шающей стадии успех (или провал) разработки по существу определяется ответом конечного пользователя на два основных вопроса: 1) сделано ли именно то, что он пытался сформулировать (самостоятельно или с помощью системного аналитика) как задачу разработки программного продукта (или результатом разработки оказалось нечто иное); 2) не изменилась ли точка зрения конечного пользователя на то, что он ждет от заказанной программы за время ее разработки.

Один из наиболее драматических парадоксов технологии программирования заключается в том, что профессионально сделанная прикладная программа с вероятностью, близкой к единице, оказывается бесполезной. В более чем 9 случаев из 10 организация-пользователь или отдельные конечные пользователи дают разочаровывающий программиста ответ по крайней мере на один из двух поставленных выше вопросов. Например, по результатам исследований, выполненных компанией «Эпплайд дейта рисеч», на американском рынке программного обеспечения ЭВМ в среднем только 1 из 10 разрабатываемых пакетов имеет шанс на успех у пользователя [13].

При этом данная оценка еще не означает, что индустрия прикладных программ к началу 80-х годов работала, пользуясь терминологией изготовителей интегральных схем, с «10%-ным выходом годных». К сожалению, и эту оценку следует считать завышенной, поскольку на усредненные «10% выхода годных» приходятся и все те программы, которые хотя и были приобретены конечным пользователем, но затем сразу осели на стеллажах с неиспользованными колодами.

Такая ситуация, как правило, складывается, когда организация приобретает пакет, оказавшийся (по отзывам) полезным в другой, близкой по производственному профилю организации. Причины возникающих затем разочарований понятны. «Жизнь сложна потому, что конкретна»,— поясняет М. И. Лазарев, один из разработчиков в НИВЦ АН СССР пакетов для прочностных расчетов. Американские эксперты предпочитают афоризмам количественные иллюстрации: «Если у вас есть пакет прикладных программ и вы устанавливаете его в 16 различных организациях, то при этом вы создаете 16 различных пакетов» [14, p. 142].

Чтобы объяснить причины, по которым пользователь на этапе внедрения нередко отказывается от сформулированных им же самим требований на программу и утверждает, что имелось в виду нечто совершенно иное, Дж. Мартин