures. *Embedded Real Time Software and Systems*. 2014, pp. 1–9.

16. Khoroshilov A., Albitskiy D., Koverninskiy I., Olshanskiy M., Petrenko A., Ugnenko A. AADL-based toolset for ima system design and integration. *SAE Int. Jour. of Aerospace*. 2012, vol. 5, no. 2, pp. 294–299.

17. Bengtsson J., Yi W. Timed automata: Semantics, algorithms and tools. *Lectures on Concurrency and Petri Nets*. LNCS, Springer, 2004, vol. 3098, pp. 87–124.

18. Andre E. Observer patterns for real-time systems. *Proc. ICECCS*. 2013, pp. 125–134.

19. Avionics application software standard interface. *ARINC specification 653*. Aeronautical Radio. Annapolis, 1997.

20. *Cheetah. The Python-Powered Template Engine*. Available at: http://cheetahtemplate.sourceforge.net/ (accessed July 13, 2017).