System Maintenance

Introduction

Here we discuss systems maintenance, the **largest systems development expenditure for many organizations**. In fact, more programmers today work on maintenance activities **than** work on new development. Your first job after graduation may very well be as a maintenance programmer/analyst. This disproportionate (too large or too small in comparison to something else) **distribution of maintenance programmers is interesting because software does not wear out in a physical manner as do buildings and machines**. **There is no single reason** why software is maintained; however, most reasons relate to **a desire to evolve system functionality in order to overcome internal processing errors** or **to better support changing business needs**. Thus, maintenance is a fact of life for most systems. This means that maintenance can begin soon after the system is installed.

Maintaining Information Systems:

Once an information system is installed, the system is essentially in the **maintenance phase** of the systems development life cycle (SDLC). When a system is in the maintenance phase, some person within the systems development group is responsible for collecting maintenance requests from system users and other interested parties, such as system auditors, data center and network management staff, and data analysts. Once collected, each request is analyzed to better understand how it will alter the system and what business benefits and necessities will result from such a change. If the change request is approved, a system change is designed and then implemented. As with the initialdevelopment of the system, implemented changes are formally reviewed and tested before installation into operational systems.

The Process of Maintaining information Systems

As we can see in Figure 14-1, the maintenance phase is the last phase of the SDLC. It is here that the SDLC becomes a cycle, with the last activity leading back to the first. This means that the process of maintaining an information system is the process of returning to the beginning of the SDLC and repeating development steps until the change is implemented.

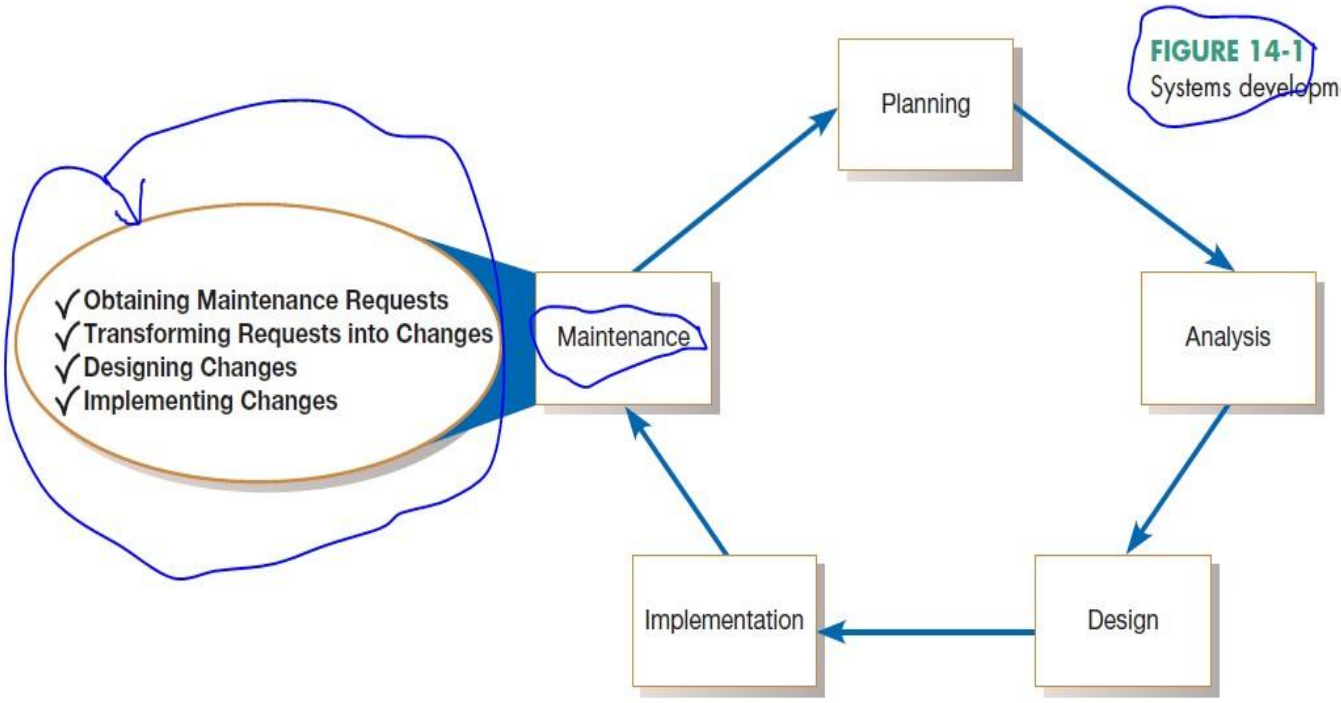

FIGURE 14-1
Systems development life cycle

Also shown in Figure14-1,
Four major activities occur within maintenance:

1. **Obtaining maintenance requests**
2. **Transforming requests into changes**
3. **Designing changes**
4. **Implementing changes**

Obtaining maintenance requests requires that a formal process be established whereby users can submit system change requests. Earlier in this book, we presented a **user request document** called a **System Service Request (SSR)**, which is shown in Figure 14-2 (in next slide). Most companies have some sort of document like an SSR to request new development, to report problems, or to request new features within an existing system. When developing the procedures for obtaining maintenance requests, organizations must also specify an individual within the organization to collect these requests and manage their dispersal to maintenance personnel.

Pine Valley Furniture
System Service Request

REQUESTED BY Juanita Lopez DATE November 5, 2017

DEPARTMENT Purchasing, Manufacturing Support

LOCATION Headquarters, 1-322

CONTACT Tel: 4-3267 FAX: 4-3270 e-mail: jlopez

TYPE OF REQUEST

[X] New System

[ ] System Enhancement

[ ] System Error Correction

URGENCY

[ ] Immediate—Operations are impaired or opportunity lost

[ ] Problems exist, but can be worked around

[X] Business losses can be tolerated until new system is installed

PROBLEM STATEMENT

Sales growth at PVF has caused greater volume of work for the manufacturing support unit within Purchasing. Further, more concentration on customer service has reduced manufacturing lead times, which puts more pressure on purchasing activities. In addition, cost-cutting measures force Purchasing to be more aggressive in negotiating terms with vendors, improving delivery times, and lowering our investments in inventory. The current modest systems support for manufacturing purchasing is not responsive to these new business conditions. Data are not available, information cannot be summarized, supplier orders cannot be adequately tracked, and commodity buying is not well supported. PVF is spending too much on raw materials and not being responsive to manufacturing needs.

SERVICE REQUEST

I request a thorough analysis of our current operations with the intent to design and build a completely new information system. This system should handle all purchasing transactions, support display and reporting of critical purchasing data, and assist purchasing agents in commodity buying.

IS LIAISON Chris Martin (Tel: 4-6204 FAX: 4-6200 e-mail: cmartin)

SPONSOR Sal Divario, Director, Purchasing

---------- TO BE COMPLETED BY SYSTEMS PRIORITY BOARD ----------

[ ] Request approved Assigned to _____
Start date _____

[ ] Recommend revision

[ ] Suggest user development

[ ] Reject for reason _____

Once a request is received, analysis must be conducted to gain an understanding of the scope of the request. It must be determined how the request will affect the current system and how long such a project will take. As with the initial development of a system, the size of a maintenance request can be analyzed for risk and feasibility.

Next, a change request can be transformed into a formal design change, which can then be fed into the maintenance implementation phase. Thus, many similarities exist between the SDLC and the activities within the maintenance process. Figure 14-3 equates SDLC phases to the maintenance activities described previously.

The first phase of the SDLC—planning—is analogous to the maintenance process of obtaining a maintenance request (step 1).

The SDLC analysis phase is analogous to the maintenance process of transforming requests into a specific system change (step 2).

The SDLC design phase, of course, equates to the designing changes process (step 3).

Finally, the SDLC phase implementation equates to implementing changes. (step 4)

This similarity between the maintenance process and the SDLC is no accident. The concepts and techniques used to initially develop a system are also used to maintain it.

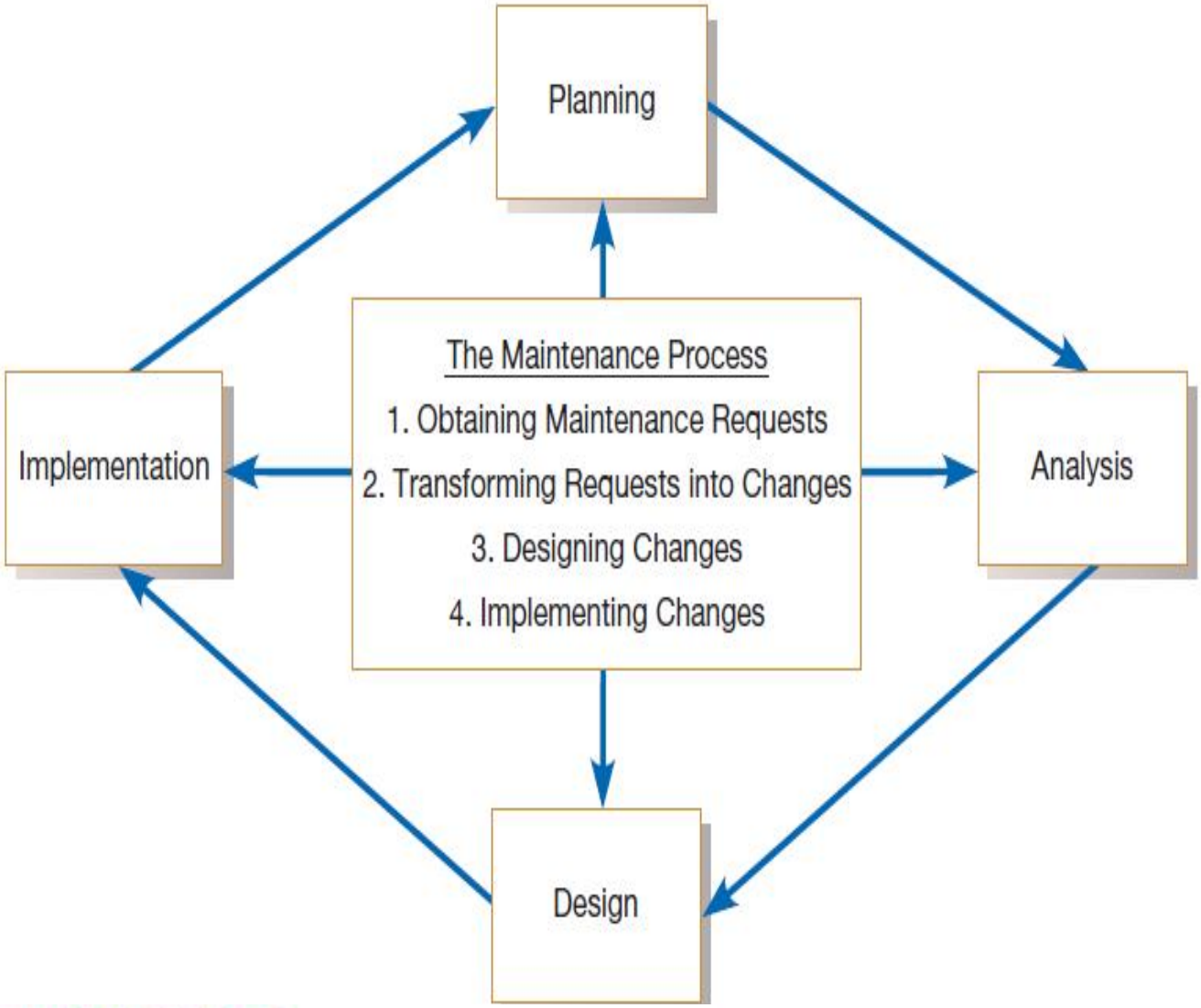

FIGURE 14-3
Maintenance activities parallel those of the SDLC

Conducting Systems Maintenance

A significant within organizations does not go to the development of new systems but to the **maintenance of existing systems**. We will describe various types of maintenance, factors influencing the complexity and cost of maintenance, and alternatives for managing maintenance.

Types of Maintenance

By maintenance, we mean the fixing or enhancing of an information system.

Corrective maintenance: It refers to changes made to repair defects in the design, coding, or implementation of the system. For example, if you had recently purchased a new home, corrective maintenance would involve repairs made to things that had never worked as designed, such as a faulty electrical outlet or a misaligned door. Most corrective maintenance problems surface soon after installation. When corrective maintenance **problems surface, they are typically urgent and need to be resolved to restrict possible interruptions in normal business activities**. Of all types of maintenance, corrective accounts for as much as 75 percent of all maintenance activity (Andrews and Leventhal, 1993; Pressman,2005).This is unfortunate because corrective maintenance adds little or no value to the organization; it simply focuses **on removing defects from an existing system without adding new functionality** (see Figure 14-4).

Adaptive maintenance involves making changes to an information system to evolve **its functionality to changing business needs or to migrate it to a different operating environment.** Within a home, adaptive maintenance might be adding storm windows to improve the cooling performance of an air conditioner. Adaptive maintenance is usually less urgent than corrective maintenance because business and technical changes typically occur over some period of time.

Contrary to corrective maintenance, adaptive maintenance is generally a small part of an organization's maintenance effort, but it adds value to the organization.

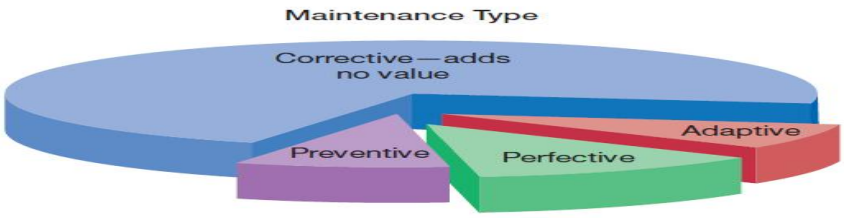

FIGURE 14-4
Value and non-value adding of different types of maintenance
(*Sources:* Based on Andrews and Leventhal, 1993; Pressman, 2005.)

Perfective maintenance involves making enhancements to improve processing performance or interface usability or to add desired, but not necessarily required, system features (bells and whistles). In our home example, perfective maintenance **would be adding a new room**. Many systems professionals feel that perfective **maintenance is not really maintenance but rather new development.**

Preventive maintenance involves changes made to a system to reduce the chance of future system failure. An example of preventive maintenance might be to increase the number of records that a system can process far beyond what is currently needed or to generalize how a system sends report information to a printer so that the system can easily adapt to changes in printer technology.

In our home example, **preventive** maintenance could be **painting the exterior to better protect the home from severe weather conditions**. As with adaptive maintenance, both perfective and preventive maintenance are typically a much lower priority than corrective maintenance. Over the life of a system, corrective maintenance is most likely to occur after initial system installation or after major system changes. This means that adaptive, perfective, and preventive maintenance activities can lead to corrective maintenance activities if not carefully designed and implemented.

The Cost of Maintenance

Information systems maintenance costs are a significant expenditure. For some organizations, as much as **60 to 80** percent of their information systems budget is allocated to maintenance activities (Kaplan, 2002). These huge maintenance costs are due to the fact that many organizations have accumulated more and more older so-called legacy systems that require more and more maintenance (see Figure 14-5). More maintenance means more maintenance work for programmers.

For systems developed in-house, on average, 52 percent of a company's programmers are assigned to maintain **existing software** (Lytton, 2001).In situations where a company has not developed its systems in-house but instead has licensed software, as in the case of ERP(enterprise resource planning) systems, maintenance costs remain high. The standard cost of maintenance for most ERP vendors is 22 percent annually (Nash,2010).

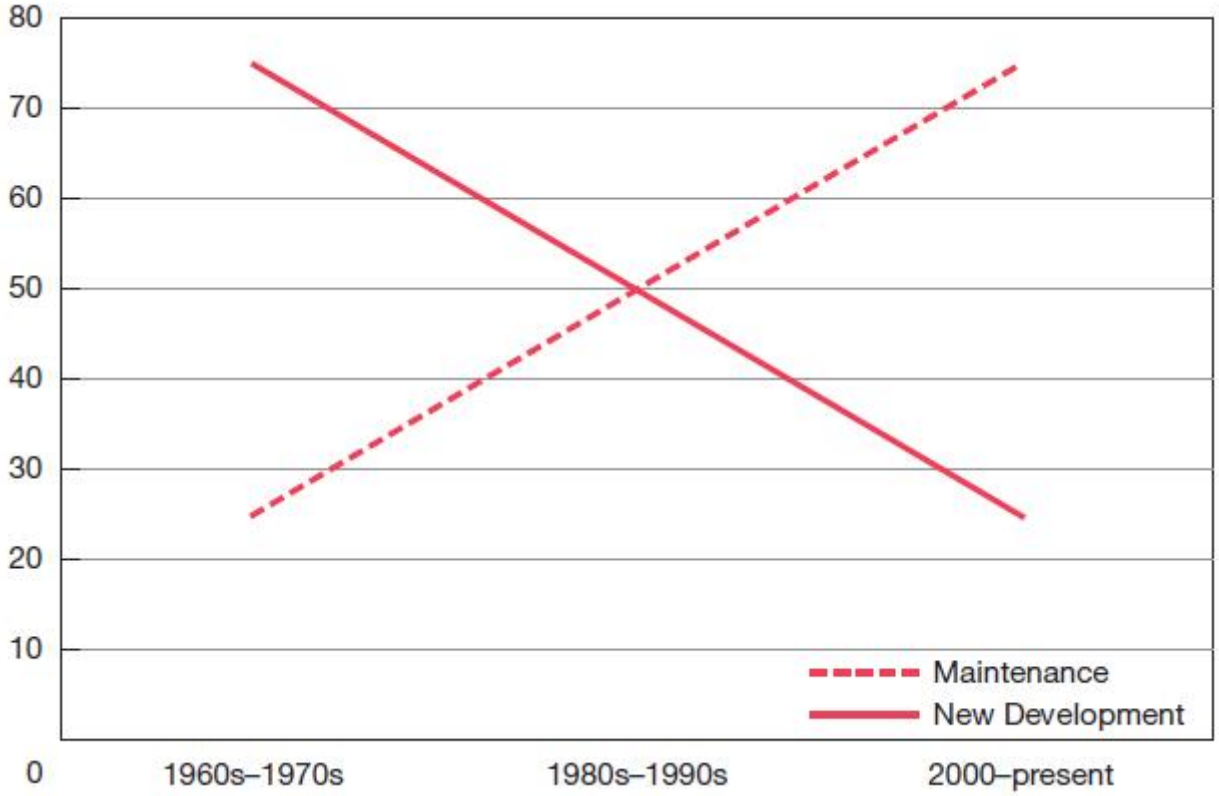

FIGURE 14-5

New development versus maintenance as a percentage of the software budget over the years

(*Source:* Based on Pressman, 2005.)

Managing Maintenance

As maintenance activities consume more and more of the systems development budget, maintenance management has become increasingly important. Today, far more programmers worldwide are working on maintenance than on new development. In other words,maintenance is the largest segment of programming personnel, and this implies **the need for careful management.**

Managing Maintenance Personnel:

One concern with managing maintenance relates to personnel management. Historically, many organizations had a "maintenance group" that was separate from the "development group." With the increased number of maintenance personnel, the development of formal methodologies and tools, changing organizational forms, end-user computing, and the widespread use of very high-level languages for the development of some systems, organizations have rethought the organization of maintenance and development personnel.

TABLE 14-2 Advantages and Disadvantages of Different Maintenance Organizational Structures

Type	Advantages	Disadvantages
Separate	Formal transfer of systems between groups improves the system and documentation quality	All things cannot be documented, so the maintenance group may not know critical information about the system
Combined	Maintenance group knows or has access to all assumptions and decisions behind the system's original design	Documentation and testing thoroughness may suffer due to a lack of a formal transfer of responsibility
Functional	Personnel have a vested interest in effectively maintaining the system and have a better understanding of functional requirements	Personnel may have limited job mobility and lack access to adequate human and technical resources

Measuring Maintenance Effectiveness: A **second management issue** is the measurement of maintenance effectiveness. As with the effective management of personnel, the measurement of maintenance activities is fundamental to understanding the quality of the development and maintenance efforts. To measure effectiveness, you must measure the following factors:

- Number of failures
- Time between each failure
- Type of failure

Measuring the number of and time between failures will provide you with the basis to calculate a widely used measure of system quality. This metric is referred to as the **mean time between failures (MTBF).** As its name implies, the MTBF metric shows the average length of time between the identification of one system failure and the next. Over time, you should expect the MTBF value to rapidly increase after a few months of use (and corrective maintenance) of the

system (see Figure 14-7 for an example of the relationship between MTBF and age of a system). If the MTBF does not rapidly increase over time, it will be a signal to management that major problems exist within the system that are not being adequately resolved through the maintenance process.

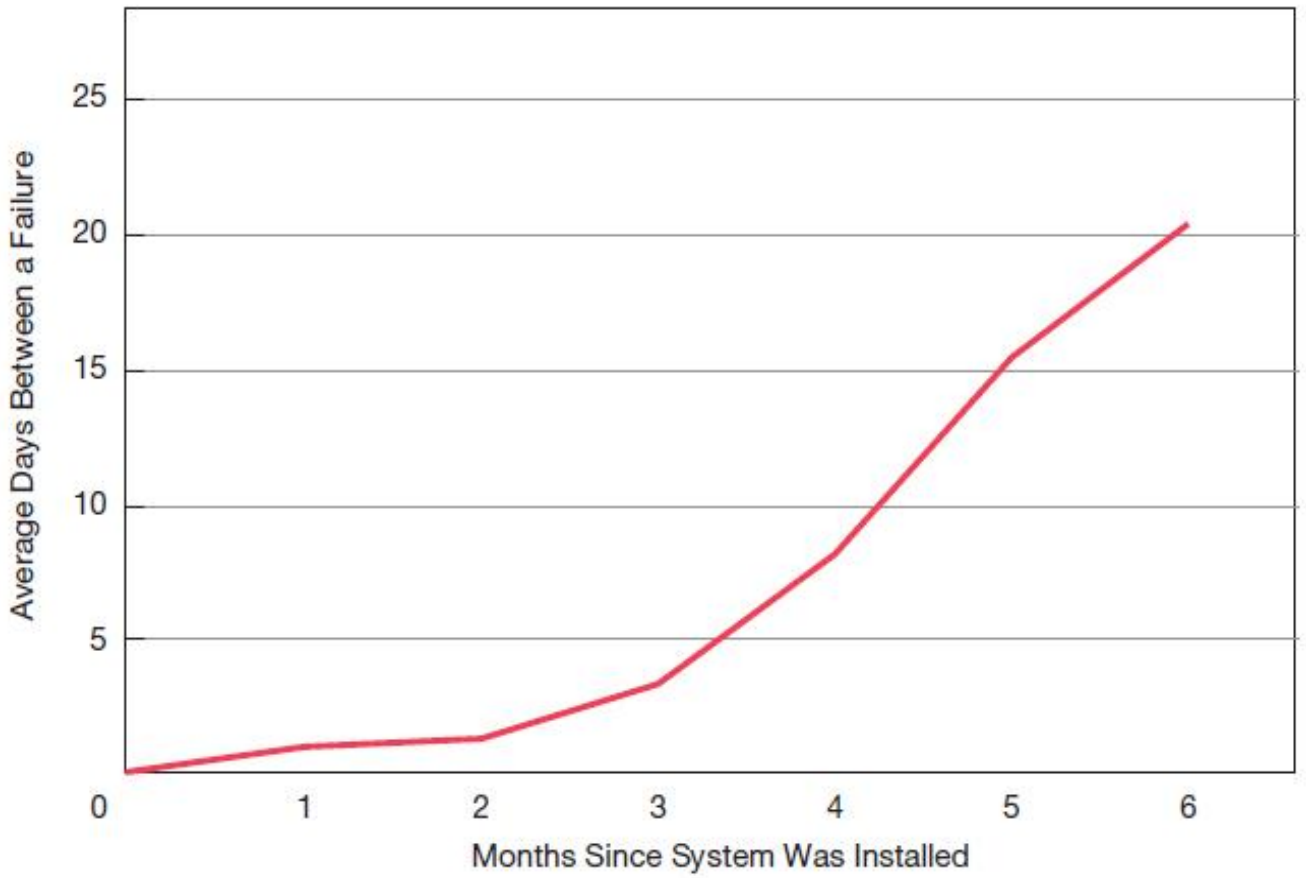

FIGURE 14-7
How the mean time between failures should change over time

Controlling Maintenance Requests

Another maintenance activity is managing maintenance requests. There are various types of maintenance requests—some correct minor or severe defects in the systems, whereas others improve or extend system functionality. From a management perspective, a key issue is deciding **which requests to perform and which to ignore**. Because some requests will be more critical than others, some method of

prioritizing requests must be determined. **Figure 14-8 shows a flowchart** that suggests one possible method you could apply for dealing with maintenance change requests. First, you must determine the type of request. If, for example, the request is an error—that is, a corrective maintenance request—then the flowchart shows that the request is placed in the queue of tasks waiting to be performed on the system. For an error of high severity, repairs to remove it must be made as soon as possible. If, however, the error is considered "**nonsevere**," then the change request can be categorized and prioritized based upon its type and relative importance.

If the change request is not an error, then you must determine whether the request is to adapt the system to technology changes and/or business requirements, perfect its operation in some way, or enhance the system so that it will provide new business functionality. Enhancement-type requests must first be evaluated to see whether they are aligned with future business and information systems' plans. If not, the request will be rejected and the requester will be informed. If the enhancement appears to be aligned with business and information systems plans, it can then be prioritized and placed into the queue of future tasks. Part of the prioritization process includes estimating the scope and feasibility of the change.

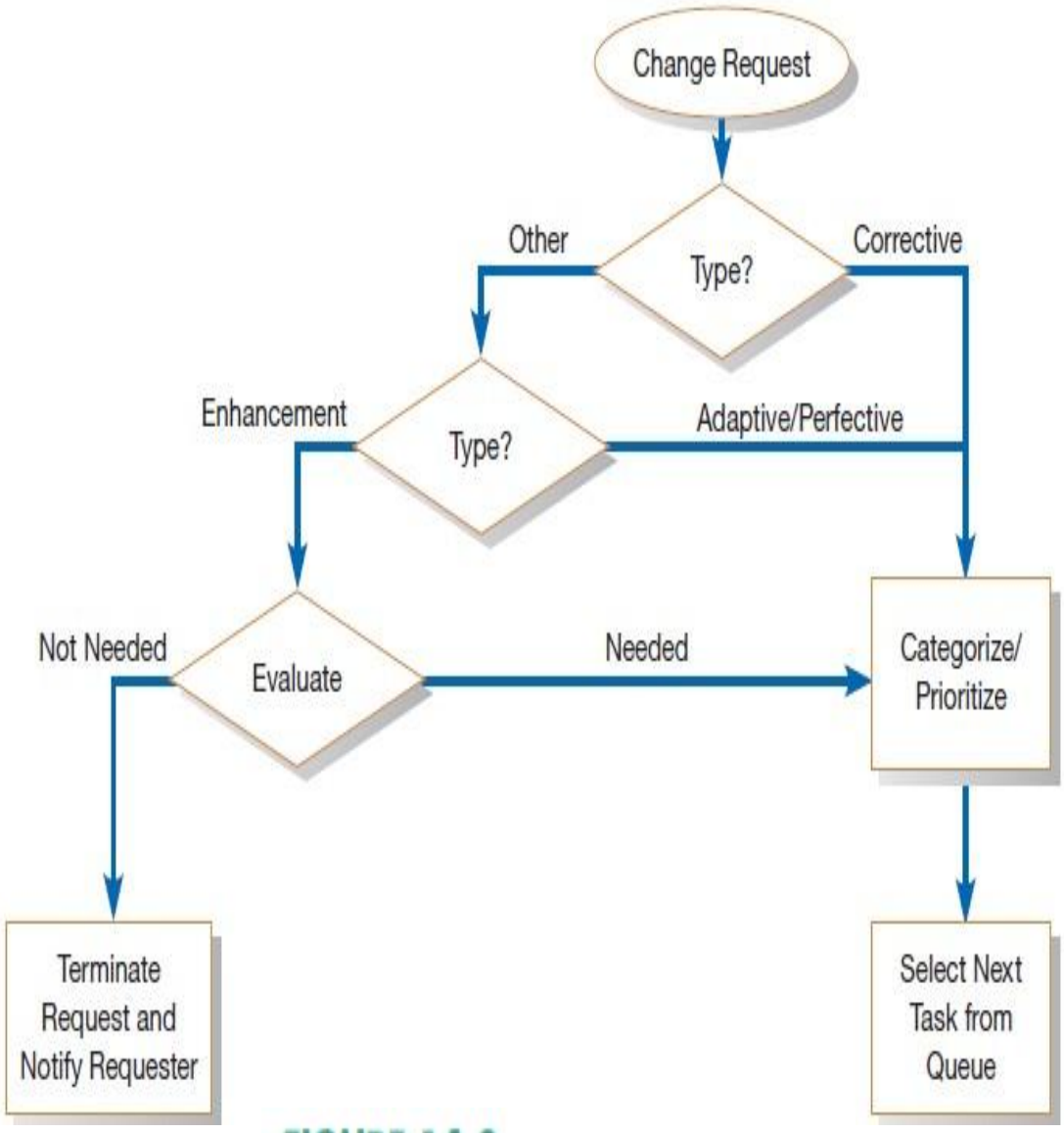

FIGURE 14-8

How to prioritize maintenance requests

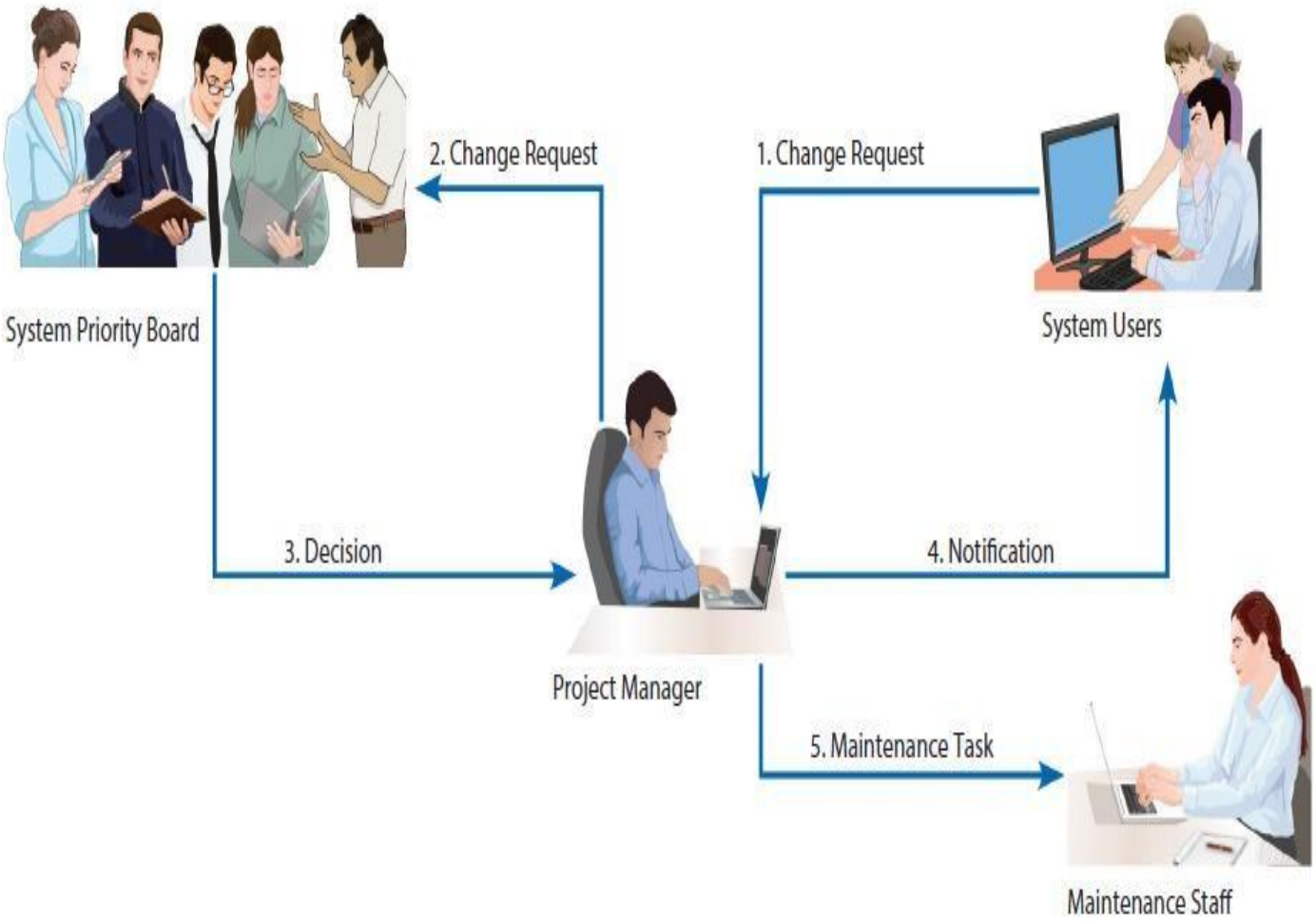

FIGURE 14-9
How a maintenance request moves through an organization

Configuration Management

A final aspect of managing maintenance is configuration management, which is the process of ensuring that **only authorized changes are made to a system**. Once a system has been implemented and installed, the programming code used to construct the system represents the **baseline modules of the** system. The baseline modules are the software modules for the most recent version of a system whereby each module has passed the organization's quality assurance process and documentation standards. A **system librarian controls the checking out** and checking in of the baseline source code modules. If maintenance personnel are

assigned to make changes to a system, they must first check out a copy of the baseline system **modules—no one is allowed to directly modify the baseline modules**. Only those modules that have been tested and have gone through a formal check-in process can reside in the library. Before any code can be checked back in to the librarian, the code must pass the quality control procedures, testing, and documentation standards established by the organization.