ТУБ №4:
Графическое представление алгоритмов и ГОСТ 19.701-90

Графическое представление алгоритмов

Когда речь заходит об общении с заказчиком и согласовании того, как должна решать поставленную задачу программа, словесное описание алгоритма действительно выручает. В конце концов от привычной речи оно отличается разве что более ограниченным набором слов да более строгим изложением, с этим может смириться и потерпеть даже самый нетерпеливый заказчик.

Но одна проблема у словесного описания всё же есть: как только алгоритм становится чуть сложнее, разбираться в нём приходится в муках. Особенно это ощутимо, если словесное описание строится прямо по ходу общения с заказчиком, когда идея алгоритма вырисовывается не сразу. Другой случай — когда Вам нужно разобраться с алгоритмом, который записан кем-то другим (а ведь для этого алгоритмы и записывают). Возьмём такой пример:

> 1. Куда := 1.
> 2. Минимальная := Куда.
> 3. Откуда := Куда + 1.
> 4. Если Масса(Откуда) > Масса(Куда), идти к 6.
> 5. Минимальная := Откуда.
> 6. Откуда := Откуда + 1.
> 7. Если Откуда ≤ КоличествоПлит, идти к 4.
> 8. Если Минимальная = Куда, идти к 10.
> 9. Поменять местами плиты Минимальная и Куда.
> 10. Куда := Куда + 1.
> 11. Если Куда < КоличествоПлит, идти к 2.
> 12. Останов.

Приведённый здесь алгоритм достаточно прост, но можете ли Вы с ходу сказать, какую задачу он решает?

Критически настроенный читатель скажет: «Ну так когда я читаю чужой алгоритм или составляю свой собственный, общаясь с заказчиком, я знаю, какая задача решается». Ну что же, давайте уточним этот момент: приведённый алгоритм решает задачу упорядочения железобетонных плит, лежащих в один ряд, по возрастанию их массы.

Если теперь Вам кажется, что алгоритм как на ладони и всё понятно, не спешите. Если заставить Вас своими силами перемещать плиты по этому алгоритму, Вы очень быстро ощутите острое желание поговорить с составителем этого алгоритма по душам. Сам алгоритм весьма неплох, но есть в нём ошибка, которая делает все старания исполнителя бесполезными. И вот вопрос: а Вы видите эту ошибку?

А находится она в 4-й строке, которая должна выглядеть вот так:

> 4. Если Масса(Откуда) > Масса(Минимальная), идти к 6.

К тому же остаётся открытым вопрос, какой вариант неравенства — строгое «больше» или же нестрогое «больше или равно» — будет предпочтительнее.

Но, пожалуй, основной проблемой словесного описания является то, что большое количество условных и безусловных переходов сильно мешает анализировать правильность записи алгоритма. К тому же при составлении словесного описания часто приходится добавлять в алгоритм новые шаги, а значит, и перенумеровывать уже имеющиеся, что не очень-то удобно.

Один из очевидных способов решения перечисленных проблем заключается в том, чтобы записывать алгоритм менее детально и более схематично, т.е. в виде рисунков. Такие рисунки — это и есть схемы алгоритмов. Конечно же, программисты по всему миру придумывали и активно использовали различные способы нарисовать алгоритм практически с самого появления профессии, если не раньше. Но мало нарисовать алгоритм — нужно ещё и сделать так, чтобы другой человек смог этот рисунок понять. С этим-то и возникали некоторые проблемы. Вот, скажем, проверка условия: кто-то обозначил прямоугольником, кто-то ромбом, кто-то показывает, куда переходить по условию линиями и стрелками, кто-то вообще обходится без линий — поди пойми, что автор схемы имел в виду!

Дальше так продолжаться не могли, и в 1985 году специально обученные люди издали международный стандарт ISO 5807-85, в котором попытались как-то упорядочить имевшийся на тот момент опыт специалистов. Вскоре после этого появилась и русскоязычная адаптация стандарта, которая была принята под именем ГОСТ 19.701-90 и наравне с международным стандартом ISO 5807-85 используется до настоящего времени.

Общие сведения о ГОСТ 19.701-90

Что же предлагает нам ГОСТ 19.701-90? В нём предусмотрено 5 видов схем, описывающих различные аспекты внутреннего устройства программного (и частично аппаратного) обеспечения.

1) Схема данных

Данные — это то, ради обработки чего создаётся программа. Схемы данных ГОСТ 19.701-90 позволяют изобразить пути, которые проходят данные в ходе своей обработки, собственно этапы их обработки и то, какие носители данных при этом используются (т.е. на чём они записываются и откуда считываются).

2) Схема программы

Схема программы отображает последовательность операций в программе. Именно эти схемы стали использовать в качестве схем алгоритмов. По сути, программы и схема алгоритма — это одно и то же. Но если стремиться к более точному смыслу названий, то схема алгоритма должна быть максимально независимой от того, как алгоритм будет реализован в виде программы. Схема же программы, как следует из названия, может содержать специфические для конкретного языка программирования приёмы, обозначения и описывать алгоритм ближе к тому, как он будет реализован в конкретной программе.

3) Схема работы системы

Схемы работы системы используются в случаях, когда в рамках проекта имеется несколько программ, которые решают поставленную задачу совместно. Такие схемы отображают управление операциями и потоки данных в системе (т.е. в этой совокупности программ), причём одна и та же программа может изображаться в пределах схемы неоднократно.

4) Схема взаимодействия программ

Как и схемы работы системы, схемы взаимодействия программ используются для отображения того, как происходит совместная работа нескольких программ. Отличие заключается в том, на чём делается акцент при построении схемы: на том, как в системе перемещаются данные, или же на том, какие вообще программы в составе системы могут взаимодействовать между собой и каким образом. Каждая программа на такой схеме изображается один раз.

Можно сравнить такую схему со схемой взаимодействия сотрудников или целых отделов в крупной организации.

5) Схема ресурсов системы

На схемах ресурсов системы изображают конфигурацию блоков данных и обрабатывающих блоков, требуемую для решения задачи. Здесь акцент уже не на том, как именно должна происходить обработка данных, а на том, какие ресурсы (отсюда и название), т.е. источники данных и устройства для хранения результатов, для этого нужны.

Для всех перечисленных видов схем используются схожие правила их выполнения и примерно одинаковый набор условных обозначений. Кроме того, часто можно встретить некоторые обозначения из ГОСТ 19.701-90 (например, фигуры, которыми обозначаются места хранения данных — оперативная память, жёсткий диск и т.п.) вне этих схем.

Чаще всего, когда говорят о схемах алгоритмов, имеют в виду именно схемы, построенные по ГОСТ 19.701-90. В этом случае под схемой алгоритма понимают графическое представление алгоритма, в котором этапы процесса обработки информации и носители информации представлены в виде геометрических символов, а последовательность процесса отражена направлением линий.

«Последовательность процесса», т.е. то, в каком порядке выполняются отдельные шаги (или этапы) алгоритма, ещё называют потоком управления. Мы начинали с того, что использование нумерации шагов для описания потока управления может сильно запутывать. Как видим, в схемах алгоритмов для этих целей используются линии, соединяющие отдельные фигуры-этапы в порядке их выполнения.

Условные обозначения, используемые в схемах по ГОСТ 19.701-90, называются символами. Стандарт определяет 4 вида символов:

1) символы данных;
2) символы процесса;
3) символы линий;
4) специальные символы.

Символы данных и процесса, как следует из названия, используются для того, чтобы отразить на схемах обрабатываемые данные и собственно этапы их обработки. Символы линий, как уже было отмечено ранее, на схемах алгоритмов показывают направление потока управления. На других видах схем ими же отображается порядок передачи данных между частями программы или системы (это ещё называется потоками данных).

Специальные символы — это балкон ГОСТ 19.701-90. К ним отнесено всё, что не поместилось в квартире, т.е. не подошло в другие группы символов. Ближайшие родственники этой группы символов — «и т.д.», «и др.» и «проч.». Тем не менее, они используются достаточно части, так что, как говорят англичане, «последние по порядку, но не последние по значимости».

Каждая из первых трёх групп подразделяется на две подгруппы:

* основные символы;
* специфические символы.

Основные символы — это тот минимум, которые позволяет строить простейшие схемы без особой детализации того, что именно Вы пытаетесь сделать в своей программе. Что-то вроде хлеба и воды: если есть хотя бы это, то шансы выжить ещё есть. Специфические символы — это масло, яйца, сахар, крупы, колбаса, мясные рулеты, тройной Биг Мак с пыльцой антарктических колибри и трюфели с сыром Пуле, т.е. всё то, что наполняет жизнь каждого из нас простыми человеческими радостями, а схемы по ГОСТ 19.701-90 — более глубоким смыслом.