Математика делает то, что *можно*, так, как *нужно*, тогда как информатика делает то, что *нужно*, так, как *можно*.

Программистский фольклор

Лекция 1.
НАДЕЖНОЕ ПРОГРАММНОЕ СРЕДСТВО КАК ПРОДУКТ ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ. ИСТОРИЧЕСКИЙ И СОЦИАЛЬНЫЙ КОНТЕКСТ ПРОГРАММИРОВАНИЯ

Понятие информационной среды процесса обработки данных. Программа как формализованное описание процесса. Понятие о программном средстве. Понятие ошибки в программном средстве. Неконструктивность понятия правильной программы. Надежность программного средства. Технология программирования как технология разработки надежных программных средств. Технология программирования и информатизация общества.

1.1. Программа как формализованное описание процесса обработки данных. Программное средство.

Целью программирования является описание процессов обработки данных (в дальнейшем – просто *процессов*). Согласно ИФИПа [1.1]: *данные* (*data*) – это представление фактов и идей в формализованном виде, пригодном для передачи и переработке в некоем процессе, а *информация* (*information*) – это смысл, который придается данным при их представлении. *Обработка данных* (*data processing*) – это выполнение систематической последовательности действий с данными. Данные представляются и хранятся на т.н. *носителях данных*. Совокупность носителей данных, используемых при какой-либо обработке данных, будем называть *информационной средой* (*data medium*). Набор данных, содержащихся в какой-либо момент в информационной среде, будем называть *состоянием* этой информационной среды. *Процесс* можно определить как последовательность сменяющих друг друга состояний некоторой информационной среды.

Описать процесс – это значит определить последовательность состояний заданной информационной среды. Если мы хотим, чтобы по заданному описанию требуемый процесс порождался *автоматически* на каком-либо компьютере, необходимо, чтобы это описание было *формализованным*. Такое описание называется *программой*. С другой стороны, программа должна быть понятной и человеку, так как и при разработке программ, и при их использовании часто приходится выяснять, какой именно процесс она по-

рождает. Поэтому программа составляется на удобном для человека формализованном *языке программирования*, с которого она автоматически переводится на язык соответствующего компьютера с помощью другой программы, называемой *транслятором*. Человеку (*программисту*), прежде чем составить программу на удобном для него языке программирования, приходится проделывать большую подготовительную работу по уточнению постановки задачи, выбору метода ее решения, выяснению специфики применения требуемой программы, прояснению общей организации разрабатываемой программы и многое другое. Использование этой информации может существенно упростить задачу понимания программы человеком, поэтому весьма полезно ее как-то фиксировать в виде отдельных документов (часто не формализованных, рассчитанных только для восприятия человеком).

Обычно программы разрабатываются в расчете на то, чтобы ими могли пользоваться люди, не участвующие в их разработке (их называют *пользователями*). Для освоения программы пользователем помимо ее текста требуется определенная дополнительная документация. Программа или логически связанная совокупность программ на носителях данных, снабженная программной документацией, называется *программным средством* (*ПС*). Программа позволяет осуществлять некоторую автоматическую обработку данных на компьютере. Программная документация позволяет понять, какие функции выполняет та или иная программа ПС, как подготовить исходные данные и запустить требуемую программу в процесс ее выполнения, а также: что означают получаемые результаты (или каков эффект выполнения этой программы). Кроме того, программная документация помогает разобраться в самой программе, что необходимо, например, при ее модификации.

1.2. Неконструктивность понятия правильной программы.

Таким образом, можно считать, что продуктом технологии программирования является ПС, содержащее программы, выполняющие требуемые функции. Здесь под «программой» часто понимают правильную программу, т.е. программу, не содержащую ошибок. Однако, понятие ошибки в программе трактуется в среде программистов неоднозначно. Согласно Майерсу [1.2, стр. 10-13] будем считать, что в программе имеется *ошибка*, если она не выполняет того, что разумно ожидать от нее пользователю. «Разумное ожидание» пользователя формируется на основании документации по применению этой программы. Следовательно, понятие ошибки в программе является существенно не формальным. В ПС программы и документация взаимно увязаны, образуют некоторую целостность. Поэтому пра-

вильнее говорить об ошибке не в программе, а в ПС в целом: будем считать, что в ПС имеется *ошибка* (*software error*), если оно не выполняет того, что разумно ожидать от него пользователю. В частности, разновидностью ошибки в ПС является несогласованность между программами ПС и документацией по их применению. В работе [1.3] выделяется в отдельное понятие частный случай ошибки в ПС, когда программа не соответствует своей функциональной спецификации (описанию, разрабатываемому на этапе, предшествующему непосредственному программированию). Такая ошибка в указанной работе называется *дефектом программы*. Однако выделение такой разновидности ошибки в отдельное понятие вряд ли оправданно, так как причиной ошибки может оказаться сама функциональная спецификация, а не программа.

Так как задание на ПС обычно формулируется не формально, а также из-за того, что понятия ошибки в ПС не формализовано, то нельзя доказать формальными методами (математически) правильность ПС. Нельзя показать правильность ПС и тестированием: как указал Дейкстра [1.4], тестирование может лишь продемонстрировать наличие в ПС ошибки. Поэтому понятие правильной ПС неконструктивно в том смысле, что после окончания работы над созданием ПС мы не сможем убедиться, что достигли цели.

1.3. Надежность программного средства.

Альтернативой правильного ПС является *надежное* ПС. *Надежность* (*reliability*) ПС – это его способность безотказно выполнять определенные функции при заданных условиях в течение заданного периода времени с достаточно большой вероятностью [1.5]. При этом под *отказом в* ПС понимают проявление в нем ошибки [1.2, стр. 10-13]. Таким образом, надежное ПС не исключает наличия в нем ошибок – важно лишь, чтобы эти ошибки при практическом применении этого ПС в заданных условиях проявлялись достаточно редко. Убедиться, что ПС обладает таким свойством можно при его испытании путем тестирования, а также при практическом применении. Таким образом, фактически мы можем разрабатывать лишь надежные, а не правильные ПС.

ПС может обладать различной степенью надежности. Как измерять эту степень? Так же как в технике, степень надежности можно характеризовать [1.2, стр. 10-13] вероятностью работы ПС без отказа в течение определенного периода времени. Однако в силу специфических особенностей ПС определение этой вероятности наталкивается на ряд трудностей по сравнению с решением этой задачи в технике. Позже мы вернемся к более обстоятельному обсуждению этого вопроса.

При оценке степени надежности ПС следует также учитывать последствия каждого отказа. Некоторые ошибки в ПС могут вызывать лишь некоторые неудобства при его применении, тогда как другие ошибки могут иметь катастрофические последствия, например, угрожать человеческой жизни. Поэтому для оценки надежности ПС иногда используют дополнительные показатели, учитывающие стоимость (вред) для пользователя каждого отказа.

1.4. Технология программирования как технология разработки надежных программных средств.

В соответствии с обычным значением слова «технология» [1.6] под *технологией программирования* (*programming technology*) будем понимать совокупность производственных процессов, приводящую к созданию требуемого ПС, а также описание этой совокупности процессов. Другими словами, технологию программирования мы будем понимать здесь в широком смысле как технологию разработки *программных средств*, включая в нее все процессы, начиная с момента зарождения идеи этого средства, и, в частности, связанные с созданием необходимой программной документации. Каждый процесс этой совокупности базируется на использовании каких-либо методов и средств, например, компьютер (в этом случае будем говорить о *компьютерной* технологии программирования).

В литературе имеются и другие, несколько отличающиеся, определения технологии программирования. Эти определения обсуждаются в работе [1.7]. Используется в литературе и близкое к технологии программирования понятие *программной инженерии*, определяемой как систематический подход к разработке, эксплуатации, сопровождению и изъятию из обращения программных средств [1.7, стр. 9-16]. Именно программной инженерии (s*oftware engineering*) посвящена упомянутая работа [1.3]. Главное различие между технологией программирования и программной инженерией как дисциплинами для изучения заключается в способе рассмотрения и систематизации материала. В технологии программирования акцент делается на изучении процессов разработки ПС (*технологических процессов*) и порядке их прохождения – методы и инструментальные средства разработки ПС *используются* в этих процессах (их применение и образуют технологические процессы). Тогда как в программной инженерии изучаются различные методы и инструментальные средства разработки ПС с точки зрения достижения определенных целей – эти методы и средства могут использоваться в разных технологических процессах (и в разных технологиях программирования).

Не следует также путать технологию программирования с методологией программирования [1.8]. В технологии программирования методы рассматриваются «сверху» – с точки зрения организации технологических процессов, а в методологии программирования методы рассматриваются «снизу» – с точки зрения основ их построения (в работе [1.9, стр. 25] методология программирования определяется как совокупность механизмов, применяемых в процессе разработки программного обеспечения и объединенных одним общим философским подходом).

Имея ввиду, что надежность является неотъемлемым атрибутом ПС, мы будем рассматривать технологию программирования как технологию разработки *надежных* ПС. Это означает, что

- мы будем рассматривать все процессы разработки ПС, начиная с момента возникновения замысла ПС;
- нас будут интересовать не только вопросы построения программных конструкций, но и вопросы описания функций и принимаемых решений с точки зрения их человеческого (неформального) восприятия;
- в качестве продукта технологии принимается надежная (далеко не всегда правильная) ПС.

Такой взгляд на технологию программирования будет существенно влиять на организацию технологических процессов, на выбор в них методов и инструментальных средств.

1.5. Технология программирования и информатизация общества.

Технологии программирования играло разную роль на разных этапах развития программирования. По мере повышения мощности компьютеров и развития средств и методологии программирования росла и сложность решаемых на компьютерах задач, что привело к повышенному вниманию к технологии программирования. Резкое удешевление стоимости компьютеров и, в особенности, стоимости хранения информации на компьютерных носителях привело к широкому внедрению компьютеров практически во все сферы человеческой деятельности, что существенно изменило направленность технологии программирования. Человеческий фактор стал играть в ней решающую роль. Сформировалось достаточно глубокое понятие качества ПС, причем предпочтение стало отдаваться не столько эффективности ПС, сколько удобству работы с ним для пользователей (не говоря уже о его надежности). Широкое использование компьютерных сетей привело к интенсивному развитию распределенных вычислений, дистанционного доступа к информации и электронного способа обмена сообщениями между

людьми. Компьютерная техника из средства решения отдельных задач все более превращается в средство информационного моделирования реального и мыслимого мира, способное просто отвечать людям на интересующие их вопросы. Начинается этап глубокой и полной информатизации (компьютеризации) человеческого общества. Все это ставит перед технологией программирования новые и достаточно трудные проблемы.

Сделаем краткую характеристику развития программирования по десятилетиям.

В 50-е годы мощность компьютеров (первого поколения) была невелика, а программирование для них велось, в основном, в машинном коде. Решались, главным образом, научно-технические задачи (счет по формулам), задание на программирование содержало, как правило, достаточно точную постановку задачи. Использовалась интуитивная технология программирования: почти сразу приступали к составлению программы по заданию, при этом часто задание несколько раз изменялось (что сильно увеличивало время и без того итерационного процесса составления программы), минимальная документация оформлялась уже после того, как программа начинала работать. Тем не менее, именно в этот период родилась фундаментальная для технологии программирования концепция модульного программирования [1.10], ориентированная на преодоления трудностей программирования в машинном коде. Появились первые языки программирования высокого уровня, из которых только ФОРТРАН пробился для использования в следующие десятилетия.

В 60-е годы можно было наблюдать бурное развитие и широкое использование языков программирования высокого уровня (АЛГОЛ 60, ФОРТРАН, КОВОЛ и др.), значение которых в технологии программирования явно преувеличивалась. Надежда на то, что эти языки решат все проблемы, возникающие в процессе разработки больших программ, не оправдалась. В результате повышения мощности компьютеров и накопления опыта программирования на языках высокого уровня быстро росла сложность решаемых на компьютерах задач, в результате чего обнаружилась ограниченность языков, проигнорировавших модульную организацию программ. И только ФОРТРАН, бережно сохранивший возможность модульного программирования, гордо прошествовал в следующие десятилетия (все его ругали, но его пользователи отказаться от его услуг не могли из-за грандиозного накопления фонда программных модулей, которые с успехом использовались в новых программах). Кроме того, было понято, что важно не только то, на каком языке мы программируем, но и то, как мы программируем [1.4]. Это было уже началом серьез-

ных размышлений над методологией и технологией программирования. Появление в компьютерах 2-го поколения прерываний привело к развитию мультипрограммирования и созданию больших программных систем. Широко стала использоваться коллективная разработка, которая поставила ряд серьезных технологических проблем [1.11].

В 70-е годы получили широкое распространение информационные системы и базы данных. К середине 70-ых годов стоимость хранения одного бита информации на компьютерных носителях стала меньше, чем на традиционных носителях. Это резко повысило интерес к компьютерным системам хранения данных. Началось интенсивное развитие технологии программирования [1.2, 1.8, 1.12-1.14], прежде всего, в следующих направлениях:

- обоснование и широкое внедрение нисходящей разработки и структурного программирования,
- развитие абстрактных типов данных и модульного программирования (в частности, возникновение идеи разделения спецификации и реализации модулей и использование модулей, скрывающих структуры данных),
- исследование проблем обеспечения надежности и мобильности ПС,
- создание методики управления коллективной разработкой ПС,
- появление инструментальных программных средств (программных инструментов) поддержки технологии программирования.

80-е годы характеризуются широким внедрением персональных компьютеров во все сферы человеческой деятельности и тем самым созданием обширного и разнообразного контингента пользователей ПС. Это привело к бурному развитию пользовательских интерфейсов и созданию четкой концепции качества ПС [1.5, 1.15-1.18]. Появляются языки программирования (например, Ада), учитывающие требования технологии программирования [1.19]. Развиваются методы и языки спецификации ПС [1.20-1.21]. Начинается бурный процесс стандартизации технологических процессов и, прежде всего, документации, создаваемой в этих процессах [1.22]. Выходит на передовые позиции объектный подход к разработке ПС [1.9]. Создаются различные инструментальные среды разработки и сопровождения ПС [1.3]. Развивается концепция компьютерных сетей.

90-е годы знаменательны широким охватом всего человеческого общества международной компьютерной сетью, персональные компьютеры стали подключаться к ней как терминалы. Это постави-

ло ряд проблем (как технологического, так и юридического и этического характера) регулирования доступа к информации компьютерных сетей. Остро встала проблема защиты компьютерной информации и передаваемых по сети сообщений. Стали бурно развиваться компьютерная технология (CASE-технология) разработки ПС и связанные с ней формальные методы спецификации программ. Начался решающий этап полной информатизации и компьютеризации общества.

Упражнения к лекции 1.

1.1. Что такое информационная среда программы?
1.2. Что такое программное средство (ПС)?
1.3. Что такое ошибка в ПС?
1.4. Что такое надежность ПС?
1.5. Что такое технология программирования?

Литература к лекции 1.

[1.1] И.Г.Гоулд, Дж.С.Тутилл. Терминологическая работа IFIP (Международная федерация по обработке информации) и ICC (Международный вычислительный центр) // Журн. вычисл. матем. и матем. физ., 1965, #2. — С. 377-386.
[1.2] Г.Майерс. Надежность программного обеспечения. — М.: Мир, 1980.
[1.3] Ian Sommerville. Software engineering. — Addison-Wesley Publishing Company, 1992.
[1.4] Э. Дейкстра. Заметки по структурному программированию / У. Дал, Э. Дейкстра, К. Хоор. Структурное программирование. — М.: Мир, 1975. — С. 7-97.
[1.5] Criteria for evaluation of software. — ISO TC97/SC7 #367 (Supersedes Document #327).
[1.6] С.И. Ожегов. Словарь русского языка. — М.: Советская энциклопедия, 1975.
[1.7] Ф.Я. Дзержинский, И.М. Калиниченко. Дисциплина программирования Д: концепция и опыт реализации методических средств программной инженерии. — М.: ЦНИИ информации и технико-экономических исследований по атомной науке и технике, 1988.
[1.8] В. Турский. Методология программирования. — М.: Мир, 1981.
[1.9] Г. Буч. Объектно-ориентированное проектирование с примерами применения. — М.: Конкорд, 1992.

[1.10] Е.А. Жоголев. Система программирования с использованием библиотеки подпрограмм / Система автоматизация программирования. — М.: Физматгиз, 1961. — С. 15-52.
[1.11] Ф.П. Брукс, мл. Как проектируются и создаются программные комплексы. — М.: Наука, 1979.
[1.12] R.C. Holt. Structure of computer programs: A Survey // Proceedings of the IEEE, 1975, 63(6). — P. 879-893.
[1.13] Дж. Хьюз, Дж. Мичтом. Структурный подход к программированию. — М.: Мир, 1980.
[1.14] Е.А. Жоголев. Технологические основы модульного программирования // Программирование, 1980, #2. — С. 44-49.
[1.15] Б. Боэм, Дж. Браун, Х. Каспар и др. Характеристики качества программного обеспечения. — М.: Мир, 1981.
[1.16] В.В. Липаев. Качество программного обеспечения. — М.: Финансы и статистика, 1983.
[1.17] Б. Шнейдерман. Психология программирования. — М.: Радио и связь, 1984.
[1.18] Revised version of DP9126 — Criteria of the evaluation of software quality characteristics. ISO TC97/SC7 #610. Part 6.
[1.19] В.Ш. Кауфман. Языки программирования. Концепции и принципы. — М.: Радио и связь, 1993.
[1.20] Требования и спецификации в разработке программ. — М.: Мир, 1984.
[1.21] В.Н. Агафонов. Спецификация программ: понятийные средства и их организация. — Новосибирск: Наука (Сибирское отделение), 1987.
]1.22] В.В. Липаев, Е.Н Филиппов. Мобильность программ и данных в открытых информационных системах. — М.: Научная книга, 1997.