УДК 004(4’242+053)

С.И. Чайников[1], А.С. Солодовников[2]
[1] ХНУРЭ, г. Харьков, Украина;
[2] ХНМУ, г. Харьков, Украина

ПРИНЦИПЫ ОРГАНИЗАЦИИ ВЫЧИСЛЕНИЙ НА БАЗЕ ГРАФ-МОДЕЛИ ПРЕДМЕТНОЙ ОБЛАСТИ

Рассмотрены преимущества использования диалоговых систем как средств автоматизации вычислительных процессов, имеющих особое значение для крупномасштабных объектов. Выработаны принципы организации вычислений на основе формализованного представления предметной области, которые позволяют оптимизировать время выполнения вычислений и достичь устойчивости архитектуры программного средства в условиях динамически изменяемых требований заказчика.

ДИАЛОГОВЫЕ СИСТЕМЫ, ГРАФ-МОДЕЛЬ ПРЕДМЕТНОЙ ОБЛАСТИ, ГЕНЕРАТОР МОДЕЛИ ПРЕДМЕТНОЙ ОБЛАСТИ

Введение

Существует разнообразное количество подходов и технологий проектирования программных средств. Все они имеют свое применение в зависимости от сложности предметной области и условий, в которых программное обеспечение должно функционировать. В связи со сложившимся многообразием готовых решений, которые могут быть взяты за основу при проектировании программных средств, позволяющих автоматизировать заданные процессы, существует проблема выбора того или иного подхода. Однако данный вопрос становится наиболее актуальным в случае возрастающей сложности объекта автоматизации, которая обусловлена иерархичностью его структуры, а также значительным количеством элементов, разнородностью связей между ее элементами, их территориальной распределенностью. Решение такой проблемы требует использования системного подхода [1].

Характеристики сложного объекта можно классифицировать следующим образом [2]:

• структурная сложность (иерархическая структура организации, территориальная распределенность);

• функциональная сложность (многоуровневая иерархия, большое количество функций, выполняемых организацией, сложные взаимосвязи между ними);

• информационная сложность (большое количество источников и потребителей информации, разнообразные формы и форматы представления информации, сложная информационная модель объекта;

• сложная динамика поведения объекта, обусловленная высокой изменчивостью внешней и внутренней среды (структурные реорганизации, текучесть кадров).

Структурная сложность обусловливает особый подход к планированию вычислительного процесса, поскольку для систем со сложной структурой требуется значительное время для решения таких проблем, как восстановление утерянных данных, планирование вычислений, восстановление истории вычислений или отмена результатов вычислительных процессов до заданного момента времени. Диалоговый режим работы систем в зависимости от подхода к их проектированию позволяют в той или иной степени оптимизировать работу пользователей.

1. Цель работы

В связи с необходимостью применения специальных методов проектирования программного обеспечения и организации вычислительных процессов для сложных объектов необходимо изложить ряд принципов построения и функционирования диалоговых систем управления вычислительными процессами, позволяющих адаптировать технологию проектирования для использования в различных предметных областях.

2. Основные свойства диалоговой системы

Класс диалоговых автоматизированных систем позволяет взять на себя бульшую часть вычислений, оставляя пользователю функцию контроля результата вычислительных процессов и ввода необходимой информации. Только те задачи, которые не имеют четкого алгоритма автоматизации, полностью лежат в зоне ответственности пользователя.

Диалоговые системы обладают определенными характеристиками по отношению к типу диалога и действиям пользователя системы.

В ДС выделяют два рода действий пользователя:

1) получение справок по теме диалога;

2) принятие решений для возобновления вычислений.

При этом действия первого рода доступны пользователю в любой момент вычислений, а действия второго рода регламентируются и контролируются системой.

Диалоговое общение разделяется на три вида:

1) директивное (диалогом управляет человек);

2) инициируемое (диалогом управляет ЭВМ);

3) смешанное [3].

Каждый из этих видов обладает своим преимуществом и должен приниматься во внимание на этапе разработки системы.

На сегодняшний день существует широкий круг предметных областей, для которых требуется применение диалоговых систем: машино- и приборостроение, консультативно-диагностические системы, системы автоматизированного проектирования. Каждая предметная область требует адаптации программного средства.

Предлагается построение диалоговой системы, работа которой основана на использовании управляющего графа-модели предметной области, обеспечивающего взаимосвязь программных модулей системы. Новизна такого подхода заключается в отказе от выделения маршрутов вычислительных процессов, используемого ранее [3]. В данном случае в диалоговой системе управление решением поставленных задач основывается на выделении подграфа, в множество вершин которого входят вершины, описывающие структуру заявленного вычислительного процесса.

3. Подход к проектированию диалоговой системы

Процесс проектирования диалоговых систем основывается на знаниях разработчиков программного средства о предметной области, в которой данная система будет функционировать. Основным моментом при разработке программного средства является то, что диалоговая система должна функционировать на базе формализованного представления – модели предметной области (МПО) в виде направленного ациклического графа. Данное представление является платформонезависимым и позволяет при необходимости использовать любой объектно-ориентированный язык программирования для адаптации программы к заданной модели.

Для построения модели предметной области за основу берется концептуальное представление системы [4], выработанное на стадии предпроектных исследований, вида

$$S=<E, R>, \quad (1)$$

где E – множество элементов, а R – это множество отношений между элементами системы. Для концептуальной модели (1) должны быть конкретизированы топологические свойства:

$$S=<E, R, T>, \quad (2)$$

где топология T определяется как $T=<T_E, T_R, T_P>$. Здесь T_E – топология элементов, T_R – топологические отношения и связи, а T_P – траектории перемещения информации, энергии или других ресурсов, которые определяются алгоритмами и технологиями функционирования системы.

В качестве входной информации для построения МПО используется набор спецификаций, включающий как требования, так и связи между структурными составляющими системы, данные о динамике поведения системы. После определения зависимостей между элементами автоматизируемой системы и определения функций подсистем строится топология элементов. На выходе получается IDEF3 модель и диаграмма DFD, на основе которых выполняется модульное проектирование программного средства. В результате структурного и объектного анализа должна быть построена граф-модель предметной области.

Процесс проектирования программного средства можно разбить на такие этапы [5]:

1) содержательное описание объекта или процесса, когда выделяются основные составляющие системы, закономерности системы;

2) формулировка прикладной задачи или задачи формализации содержательного описания системы (прикладная задача содержит в себе изложение идей исследования, основных зависимостей, постановку вопроса, решение которого достигается посредством формализации системы);

3) построение формализованной схемы объекта или процесса, что предполагает выбор основных характеристик и параметров, которые будут использованы при формализации;

4) моделирование предметной области на основе формализованной схемы (структурно-функциональное моделирование объекта или процесса).

Программное обеспечение разработчика системы и технологии проектирования определяют структуру генератора, позволяющего сформировать МПО (рис. 1).

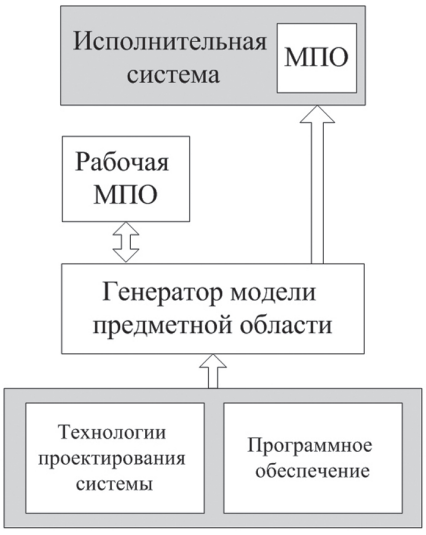

Рис. 1. Формирование и использование МПО

Сформированная модель проверяется на наличие ошибок и тестируется на адекватность по отношению к объекту автоматизации. После того как исправлены все обнаруженные ошибки, модель может использоваться заказчиком программной системы. Исполнительная система состоит из

набора программных модулей, связанных по данным. Каждый из модулей закреплен за определенной функцией, требующей автоматизации.

4. Требования к граф-модели предметной области

К началу процесса моделирования предметной области необходимо использовать два подхода – в ширину и в глубину [6] с тем, чтобы в итоге получить древовидную иерархическую граф-модель.

Исходная информация, а именно МПО, в виде графа вводится пользователем в базу данных САПР.

Граф $G = <V, U>$ состоит из двух множеств: непустого конечного множества вершин V, которым соответствуют вычислительные процессы, закрепленные за разными подразделениями организации, и конечного множества ребер U, которым соответствуют информационные связи между вершинами (рис. 2). При этом граф должен быть приведен к ярусно-параллельной форме для оптимизации процессов управления программы согласно правилу $N_{яр} = L_{кр} + 1$, где $N_{яр}$ – количество ярусов, а $L_{кр}$ – длина критического пути.

Верификация (проверка структуры на наличие ошибок) построенной модели осуществляется на базе электронных спецификаций, которые описывают стандарты сборки модулей. Граф-модель, благодаря своей структуре, позволяет обнаруживать зависимости от данных других модулей, которые необходимы для выполнения текущей операции (рис. 3).

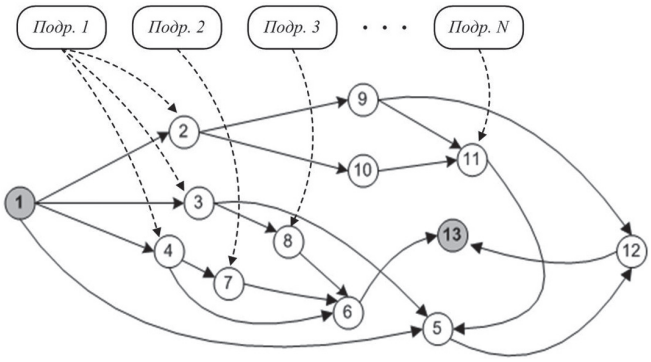

Рис. 2. Граф-модель предметной области

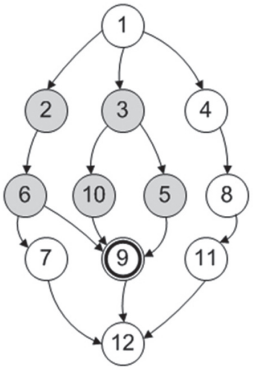

Рис. 3. Активные функции, необходимые для запуска вычислительного процесса

Пользователь, который является ответственным за выполнение определенной задачи и закрепленным за определенное подразделение, запускает соответствующую функцию (вершина графа под номером 9). Программа, согласно хранящемуся в базе данных описанию предметной области, обнаруживает зависимые модули. Происходит выделение подграфа $G' = <V', U'>$ на графе $G = <V, U>$ модели предметной области, для которого $V' \in V$ и $V' \in V$.

В случае, если выполнение зависимых вычислительных процессов не завершилось к текущему моменту, программа обязана проинформировать об этом пользователя. Такое свойство программы дает преимущество контролировать весь вычислительный процесс на всех стадиях и обеспечивать ответственное лицо соответствующей информацией. Сама модель, которая разрабатывается архитектором системы, может рассматриваться на различных уровнях детализации (рис. 4). Согласно этим уровням программное средство позволяет оценить процент завершенности того или иного процесса.

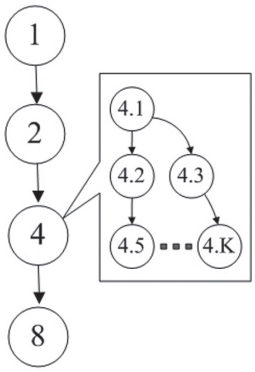

Рис. 4. Укрупненная и детализированная схема МПО

5. Требования к исполняющей системе

Исполняющая система имеет модульную структуру, что дает возможность изменять функционал или исправлять ошибки без ущерба целостности системы. Программа разрабатывается с учетом установленного однозначного соответствия между вершиной граф-модели и программным модулем. Структура ПС предоставляет возможность использования того или иного шаблона при проектировании. Это позволяет упростить процесс адаптации готовых решений к новым моделям предметных областей и обеспечивает соблюдение принципа сборочного создания программного обеспечения [7, 3].

Исполняющая система позволяет оптимизировать вычислительный процесс путем резервирования как исходных данных, так и результатов каждой узловой точки глобального вычислительного процесса. Тем самым обеспечивается возможность восстановления истории вычислений, например, при обнаружении ошибок вычислений. При этом

программе в диалоговом режиме даются директивы, какие именно процедуры необходимо повторить. Множество вершин, связанных с выбранной вершиной (рис. 5), запоминается для текущих вычислений и служит, при необходимости, основой для отката вычислительного процесса.

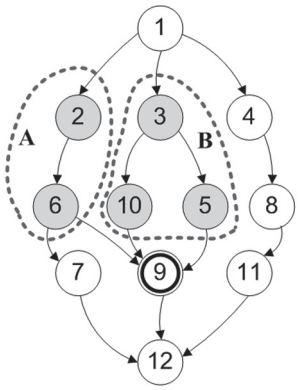

Рис. 5. Принцип восстановления данных (backtracking)

Такой принцип называется backtracking и позволяет оптимизировать время выполнения вычислительной задачи.

Выводы

Преимущество метода заключается в том, что, во-первых, формируется стандартный обобщенный подход к построению крупных программных систем, оперирующий набором архитектурных шаблонов, позволяющий использовать уже готовые типизированные решения для новых предметных областей, во-вторых – описание предметной области с помощью граф-модели позволяет структурировать процесс разработки программы и уменьшает степень связанности программных модулей. Структура ПС, основанная на граф-представлении, позволяет использовать спиральную модель жизненного цикла системы [8], что дает преимущество при уточнении или модификации в связи с измененными требованиями заказчика и позволяет достичь определенной устойчивости программной системы к изменениям среды функционирования и воздействия внешних факторов. Метод имеет практическое значение, прежде всего, при проектировании компьютерных систем, предназначенных для автоматизации вычислительных процессов крупномасштабных объектов, таких как проектные организации, производственные компании.

Перспективы дальнейших исследований заключаются в разработке для диалоговой системы механизмов сбора, хранения и отмены результатов вычислительных процессов при их параллельной обработке.

Список литературы: 1. *Чайников, С.И.* Методы и алгоритмы априорной оценки параметров вычислительных процессов [Текст] / С.И. Чайников – автореф. дисс. канд. тех. наук по специальности 05.13.01 – техническая кибернетика и теория информации. – Харьков, 1983. – 16 с. **2.** *Вендров, А.М.* Методы и средства моделирования бизнес-процессов [Текст] / А.М. Вендров // Jet Info – информационный бюллетень. – 2004. – № 10(137). – С.5 – 28. **3.** *Перевозчикова, О.Л.* Диалоговые системы [Текст] / О.Л. Перевозчикова, Е.Л Ющенко/ ин-т кибернетики им. В.М. Глушкова. – К.: Наук. Думка, 1990. – 184 с. **4.** *Бескоровайный, В.В.* Системологический анализ проблемы структурного синтеза территориально-распределенных систем [Текст] / В.В. Бескоровайный // Автоматизированные системы управления и приборы автоматики. – 2002. – №120. – С.29-37. **5.** *Черемных, С.В.* Структурный анализ систем: IDEF-технологии [Текст] / С.В. Черемных, И.О. Семенов, В.С. Ручкин – М.: Финансы и статистика, 2003. – 208 с. **6.** *Сурмин, Ю.П.* Теория систем и системный анализ [Текст] / Ю.П. Сурмин: Учеб. пособие – К.: МАУП, 2003. – 359 с. **7**. *Буч, Г.* Объектно-ориентированный анализ и проектирование с примерами приложений на C++ [Текст] / Г. Буч. – Спб.: Бином, Невский Диалект, 1998. – 560 с. **8.** *Чайніков, С.І.* Сучасні структурні методології проектування ІКС і особливості використання DFD та ERD [Текст] / С.І. Чайніков // Автоматизированные системы управления и приборы автоматики. – 2002. – №120. – С.63-72.

Поступила в редколлегию 19.07.2012

УДК 004(4’242+053)

Принципи організації обчислень на базі граф-моделі предметної галузі / С.І. Чайніков, А.С. Солодовніков // Біоніка інтелекту: наук.-техн. журнал. – 2012. – № 2 (79). – С. 72–75.

Пропонується побудова діалогової системи, структура і функціонування якої визначається моделлю предметної галузі, формалізованої у вигляді графа. Розглянуто принципи організації обчислень, що враховують можливість збереження і відновлення результатів обчислень з оптимізацією витрат часу на їх виконання. Також розглянута структура програмного засобу, призначеного для проектування діалогових систем, яка дозволяє на базі граф-моделей предметних галузей організовувати взаємодію програмних модулів.

Іл. 5. Бібліогр.: 8. найм.

UDC 004(4’242+053)

Computing organization principles based on problem domain graph-model / S.I. Chaynikov, A.S. Solodovnikov // Bionics of Intelligense: Sci. Mag. – 2012. – № 2 (79). – P. 72–75.

It is proposed to build a dialog system which has the structure and functionality based on the model of problem domain formalized by graph. The principles of computing organization are considered and take into account the ability to keep and restore calculus results to optimize execution time. The structure of dialog system design software is considered in order to provide interoperability to the program modules on the base of problem domain graph-model.

Fig. 5. Ref.: 8 items.