Е.М. Лаврищева

ПОДХОД К ПРОМЫШЛЕННОЙ ТЕХНОЛОГИИ ИЗГОТОВЛЕНИЯ БОЛЬШИХ ПРОГРАММ

Основу рассматриваемого подхода к изготовлению больших программ составляет модульный принцип, при котором большую программу будем трактовать как некоторую композицию модулей с высокой степенью сложности.

В процессе разработки больших программ решаются такие задачи: нужно реализовать требуемые функции и при этом уложиться в намеченные сроки, не выйдя за пределы стоимости; обеспечить надежность, изменяемость, а также способность функционировать в соответствующей операционной среде и т.д. Эти требования в той или иной степени не выдерживаются в процессе реализации, и поэтому изготовленная большая программа оказывается функционально неполной, трудно изменяемой и недостаточно надежной.

Причина заключается в том, что до настоящего времени еще не сложились общие формальные правила задания процесса объединения отдельных модулей в единый программный агрегат, хотя изготовление отдельных модулей полностью автоматизировано.

В ОС ЕС реализован модульный принцип объединения на уровне их объектного представления. Однако пользователей языков высокого уровня это не устраивает, так как от них требуется еще и знание языка низкого уровня [I]. Это приводит к тому, что отдельные объекты теряют свойство мобильности, а изготовление большой программы из отдельных разноязыковых модулей становится трудоемким и сложным процессом.

Наши усилия направлены на обеспечение технологичности сборки программных агрегатов из модулей, записанных в любых языках ОС ВС, а также на их отладку и хранение.

При этом основной предпосылкой промышленной технологии изготовления программ из модулей нам представляется стандартизация.

Многие отрасли промышленности имеют стандарты и ГОСТы, которые позволяют собирать более крупные детали из заготовок и заменять одни детали другими, более качественными. В частности, проектирование и изготовление ЭВМ поставлено на промышленную основу. Для того, чтобы к программе подходить как к промышленному изделию, необходима единая система правил и ограничений на задание отдельных объектов и на их сопряжение с другими, т.е. должен вводиться стандарт на объект - так же, как для промышленных изделий создается конструкторская документация на изготовление.

Стандарт позволит, с одной стороны, иметь возможность применять объекты в различных системах без участия разработчика и, с другой стороны, сводить создание сложных комплексов программ к сборке объектов в виде комплектующих изделий, как при промышленном изготовлении. С этой точки зрения стандартизация объектов и правил объединения в целом повышает технологичность сборки и служит шагом к созданию промышленной технологии изготовления программных агрегатов из исходных объектов, записанных в любых ЯП.

1. В в е д е н и е с т а н д а р т а. Для принятия стандарта на объекты сборки определим на интуитивном уровне понятие объекта сборки; в качестве него применим модуль - поименованный текст на языке программирования со списком входных параметров. Модуль является самостоятельной программной единицей, беря свои истоки у широко используемых в практике программирования понятий подпрограммы и функции. Технология как ручной, так и автоматизированной сборки программ из модулей, как показывает опыт, требует знания целого ряда характеристик, с помощью которых можно провести сборку (по управлению, данным, управлению и данным, используемой памяти и т.д.) Класс этих характеристик весьма общий и может быть приведен к единой стандартизованной форме для всех широко используемых ЯП. Среди основных характеристик выделим те, которые определяют связь его с внешним миром. К ним

относятся список входных и выходных параметров, внешних модулей и областей.

В силу того, что языки описания исходных модулей могут быть разными и в них приняты не всегда совпадающие типы описаний для передаваемых данных, принято дополнительное специфицирование передаваемых через параметры данных. Это позволяет проводить автоматическое преобразование как типов данных, так и их структур [2].

Для обеспечения независимости описания модуля от его характеристик последние выделены в специальный раздел, называемый паспортом модуля. Его проект предложен в [2] и используется нами в качестве основы.

2. З а д а н и е п р о ц е с с а с б о р к и . Для обеспечения сборки более сложных объектов из стандартизированных требуется такие приемы, которые сделали бы этот процесс технологичным, т.е. таким, чтобы по формализованному заданию "каркаса" для создаваемого программного агрегата на множестве стандартизированных объектов можно было формально изготавливать сложные объекты с закрепленными или подвижными (динамическими) объектами. В качестве каркаса предлагается использовать структурную граф-схему агрегата (дерево с модулями в его вершинах, связанными дугами по управлению и данным), задаваемую на этапе проектирования и являющуюся основной формой его первоначального представления и до полного изготовления. Последним она отличается от блок-схем, создаваемых для удобства представления лишь на этапе проектирования. В структурной схеме существующие между элементами связи задается дугами, идущими от вызывающего модуля к вызываемому.

То есть, проектируемая программа представляется в виде иерархического дерева, на самом нижнем уровне которого находятся элементарные конструктивные элементы - модули. Они ориентированы на выполнение определенных функций, и их решение не зависит от других модулей этого уровня или от модулей более высокого уровня. На следующем уровне иерархии находятся модули управления модулями нижнего уровня и т.д. Наконец, на самом верхнем уровне располагается главный модуль, обеспечивающий настройку и выполнение модулей, расположенных на разных ветвях дерева.

Таким образом, дерево есть не что иное, как ориентированный граф G (M, S, P), определенный на множестве вершин $M = \{M_1, M_2, \dots M_n\}$, каждая из которых является модулем, и на множестве дуг $S = \{S_1, S_2, \dots S_r\}$, задающий совокупность связей данного модуля с другими модулями. Функция F отображает реальные связи между элементами множеств (M, S), т.е. эта функция каждой дуге графа соотносит упорядоченную пару вершин $(M_k, M_n) \in M$, в которой $M_к$ соответствует вызывающему модулю, а M_n - вызываемому модулю. Условимся называть в ы з ы в а ю щ и м м о д у л е м такой модуль, который обращается к другому модулю, расположенному на более низких уровнях иерархии. В качестве обращения может использоваться оператор вызова CALL [1]. Модуль, к которому имеется обращение, заданное дугой графа, будем называть в ы з ы в а е м ы м м о д у л е м .

Таким образом, между элементами схемы и графа существует взаимно-однозначное соответствие; т.е. каждому оператору CALL в схеме всегда соответствует дуга графа с прилегающими вершинами в графе, а количество вершин графа не меньше количества модулей в схеме.

Сборка программ на основе граф -схемы реализуется в системе АПРОП [3] посредством

1) описания ее правилами,
2) заданием схемы с экрана.

Для случая описания правил сборки разработаны специальные операторы, позволяющие восстановить исходную граф-схему и вывести ее на экран.

В случае 2) пользователь сам вводит схему в требуемом виде.

Схема, введенная аналитически или графически, служит основой для построения по ней программы или отдельных ее частей (сегментов). При этом создается машинное представление графа в виде матрицы соотношений (МС) множества модулей и множества

связей между ними. В результате МС становится неотъемлемой компонентой, создаваемой программы и сохраняется до конца изготовления программного агрегата. Всякие изменения, вносимые в схему на экране, переносятся в ее машинное представление. Таким образом, по графу создается композиция элементов, настроенная на одновременную работу ветвей или на перекрытие памяти для совмещаемых объектов, то есть решается задача минимизации графа, обеспечивающая получение программной структуры, критичной по памяти или быстродействию.

Описание схемы инициирует операцию сборки. При этом элементы схемы (объекты, подграфы) могут особым образом отмечаться для управления памятью и быстродействием в создаваемом подпрограммном агрегате. В общем случае система делает прикидки и выдает сообщения, если критерий памяти не выдерживается. При реализации сборки по схеме решались задачи межъязыковых связей, отладки разноязыковых агрегатов и др. В плане решения первой задачи создана библиотека межъязыкового интерфейса, обеспечивающая преобразование разных типов данных, сохранение областей и обратное восстановление типов данных и областей вызывающего модуля после работы вызываемого модуля.

3. О т л а д к а и з г о т о в л е н н о г о а г р е г а т а . В связи с тем, что созданный агрегат необходимо отлаживать, то для этого используется структурная схема, представленная в машине матрицей МС. Кроме того, создаются унифицированные таблицы описаний величин для каждого ЯП. С их помощью осуществляется трассирование агрегата и выдача пути выполнения элементов на экране, в виде индикации тех вершин схемы, которые выполнялись в данный момент времени. Одновременно с этим могут выдаваться и значения требуемых величин. Для обеспечения отладки агрегатов по схеме разработан специальный набор средств, направленный на проверку правильности связей объектов и передач данных между ними; отладка же отдельных объектов схемы обеспечивается средствами самих языков программирования, а также системными диалоговыми средствами.

4 . Б а н к о б щ е г о п о л ь з о в а н и я . Для повышения уровня технологии изготовления больших программ из исходных модулей создано эффективно организованное хранение готовых объектов в Банке общего пользования. Паспортные данные помещаемых в него объектов приводятся к системному виду и используются как при автоматизированной сборке, так и при формировании ответов на запросы пользователей о реализованных в модулях алгоритмах и правилах их использования в других программах.

5. В ы в о д ы . Таким образом, предлагаемый подход к промышленной технологии опирается на стандартизацию исходных объектов, запоминаемых в централизованном хранилище ("складе деталей"), а также на описание связей объединяемых объектов, задаваемых по структурным граф-схемам.

Л и т е р а т у р а

1. ЛЕБЕДЕВ В.Н., СОКОЛОВ А.П. Введение в систему программирования ОС ЕС. М., "Статистика", 1978.
2. ГРИЩЕНКО В.Н., ЛАВРИЩЕВА Е.М. О создании межъязыкового интерфейса для ОС ЕС ЭВМ. Ж. УСиМ № I, 1978.
3. Система автоматизации производства программ, РФАП, Киев, 1976.