УДК 681.51:629.78

МЕТОДОЛОГИЯ И ПРОГРАММНЫЙ КОМПЛЕКС ДЛЯ АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ УПРАВЛЯЮЩИХ АЛГОРИТМОВ РЕАЛЬНОГО ВРЕМЕНИ

© 2005 А.А. Тюгашев

Самарский государственный аэрокосмический университет

Статья посвящена рассмотрению интегрированной среды, позволяющей проектировать управляющие программы реального времени с использованием оригинальной методологии и средств автоматизации. Среда обеспечивает повышение качества и надежности программ, снижение трудоемкости и стоимости разработки. Предлагается специальная математическая модель для представления управляющих алгоритмов реального времени, разработаны языки для их описания. Приводится методика автоматизированного расчета временных характеристик управляющей программы. Рассказывается о структуре программного комплекса ГРАФКОНТ/ГЕОЗ.

Введение

Весьма актуальной задачей в настоящее время является проектирование комплексов надежных управляющих программ для бортовых вычислительных систем (БВС) космических аппаратов [1]. При этом надежность управляющей программы определяется предварительным проектированием надежного алгоритма управления реального времени, реализующего все поставленные целевые задачи.

К сожалению, создание надежного бортового комплекса управления остается одной из наиболее сложных и трудоемких задач при создании всего космического аппарата (КА). При этом большие проблемы представляет собой создание не только надежного комплекса бортовой аппаратуры (БА), что является в настоящее время достаточно формализованной задачей, но и надежного бортового программного обеспечения. Последняя задача гораздо менее изучена, и можно констатировать отсутствие в настоящее время общепринятых подходов её точного решения.

В случае бортовых алгоритмов управления ситуация осложняется тем фактом, что работа алгоритма должна происходить в режиме реального времени, и в случае ошибки, как правило, нельзя рассчитывать на возможность повторного прогона. А цена ошибки, определяемая стоимостью космического аппарата, может быть неприемлемо высокой.

Широко применяемой и достаточно апробированной методикой повышения надежности программ является тестирование.

Однако проверить все возможные варианты исполнения при достаточно сложной схеме программы и количестве логических условий в ней, равно как и правильность исполнения программы при всех возможных комбинациях исходных данных (ИД), практически невозможно. Даже полностью успешное тестирование на тестовом подмножестве вариантов исполнения управляющего алгоритма (УА) и на некотором подмножестве возможных исходных данных, не дает гарантии правильности программы.

В настоящей работе ставится задача поиска методики, обеспечивающей повышение надежности программы при ее проектировании при одновременном снижении трудоемкости, сроков и стоимости разработки бортового программного обеспечения.

Предлагается основанная на оригинальной математической модели управляющего алгоритма реального времени (УА РВ) методика проектирования, поддержанная специальным инструментальным программным комплексом (интегрированной программной средой разработки).

Применение данной методики и реализованных на её основе программных средств (которые по классификации относятся к CASE/CALS средствам – Computer Aided Software Engineering/Continuous Acquisition Lifecycle Support), может позволить повысить производительность труда, сократит стоимость разработ-

ки, повысить вероятность безотказной работы управляющих комплексов программ реального времени, причем не только в области авиации и космонавтики, но и в широком спектре применений. К возможным областям относятся, в частности, судостроение, управление технологическими производственными комплексами, осуществляемое в реальном времени, в химической и нефтяной промышленности, энергетике и др.

Постановка задачи

Сформулируем теоретический базис предлагаемой технологии.

Когда проводят исследование обычного алгоритма (либо программы), то с точки зрения описания семантики, его ценность определяется результатом, получающимся по окончании его работы. Под результатом подразумевается некоторый набор выходных данных, полученных в результате обработки набора ИД. Состояния программы (алгоритма), как начальное и конечное, так и промежуточные, определяются при этом совокупностью значений в программной памяти. Известна общая формализация понятия программы, исходящая из формулировки так называемых постусловий и предусловий [2]. Данные условия есть часть языка алгоритмических логик, предложенного практически одновременно в 60-х годах прошлого века Флойдом, Хоаром, математиками польской школы, и включающего условия вида:

$$\{U\}\ S\ \{B\},$$

читающиеся следующим образом: " U – условие, относящееся к исходным данным программы S, истинное до ее выполнения, B – условие, относящееся к выходным данным программы S, которое должно быть истинным после ее исполнения".

Данный подход не может быть непосредственно применен для случая управляющих алгоритмов реального времени (УА РВ), поскольку в этом случае важным условием правильности является осуществление корректного управления бортовой аппаратурой на всем промежутке времени активного существования КА. Более того, для управляющей программы неприменим классический подход к алгоритму как к набору действий для получения по завершению его работы определенного результата.

С точки зрения правильности работы УА РВ его корректность может быть определена как успешное исполнение КА его целевой задачи при любом возможном развитии ситуации, или в обозначениях пред- и постусловий:

$$\{U(D_0,t_0)\}\ YS\ \{B(D_k(t_1,t_2,\ldots t_k),t_k)\}$$

В момент времени t_0 начала функционирования КА истинно условие корректного задания исходных данных D_0, а к моменту t_k завершения работы управляющего алгоритма YS истинно условие B, означающее успешную отработку всех целевых задач D_k в заданные моментах времени $t_1,t_2,\ldots,t_k$ на всем промежутке времени функционирования КА.

Отсюда логически следует предложенное автором [3] представление семантики УА РВ в виде набора четверок объектов:

$$\text{УА РВ} = \{ < f_i,\ t_i,\ t_{\tau},\ \overline{l_i} > \},\ i=\overline{1,N},$$

где f_i – функциональная задача (действие); t_i – момент начала исполнения действия; t_{τ} – длительность действия; $\overline{l_i}$ - логический вектор, обуславливающий действие.

Таким образом, управляющий алгоритм должен обеспечивать на некотором непустом множестве временных меток ("включений") выполнение определенных действий по управлению КА, зависящих от текущей ситуации на борту, отражаемой вектором значений логических переменных. В этом заключается целевая задача, решение которой обеспечивается алгоритмом [4].

Основная идея предлагаемой технологии проектирования надежных алгоритмов управления заключается в постепенном конструировании сложного УА из априори надежных блоков – более простых программ управления "базового" уровня – так называемых функциональных программ (ФП) [5].

Если при этом обеспечивается правильность алгоритма конструирования, т.е. соединения ФП в единое целое, то обеспечивается повышение надежности получаемого в результате управляющего алгоритма.

Основой принципов надежного соединения функциональных программ, адекватно описывающего все возможные варианты в рамках логики и внутренних временных соотношений управляющего алгоритма, являются предложенные

Калентьевым А.А. исчисление УА РВ и операции алгебры УА [6].

К функциональным программам и получаемым промежуточным результатам их объединения по правилам алгебры УА применяются операции, показанные на рис. 1.

Последовательное применение приведенных операций позволяет описать все возможные варианты сочетаний моментов запуска ФП по времени, и все возможные варианты исполнения УА РВ.

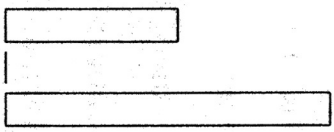
Задача построения (генерации) управляющей программы, реализующей управляющий алгоритм с заданной семантикой, сводится к построению логико-временной схемы функционирования, то есть фактически в терминах теории схем программ – схемы программы, но с учетом аспекта реального времени.

Будем называть такую схему многовходовой моделью управляющего алгоритма.

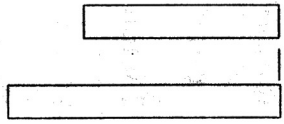
Таким образом, задача генерации управляющей программы сводится к построению отображения

$$S \Rightarrow MWM,$$

Операция совмещения по началу



Операция совмещения по концу



Операция непосредственного следовани

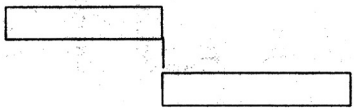


Рис. 1. Операции алгебраической системы управляющих алгоритмов

где S – семантика УА в виде набора четверок (см. выше); MWM – многовходовая модель УА.

Многовходовая модель представляет собой набор входов (включений) УА в определенные моменты времени. При этом считаем, что длительность входа равна нулю в условной временной шкале исполнения управляющей программы, т. е. длительность операций – функциональных задач внутри входа стремится к нулю по шкале бортового времени и несоизмерима с интервалами времени между входами, например, миллисекунды и секунды. Этот факт приводит к тому, что внутри входа генератор может произвольно переставлять моменты запуска f_i, и при необходимости сохранения определенного порядка действий, их последовательность надо задавать дополнительно в виде информации о существовании отношения предшествования на множестве функциональных задач f_i (обуславливаемого, например, тем, что одна программа использует данные, формируемые другой, то есть между ними существует информационная связь).

В наиболее простом случае число включений равно просто количеству различающихся друг от друга моментов времени t_i, содержащихся в семантике УА и соответствующих моментам запуска входящих в управляющий алгоритм отдельных функциональных программ f_i.

Соединенная передачами управления между входами совокупность входов образует граф:

$$MWM = < W, U >,$$

где W – множество вершин (входов); U – множество дуг (передач управления между входами), то есть бинарное отношение W x W.

В общем случае, при наличии циклических действий в УА граф может содержать циклы. В случае наиболее распространенного ациклического варианта, должно выполняться условие последовательного запуска при передаче управления: если существует дуга (W_i, W_j), то $t_{owi} < t_{owj}$, где t_{owi} – момент включения входа W_i t_{owj} - момент включения входа W_j.

Каждый вход представляет собой, в свою очередь, логическую схему (последовательность проверок условий и действий f_i), реализующих логику входа.

Вход можно рассматривать как совокупность линейных участков и ветвлений по резуль-

татам проверки логических условий [2].

Линейным участком будем называть последовательность функциональных задач (ФЗ) без проверок логических условий и передач управления. Линейный участок может включать несколько f_i, или ни одной.

Тогда логическая схема входа будет представлять собой граф (дерево):

$$W = < LU, Y >,$$

где LU – множество вершин графа (линейных участков); Y – множество дуг - передач управления (ветвлений).

Каждая вершина логической схемы входа (помимо листьев, соответствующих заключительным линейным участкам входа) будет иметь две исходящих из нее дуги, соответствующие вариантам выполнения и невыполнения логического условия, то есть дерево будет бинарным. Пример графа входа приведен на рис. 2.

Ясно, что если внутри линейного участка важна последовательность выполнения ФЗ f_i и f_j, то должно выполняться условие: $t_i <= t_j$.

Итак, управляющий алгоритм в виде набора действий, которые необходимо предпринять в заданные моменты времени в зависимости от заданных логических условий, необходимо реализовать в контексте конкретной операционной системы БВС реального времени.

Логические условия, фигурирующие в многовходовой модели, по времени их значимости (актуальности) могут относиться к одному из трех типов [2]:

1. $\alpha(t)$, то есть интересует значение логической переменной в текущий момент времени t, и с течением времени значение а может меняться;

2. $\alpha(t_0)$, когда значение переменной может меняться, но интересует ее значение в указанный момент времени t_0 (часто это момент начала исполнения УА), а изменения значения во времени не должны влиять на исполнение ПКФ;

3. $\alpha_{стат}$, когда значение логической переменной не претерпевает изменений в процессе исполнения УА.

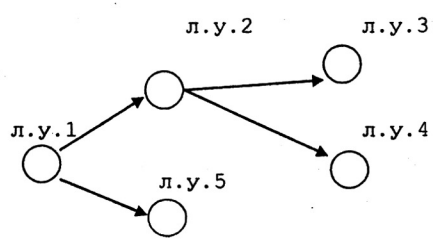


Рис. 2. Вариант взаимосвязи линейных участков внутри входа

Очевидно, что подход при интерпретации этих трех типов логических переменных при построении многовходовой модели, должен быть различным. Для случая $\alpha(t)$ необходима проверка актуального значения в каждый текущий момент, то есть внутри каждого входа должен полностью при обусловленности некой f_i логическим вектором проверяться весь вектор значений логических переменных.

Для случая $\alpha(t_0)$ необходимо один раз в начале выполнения УА выяснить, к какому варианту необходимо прибегнуть при исполнении, то есть один раз в момент времени t_0 проверить все логические условия, фигурирующие во всех логических векторах, обуславливающих выполнение всех f_i, входящих в данный УА.

Для случая $\alpha_{стат}$ проверку можно проводить в любой удобный момент и соответствующим удобным и гибким образом распределять проверки логических условий между входами и организовывать передачи управления между входами в многовходовой модели.

Так, для случаев $\alpha(t_0)$ и $\alpha_{стат}$ можно считать, что от входа к входу логическая обусловленность "наследуется" и образуется цепочка входов, логически обусловленная аналогично случаю отдельной функциональной задачи f_i в модели семантики УА. При этом распределенная во времени цепочка входов является аналогом логической ветви исполнения обычного алгоритма, не выполняющегося в реальном времени.

По ряду причин, включающих, в частности, простоту коррекции БПО и добавления в него дополнительных задач; возможность оперативного дистанционного изменения состава решаемых БПО задач; более оптимальную загрузку вычислительных ресурсов для сложных многофункциональных комплексов БПО, в которых моменты начала и окончания решения задач могут меняться в широких пределах в зависимости от временных разбросов работы бортовой аппаратуры (БА) и исходных данных, передаваемых с

Земли, предпочтительно использование приоритетной динамической асинхронной организации вычислительного процесса на борту космического аппарата [4].

Особую важность представляет возможность автоматизированного получения временных характеристик конструктивным путем по результатам проектирования управляющих программ комплексного функционирования БА в рамках предлагаемой технологии. Поскольку система предполагает предварительное описание характеристик каждой из базовых функциональных задач, включая время исполнения, то, имея полученную по результатам трансляции логико-временную схему исполнения управляющей программы, отраженную во временной диаграмме, сделать это нетрудно.

В рамках CASE-системы автоматизированного проектирования УА РВ применяются файлы описания функциональных программ (специальный формат *.ops*), в которых содержатся данные по временным характеристикам исполнения отдельных действий. Это дает возможность подсчета временных характеристик линейных участков путем суммирования времен исполнения ассемблерных команд передач управления, выдачи команд управления (КУ) бортовой аппаратуре, а также времени выполнения функциональных программ, исполняемых с возвратом. При этом необходимо учесть все возможные варианты выполнения (маршруты) программы управления, определяемые значениями вектора логических переменных $l=(\alpha_1,\alpha_2,...,\alpha_M)$. Необходимо рассматривать логику выполнения алгоритма, и тогда максимальное время выполнения входа определится формулой

$$t_{вх\,max} = \max_{\substack{\text{по всем} \\ \text{маршрутам}}} \sum_{i=1}^{N} t_{лу\,i},$$

где N – число линейных участков на маршруте (варианте) исполнения входа, $t_{лу\,i}$ - длительность i-го линейного участка.

В свою очередь, длительность выполнения линейного участка определяется суммарным временем выполнения присутствующих на нем операций (функциональных программ):

$$t_{лу} = \sum_{j=1}^{L} t_j,$$

где L – число операций линейного участка, t_j - время исполнения j-ой команды или функциональной программы.

При более крупномасштабном рассмотрении можно подсчитывать также временные характеристики УА в целом, по всем входам. В таком случае для описания передач управления при различных значениях компонент логического вектора можно применить в качестве модели ациклический граф, в котором вершинами являются входы УА, а направленные дуги соответствуют передачам управления между входами, как показано на рис. 3.

Выше описан пример многовходовой модели УА, при котором точка передачи управления из каждого входа фиксирована. Модель становится более сложной при учете возможности передачи управления из произвольного места внутренней структуры входа.

Та же самая модель может быть использована и для представления логической структуры передачи управления внутри входа. Отличия заключаются только в том, что в данном случае на граф передач управления накладывается дополнительное ограничение, а именно: он должен представлять собой дерево с одним корнем.

Таким образом, при применении описанной технологии возможен автоматизированный синтез временных характеристик исполнения построенных алгоритмов управления (как это видно из приведенного примера), легкость и оперативность сопоставления вариантов при внесении изменений в управляющие алгоритмы.

Описание среды программной разработки ГРАФКОНТ/ГЕОЗ

Описанная технология проектирования поддерживается специально разработанной программной системой, функционирующей на платформе Windows 95/98/2000/XP и созданной в основном средствами языка C/C++.

Структура программного комплекса ГРАФКОНТ (от слов "графическое конструирование") представлена на рис. 4.

Пользователь системы взаимодействует с несколькими интерфейсными модулями – интег-

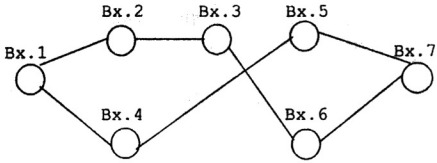


Рис. 3. Вариант взаимосвязи входов алгоритма по управлению

рированной средой разработчика, являющейся аналогом IDE современных языков программирования, таких как C++ (Borland C++ Builder, Visual Studio) или Java (Eclipse, IDEA, JBuilder), объединяющей все модули комплекса в единое целое и позволяющей получить доступ к любой функции системы; графическим конструктором управляющих алгоритмов на основе применения операций алгебраической системы УА РВ; графическим редактором временных диаграмм; оболочками генератора ассемблерного текста управляющей программы и отладочных заданий.

После этапа ввода исходного текста на языке описания УА (или формирования его с помощью графического конструктора), работает модуль трансляции. В результате работы транслятора, включающего модули лексического и синтаксического анализатора, а также модуль выявления семантики, строится многовариантная модель УА, отражающая все возможное многообразие вариантов исполнения.

Говоря о методах обеспечения надежности программ путем их тестирования, необходимо отметить, что несмотря на повышение надежности проектируемого алгоритма при использовании технологии и программного комплекса ГРАФКОНТ, система не может парировать грубые семантические ошибки, допущенные при конструировании УА в графическом конструкторе, например, пробелы или упущение раздела в материалах по логике управления. Поэтому нецелесообразно отказываться от комплексного поэтапного тестирования полученного программного продукта, в ходе которого путем моделирования бортового функционирования УА выявляется его соответствие исходным требованиям.

На протяжении длительного периода времени на предприятии-заказчике (ГНПРКЦ "ЦСКБ-Прогресс", г. Самара) выработана методика поэтапной отладки управляющих программ на специальном наземном отладочном моделирующем комплексе, в который входят ЭВМ, эмулирующие БВС и ЭВМ для моделирования внешних воздействий космического пространства и других факторов среды [4].

Отладка включает автономную отладку и комплексную отладку. При этом для каждой управляющей программы составляется отладочное задание на специальном символьном языке отладки. В отладочное задание входят перечень отслеживаемых по значению переменных, необходимых точек останова и т. д.

Созданная система автоматизированной генерации отладочных заданий ГЕОЗ позволяет на базе внутренних структур данных программного комплекса ГРАФКОНТ автоматически формировать отладочное задание на автономную отладку УА.

Важными задачами при проведении отладки УА РВ являются автоматизированное выявление всех возможных вариантов исполнения ("маршрутов") алгоритма в зависимости от значений компонент вектора логических перемен-

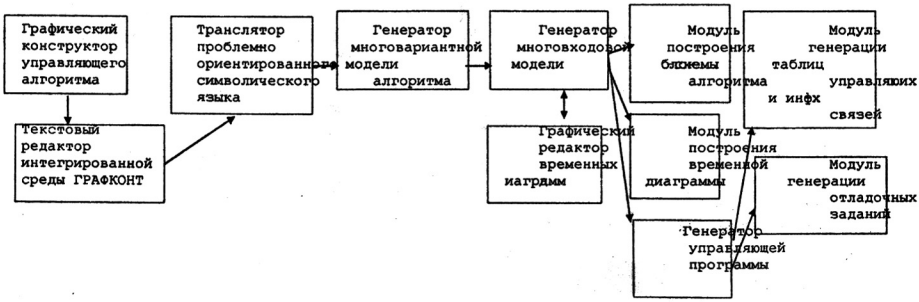


Рис.4. Структура программного комплекса ГРАФКОНТ/ГЕОЗ

ных, а также выявление всех имеющихся в алгоритме информационных и управляющих связей с другими управляющими программами. Эти задачи также решаются в рамках программной системы ГЕОЗ

Заключение

Таким образом, в работе приводится математическая модель семантики управляющих алгоритмов реального времени и предлагается использование алгебраической системы как основы для возможности пошагового синтеза (конструирования) управляющих алгоритмов из более простых блоков – функциональных задач.

Использование технологии ГРАФКОНТ/ГЕОЗ и поддерживающего её программного комплекса дает возможность обеспечить:

1. Снижение трудоемкости разработки бортовых алгоритмов управления реального времени.
2. Сокращение сроков и стоимости разработки комплексов управляющих программ.
3. Уменьшение зависимости от уникальной квалификации человека-проектировщика при создании бортовых программ управления комплексного функционирования, за счет использования формально определенных регулярных процедур.
4. Повышение надежности бортового управляющего ПО.

Предлагаемая методика может быть использована, как представляется, не только в космической отрасли для управления аппаратами различного назначения, но и в авиации, а также при управлении сложными техническими комплексами и технологическими установками в различных отраслях промышленности и транспорта.

СПИСОК ЛИТЕРАТУРЫ

1. "Авиастроение". Том 6 (Итоги науки и техники, ВИНИТИ АН СССР). М., 1978.
2. Логика и компьютер. Моделирование рассуждений и проверка правильности программ / А.М. Анисов, П.И. Быстров, В.А. Смирнов и др. М: Наука, 1990.
3. *Тюгашев А.А.* Проблема неоднозначности при порождении логико-временной структуры управляющего алгоритма по многовходовой модели реального времени. // Сб. трудов Третьей международной молодежной школы-семинара БИКАМП-01. СПб, 2001.
4. Управление космическими аппаратами зондирования Земли: Компьютерные технологии / Д.И. Козлов, Г.П. Аншаков, Я.А. Мостовой, А.В. Соллогуб. М.: Машиностроение, 1998.
5. *Калентьев А.А., Тюгашев А.А.* Разработка подсистемы синтеза управляющих алгоритмов на базе исчисления УА // Сб. трудов Всероссийской научной школы "Компьютерная алгебра, логика и интеллектное управление. Проблемы анализа стратегической стабильности". Иркутск: ИрВЦ СО РАН, 1994. Т.4.
6. *Калентьев А.А.* Автоматизированный синтез алгоритмов асинхронного управления технологическими системами с множеством дискретных состояний. Самара: Самарский аэрокосмический университет, 1998.

METHODOLOGY AND SOFTWARE FOR AUTOMATED DESIGN OF REAL-TIME CONTROLLING ALGORITHMS

© 2005 A.A. Tyugashev

Samara State Aerospace University

Article is devoted to review of integrated software development environment for real-time controlling algorithms. Integrated environment is based on original methodology and mathematical models. Environment provides improvement of quality and reliability of programs, decrease in labor input and costs of development. The special mathematical model for representation of real-time controlling algorithms is formulated; special languages based on this model are developed. The technique of the automated calculation of time characteristics of the operating program is resulted also. It is told about structure of integrated development software.