2.2. Надежность систем автоматизации

Наиболее остро проблема надежности проявляется в системах автоматизации, поскольку выход одного (на первый взгляд не существенного) компонента может привести к сбою работы всей системы в целом.

Одним из важнейших параметров системы является **коэффициент готовности** (*availability*), который определяется как вероятность того, что элемент, устройство или система в данный момент времени работает правильно. Эта вероятность представляет собой отношение времени, в течение которого элемент исправен, ко всему сроку службы. Коэффициент готовности элемента или устройства – это функция вероятности отказа в течение заданного периода времени, за которое элемент или устройство приводится в рабочее состояние после отказа.

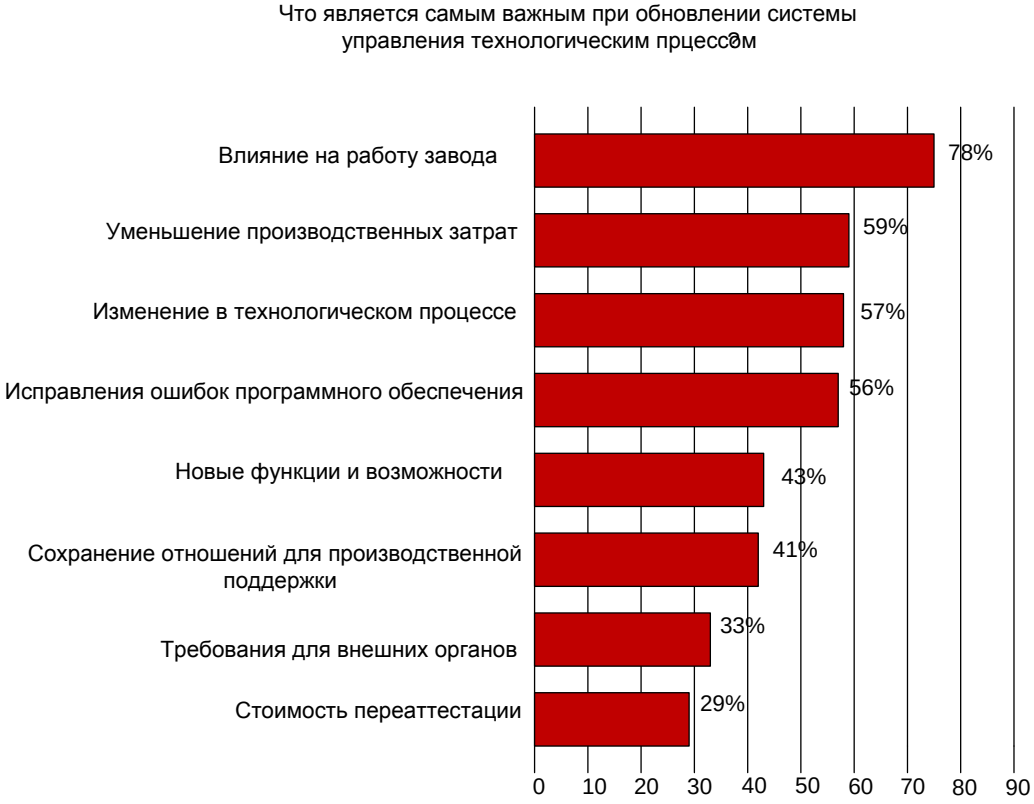

Рис. 2.3. Важнейшие показатели
при обновлении системы управления технологическим процессом

Из-за взаимодействия между частями и компонентами полная надежность системы может оказаться достаточно малой, если не все составляющие имеют высокую надежность. В качестве примера, можно рассмотреть производственную линию, включающую десять последовательно соединенных станков. Если каждый станок все время повторяет одни и те же операции и делает одну ошибку в среднем на каждую сотню операций, то

вероятность того, что станок не сделает ошибку за цикл, равна 99 %. Для получения приемлемого конечного продукта все станки должны работать правильно, так что общая вероятность $0{,}99^{10} = 0{,}904$, т.е. вероятность безошибочной работы всей линии всего лишь около 90 %. Ввиду этого на производстве широко используемый метод улучшенной суммарной надежности на конвейерах – создание промежуточных складов между станками. В этом случае удается избежать остановки всего конвейера при отказе одного станка.

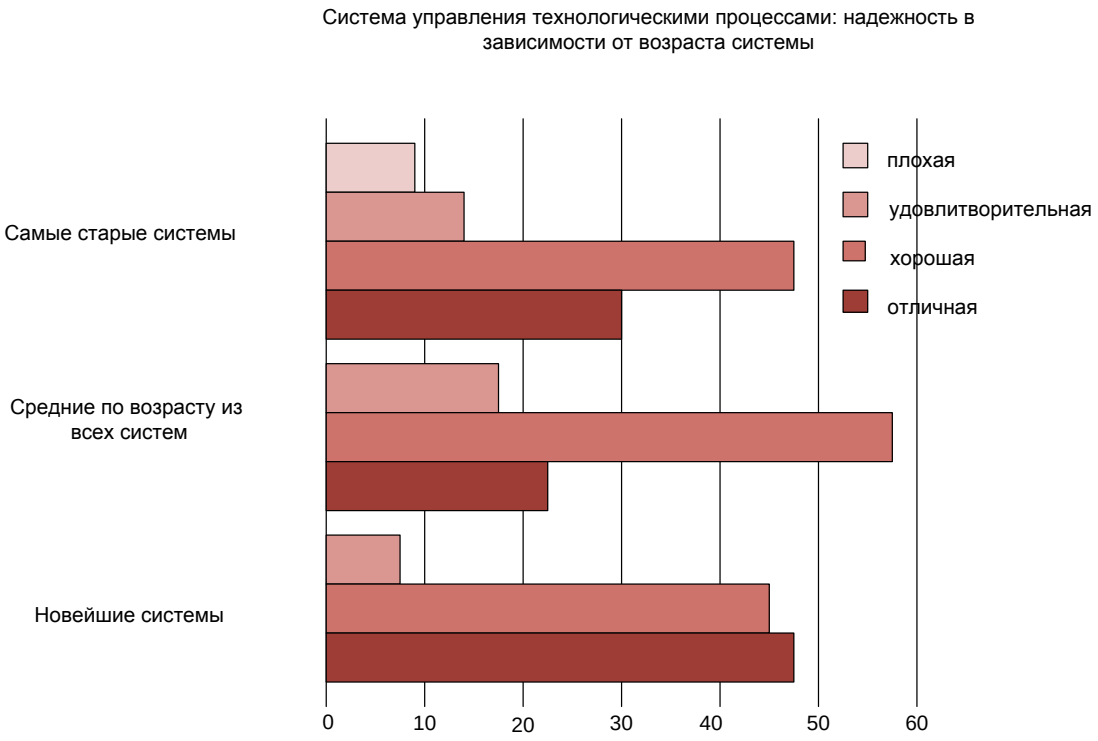

Рис. 2.4. Надежность в зависимости от возраста системы

2.2.1. Модели для расчета надежности

При расчетах надежности сложной системы обычно полагают, что возможные ошибки не коррелируют, т.е. являются независимыми событиями. Это предположение верно при условии, что неисправный элемент не влияет на другие.

Для n элементов

$$n = n_h(\ ) + n_f(\ ),$$

где $n_h(\ )$ – число правильно функционирующих элементов, а $n_f(\ )$ – число неисправных элементов. Оба слагаемых– суть функции времени,

а их сумма n при этом постоянна. **Функция надежности** определяется следующим образом:

$$R(t) = \frac{n_h(t)}{n} = 1 - \frac{n_f(t)}{n}.$$

Интенсивность отказов определяется следующим образом:

$$z(t) = \frac{f(t)}{R(t)} = -\frac{1}{R(t)} \cdot \frac{d}{dt} R(t).$$

Если элемент остается в работе до времени t, интенсивность отказа показывает вероятность того, что этот элемент откажет сразу после момента t. Функцию интенсивности отказов $z(t)$ можно оценить, наблюдая большое число элементов в течение длительного периода времени. Несколько упрощенный вид функции $z(t)$ представлен на рис. 2.5. Из-за своей формы эта кривая называется «корытообразной» функцией.

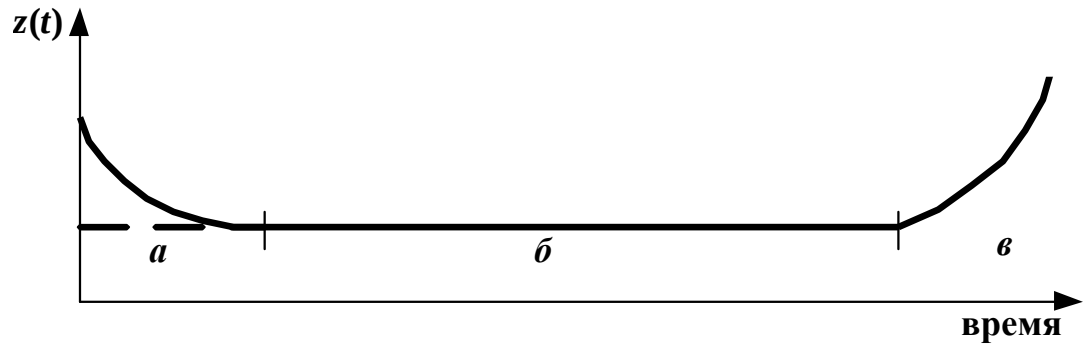

Рис. 2.5. Интенсивность отказов элементов $z(t)$ как функция времени:
а – отказы на первоначальных этапах работы; *б* – случайные отказы;
в – отказы, связанные со старением

При рассмотрении системы обычно анализируется центральная часть кривой интенсивности отказов. При этом полагают, что система функционировала достаточно долгое время для того, чтобы избежать «детские болезни». С другой стороны, система не должна находиться в эксплуатации так долго, что ее компоненты уже износились и уровень отказов снова увеличился. С этими ограничениями $z(t)$ можно принять за константу $z(t) = \lambda$. Тогда решением уравнения для интенсивности отказов будет

$$R(t) = e^{-\lambda t}.$$

Основной интерес для пользователя или производителя системных компонентов представляет собой время, в течение которого компонент

может функционировать в нормальных условиях до появления неисправности. Мерой этого является **среднее время наработки на отказ** (Mean Time to Failures – MTTF), т.е. математическое ожидание экспоненциального распределения

$$MTTF = \int_0^\infty R(t)dt = \frac{1}{\lambda}.$$

Мера готовности системы получается на основе среднего значения промежутка времени, в течение которого система функционирует правильно. Это значение называется **средним временем между отказами** (Mean Time Between Failures – MTBF). Аналогично мера времени, в течение которого система не функционирует, называется **средним временем восстановления** (Mean Time to Repair – MTTR) и представляет собой время от появления неисправности до восстановления работоспособности системы.

Коэффициент готовности A элемента или подсистемы можно определить выражением

$$A = \frac{MTBF}{MTBF + MTTR}.$$

Для систем, состоящих из одного устройства или элемента, коэффициент готовности легко подсчитать. Для систем с параллельным соединением одинаковых элементов увеличивается надежность, так как только если все компоненты, соединенные параллельно, неисправны, функция системы больше не обеспечивается.

2.2.2. Надежность систем управления процессами

Частота отказов существенно влияет на надежность системы. При этом существенную роль играют ремонты. С этой точки зрения электронным системам управления технологическими процессами отдается предпочтение. Например, в случае отказа релейного оборудования поиск неисправности с помощью вольтметра занимает длительное время. В случае же применения ПЛК тестирование и отладка с помощью ручного программатора и соответствующего ПО гораздо проще и быстрее. Однако при этом уровень обслуживающего персонала должен иметь более высокую подготовку.

Помимо этого, суммарная надежность зависит от структуры системы управления технологическим процессом. Например, при централизованном управлении единственная ЭВМ выполняет функции сбора, обработки и выдачи управляющих воздействий на исполнительный механизм и его

поломка вызывает полный останов всей системы управления и технологического процесса. При распределенном управлении функции управления и регулирования выполняются локальными устройствами, расположенными в непосредственной близости от технологических процессов. Поломка локальной или даже центральной ЭВМ влияет только на часть функций, поскольку системные компоненты независимы.

Опыт разработки таких систем показал, что отказы вызываются либо неправильным функционированием отдельных элементов, либо нарушением взаимосвязи между элементами.

В связи с этим появилось понятие «отказоустойчивое решение». **Отказоустойчивое решение –** это решение, гарантирующее, что система как целое будет функционировать даже при наличии неисправностей. В первую очередь это означает, что система строится не только с применением высоконадежных компонентов, но и проектируется таким образом, что отдельные неисправности не повлияют на ее работу. Здесь учитывается не только аппаратная составляющая, но и программная, которая также может содержать ошибки либо непредсказуемо реагировать на поступающую непредусмотренную информацию.

В отказоустойчивую систему закладываются элементы избыточности. Выделяют следующие типы избыточности:

- физическая избыточность;
- информационная избыточность;
- избыточность по времени.

Физическая избыточность обычно достигается дублированием некоторых элементов системы. Когда элемент перестает работать, его заменяют другим элементом. В случае ограничения на стоимость системы обычно дублируют лишь необходимую составляющую системы. Например, покупка базового комплекта ZIP.

Информационная избыточность используется, например, в коммуникационных протоколах в виде служебной информации, добавляемой к пакету для того, чтобы обеспечить восстановление искажений в полезной информации, резервирование данных на внешних носителях информации или теневое хранение переменных.

Избыточность по времени заключается в том, что сначала выполняется действие, а затем оценивается его результат. Если результат неправильный, то действие выполняется заново. Применение тайм-аутов и ограничений максимального количества повторений помогают избежать бесконечных циклов.

Например, отказоустойчивость в коммуникационных протоколах обмена достигается комбинацией информационной и временной избыточностью. Применение контрольной суммы в пакетах данных обеспе-

чивает информационную избыточность, а процедуры подтверждения приема сообщения, запросы на новую передачу являются примерами избыточности.

2.2.3. Надежность программного обеспечения

Ошибки в программах часто могут вводить в заблуждение, их труднее найти, чем неисправности аппаратной части.

Проблемы с программными ошибками зависят от сложности системы, могут многократно воспроизводиться и остаться незамеченными в конечном продукте. Программные ошибки являются обычным делом в сложных проектах, в частности в автоматизированных системах, а сообщается о них только в некоторых очень громких случаях.

В отличие от аппаратной части программное обеспечение не изнашивается. Дефекты программы возникают на стадии разработки, так что теоретически они могут быть устранены с самого начала. Самая главная проблема заключается в том, как их обнаружить. Математические и логические методы помогают разрабатывать не содержащие ошибок программы. Однако на практике, несмотря на многочисленные тесты, большая часть программ все-таки содержит ошибки на начальном этапе их эксплуатации. В случае появления непредусмотренных входных данных программа может повести себя так, как не запланировано.

На практике достаточно часто требования к программе меняются в процессе разработки, что может привести к появлению дополнительных ошибок.

Надежность программы, в соответствии с функциональными требованиями, должна быть определена, например, с помощью тестов. Так, например, корпорация Microsoft привлекает тысячи тестеровщиков по всему миру для отладки своего ПО. Такой метод применим, если требования не являются слишком жесткими, например, как в авиастроении, где специфика надежности выражается цифрами 10^{-9}.

Метод, применяемый для повышения показателей надежности в технике управления авиационным и железнодорожным движением, – это использование избыточных систем. Несколько одинаковых систем создаются параллельно разными рабочими группами. Основное допущение – ошибки не должны повторяться. Результат – это комбинация нескольких решений. Например, на бортовой ЭВМ полностью управляемого электроникой аэробуса А-320 установлено пять разных систем, которые были разработаны на основе одинаковых требований пятью независимыми рабочими группами. Выбор окончательного управляющего воздействия выбирается с помощью электромеханического селектора.

Надежность ПО – это свойство программного обеспечения своевременно выполнять в заранее указанных условиях эксплуатации вперед установленные функции.

В самом общем случае основную функцию ПО АСУТП можно рассматривать как своевременное получение некоторого результата или решения y при переработке входной информации x из множества X.

Под x понимается контрольная информация от технологического объекта управления (ТОУ), сигналы о состоянии технологического оборудования и ТСА, команды управленческого персонала и вышестоящих АСУ и т.п. Результат y зависит как от случайного $x \in X$, так и от свойств ПО, носящих во многом стохастический характер. Поэтому установление каких-либо диапазонов изменения y и тем более границ допустимых или разумных результатов Y оказывается в этом случае невозможным. Вследствие этого становится затруднительной строгая качественная оценка принадлежности данного y множеству «разумных» результатов Y.

Решение о выполнении или невыполнении функций ПО вынужден принимать пользователь и, в меньшей степени – разработчик программы или программист. Таким образом, надежность – это свойство программ обеспечивать «разумные», по мнению пользователя и программиста, решения при переработке входной информации x из условного множества X и нормальном функционировании управляющей вычислительной машины (УВМ).

Отказы ПО делятся на случайные и неслучайные:

Неслучайные отказы ПО обусловлены действием так называемых компьютерных вирусов.

Случайные отказы ПО наблюдаются в случайные моменты времени работы УВМ или процессора. По своим последствиям эти отказы классифицируются на случайные сбои программ и устойчивые отказы ПО.

Под **сбоем ПО** понимают случайное событие, заключающееся в появлении «неразумного» результата $y \in Y$ и исчезающее при последующих прогонах (запусках) программ.

Сбой ПО – это самоустраняющийся (перемежающийся) отказ программы, возникающий при некоторых, возможно случайных, состояниях УВМ и информации $x \in X$, наблюдаемый пользователем в случайные моменты времени и исчезающий без вмешательства программиста.

Устойчивый отказ ПО наблюдается в случайный момент процессорного времени в форме «неразумного» результата $y \in Y$ при $x \in X$ в нормальном функционировании УВМ.

Причиной отказа ПО служит некоторая систематическая ошибка программы, после устранения которой программистом данный отказ исчезает, т.е. имеет место восстановление ПО.

Различают ошибки первичного и вторичного типа.

Ошибки первичного типа связаны с неточностями в текстах программ и возникают при подготовке носителей и документации ПО, при записях кодов на алгоритмических языках и трансляции программ на машинный язык, а также из-за неточностей алгоритмов и при неверных или некорректных постановках решаемых на УВМ вычислительных задач.

Ошибки вторичного типа во многом являются следствием первичных ошибок программ. К ним относят ошибки:

− вычислительные (неверная индексация и подсчет временных параметров, расхождение результата ручного и машинного счета, появление неустойчивых операций и т.п.);

− логические (пропуск логических условий, неверные краевые условия и др.);

− сопряжения интерфейсов (межмодульных, программно-технических, информационных).

Ошибки первичного и вторичного типов порождаются на этапах разработки спецификаций на ПО; проектирования ПО; реализации программ.

Устранение ошибок или восстановление программ осуществляется программистом на этапе отладки ПО, который заканчивается сдачей готовых программ в эксплуатацию. Однако, как показывает опыт исследования надежности сложных ПО, около половины ошибок программ не выявляется на стадии отладки и сдачи ПО в эксплуатацию. Эти ошибки (преимущественно вторичные) проявляют себя в процессе эксплуатации ПО в случайные моменты времени и приводят к отказам программ.

Отказы ПО, при его эксплуатации, имеют ряд отличий от отказов технических элементов:

- Отказ ПО не приводит к разрушению или поломке программного элемента. Отказы ПО не связаны с физическим износом элемента (в частности, носителя программ).
- Отказ ПО не коррелирован с процессорным и тем более астрономическим временем или числом прогонов ПО программ пользователем.
- При длительной эксплуатации ПО все его ошибки могут быть устранены и программы становятся абсолютно надежными. Если обозначить через $N(t)$ число невыявленных ошибок ПО в произвольный момент процессорного времени t, то формально имеет место соотно-

шение $\mathrm{Lim}N(\tau) = 0$, справедливое при условии, что в процессе восстановления программ в них не вносятся новые ошибки.

Опыт создания и эксплуатации ПО реального времени показывает, что при устранении одних ошибок вносятся другие. Поэтому, при длительной эксплуатации ПО, общее число ошибок может оставаться постоянным или даже возрастать.

Основные показатели надежности ПО:

- **функция ненадежности (или отказа) ПО** $Q(t)$ Вероятность того, что отказ ПО появится до момента времени t;
- **функция надежности ПО** $P(t)$ Вероятность того, что отказ ПО появится после момента времени t;
- **интенсивность отказов ПО** $\lambda(t) = dQ/dt$;
- **средняя наработка на** отказ **ПО** $t = \int P(t)dt$.

В сложном ПО надежность определяется надежностью отказов самой «ненадежной» программы, имеющей наибольшее значение интенсивности отказов λ.

Для повышения надежности ПО следует в первую очередь улучшить характеристики самых «ненадежных» программ (более жесткое динамическое тестирование «ненадежных» программ, расширяя при этом набор тестовых задач). Если тестирование не уменьшает интенсивность проявления ошибок, то переписывают «ненадежную» программу, стремясь усилить ее структурированность путем увеличения числа готовых и хорошо изученных программных модулей и стандартных подпрограмм и применения апробированных межмодульных интерфейсов. Понижению интенсивности λ способствует и переход на другой, более высокий, язык программирования.

Другой путь повышения надежности ПО связан с резервированием и введением в программную систему некоторой избыточности.

Применительно к ПО АСУТП различают три вида резервирования:

− временное;
− информационное;
− программное.

Временное резервирование ПО заключается в многократном прогоне одних и тех же «ненадежных» программ и сравнении результатов расчета. Такое нагруженное резервирование позволяет устранять влияние случайных сбоев и выявлять случайные ошибки, требующие восстановления программ.

Информационное резервирование ПО основано на дублированных исходных и промежуточных данных. Эти данные могут проходить

дополнительную обработку, например усреднение, до ввода в ПО, где они обрабатываются один раз; или обрабатываться одной и той же программой дважды, т.е. информационное резервирование подкрепляется временным.

Программное резервирование предусматривает наличие в ПО двух или больше разных программ для получения одного и того же результата или реализации одной функции. Здесь возможно нагруженное и ненагруженное резервирование.