BFB-IS-10: Systems Development Standards

Responsible Officer:	VP - Chief Information Officer
Responsible Office:	IT - Information Technology Services
Issuance Date:	5/18/2001
Effective Date:	5/18/2001
Scope:	[Scope]

Contact:	Stephen Lau
Email:	Stephen.Lau@ucop.edu
Phone #:	(510) 987-0409

I. POLICY SUMMARY

This manual describes standards for developing (or purchasing and installing) and maintaining computer applications for administrative purposes at the ten campuses and at the Office of the President (OP) of the University of California. The degree to which the responsibility for development, implementation and maintenance of systems is centralized in a single administrative computing department versus decentralized and handled by the functional offices varies from campus to campus, with each site having some degree of decentralization. While these standards will sometimes refer to the "administrative computing department", these standards apply to any department or any vendor engaged by the campus (or OP) that undertakes development, installation or maintenance of administrative applications. The determination for when these standards apply depends on the nature of the application, not on who is responsible for the development.

II. DEFINITIONS

III. POLICY TEXT

TABLE OF CONTENTS

A. INTRODUCTION TO SYSTEMS DEVELOPMENT

1. WHEN DO THESE STANDARDS APPLY

This manual describes standards for developing (or purchasing and installing) and maintaining computer applications for administrative purposes at the ten campuses and at the Office of the President (OP) of the University of California. The degree to which the responsibility for development, implementation and maintenance of systems is centralized in a single administrative computing department versus decentralized and handled by the functional offices varies from campus to campus, with each site having some degree of decentralization. While these standards will sometimes refer to the "administrative computing department", these standards apply to any department or any vendor engaged by the campus (or OP) that undertakes development, installation or maintenance of administrative applications. The determination for when these standards apply depends on the nature of the application, not on who is responsible for the development.

There are two main characteristics of applications which must be considered when determining whether or not these standards apply:

1. The size and complexity of the application; and

2. Some measurement of how essential the system is to the operation of the campus or OP, or the University as a whole, and therefore how much risk is associated with whether or not the development efforts are successful.

A system should be considered high risk if a failure of the system to function correctly and on schedule could result in a major failure by the University to perform essential functions, a significant loss of funds to the University, or a significant liability or legal exposure to the University.

For the purposes of determining whether or not these standards apply to the development (or installation) of a system, a small, less complex system is one that would take less than 1 year of effort (staff time) to develop and implement, or less than $150,000 in cost.

For the purposes of determining whether or not these standards apply to the maintenance of a system, a small, less complex maintenance project is one that would take less than 1 month of effort (staff time) to develop and implement, or less than $15,000 in cost.

The following matrices outline when these development and maintenance standards must be followed (required), should be followed (recommended), or may be followed (optional), depending on the relative size and/or complexity of the system, and on the risk factor related to the application:

	Low Risk	**High Risk**
Small/ Simple	Optional	Required
Large/ Complex	Recommended	Required

	Low Risk	**High Risk**
Small/ Simple	Section 4.2 applies	Section 4.3 applies
Large/ Complex	Section 4.3 & 4.4 apply	Section 4.3 & 4.4 apply

2. SIMPLIFICATION OF BUSINESS PROCESSES

The development of new systems or major enhancements to existing systems is often the result of significant changes made to the business processes supported by the systems. In the late 20th century and into the beginning of the 21st century, organizations are being faced with the requirement to meet higher quality standards, but often with fewer resources. Basically, they need to do more with less and do it better. This trend affects the administration of institutions of higher education as well as other types of organizations. This trend often forces administrative units to find simpler and more efficient methods for performing their work.

Usually the effort to simplify the business processes themselves precedes any major systems development effort. It is appropriate that the business processes be reviewed before systems work begins, to avoid the unfortunate mistake of simply automating existing cumbersome processes. Ideally the efforts to simplify business processes will be done by the functional office in conjunction with technical personnel, so that current technology can be considered as the business processes are reviewed. In some cases, particularly when a vendor package is selected for implementation, the simplification of business processes may occur during the systems development or installation process.

This standards manual provides standards and guidelines for the systems development process. This manual does not provide standards for reviewing the functional offices' business processes, but it is recognized that this review will frequently precede or may coincide with the systems development process.

3. SYSTEMS DEVELOPMENT TRACKS

a. Three Development Tracks

This manual, Systems Development Standards, describes the phases of a systems development project. The exact methods employed for systems development will vary depending on the specific project. Although every systems project is unique, there are three key characteristics which will influence the overall approach chosen for systems development:

1. The overall size and complexity of the application;
2. The technology to be used for developing the application; and
3. Whether the system will be a purchased package, or custom developed.

Clearly, a smaller, less complex application will most likely not require as many separate phases as a larger, more complex application. In addition, some of the newer development technologies such as GUI-based development tools allow a different approach than a more traditional technology such as 3GL languages. Finally, if the system will be a purchased vendor package, then the phases of the development process will differ from those for a system being developed from scratch.

Three separate development approaches or “tracks” have been identified:

- the first, involving prototyping, is appropriate for custom development of smaller systems or systems that use newer technology such as GUI-based development tools;
- the second, involving a more traditional life cycle approach, is more suited to custom development of larger systems using 3GL languages; and
- the third is tailored for the purchase and implementation of vendor packages.

Some phases of project development apply to two or all three approaches, whereas others are unique to one approach or another. The phases in these three approaches, or tracks, are listed below:

Track 1: Prototyping	Track 2: Traditional Life Cycle	Track 3: Vendor Package Purchase
Project Proposal	Project Proposal	Project Proposal Request for Information
System Definition	Requirements Definition	Requirements Definition Request for Proposal
Feasibility Study	Feasibility Study	Feasibility Study Vendor Contract and Installation Plan
Prototyping	General Design Detail Design Programming and Unit Testing	
System Testing	System Testing	System Testing
Implementation	Implementation	Implementation
Final Documentation	Final Documentation	Final Documentation
Post-Implementation Review	Post-Implementation Review	Post-Implementation Review

There are several important points to be noted regarding the phases listed above for each of the three approaches to system development:

1. The development of any one system may not fall neatly into one of the three categories listed above.

2. Depending on the individual project, not every phase may be needed. For example, when developing a custom-built application where there is general agreement as to the overall approach, there most likely will be no need for a Feasibility Study.

3. Although the phases listed above are listed in the general sequence in which they occur, it is important to note that phases will overlap. For example, Final Documentation must be completed at about the same time as implementation of the system. To complete the documentation by the time the system is implemented (or, shortly thereafter), the documentation must be started much earlier in the development process.

The above phases for each of the three tracks are described in Chapter 2, Phases of Systems Development.

b. How to Choose the Right Track

Prototyping Track. As noted above, the prototyping methodology is best suited to developing systems (or portions of systems) where:

- the system is small to medium in size;
- the technology to be used for development (e.g., GUI-based development tools) is not amenable to the traditional approach to systems development normally used with 3GL systems development; and/or
- the system involves heavy interaction with the functional office users (e.g., online processing).

Since prototyping is based upon a high degree of involvement by the functional office in the specification of system functions, it is most useful for developing system functions with which end users will interact directly, such as online systems or online portions of systems. Also, this method works best when a prototype may be developed in a relatively short period, and the time between iterations of the working model is not overly long. For this reason, the method is best used for small to medium scale systems, or small to medium scale portions of larger-scale systems. This track is also appropriate for systems or portions of systems where a Joint Application Development (JAD) or Rapid Application Development (RAD) methodology will be employed.

Traditional Life Cycle Track. The Traditional Track relies more heavily than the Prototyping Track on formal written specifications, particularly for the design phases specifying the programming to be performed. This Track is appropriate for systems where:

- the system is large, and multiple application developers will be working simultaneously on the project, particularly during the programming and unit testing phase;
- the system will use 3GL technology, which requires application developers to explicitly code more than may be necessary with 4GL or GUI-based development tools;
- the system has little direct interaction with the users (e.g., much of the processing is batch); and/or
- it is difficult for the functional office(s) to confirm the correctness of the programming directly due to the complexity of the logic and/or the nature of the processing, making written specifications vital for functional office review (e.g., in the case of a complex calculation).

It is particularly important with a large system development project involving many people in design, programming and testing efforts to have written design documents, to help prevent misunderstandings among the application developers.

Vendor Package Purchase and Installation Track. Purchasing and installing a vendor package rather than developing a system from scratch can save significant time, and often significant resources, in terms of cost. There are many issues to consider when deciding whether to purchase a product or develop a custom system, but the following guidelines help determine if purchase of a vendor package should be considered:

- Whether packages are available on the market that can satisfy at least 80% of the functional requirements of the system.

  While it may not be possible to find a package that exactly matches the specific functional requirements, if less than 80% of the requirements are satisfied then purchase of a package is likely to be more costly than custom development.

- If packages available on the market that meet the functional needs adequately are in production, not planned for future release.

- If packages are available which are compatible with the technical environment currently available at the campus (or OP) (or planned for installation in the immediate future).

- Whether the overall cost of purchasing and installing (and maintaining) a package will be at least no more than (and preferably less than) the cost of custom development and maintenance.

- Whether or not the vendors for the packages available have a proven track record of installation and support and can show evidence of financial stability (this is particularly critical for large systems).

Selection of One or More Tracks. Most systems will use one track as the primary track, but some systems may use a combination of two, or possibly even all three, tracks. One common instance where more than one track might be used involves the installation of a large vendor package. Most likely, interface processes will need to be heavily revised or rewritten when a vendor package is installed. In addition, some customization of the product may be planned before installation. The development of heavily modified or rewritten interface processes plus any customized modifications may be treated as separate, custom-developed projects, which would follow the phases listed under the Prototyping Track or the Traditional Life Cycle Track, although the larger, overall project would follow the Vendor Package Purchase and Installation Track.

The Project Leader is responsible for making a recommendation to his/her manager regarding which of the development tracks or combination of tracks is the best approach for a project. The manager is responsible for final determination of which track or tracks will be used for a project.

ROLES AND RESPONSIBILITIES

a. Functional Office and Administrative Computing Department Responsibilities
One of the first tasks of any system development project is to identify the participating departments, and clarify responsibilities. The two obvious participants are the functional office identified as the key user of the application (assuming there is an identifiable primary user), and the administrative computing department, if the campus or OP administrative computing department is going to develop or install the system. The possible split of responsibilities between the functional office (or offices) and the administrative computing department varies along a continuum from the administrative computing department taking primary responsibility throughout the project development effort, to the functional office taking primary responsibility with the administrative

computing department serving in the role of a consultant. While many different models are viable, it is critical that the split of responsibilities is clearly understood from the beginning of the project. Assignment of responsibility for systems development must be accompanied by a corresponding commitment of staff time from each participating office.

As responsibilities are being defined, it is strongly recommended that a single entity (e.g., one manager or executive in the functional office, or possibly a task group or steering committee), is assigned clear ownership or sponsorship of the project. This person or group will be responsible for making key decisions, such as determining whether to increase the scope or budget of the project during development if changes arise, or to stay within the originally-allocated budget and schedule and forego the proposed changes.

It is also recommended that procedures for handling changes that may arise during the systems development process be agreed upon as part of the determination of responsibilities.

The level of formality by which responsibilities and the method for handling changes are delineated may vary widely, in some cases consisting of an informal agreement while in other cases the decisions may be codified in a formal “governance contract” that all parties adhere to throughout the development process.

In the case where the administrative computing department will lead the project through most or all of the development phases, it remains critical that the functional office is heavily involved, particularly during the earlier stages of the process (i.e., Project Proposal, System Definition or Requirements Definition, and Requests for Information and Proposals for Package purchases), for the project to be successful. It is also critical that senior-level management within the functional office(s) are involved at pivotal points in the project development process to ensure that the objectives of the project have been accurately identified and are being fulfilled.

The general question of who will be responsible for the on-going support for a system should be addressed along with the determination of responsibilities during the development process. Responsibility for on-going support of required hardware (including servers, workstations, printers, and networking) and maintenance of the software should be addressed.

Clarifying the roles and responsibilities for a project is particularly important if the project may involve the purchase and installation of a vendor package. At the point when a vendor is selected, there is yet another group to coordinate with, making project management even more complex. It is also critical that the role and responsibilities of the vendor, both during system installation and for on-going support, are clearly delineated in the contract (see section 2.8 Vendor Contract and Installation Plan).

b. Steering Committee and Project Team

The organizational structure for handling projects can vary from campus to campus and from project to project. As noted earlier, ownership of a project may be assigned to a Steering Committee. For larger projects, it is often effective to have a high-level steering committee composed of senior-level management from functional offices, the administrative computing department and, if appropriate (e.g., for financial systems), internal audit. The steering committee provides overall strategic direction and decisions for the project, including establishing the business case for system development and the planning principles, defining the overall scope of the project, communicating with all stakeholders in the system, developing a decision framework, and providing management oversight throughout development and implementation. A Project Leader is appointed who will be responsible for the day-to-day management and oversight of the project. For larger projects, the Project Leader will lead a project team, made up of representatives from the functional offices and the administrative computing department. (This project team is in addition to the technical design and programming team assigned to the project within the administrative computing department.) The project team is responsible for performing much of the work during the analysis phase (Project Proposal, System Definition or Requirements Definition, and Request for Information, Request for Proposal, and Feasibility Study for vendor package solutions), and continued project management throughout the prototyping, design or installation phases and final implementation. In some cases, the project team may continue after implementation to assist with on-going maintenance issues.

c. Internal Audit

Internal Audit's involvement during high risk systems development is important. Internal Audit can provide advice and assistance in the identification and development of adequate system and process controls. Accordingly, Internal Audit must be notified of all systems development projects early in the development process (normally at the Project Proposal phase). Based upon the risk and scope of the project, Internal Audit will establish a plan with the Project Leader that details Internal Audit's involvement during each phase of the project.

Internal Audit may elect to participate in the development process for new systems or major enhancements to existing systems in the case where the project is not considered high risk. The level of involvement for non-high risk applications will be determined by Internal Audit.

d. Office of the President Review

Campus Responsibility. For major, campus-wide applications the Project Plan highlighting the major milestones established for the project must be forwarded to the Associate Vice President for Information Resources and Communications (IR&C) at the Office of the President, for information and review. Upon reaching major milestones in the project, an updated Project Plan showing progress measured against major

milestones should be forwarded to the Associate Vice President for IR&C. Office of the President review is only necessary for major campus-wide applications, such as a new, campus-wide financial system. The review by Office of the President will help assure the uninterrupted flow of corporate data from campus systems, and will help assure continued smooth University operations.

Implementation Procedures. Applications meeting the criterion of "major, campus-wide" are those which supply information to the corporate financial (e.g., campus budget, general ledger/financial, chart of accounts, etc.), the corporate contracts and grants, the corporate student or the corporate personnel (except base Payroll/Personnel) systems.

The Project Plan should be presented in a form which adheres to the steps in these Systems Development Standards and which indicates major milestones. Preferably the plan should be the same plan used at the campus level. Additionally, the Associate Vice President should be invited to participate in major campus project milestone reviews. It should be noted that the Associate Vice President will coordinate reviews of major campus applications with UCOP functional officers (in Business and Finance, Student Services, etc., as appropriate) and with the University Auditor.

e. Hardware and Software Acquisition by Functional Offices

Some sites may have specific policies regarding acquisition of hardware and software. For example, at Office of the President, the "Policies and Procedures for Acquisition by Systemwide Administration of Computers, Related Equipment, and Software" (dated January 18, 1981) must be followed and appropriate approvals obtained.

1. PROJECT PLANNING AND MANAGEMENT

Project management is a task which spans across all phases of systems development. The Project Plan, described below, is a key tool for effective project management. A Project Plan, used as part of the management of a project, can reduce the likelihood of major, unexpected schedule or cost overruns.

a. Project Plan

The Project Plan is written very early in the development process. The Plan includes a list of tasks necessary to complete the project, an estimate of elapsed time for each task, an estimate of administrative computing department staff time required to complete each task (and possibly of functional office staff time as well), an indication of who (or which department(s)) is responsible for each task, and information on the sequence and dependencies of the tasks. For all but the smallest of projects, tasks will be put into logical groups of related tasks, and major milestones will be identified as the completion of a group of related tasks. The original version of the Project Plan may be quite general, and can be refined with more detail as the project progresses. For the Plan to continue to serve as a useful project management tool, it must be updated

regularly throughout the development phases to reflect any changes or refinements. The format of the Plan is not critical, as long as the required information (basically what, when, and who) is included.

Each project that requires more than 1 year of effort (staff time) to develop and implement or more than $100,000 in cost must have a Project Plan, unless an exception is granted by the Director of the administrative computing department or equivalent.

b. Staff Time Estimates

As noted above, the Project Plan should include estimates for the time needed for the administrative computing department staff to complete the tasks. Estimates for staff time should be developed by the completion of the System Definition phase (Prototyping Track) or Requirements Definition phase (Traditional or Purchase Track), although these estimates may be adjusted as needed during the Design, Prototyping, or Installation phases. If on-going support for the system after completion of the project is expected to be significant, then estimates of the required staff time for on-going support should be made.

Estimates for functional office staff time may also be included in the Project Plan. It is important that adequate resources in terms of functional office staff time are committed to the project. Time required for functional office staff training should also be considered.

c. Project Scope

The overall scope of the project is established early in the development cycle. As analysis progresses, the scope may legitimately change over time, usually increasing. To some extent, changes are inevitable in the development of a system. However, the project becomes unmanageable if the number or magnitude of the changes are too great. It is the Project Leader's responsibility, in conjunction with staff in the functional office, to agree early in the process on the overall scope, and to control the changes to a reasonable level. If major issues arise regarding the scope of the project, then senior level management may need to become involved to determine whether the scope should be expanded significantly (with a corresponding expansion in schedule and resources), or remain at the original level.

When the project involves purchase and installation of a vendor package, the scope issue can become more complicated. It will be up to the functional office representatives and the administrative computing staff to review packages in light of the agreed-upon scope of the project, or to explicitly consider (with appropriate high-level management review) expansion of the scope. The common problem of the scope of a system gradually increasing as the project progresses (often referred to as "scope creep") can become particularly expensive when a vendor is involved. This issue should be recognized when developing the contract with the vendor, so that efforts may be made both to control "scope creep", and also to control the costs to the campus or OP of changes, in the case when a change (after the contract is finalized) is determined to be necessary.

d. Project Status Reporting

As part of the management of projects within the administrative computing department, regular status reports must be provided to the Director by the Project Leader for each significant development project (i.e., any project where a Project Plan is required). While the detail of reporting will vary from project to project, and will vary within a single project over time, the following should be addressed:

- what has been completed since the last report;
- progress on meeting major milestones, plus any changes in estimated completion dates for future milestones;
- any issues that have arisen, such as difficulties in reaching agreement among the users of the system on a design issue, or a recently discovered interface file problem; and
- review of the remaining tasks and the next milestone, together with the estimates for staff time required.

The format for presenting the status of a project is up to the Project Leader.

For major, campus-wide application development efforts, the Associate Vice President for Information Resources and Communications must be notified at key points (major milestones) during the project. This notification must include information on progress on the project, measuring progress against the milestones established in the Project Plan. This enables the Office of the President to track overall progress of major development efforts at the campuses.

e. Employee Time Reporting

The administrative computing department may require staff to report their time by project to their manager. This time reporting by project provides information which may then be compared to the staff estimates made for each phase of a development project. This allows the Project Leader to monitor the progress of a project, and provides information that may prove extremely useful during the Post-Implementation Review for a project.

The format for employee time reporting is up to the manager.

B. PHASES OF SYSTEMS DEVELOPMENT

1. PROJECT PROPOSAL

a. Purpose

The Project Proposal document is a clear statement of the problem to be addressed, and a brief description of the proposed solution. The document may be used to formally present notice that a development need exists. If there is a campus (or OP) procedure for review of projects for the purpose of setting priorities, then the Project Proposal may be used for consideration of this project during this review.

If further work on the project is undertaken, the Project Proposal should serve as a base from which a full Requirements Definition (Traditional or Purchase Track) or System Definition (Prototyping Track) is developed. It may also be used as the basis for a Request for Information, if this phase is undertaken (Purchase Track). For more information regarding the different development tracks, see section 1.3 Systems Development Tracks.

b. Project Proposal Document

The Project Proposal document may be quite general. The document is usually prepared by the functional office. The following sections are ideally included in a proposal document:

- Problem
  A brief discussion of the reason for the proposed new system or system modification should be given. Among questions to be addressed are: what benefits are to be derived? Why should the system be developed or modified?

- Description
  A brief description of the proposed system or proposed modification to an existing system should be included in the Proposal. If the proposed system replaces an existing system, this should be noted.

- Users
  The users of the new system or the system modification should be identified.

- Special Considerations
  Any special requirements or constraints should be stated. For example, if Federal regulations or University policy requires implementation by a certain date, this should be stated.

- Funding

The intended source of funding for both one-time development and acquisition costs and any on-going costs (for staff to maintain the system, and maintenance and/or upgrades of software and hardware) should be indicated.

- Interfaces
  If the system includes any external interfaces, such as a campus-to-OP interface, Electronic Transfer of Funds (EFT) to banks, or an Electronic Data Interchange (EDI) interface with vendors, then this should be explained.

c. Internal Audit Notification

Internal Audit should be notified, either by forwarding a copy of the Project Proposal or by notification via email or other means, that a project is being proposed.

d. Review and Approval

If the Proposal was not written by the functional office, then the functional office must approve the Project Proposal document. Approval of the Project Proposal document by the functional office does not imply a commitment to proceed with the remaining phases of development for the project.

2. REQUEST FOR INFORMATION (PURCHASE TRACK ONLY)

a. Purpose

The purpose of issuing a Request for Information (RFI) early in the development process is to gather information regarding currently available technology and products. This phase is only applicable to projects where purchase of a vendor product is being considered (i.e., this phase is part of the Purchase Track; for more information regarding development tracks, see section 1.3 Systems Development Tracks.) Even for those projects where purchase of a vendor product is likely, this phase is not required, but may prove useful. Specifically, gathering vendor information prior to performing the detailed analysis of the functional requirements (Requirements Definition phase) may help to:

- establish the overall scope of the project, depending in part on the scope of the products available;
- identify whether or not a vendor package solution is feasible;
- clarify whether or not the business processes must be reviewed and possibly revamped, and if so, whether it is expected that business processes must be tailored to fit a vendor package (once a package is selected), or that processes will be modified, which could result in a package solution no longer fitting the business practices implemented;
- allow participants to learn about newer technologies available, that may, if utilized, affect business processes; and

- allow participants to learn about the approximate cost of newer technologies.

Note: If it is determined as the result of reviewing the responses to the Request for Information that the business processes must be reviewed and possibly significantly modified, then this review should be undertaken before the Requirements Definition phase (see section 1.2 Simplification of Business Processes).

Note: A Request for Information document may also be issued if a vendor may be engaged to perform a feasibility study or custom development of an application. This section may be used for guidelines in these cases, although the exact content of the RFI document will differ somewhat.

Some information regarding vendor products may be gathered without a formal Request for Information, by researching available products, and contacting the vendors for standard information and brochures. For some projects, this informal approach to gathering information will suffice, and the more formal Request for Information process may be skipped. Other sources in addition to the vendors themselves may also be explored for information regarding available packages, including other UC campuses or laboratories and other universities.

b. Request for Information Document

The Request for Information (RFI) document may be based largely on the Project Proposal document. At this point in the process, and for the purposes of gathering information from vendors, it is not expected that a detailed analysis of the functional requirements have been performed, and the level of detail provided in the Project Proposal should be adequate.

In addition to the information contained in the Project Proposal, the following should also be included in an RFI:

- An overview of the organization, and the context within which this system will be implemented (often an internal Project Proposal document assumes an understanding of the organization, and would not include the type of overview that an external vendor might need). This should also identify the functional office or offices that will be the primary users of the system.
- An overview of the technical infrastructure within which the system must fit, if this has been determined. For example, if the system must run on specific hardware that is already installed, then this must be stated. However, unnecessary restrictions should be avoided at this stage.
- A description of all interfaces this system has with other applications.
- Any assumptions that have been made regarding the project. For example, if the intended new application will be an enhancement to an already existing system, where the assumption is that the existing system will remain in production more-or-less unchanged, then this should be explained.

- A clear iteration of the type of information desired in the responses to the RFI, regarding both the vendor product itself (the functionality, technical requirements, etc.), and also regarding the vendor and the support provided by the vendor. The RFI may also include whether or not vendors will be expected to provide a demonstration of their product.

c. Process for Issuing a Request for Information

A Request for Information (RFI) must be submitted to the Purchasing Office for distribution. The department requesting issuance of an RFI may provide Purchasing with a list of companies to which the RFI should be sent. The Purchasing Office may add more companies, and may also post the RFI publicly.
The deadlines for responses established in the RFI must allow vendors a reasonable time period in which to respond. While this will vary depending on the complexity of the request and the project itself, generally between 30 and 60 days is reasonable. The RFI should include the name and number of a person at UC responsible for answering any questions vendors may have regarding the RFI.

d. Review of Responses

The vendors' responses to the RFI must be reviewed by those responsible for the project's next phase: Requirements Definition. Normally there is a project team composed of the Project Leader (and possibly other staff from the administrative computing department) and representatives from the functional office(s). For high risk systems, internal audit may also be involved. If demonstrations of vendor products are given, then the same team should be present. While the review of RFI responses need not be as formal as a review of responses to a Request for Proposal (see sections 2.7 Request for Proposal and 2.8 Feasibility Study), the project team should agree in advance how the review should proceed. Once all responses have been reviewed, the team should discuss the results and how this may affect the next phases.

One risk of researching vendor products before performing a detailed analysis of the functional requirements is having a specific product in mind during the Requirements Definition phase. The participants must be aware of this risk, and avoid premature attachments to specific vendor products.

3. System Definition (Prototyping Track only)

The System Definition phase can be undertaken as soon as the Project Proposal is approved, or as the first phase of a project where there is no formal Project Proposal document. The System Definition phase is part of the Prototyping Track (vs. the Traditional or Purchase Track) for systems development. (See 1.3 Systems Development Tracks for further information regarding development tracks.)

a. Purpose

The System Definition phase is intended to define the proposed system in sufficient detail for prototyping to begin. The most important outcome from this phase is an understanding and agreement with the functional office or offices on the proposed systems solution. Other outputs from this phase include:

- The System Definition document, which must include a description of the functional needs that the system must satisfy, and a high-level design of the proposed solution; and
- Data element definitions.

The Systems Definition document and the data element definitions may be completed iteratively, overlapping with the Prototyping phase. Prototyping of the system may begin based on a less detailed version of the document and data elements, and the document, data element definitions, and prototyped system may be refined in parallel.

b. System Definition Document

The primary purpose of the System Definition document is to describe the system from both technical and functional perspectives so that both the functional office and the application developer can understand what it will look like and do. The document may consist mostly or entirely of text, or may incorporate alternative descriptive methods, including charts or output from modeling tools. The following sections may be included in this document:

Scope and Objectives. The scope of the proposed system and the overall objectives are described in this section:

- The section should include a statement of the general purpose of the proposed system or system modification. If a Project Proposal was written for the project, then the general purpose should coincide with what was stated in the Project Proposal. If the purpose or objectives of the project have changed since the Project Proposal, then this should be noted, and the revised purpose and objectives stated.

- Every functional office which will use the new system should be identified, and their use of the system described (e.g., one office may have read-only access to data for reporting, whereas another office has responsibility for update).

Current System or Procedures. This section describes the system (if any) or procedures currently in place. This section may also highlight the differences between the current procedures and the proposed procedures.

System Architecture. A description of the system architecture should be included, i.e., standalone PC-based, client/server, web-based, etc. If remote access to data is required (e.g., from other UC sites), describe how this will be provided.

Database Design. Every database within the system should be described. In as much detail as is possible at this stage of development, the structure and data elements should be included. For relational databases, the tables and the data elements within each table should be listed, as well as the relationship between tables, and any required

referential integrity. Where useful, a picture of the database(s) may be included. Key fields should be identified, and indexed fields, both unique and non-unique, should be identified.

Any sequential files should be described. For sequential files, the data elements should be listed, the record format and length given, key fields identified (and whether or not the key fields are unique), and the sort sequence specified.

Sources of Data. Input sources for the proposed system should be fully described. The method for input should be identified, e.g., online data entry by the functional office, interfaces with existing systems, etc. For each input source, the functional office owning the data source (and responsible for the accuracy of the data) should be identified. Any verification of input data should be described (see also Item 8, System Control and Data Verification). The frequency of data input and the approximate volume of data should be identified.

Some possible input sources include:

- Online data entry

  The screens for online data entry should be described in general, and the processing for these screens described. (Details of the exact design of the screen may be left to the Prototyping phase.) Data edits to be performed online for each data element entered by the user should be described. Error conditions (data verification, and errors when updating the database) may be described.

- Interfaces TO the system

  Any interfaces TO the system should be described, including record layout (if available) and method of transmission of the data. If an interface file is used for a batch update process, the database update logic should be described, including data editing procedures and error handling.

  For interfaces from other UC sites or from external sources, any issues regarding the definition of the interface or testing of the interface should be described. Sufficient time for external sites to develop or modify the interface and provide test files should be incorporated into the Project Plan.

- Other Batch Data Inputs

  Any other batch update processes (other than for interfaces TO the system) should be described, including: the database update logic, data editing, and error handling. Examples of other batch data input include: data originally collected on hardcopy forms and subsequently entered as transactions; or data files created by one process within the application, and input to a subsequent process.

Processing Logic. The processing logic for the system should be described in general. The logic for updating the database, via either online data entry or batch update, should be described (see Sources of Data, above). Processing for outputs, whether reports, interface files FROM the system to other systems, or online inquiry, should be described (see Major Outputs, below). Any processing, not directly related to input or output of data, should also be described. For example, for a system that will perform a modeling function, the logic for the modeling, including what data is used from the database and what data the user may enter or modify, should be described. Other logic, such as calculations to be performed, or workflow logic required, should also be described.

Major Outputs. All major outputs from the system should be described. This includes the following:

- Reports (online or hardcopy)
  Any reports from the system should be described, in general. Whether or not they are available online or via batch processing should be included in the description.

- Interfaces FROM the system
  Any interface files required FROM the system to another system should be described. If the record layout is known (e.g., the record layout is determined by the other system), then this should be included. If the frequency of required interfaces is known, this should be included. The processing to produce this file should be described in general terms.

- Online Inquiry
  Any online inquiry to data within the system should be described, in general terms. The method for providing online inquiry, and the types of screens to be made available should be described.

- Ad-Hoc Inquiry and Reporting
  If the system will provide ad-hoc inquiry and reporting access for users, then the general method for providing this access should be described. For example, if PC-based ad-hoc reporting tools will be used, this should be noted. If a specific product is expected to be used, this should be noted.

System Control and Data Verification. This section should describe how the integrity of the data is maintained within the system. The methods for verification of data accuracy should be described. Any verification of data that will occur outside of programs should be described. For example, if the functional office is responsible for reviewing certain reports before a specific process can be performed, this needs to be explained. Another example is when input data (e.g., from another system) is assumed to be accurate.

Audit and Authorization Features. This section should identify any audit, control or security and authorization considerations required. Examples are:

- appropriate record retention schedules to be followed to meet any legal or other data retention requirements (see section 3.1 Data Retention for further information);
- audit trails to meet legal or audit requirements[1]; and
- levels of authorization required to satisfy privacy standards and/or the requirements of the functional office for security protection of the data. For example, this might include read-only access to the data for selected users, or access limited to certain data elements for selected users. Authorization of users will depend on the authentication of the userid used for access into the application. With respect to privacy policies, it is particularly important that any data elements relating to personal information (individually identifiable data) be identified in order for appropriate security measures to be addressed in the system.

Privacy Consideration. The privacy requirements which should be addressed in the development of each system are contained in section 3.2 Privacy. In development of the system definition, these privacy standards should be reviewed. Any system requirements necessary for ensuring adherence to privacy standards should be described fully in this section of the System Definition document.

Backup and Recovery. This section describes the backup and recovery requirements in general terms. For example, if up-to-the-minute recovery is required for an online system (vs., for example, recovery to the end of the prior day), this should be noted.

Hardware/Software. For any shared resources, such as shared servers or printers, the impact of the proposed system should be evaluated to ensure that adequate resources are available to handle the new system without adversely impacting existing systems. Any significant upgrades or new hardware resources required for the new system should be identified as soon as possible. The following types of resources should be considered:

- memory allocation requirements on any shared server;
- disk space usage on shared server machines;
- Client workstation requirements (e.g., for PC/Windows clients, minimum RAM required, disk space required, level of operating system software required, and application software required);
- printer requirements, including estimates of pages printed, and determination as to whether or not existing printers will provide sufficient speed for the output expected; graphics printing capabilities; etc.;

[1] One common method for providing an audit trail of online update activity to a database is to include the userid, date and time as fields in the database, and record this information whenever an online update occurs.

- networking requirements, including: workstation networking capabilities required (cabling and software), requirements for a LAN or access to a file server, etc.; and
- level of client application software and server application software[2].

Impact on Existing Functional Office Operations. If there will be major impacts on the existing operations of the functional office with the implementation of the proposed system, then these should be described in general terms. Some examples of the possible impact of a new or revised system include impact on personnel (elimination of jobs or creation of new tasks), or reallocation of resources, such as from a centralized model to a decentralized model.

Conversion. This section should outline any conversions required for the system. If this system replaces an existing system, then the general approach for converting the existing data to the new system should be described. If there are major one-time startup efforts (e.g., one-time data entry of historical data), then this should be described.

c. Data Elements

Draft data element definitions should be written during the System Definition phase. The data elements should be categorized by the major data entities identified (i.e., for relational databases, the table(s) in which each data element belongs should be identified). Although these data elements may be further refined during the Prototyping phase, the application developer cannot begin a useful version of a prototype until the data elements are reasonably well defined.

At a minimum, the data element definitions should include the following information:

- Data Element Name
- Type of data element (numeric or alphanumeric)
- Format of the data element (e.g., for numeric fields, the number of digits allowed, such as 3v2 (meaning a number of the format nnn.nn))
- General description or definition of the data element
- If the data element is a coded value, then the possible codes and the corresponding interpretation of each code
- Programming name (i.e., the name of the field in the database)

Additional information, such as data editing requirements and source of the data element, may also be included in the data element definitions.

[2] It is possible, especially if the development time is long, that application development software or possibly operating system software may need to be upgraded during development efforts. If needed, the time for upgrading and testing the underlying software used for developing the application must be taken into account in the project planning and schedule.

If there is an official review process for data elements by a data administration group at the campus or at OP, then this review process should be initiated during this phase. Such a review process may include a review for policy implications, and a review for consistency and standardization of data across administrative applications at the campus or OP. Issues of consistency and standardization are particularly important for applications where data will be shared by multiple departments, such as data warehouse applications. The data elements and any required review processes should be finalized by the end of the Prototyping phase.

d. Review and Approval

The Project Leader is responsible for determining when the System Definition document may be distributed. When distributed, copies of the System Definition document should be given to:

- the functional office(s) identified as users of the system
- those within the administrative computing department who should review the document
- Internal Audit (for high risk systems)
- others, for informational purposes

The functional office should agree that the System Definition document accurately reflects their functional requirements for the new system or system modification, and accurately describes the agreed-upon systems solution. If the administrative computing department took the lead on producing the document, then a formal acceptance from the functional office (or offices) may be needed to ensure their agreement.

For some projects, the Project Leader may elect to require review and approval of the System Definition document from a working group of end-users in offices outside of the functional office(s) comprising the primary users of the system. This may be appropriate, for example, in the case of a campus-wide financial system that will impact offices outside of the Accounting Office, which is perhaps the project sponsor or major user.

The System Definition document is also given to Internal Audit for review for high risk applications. (Copies of the System Definition documents for applications not considered high risk are also available to Internal Audit upon request.) Audit's review of the System Definition document should provide feedback regarding whether or not necessary audit requirements have been identified (this is not to imply that Internal Audit is responsible for certifying the adequacy of the design and controls). In addition, Internal Audit should identify the degree of involvement desired by their office for the remaining phases of development.

4. PROTOTYPING (PROTOTYPING TRACK ONLY)

a. Purpose

Prototyping is a method of designing applications through iterative refinement of an actual working model of the system, followed by the acceptance of the actual working model of the system by the functional office. It involves developing a working model of a system (or a portion of a system), demonstrating the model to the prospective users, and modifying the model based on the users' responses.

The Prototyping phase is part of the Prototyping Track for systems development. (See 1.3 Systems Development Tracks for further information regarding development tracks.)

The major outputs from the Prototyping phase include:

- a working version of the system;
- specifications for interfaces with other UC sites or an external organization, if applicable; and
- updated Data Element definitions.

b. When to Use Prototyping

Prototyping can be applied to almost any system. However, since it is based upon a high degree of involvement by the functional office in the specification of system functions, it is most useful for developing system functions with which end users will interact directly, such as online systems or online portions of systems. Also, this method works best when a prototype may be developed in a relatively short period, and the time between iterations of the working model is not overly long. For this reason, the method is best used for small to medium scale systems, or small to medium scale portions of larger-scale systems.

The following system functions, if applicable, may be difficult to address via prototyping, and the Project Leader may choose to document these functions in a more traditional design document:

- Any interfaces to or from other systems; and
- Backup and recovery of data.

c. Programming

The programming standards and guidelines in effect for the campus or OP for the language being used for systems development must be followed -- refer to local standards manuals for further information.

d. Testing Each Prototype

The application developer is responsible for determining what testing is required for each iteration of a prototype, before presenting the prototype to the functional office for

review. A test plan, itemizing the test conditions to be exercised during testing, may be developed. The test plan can be an informal list for use by the application developer only, or may be a more formal list if functional offices will be involved in testing the prototype or test verification.

In general, testing should address the following:

- every function performed by the prototype should be tested;
- for event-driven portions of the system (e.g., online processing), every event should be tested;
- any error conditions should be tested; and
- "boundary" conditions should be tested (for example, if an online program is designed to handle up to ten entries on a screen, then a test of entering ten or eleven entries and a test of entering zero or one entry should be performed.)

Test results must be reviewed carefully. The review of test results may be completed by the application developer, by the Project Leader, by the functional office, or by a combination of these staff.

e. External Interface Specifications

If there is an interface between the new (or modified) system and either another UC site (a campus or OP) or an external organization (e.g., a health insurance carrier), then specifications for this interface must be written. If the interface is FROM the other site or organization, then the specifications must be sufficiently detailed to enable the other organization to develop the interface file processing. If the interface is TO the other site or organization, then the specifications must provide sufficient information for the other organization to effectively use the interface file in their application. At this point in the development process, any general issues regarding the interface should have been resolved.

Interface specifications must include the following:

Overview of the data required for each file specified, including a description of the data.

- Frequency of data submission (i.e., monthly, annually, etc.), effective dates of the submissions, and due dates (e.g., 5th working day of the month).
- A record layout for each file transmitted must be included.

- A list of data elements, including the format for each data element, must be included.

- Physical file characteristics, including the data set name, record length and record format
  OR
  The file name to be transmitted for files to be transferred via ftp, and the target system for the ftp transmission.

- The procedure for notifying the receiving site when the file has been submitted.

f. Data Elements
An updated list of data elements and data element definitions must be completed.

g. Review and Approval
When the actual working model of the system is completed, it must be presented to the functional office for sign off. For high risk applications, the completed model must also be made available for review to Internal Audit.

5. REQUIREMENTS DEFINITION (TRADITIONAL OR PURCHASE TRACKS)

The Requirements Definition phase can be undertaken as soon as the Project Proposal is approved, or as the first phase of a project where there is no formal Project Proposal document. The Requirements Definition phase is part of the Traditional Track or Purchase Track for systems development. (See 1.3 Systems Development Tracks for further information regarding development tracks.)

a. Purpose
The Requirements Definition phase of systems development must define the functional requirements of the new system or system modification. The requirements must provide sufficient details for subsequent phases: the design of the system, or selection of a vendor package.

Regardless of the split of responsibilities between the functional office(s), the administrative computing department, and any outside vendor for the Requirements phase (see Roles and Responsibilities in Chapter 1), the participation of the functional office(s) during this phase is critical. A system will not meet the needs of the functional office if staff from the functional office(s) are not heavily involved in defining the functional requirements at this stage of the project.

b. Requirements Definition Document
Simply stated, the requirements definition document describes what the system is expected to do. While technical issues are not necessarily addressed during the Requirements phase, if certain technical requirements are already known, they may be included in the Requirements Document. For example, if a new interface is required with an outside agency, and the specifications are determined by the outside agency, then the details of the new interface (e.g., record layout, and frequency of the interface) may be included. Normally, however, technical details should not be determined before the functional needs are clearly identified.

The Requirements Definition document should include the type of information described in the sections below. The document need not include the specific sections listed, as long as the information is included. Certain sections may not be needed for some projects (e.g. privacy may not be a consideration for a system if no individually-identifiable data is included). The document may consist mostly or entirely of text, or may incorporate alternative descriptive methods, including charts or output from modeling tools.

Scope and Objectives. The scope of the proposed system and the overall objectives are described in this section:

- The section should include a statement of the general purpose of the proposed system or system modification. If a Project Proposal was written for the project, then the general purpose should coincide with what was stated in the Project Proposal. If the purpose or objectives of the project have changed since the Project Proposal, then this should be noted, and the revised purpose and objectives stated.

- Every functional office which will use the new system must be identified, and their use of the system described (e.g., one office may have read-only access to data for reporting, whereas another office has responsibility for update).

Current System or Procedures. This section describes the system (if any) or procedures currently in place. This section may also highlight the differences between the current procedures and the proposed procedures.

Sources of Data. Input sources for the proposed system should be fully described. The method for input must be identified, e.g., online data entry by the functional office, interfaces with existing systems, etc. For each input source, the functional office owning the data source (and responsible for the accuracy of the data) must be identified. The method for verification of the accuracy and integrity of the data must be specified. It should be noted if data accuracy and integrity is assumed to be outside the scope of the system (e.g., where data from another system is assumed accurate).

The frequency of data input and the approximate volume of data should be identified.

For interfaces from other UC sites or from external sources, any issues regarding the definition of the interface or testing of the interface should be described. Sufficient time for external sites to develop or modify the interface and provide test files must be incorporated into the Project Plan.

Processing. The proposed processing to be performed by the system must be described in general. The logic for updating the database, via either online data entry or batch update, must be described. Processing for outputs, whether reports, interface files FROM the system to other systems, or online inquiry, should be described. Any processing, not directly related to input or output of data, should also be described. For

example, for a system that will perform a modeling function, the logic for the modeling, including what data is used from the database and what data the user may enter or modify, should be described. Other logic, such as calculations to be performed, or workflow logic required, should also be described.

Major Outputs. All major outputs of the system should be described. This includes reports, and required interfaces to other systems, either within UC or to outside agencies. If the format and frequency of required interfaces from the proposed system to other systems are known, then this information should be included.

For systems that are replacing existing systems, or for system modifications, any changes to existing reports or elimination of any existing reports should be noted.

Interfaces with Other Systems. Generally interfaces with other systems will be identified in earlier sections either as a source of data (where data comes from another system to the new system), or as output (where data is sent to another system). Any interfaces not already described in earlier sections must be identified.

Database. While it may be premature at this phase of the project to complete a detailed database design, the basic data entities required for the system must be identified, and the relationship between entities should be identified. For example, in personnel systems within the University, an employee is an entity, and appointment is often another entity. There is a one-to-many relationship between employee and appointment, since UC employees may hold multiple appointments.

Draft data elements (see section below) should be grouped by the data entities identified.

Audit and Authorization Features. This section should identify any audit, control or security and authorization considerations required. Examples are:

- appropriate record retention schedules to be followed to meet any legal or other data retention requirements;
- audit trails to meet legal or audit requirements[3]; and
- levels of authorization required to satisfy privacy standards and/or the requirements of the functional office for security protection of the data. For example, this might include read-only access to the data for selected users, or access limited to certain data elements for selected users. Authorization of users will depend on the authentication of the userid used for access into the application. With respect to privacy policies, it is particularly important that any data elements relating to personal information (individually identifiable data) be

[3] One common method for providing an audit trail of online update activity to a database is to include the userid, date and time as fields in the database, and record this information whenever an online update occurs.

identified early in the development cycle in order for appropriate security measures to be addressed during system design.

Privacy Consideration. The privacy requirements which should be addressed in the development of each system are contained in section 3.2 Privacy. In development of system requirements, these privacy standards should be reviewed. Any system requirements necessary for ensuring adherence to privacy standards should be described fully in this section of the Requirements Definition document.

Backup and Recovery. This section describes the backup and recovery requirements from a functional standpoint. For example, if up-to-the-minute recovery is required for an online system (vs., for example, recovery to the end of the prior day), this should be noted.

Hardware/Software. If there are hardware and/or software requirements for the system that are known at this point of the development process, these must be noted. For example, if there is a minimum configuration for a PC for use with the system (e.g., amount of memory required and level of operating system required), then this should be described.

Impact on Existing Functional Office Operations. If there will be major impacts on the existing operations of the functional office with the implementation of the proposed system, then these should be described in general terms. Some examples of the possible impact of a new or revised system include impact on personnel (elimination of jobs or creation of new tasks), or reallocation of resources, such as from a centralized model to a decentralized model.

Conversion. This section should outline any conversions required for the system. If this system replaces an existing system, then the general approach for converting the existing data to the new system must be described. If there are major one-time startup efforts (e.g. one-time data entry of historical data), then this must be described.

c. Data Elements

Draft data elements and definitions should be written during the Requirements Definition phase. The data elements should be categorized by the major data entities identified (see the earlier section on Database). These data elements will be further refined during the design phases. For projects where a vendor package solution will be pursued, the data elements and the definitions will depend largely on the package selected, and should be finalized during the Installation phase.

For systems that will be developed in-house, if there is an official review process for data elements by a data administration group at the campus or at OP, then this review process should be initiated during this phase. Such a review process may include a review for policy implications, and a review for consistency and standardization of data across administrative applications at the campus or OP. Issues of consistency and standardization are particularly important for applications where data will be shared by

multiple departments, such as data warehouse applications. The data elements and any required review processes should be finalized by the end of the design phases.

d. Review and Approval

The Project Leader is responsible for determining when the Requirements Document may be distributed. When distributed, copies of the Requirements Document should be given to:

- the functional office(s) identified as users of the system
- those within the administrative computing department who must review the document
- Internal Audit (for high risk systems)
- others, for informational purposes

The functional office should agree that the Requirements Definition document accurately reflects their functional requirements for the new system or system modification. If the administrative computing department took the lead on producing the document, then a formal acceptance from the functional office (or offices) may be needed to ensure their agreement.

If the Requirements Definition document was primarily or entirely prepared by the functional office and the administrative computing department will be involved in design and/or programming (or maintenance) phases, then the administrative computing department must review the document to ensure that the requirements are adequately defined and that the proposed project is consistent with the overall technical strategies of the campus or OP.

For some projects, the Project Leader may elect to require review and approval of the Requirements Definition document from a working group of end-users in offices outside of the functional office(s) comprising the primary users of the system. This may be appropriate, for example, in the case of a campus-wide financial system that will impact offices outside of the Accounting Office, which is perhaps the project sponsor or major user.

The Requirements Definition document is also given to Internal Audit for review for high risk applications. (Copies of the Requirements Definition documents for applications not considered high risk are also available to Internal Audit upon request.) Audit's review of the Requirements Definition document should provide feedback regarding whether or not necessary audit requirements have been identified (this is not to imply that Internal Audit is responsible for certifying the adequacy of the design and controls). In addition, Internal Audit should identify the degree of involvement desired by their office for the remaining phases of development.

6. REQUEST FOR PROPOSAL (PURCHASE TRACK ONLY)

a. Purpose

The purpose of the Request for Proposal phase is to engage in a formal process for eliciting bids from vendors when the project involves a vendor package solution. This phase is part of the Purchase Track (for more information regarding the different development tracks, see section 1.3 Systems Development Tracks).

Note: A Request for Proposal document may also be issued if a vendor may be engaged to perform a feasibility study or custom development of an application. This section may be used for guidelines in these cases, although the exact content of the RFP document will differ somewhat.

b. Request for Proposal Document

The core of the Request for Proposal (RFP) document is describing the functional requirements for the system, which has been accomplished as part of the Requirements Definition phase. The Requirements Definition document forms the basis of the RFP document.

In addition to the functional requirements contained in the Requirements Definition document, the RFP should include the following:

General Information.

- an overview of the overall campus or OP organization and a description of the functional offices that will be using the system, plus a description of the overall technical environment in which the system will operate and of the administrative computing organization (whether it is centralized or decentralized).

- a schedule of key events, both for the proposal process (deadline for proposals (vendors must be given a reasonable amount of time to prepare responses), schedule for UC's decision, etc.) and planned implementation date (subject to change at the discretion of UC)

- response terms and conditions, to provide information regarding the formal procedure surrounding the RFP process, including:
    - who to contact for clarification of the RFP;
    - date and location for delivery of proposal;
    - official approval process and notification of award; and
    - some definitions for the purpose of legal protection of UC, such as:
        - the right for UC to amend or supplement the information in the RFP;
        - the right for UC to not make any award as a result of this RFP process;

- assignment or subcontracting (e.g., will the vendor subcontract any of the work); and
- any issues regarding news releases and confidentiality.

Technical Requirements.

- Any restrictions regarding the hardware or software to be used for the vendor package must be listed. If the product must run on currently-installed hardware and operating system and work within the current network configuration, then this must be stated. If the database management software, application development and/or language software, and system software (e.g., a specific teleprocessing monitor such as CICS) must match that already installed at the UC campus or OP, then this must be stated. If a required environment is described and there are known upgrades or major changes in the future, then these must be outlined as well.

  Whether or not UC has specific hardware or software restrictions, all responses to the proposal must include a complete description of the technical environment required for the product. In addition, the vendor response must state how long the package will be supported in the given environment, and what major changes to the required environment are in the foreseeable future.

- Any restrictions regarding how security and authentication must be handled by the package must be explained in the RFP. For example, if it is a requirement that the package handle security using DCE, then this must be stated.

  Whether or not UC has specific security and authentication restrictions, the vendor must supply information regarding how security and authorization are handled within the product, if this is a requirement for the application.

- Support requirements, both during the installation process and on-going support, must be outlined in the RFP.

  The vendor responses must outline in detail the level of support that will be provided (and the associated costs), including: the expected staffing level during the installation process; the amount of training to be provided during the installation process both for the functional office(s) and for internal technical staff; on-going support, both for functional office questions (on-site and by telephone), and for technical problem solving and on-going maintenance of the application.

- Vendor responses must state whether or not UC-specific customization of the package is possible, and whether the vendor undertakes responsibility for custom changes, and the on-going maintenance of the custom changes. How customization of the package, if allowed, would affect implementation of future upgrades to the package must also be stated in responses.

- Vendor responses must include a description of how upgrades to the package are handled (both responsibility for installation and the cost of upgrades); whether or not the vendor will undertake future customization of the package at the request of UC and how such customization would affect future upgrades; and how the vendor determines what enhancements are made to the base package.

Other Restrictions or Vendor Response Requirements. If there are other specific restrictions known at this point related to the anticipated vendor relationship or other information required from the vendor, then these should be explained. Possible additional items include:

Information regarding the vendor, such as:

- number of similar applications implemented for clients;
- years the vendor has been providing the types of services or software outlined;
- financial status; and
- client references.

Any requirements related to cost and/or payment schedule, e.g., that the payments will be phased with a specified percent held until after final completion.

Any insurance that the vendor is required to carry.

Proposal Format and Instructions. The RFP should outline the format of the responses. A possible outline for the standard format of responses follows:

- Proposed Solution:
- Table of contents
- Description of system solution
- Application software requirements
- System-level software and technical requirements
- Hardware requirements
- Proposed implementation schedule and level of effort
- On-going support
- Exceptions to the RFP (i.e., identification of any areas that the vendor cannot address)
- Sample contracts
- Sample documentation
- Vendor Information:
- Vendor qualifications
- Vendor financial data and background information
- Client list
- Cost Proposal:
- Cost information, including: cost of the package, installation, and on-going support

c. Process for Issuing a Request for Proposal

Once a decision is made to proceed with a Request for Proposal process, the campus Purchasing Office should be notified. The schedule for preparation and issuance of the RFP document must include time for the Purchasing Office to review the RFP, if needed, and to perform the official distribution of the RFP. The Purchasing Office should be consulted during the preparation of the RFP, for example, if changes to the schedule occur, or if questions arise regarding the process.

Once the RFP document is completed (normally by the project team), it is forwarded to Purchasing together with a recommended list of companies to which it should be sent. Purchasing then reviews the RFP, and may include additional companies in the distribution list. Purchasing distributes the RFP, and also posts the RFP publicly.

d. Review of Responses

It is important to establish an evaluation framework within which the responses to the Request for Proposal can be reviewed. This framework must take into account the requirements for the system as defined in the RFP document. It is helpful to have a weighting mechanism, so that the requirements may be prioritized. The review process is summarized, together with any forms used for the evaluation and a final recommendation for a selection, in the Feasibility Study (see next section: 2.7 Feasibility Study).

7. FEASIBILITY STUDY (REQUIRED FOR PURCHASE TRACK)

a. Purpose

The feasibility phase is only necessary under circumstances where significantly different options are available for providing a systems solution to the functional needs identified in the System Definition phase (Prototyping Track) or the Requirements Definition phase (Traditional or Purchase Track). For example, a feasibility study would be required:

- when significantly different approaches by the administrative computing department in meeting the functional office's requirements are possible;
- when there is a choice of either in-house development or purchase of a vendor product ("make or buy"); and
- when vendor development or purchase of a vendor product is considered. In this case, the feasibility study should be used to evaluate different vendor responses to Requests for Proposals (see previous section).

b. Feasibility Study Document

The Feasibility Study may be prepared by the functional office or by the administrative computing department, although both offices will be involved in its review. In some

cases, particularly for UC-developed applications where different options are possible, the Feasibility Study may be incorporated into the Requirements Definition or System Definition document.

The contents of the document will vary somewhat depending on the alternatives available for a particular project. Generally, the following should be addressed in the study:

Overview of the Proposed System. A brief overview of the proposed system should be included in the Feasibility Study, although the System Definition or Requirements Definition document should be referenced for further details.

Description of Alternatives. A description of each of the alternatives, or vendor proposals, should be made, outlining in broad terms the advantages and disadvantages of each alternative.

Features Comparison. A more detailed comparison of how each of the various alternatives meets the requirements should be presented. Often it is necessary to assign a priority to each major feature, since some features are critical or mandatory, while others may be only “nice to have”. Any option which does not meet the mandatory requirements will be eliminated from consideration.

The attached form, “Features Comparison”, provides an example of how this comparison might be displayed. On this form (or something similar), each major feature of the system is listed under the “Feature” column, each feature is rated in priority as Mandatory, Desired or Optional, and each alternative is ranked as to how it satisfies the system feature. For example, for a Human Resources application, one major feature might be retention of historical data, which the functional office considers a required, or Mandatory feature. If there were three alternatives being considered, e.g., three different vendor solutions, each vendor’s product would be rated, perhaps on a basis of 1 to 5 (where 1 indicates that the feature is inadequate or missing from the product, and 5 indicates that the product provides an excellent facility for the given feature). This type of comparison provides a method for quantifying, to some extent, how well each alternative identified meets the system requirements.

Cost Comparison. The costs of each alternative should be fully indicated. Included should be both one-time costs (e.g., software development or acquisition and hardware acquisition or upgrade) and on-going costs (e.g., for staff to maintain the system, and maintenance and/or upgrades of software and hardware).

Compatibility. The compatibility or consistency of each approach with other existing applications and hardware or management’s intended hardware and software environment should be considered. For example, an Oracle version of a software package with desired features may be available, but if all other applications currently in place are Sybase-based, then the impact, including the cost of training for and supporting an Oracle software package, may overshadow the advantages of the

software. The compatibility of the proposed solutions with current procedures and operations should also be addressed.

Evaluation of Vendor. When the evaluation of alternatives includes consideration of vendor proposals, the following additional types of issues should be addressed:

- Vendor Profile/History
    - How many applications similar to that proposed has the vendor completed?
    - How many years has the vendor been providing the types of service or software which is proposed?
    - How closely related to the vendor's primary installations/services is the proposed solution?
        - What is the financial status of the vendor? Financial statements can be requested to ascertain the financial stability of the vendor.
        - What is the level of satisfaction of current clients? Client lists can be obtained to gather this information.
- Vendor Support

  If the vendor will continue to provide support after installation, the following issues should be addressed:

    - The level of support provided should be evaluated. Typical measures are the number of support staff in each geographical area, the expertise of the support staff, etc.
    - The normal range of response time for requested service should be identified, if relevant for the proposed application.
    - The cost for the on-going vendor support and/or product maintenance should be identified.
    - The cost of new releases of vendor software as they become available should be identified.

Schedule. Differences in the implementation time frame of the various options should be noted.

Recommended Alternative. A recommendation should be made for which alternative should be selected, with reasons given to support the recommendation.

c. Review and Approval

Functional office agreement with the recommended alternative and approval to proceed is required before work can begin on the next phase of systems development.

If the Feasibility Study was primarily or entirely prepared by the functional office and the administrative computing department will be involved in design and/or programming and installation (or maintenance) phases, then the administrative computing department must review the document to ensure that the recommended alternative is consistent with the overall technical strategies of the campus or OP.

FEATURES COMPARISON FORM

Feature	Priority			Alternative 1	Alternative 2	Alternative 3
	Mandatory	Desirable	Optional			

8. VENDOR CONTRACT AND INSTALLATION PLAN (PURCHASE TRACK ONLY)

a. Purpose

Once a selection of a vendor has been made (see 2.7 Feasibility Study), a contract must be negotiated and completed with the vendor. Once the contract is in place, an Installation Plan (basically, a Project Plan specifically for package installation) must be developed. These steps are only required when purchasing and installing a vendor package (i.e. these steps are part of the Purchase Track; for more information regarding development tracks, see section 1.3 Systems Development Tracks).

b. Vendor Contract

The terms of a contract with a vendor of a systems package are extremely important, particularly if the relationship with the vendor will be on-going (e.g., the vendor will provide on-going support for the package after installation). An inadequate contract could result in unexpected costs to UC or problems with providing adequate support for the system, particularly in the event of difficulties arising in the relationship with the vendor.

The terms of the contract must clearly delineate the responsibilities of the vendor for the project. This must include one-time tasks, such as the following: customization or specialized programming; conversion processing; testing; installation of the package; training of both functional and technical staff; and any follow-up issues with respect to training or problem solving. If installation of the package will involve any modifications or additions to the technical environment (hardware, systems software, or networking capabilities), then the vendor's responsibilities in these areas must be listed. If the vendor will be involved in any on-going support, then these responsibilities must also be explained in the contract, including, for example: answering questions from the functional office on use of the package; technical problem resolution; further customization or specialized programming; installation of maintenance releases or upgrades to the product; and on-going maintenance of the product.

In addition to clearly explaining the vendor's responsibilities, the following issues must also be addressed in the contract:

- The costs and payment schedule must be included. It is recommended that the payment schedule be phased, and dependent on specified milestones or deliverables. It is also recommended that some portion of the total payment is held (e.g., 10%) until the installation is complete, to ensure the vendor's completion of the project. Often contracts will include a good faith deposit made by UC to the vendor at the beginning of the project.

- Requirements for the vendor to carry insurance must be specified in the contract. This would include, for example, Worker's Compensation insurance for their staff, liability insurance, and possibly insurance related to the product itself.

- The contract should specify how changes to the contract will be handled. In any major project, some changes must be expected, although vigilance should be exercised by both functional office and administrative computing staff to limit the number and the extent of changes that occur at this stage of the project. However, the contract should protect UC as much as possible from excessive additional expenses being charged by the vendor in the case where some reasonable changes (albeit possibly outside the scope of the contract) are requested. This portion of the contact should specify an hourly rate for vendor staff to perform changes or additional work beyond that anticipated at the time of the contract negotiation.

c. Development of Interfaces and Customization of the Package

In most cases, installation of a package will involve some programming. The two most likely categories is modifying or rewriting interfaces between the package and other installed applications, and any customization that will be done to the package prior to installation. Any programming required as a result of installing a package must follow the phases for one of the other two development tracks, Prototyping or Traditional Life Cycle, which include phases for analysis, design, programming and unit testing. For these phases, the work may be handled as a completely separate project (or multiple projects), with separate Project Plans. These projects must then converge prior to the System Testing and Implementation phases.

d. Installation Plan

While an Installation Plan does not differ in concept from the Project Plan that exists for every project, the installation of a vendor package will often consist of a large number of tasks performed by external vendor staff, internal administrative computing staff and functional office staff, often under strict time frames, and requires careful oversight for successful completion. Certain aspects of the Installation Plan are specifically related to installation of a vendor product.

As for any Project Plan, the Installation Plan primarily consists of a list of tasks required to complete installation of the package, together with who is responsible for performing each task and a deadline for completion of the task. Logical sets of tasks may be grouped, and completion of a set of related tasks is a project milestone. (See section 1.5 Project Planning and Management for further information regarding the Project Plan).

In addition to containing the tasks required for testing and installation of the product, the Installation Plan must combine implementation dependencies and dates for all “projects” that may be related to the installation of the package. This would include any programming performed separately from the package installation for interfaces or customization of the package. All related projects must converge, prior to System Testing and the final Implementation. The Installation Plan is a key tool to help ensure that this convergence occurs smoothly. The Project Leader is responsible for

coordinating across all related projects, and ensuring that work is progressing satisfactorily on all fronts.

e. Coordination with Vendor Staff

Even when responsibilities of all participating groups (functional office or offices, administrative computing department, and vendor staff) are clearly defined and well understood, on-going coordination between all groups, and particularly with vendor staff, is critical to a successful installation. The Installation Plan is an important tool in ensuring that everyone understands the tasks and responsibilities, and that tasks are completed on time. In addition, other tools may be used to assist in the coordination process, including, for example:

- regular (e.g., weekly) meetings between representatives from each group, to review the Installation Plan and progress on the active tasks, and to resolve any issues that have arisen;
- regular conference calls, daily during periods of intense work, to ensure that work on the tasks is progressing, and that any problems are dealt with quickly; and
- an on-going list of any outstanding issues that arise, together with the person or group responsible for the resolution (or for the next action required for resolution). This issues list differs from the task list on the Installation Plan, since generally outstanding issues require research or discussion before they are resolved, rather than well-defined tasks to be performed. Resolution of an issue may result in additional tasks being added to the Installation Plan.

9. GENERAL DESIGN (TRADITIONAL TRACK ONLY)

a. Purpose

The General Design phase results in development of an overall system design based on the approved Requirements Definition document, and on the chosen approach selected in the Feasibility Study, when present. The General Design expands and refines the requirements. Output from this phase includes a General Design document which describes to the functional office(s) what the system will look like when completed, and establishes a basis within which the administrative computing department (or the vendor) staff completes the Detail Systems design and subsequent development of the system.

The General Design phase is part of the Traditional Track for systems development. (See 1.3 Systems Development Tracks for further information regarding different development tracks.)

The level of detail in a General Design document will vary depending upon the size and complexity of the system. In some cases, it may be appropriate to combine the General and Detail design phases.

The General Design is completed by the administrative computing department or a vendor, as appropriate. Extensive involvement of the functional office must continue through the General Design phase, to ensure that the system design meets the functional requirements identified during the Requirements Definition phase.

The outputs of this phase are: a General Design document, an updated Project Plan, and an updated list of the data elements (initially produced in the Requirements Definition phase as a preliminary list of elements).

b. General Design Document

The primary purpose of the General Design document is to describe the system from both technical and functional perspectives so that both the functional office and the application developers can understand what it will look like and do. The following sections may be included in the document.

Introduction. The introduction should reiterate the purpose of the system or analysis of the problem and the objectives to be met by the proposed system (or system modification), as stated in the Project Proposal and Requirements Definition documents.

Assumptions. Any technical or functional assumptions upon which the design is based should be stated. Also stated should be policy assumptions made during the General Design phase.

Differences from Requirements Definition. During the course of developing the General Design it is possible that some changes to the Requirements may be made. Such changes should be highlighted for review by the functional office.

Database Design. The design and structure of the database should be described. For example, if the database is relational, then the tables and the relationships between tables (including any required referential integrity) should be described.

How each data element fits into the database structure should be specified. For example, for a relational database, the data elements within each table may be listed, and the key fields identified.

Source(s) of Data. Sources of input data and method(s) for loading data (batch or online update) should be described. Required edits for data elements should be listed.

For interfaces from other UC sites or from external sources, consultation with the external organization must have occurred to ensure their willingness and ability to supply required data within the required timeframes. An outline of future involvement by the external organization, including providing test and production files, should be specified, either in the Design document, or in the Project Plan.

Processing Logic. Processing logic for batch and online processing must be described. Processing logic includes, for example, loading data into the database via

online update or batch processing, handling of input files from other sources (interfaces TO the system), producing output files for other applications (interfaces FROM the system), producing reports, providing access to data for inquiry or ad-hoc requests, etc. An explanation of error or exception conditions and how they will be handled should be included.

Reports. Any reports required should be described, including: frequency (daily, monthly, on request, etc.), general purpose, selection criteria, and data elements to be included.

Output Files. Outputs of the system other than reports should be described. This may be files required for interfaces to other systems, or possibly files required for providing ad-hoc access to data by the functional office.

Interfaces. Interfaces with other applications may be described under Sources of Data (for interfaces TO the system) or Output Files (for interfaces FROM the system). Whether interface files are described under earlier sections or in a separate interfaces section, the following information should be included for each interface file:

- Will the interface be TO the system (i.e. the system will receive the data from another application), or FROM the system (i.e. the system will produce the data for another application)?

- The format of the interface file should be described.

- Are the interfacing systems external to UC, at another UC site, or local to the campus (or OP)?

- What data elements are required for interfaces FROM the system?

- How will the interfacing system be tested?

- What is the proposed timing of the interface in a normal cycle?

Ad-Hoc Access to Data. If the proposed system will be required to support ad-hoc access to data by the functional office(s), then how such access will be accomplished should be described.

System Architecture. A description of the system architecture should be included, i.e., standalone PC-based, client/server, web-based, etc. If remote access to data is required (e.g., by functional offices at other UC sites), describe how this will be provided. Issues such as local printing capabilities, downloads of data for specific purposes, and other system architecture requirements should be addressed.

System Controls. Design considerations to satisfy system control requirements must be described. For example:

- for batch processing, record counts should be provided as needed (e.g., record counts of records contained on an input interface file vs. records updated, added or deleted to a database, or rejected for errors)

- any processing controls that are necessary to ensure accuracy of arithmetic calculations (e.g., verification that the sum of monthly totals equals the annual total) should be included

Security and Audit. Security, audit, and privacy considerations identified in the Requirements Definition must be addressed. Data access restrictions should be described, including identification of restrictions of selected users to particular functions (e.g. inquiry, or limited update access), or to particular data.

Audit considerations should be addressed in this section. How the audit needs identified in the Requirements Definition phase will be satisfied should be described.

Conversion Planning. The process for converting existing data (either hardcopy or machine-readable) to the new system should be described. Also, the means of initially loading data should be described.

User Training. If training will be required for users within the functional office, then the training approach and timeframe should be outlined.

Documentation. The documentation to be provided with the system should be listed and described (also see the section on Documentation).

c. Data Elements

An updated list of data elements including definitions should be completed during the General Design phase. The data elements should be categorized by the major data entities identified (i.e., for relational databases, the table(s) in which each data element belongs should be identified).

At a minimum, the data element definitions should include the following information:

- Data Element Name
- Type of data element (numeric or alphanumeric)
- Format of the data element (e.g., for numeric fields, the number of digits allowed, such as 3v2 (meaning a number of the format nnn.nn))
- General description or definition of the data element
- If the data element is a coded value, then the possible codes and the corresponding interpretation of each code
- Programming name (i.e., the name of the field in the database)

Additional information, such as data editing requirements and source of the data element, may also be included in the data element definitions. Data element definitions will be finalized during the Detail Design phase.

d. Review and Approval

The Project Leader is responsible for determining when the General Design Document may be distributed. When distributed, copies of the General Design Document should be given to:

- Functional office(s)
- Those within the administrative computing department who must review the document
- Internal Audit (for high risk systems)
- Others, for informational purposes

For high risk applications, copies of the design should be given to Internal Audit, unless otherwise indicated during the review by Internal Audit at the Requirements Definition phase. Internal Audit review focuses on audit considerations identified in the Requirements Definition.

The functional office(s) may be asked for a formal acceptance of the General Design document, indicating that they agree that the design accurately reflects their needs.

10. DETAIL DESIGN (TRADITIONAL TRACK ONLY)

a. Purpose

The purpose of the Detail Design phase is to expand and refine the system design outlined in the General Design phase. Included in this phase is the development of program specifications from which programs can be designed and coded, development of test plans, and identification of networking and equipment needs.

The Detail Design phase is part of the Traditional Track for systems development. (See 1.3 Systems Development Tracks for further information regarding different development tracks.)

The Detail Design is completed by administrative computing staff, or by a vendor for vendor-developed projects.

The output of the Detail Design phase include the following:

- Detail Design document, including program specifications
- Updated Project Plan
- External interface specifications, where applicable
- Data Element definitions (refined from earlier versions of the Data Element dictionary)

b. Detail Design Document

The contents of the Detail Design document should include the following information:

Introduction. The objectives of the system should be briefly reiterated. An overview of the system with a description of the major components of the system should be included.

Assumptions. Any assumptions on which the Detail Design is based should be listed. If any of the assumptions made in the General Design have changed, the changes should be noted and the reasons for the changes given.

System Overview. The system overview should include the following:

- A high-level description of the various portions of the system and how they interrelate
- A description of the major databases and files
- Every data source identified

- Every major output identified, including: reports (with report identifier), and interface files

Databases and Files. Every database within the system must be described. The structure and data elements must be included. For relational databases, every table and the data elements within each table must be listed, as well as the relationship between tables, and any required referential integrity. Where useful, a picture of the database(s) should be included. Key fields must be identified, and indexed fields, both unique and non-unique, must be identified.

Any sequential files must be described. For sequential files, the data elements must be listed, the record format and length given, key fields identified (and whether or not the key fields are unique), and the sort sequence specified.

Flowcharts or Data Flow Diagrams. Where appropriate, flowcharts or data flow diagrams showing the overall processing should be included. These flowcharts should include programs, files, utilities, etc.

Program Specifications. Program specifications will be used to code each program in the next phase: Coding and Unit Testing. Each program specification must include a description of the program function, input/output files, reads or updates to databases, reports, and a description of the processing logic. Specifications for online programs must also include screens, and event-driven processing (i.e. processing triggered by actions performed online by the user, such as pressing certain keys or selecting options).

See the end of this section for a sample detail design program specification for a batch COBOL program.

Data Retention. Required data retention periods for reports (hardcopy, fiche, image, or disk copy), sequential files, and, if applicable, data residing in databases, should be listed.

System Control and Data Verification. This section should describe how the integrity of the data is maintained within the system, from program to program, from cycle to cycle, and from system to system where there are interfaces with other systems.

Any verification of data that must occur outside of programs should be described. For example, if the functional office is responsible for reviewing certain reports before a specific process can be performed, this needs to be explained.

Backup and Recovery. The methods and frequency of backups of the databases and any sequential files must be described. The method and frequency for creating backups for offsite storage should also be described.

Methods for recovery of data in the event of system problems must be outlined.

Conversion. Any necessary one-time conversion of data must be described in detail. If conversion programs are needed, they must be specified. Any potential conversion problems or issues should be identified.

Hardware and Software Resource Requirements. For any shared resources, such as shared servers or printers, the impact of the proposed system must be evaluated to ensure that adequate resources are available to handle the new system without adversely impacting existing systems. Any significant upgrades or new hardware resources required for the new system must be identified. The following types of resources should be considered:

- memory allocation requirements on any shared server;
- disk space usage on shared server machines;
- Client workstation requirements (e.g., for PC/Windows clients, minimum RAM required, disk space required, level of operating system software required, and application software required);
- printer requirements, including estimates of pages printed, and determination as to whether or not existing printers will provide sufficient speed for the output expected; graphics printing capabilities; etc.;
- networking requirements, including: workstation networking capabilities required (cabling and software), requirements for a LAN or access to a file server, etc.; and
- level of client application software and server application software[4].

c. External Interface Specifications

If there is an interface between the new (or modified) system and either another UC site (a campus or OP) or an external organization (e.g., a health insurance carrier), then specifications for this interface must be written. If the interface is FROM the other site or organization, then the specifications must be sufficiently detailed to enable the other organization to develop the interface file processing. If the interface is TO the other site or organization, then the specifications must provide sufficient information for the other organization to effectively use the interface file in their application. At this point in the development process, any general issues regarding the interface should have been resolved.

Interface specifications must include the following:

- Overview of the data required for each file specified, including a description of the data.

[4] It is possible, especially if the development time is long, that application development software or possibly operating system software may need to be upgraded during development efforts. If needed, the time for upgrading and testing the underlying software used for developing the application must be taken into account in the project planning and schedule.

- Frequency of data submission (i.e., monthly, annually, etc.), effective dates of the submissions, and due dates (e.g., 5th working day of the month).

- A record layout for each file transmitted must be included.

- A list of data elements, including the format for each data element, must be included.

- Physical file characteristics, including the data set name, record length and record format

OR

- The file name to be transmitted for files to be transferred via ftp, and the target system for the ftp transmission.

- The procedure for notifying the receiving site when the file has been submitted.

d. Data Elements

An updated list of data elements definitions must be completed.

e. Review and Approval

The Project Leader is responsible for determining when the Detail Design Document may be distributed. When distributed, copies of the Detail Design Document should be given to:

- functional office(s) using the system
- those within the administrative computing department who must review the document
- Internal Audit (for high risk systems)

The functional office(s) may be asked for a formal acceptance of the Detail Design document, indicating that they agree that the design accurately reflects their needs.

f. Sample Program Specifications for COBOL

Program Name: IAD030

Overview

The purpose of this program is to identify duplicate records in the input file and eliminate them from the output file. Information about duplicate records is included in appropriate reports.

Input

Ddname: IN
Record Length: 1504
Record Format: Fixed
Organization: Sequential

Output

1. Files

   Ddname: OUT
   Record Length: 1504
   Record Format: Fixed
   Organization: Sequential

2. Reports

   Ddname: IAD0301 - IAD0307
   Record Length: 133
   Record Format: Fixed with ASA control character
   Organization: Sequential
   Report Names: IAD0301 - IAD0307

   Ddname: SYSPRINT, PLIDUMP
   Attributes: Default

SYSTEM:
LRECL: DATASET/DD NAME: IN, OUT
BLKSIZE:
RECFM: Fixed COPYLIB MEMBER: IADDCL
DSORG: Sequential

RELATIVE FIELD POSITION	DATA ELEMENT	LOGICAL LENGTH	PHYSICAL LENGTH	DATA TYPE	COMMENTS	ELEMENT NUMBER
0	SS#	9	6	PACKED		
7	FILL_1	1	1	CHAR		
8	CAR_APPL	2	2	CHAR		
10	FILL_2	1	1	CHAR		
11	UND_GR_LV	1	1	CHAR		
12	FILL_3	5	5	CHAR		
17	ALPHA_NO	10	10	CHAR		
27	NAME	35	35	CHAR		
62	FILL_4	1	1	CHAR		
63	KEY	10	10	CHAR	/*NOT IN SBR 102	
73	NEW_ELIG	2	2	CHAR	/*NOT IN SBR 102	
75	NEW_SPEC_ACCT	2	2	CHAR	/*NOT IN SBR 102	
77	FILL_5	22	22	CHAR		
99	BIR_DT	6	6	CHAR		
105	SEX	1	1	CHAR		
106	FILL_6	1	1	CHAR		
107	VET	1	1	CHAR		
108	HANDICAP	1	1	CHAR		
109	FILL_7	26	26	CHAR		
135	US_CIT	1	1	CHAR		
136	FILL_8	1	1	CHAR		
137	COUNTRY_CIT	15	15	CHAR		
152	FILL_9	1	1	CHAR		
153	QTR_AP	1	1	CHAR		
154	YR_AP	2	2	CHAR		
156	COL_CODE	2	2	CHAR		

Report Layouts

Files IN = input

Files OUT = output

Report Layouts

1. IAD0301

2. IAD0302 - IAD0306

3. IAD0307

Processing Logic

1. Input

   Every record in the input file, IN, is processed. If the file contains zero records, processing is terminated with a condition code of 16. Key for the file is defined as YR_AP QTR_AP KEY. The file must be in ascending key sequence. If not, processing is terminated and a condition code of 16 set.

2. Identifying Record Groups

   A record group is defined as a set of input records with the same key. A record group may contain only one record. In the interests of efficiency, groups containing more than ten records are subdivided into subgroups of ten records each (or fewer if less than ten remain). Each subgroup will be treated in processing as if it were a separate group, as if its key were different than that of any other subgroup. For each subgroup a detail line for report IAD0307 will be generated. The effect of dividing into subgroups is that duplicates across subgroups will not be identified. The size of ten records is somewhat arbitrary and must be easily changeable by recompilation.

3. Identifying Duplicate Records and Generating Output

   For each group or subgroup process as follows:

   One Record in Group

Write record to file OUT

Two Records in Group

a. SS#'s blank or unequal and NAME's unequal

Write records to file OUT.
Generate detail lines for report IAD0305.

b. SS#'s not blank and equal or NAME's equal

Do not write records to file OUT.

1) CAMM_APPL's equal

Generate detail lines for report IAD0304.

2) CAM_APPL's unequal

a) CHNG_TYPE in one record not blank and LOSING_CAMP in that record equal to CAM_APPL in other record. Generate lines for report IAD030.

b) Converse of above.

Generate detail lines for report IAD0302.

Three or More Records in Group

Generate detail lines for report IAD0306.

a. Records where NAME equals NAME in at least one other record in group or where nonblank SS# equals SS# in at least one other record in group. Do not write records to file OUT. Set field k. in detail lines to 'R'.

b. Records not meeting condition in 1. Write records to file OUT. Set field k. in detail lines to 'A'.

Note: When it is indicated that detail lines are to be generated, generate one line for each record in group.

4. Report Production

a. IAD0301 - Status Report

Report is produced upon normal completion of processing.

Print message 1 if sequence error detected, or message 2 if input file empty, or message 3 if normal completion of processing.

If normal completion produce control totals.

Records Read: records read from file IN.
Records Rejected: records that caused detail lines to be generated in reports IAD0302-IAD0304 or caused detail lines in IAD0306 to be generated with field k. set to 'R'.
Records Written: records written to file OUT.

The other totals shown in layout indicate the number of detail lines written in each of the reports IAD0302-IAD0307.

b. IAD0302-IAD0306 - Error and Advisory Reports

All reports are of same format.
Page numbering is continuous in each report.
Break to new page if QTR_AP or YR_AP changes.
Interpret QTR_AP as follows:

CODE	INTERPRETATION
'2'	'bb FALL'
'3'	'WINTER'
'4'	'SPRING'

Produce each report only if at least one detail line is to be written for that report. Each group of detail lines with equal keys are to be single spaced and separated from other groups by double spacing. The maximum is 60 lines per page. If there is insufficient space remaining on a page to print all detail lines in a group, a new page will be initiated. BIR_DT and SS# require insertion of special characters as per report layout. Interpret SEX and CHNG_TYPE as follows:

SEX	INTERPRETATION
'1'	'M'
'2'	'F'

CHGN_TYPE	INTERPRETATION
'1'	'CH_OF_CH'
'3'	'REDIRECT'

Default interpretation is blank.

Include field k. on report layout only on report IAD0306.

c. IAD0307 - Implementation Limits Exceeded Report

Produce report only if at least one detail line is to be written. Maximum is 60 lines per page. Page numbering is continuous.

11. PROGRAMMING AND UNIT TESTING (TRADITIONAL TRACK ONLY)

a. Purpose

The purpose of the programming and unit testing phase of a project is to complete the design and programming for each database and each program[5] or portion of the system. The application developer is responsible for testing each portion of the system as it is completed. In some cases it may be appropriate to involve the functional office in testing portions of the system as they are completed (e.g., testing a screen as the processing for that screen is completed), whereas for other projects, functional office involvement in testing may be primarily during the System Testing phase.

The Programming and Unit Testing phase is part of the Traditional Track for systems development. (See 1.3 Systems Development Tracks for further information regarding different development tracks.)

b. Programming

The design of the databases and programs or portions of the system is based on the Design document (either the Detail Design, or the General Design for those projects where the Detail and General Design phases were combined). The programming standards and guidelines in effect for the campus or OP for the language being used for systems development must be followed -- refer to local standards manuals for further information.

c. Unit Testing

The application developer is responsible for determining what testing is required for a program. A test plan, itemizing the test conditions to be exercised during testing, may be developed. The test plan can be an informal list for use by the application developer only, or may be a more formal list if functional offices will be involved in testing or test verification.

In general, testing should address the following:

- every function performed by the program should be tested;
- for event-driven programs (e.g., online programs), every event should be tested;
- any error conditions should be tested; and

[5] The term “program” is no longer applicable to many application development software environments. For example, within the PowerBuilder application development environment a programmer develops windows, menus, objects attached to these windows or menus, and scripts for the objects or menu items. Since the terms vary across the different application software products, the older term “program” will continue to be used in this document, and is intended to encompass the wider variety of terms used today, including: scripts, objects, windows, menus, stored procedures, etc.

- "boundary" conditions should be tested (for example, if an online program is designed to handle up to ten entries on a screen, then a test of entering ten or eleven entries and a test of entering zero or one entry should be performed.)

Test results must be reviewed carefully. The review of test results may be completed by the application developer, by the Project Leader, by the functional office, or by a combination of these staff.

d. Review and Approval

Review and approval of programming and testing is the responsibility of the Project Leader. Upon completion of the Programming and Unit Testing phase, the system must be ready for the System Testing phase.

12. SYSTEM TESTING

a. Purpose

The purpose of performing System Testing for a project is to ensure that all portions of the system work correctly (and as specified) and work together. Often portions of a system are initially developed and tested as independent pieces, and the system tests must ensure that the entire system works.

b. System Test Plan

The test plan must identify the steps to be performed and the test conditions to be exercised in order to test the system. The plan should identify who is responsible for developing the test data, performing the testing, and verifying the results. Normally, the functional office will be heavily involved in developing test data, verifying test results, and testing any online portions of the system. For some projects, the functional office may be heavily involved in preparation of the test plan as well.

The test plan should also identify the criteria for determining when testing is completed, and the person responsible for determining the completion of testing. A schedule should be established for the steps identified in the test plan.

c. System Testing

The types of testing that might be included in system testing, depending on the application, are as follows:

- test of batch and online processes in conjunction with one another (e.g., perform online testing of data entry, and subsequent batch processing using the data entered online);
- different cycles of any processing (e.g., daily vs. monthly, fiscal year-end closing processes, etc.);
- volume testing;
- performance testing, e.g., to confirm that online response time is adequate;
- backup and recovery procedures;
- test procedures and processing for receipt of files from other UC sites or external organizations, and creation of files for other UC sites or external organizations;
- test of network interfaces (e.g., lpr printing, ftp transmissions, and any other client/server processes); and
- test of workflow queue processing.

In addition, the following types of testing should be performed as part of system testing, if appropriate for the given application:

- testing of any conversion processes or one-time data load processes;
- parallel testing, for systems that will replace all or part of an existing system; and
- test of interface processing, both for incoming interface files, and the creation of outgoing interface files.

These three types of testing are discussed in more detail in the following sections.

Conversion Testing. For systems which will replace all or part of an existing system, it is often necessary to code one-time conversion programs, to convert the data from the format used in the existing system to that of the new system. This one-time process must be tested as part of the system test. It is important to identify how the conversion process will be verified, i.e. how it will be determined whether or not the data was correctly converted.

Note: Often parallel testing is a valid approach for verifying that the conversion programs are correct. The steps for using a parallel test to verify conversion processing are listed below:

- run the conversion programs, to convert the data from the existing format to the new format
- run both the existing system and the new system, in parallel
- convert the processed data from the new system back to the old system's format, and perform a comparison of data from the old system to data from the new system (this requires a one-time backwards conversion process)

Parallel Testing. For systems which will replace all or part of an existing system, parallel testing is normally performed. Parallel testing usually involves continuing with the existing system for production purposes, and running the new system "in parallel", using identical input data. A parallel test plan should be developed, outlining the steps necessary to complete the parallel test. The plan should include the following:

- how input data will be duplicated for the new system;
- how verification of data and reconciliation between the outputs from the two systems will be accomplished;
- how any inconsistencies in outputs will be handled (i.e., are some inconsistencies acceptable, and if not, how will the discrepancies be resolved);
- areas of the system that will be parallel tested, and those that will not be parallel tested; and
- methods for ensuring that test data from the new system does not interfere in any way with the continuing production of the existing system.

Interface Testing. For most systems, every interface to and from the system must be tested as part of the system test. These tests must be coordinated with those responsible for the other applications (particularly applications which receive data from

the new system), so that appropriate testing will be performed in the other application as well. The methods for verification of the interface tests must be identified, both within the new system, and in those applications receiving and testing interface files. The number of tests to be performed must be identified (e.g., if different cycles need to be tested, or special year-end versions of the interface need to be tested).

Acceptance Testing. As noted above, the functional office(s) must be heavily involved in many aspects of system testing. The functional office(s) may be involved in preparing the test plan and preparing test data, and will most certainly be involved in testing any online portions of the system and in verification of test data.

In some cases it may be useful to handle the functional office testing of online portions of a system as a separate, identifiable portion of the System Testing, often referred to as Acceptance Testing. By somewhat isolating this portion of the testing, it may be easier for the functional office to concentrate on the online portions of the system, and ensure that these portions of the system work as specified. Often it is helpful to isolate the acceptance testing technically as well (in a separate testing region, for example), to enable multiple types of system testing to occur simultaneously.

The Project Leader may elect to require a formal signoff from the functional office(s) upon completion of Acceptance Testing.

13. IMPLEMENTATION

a. Purpose

The purpose of the Implementation phase is to move the system into production.

b. Description

Prior to the start of the Implementation phase, an Implementation approach must be established. For some projects, implementation is straightforward. For others, especially in the case where an existing system is being replaced, the cutover process may be quite involved, and often must be performed under strict time-constraints.

The Implementation Plan should include a schedule of dates and who is responsible for each cutover step. Steps may include the following:

- Moving the code into production (e.g., into central production libraries);
- Distributing executable code on workstations within the functional office(s), as required;
- Moving data into production databases, including any conversion processes for converting existing data into the new system;
- Setting up user accounts or id’s and associated authorizations for access to production programs and/or data, as needed; and

- Any required announcements that may need to accompany the startup of the system.

If the system provides the capability for direct access by the functional office(s) to the data for ad-hoc reporting, then the functional office(s) is responsible for ensuring that privacy regulations and policies are adhered to for any individually-identifiable data elements (see Section 3.2 for further information regarding Privacy).

14. DOCUMENTATION STANDARDS

a. Overview of Written Documentation

The following documentation, if applicable, must be completed for each application:

- Operations Manual

  The Operations Manual contains information required for the production control group to run batch portions of the system, if any. The Operations Manual must be reviewed by the manager of the production control group. (The production control group is the group within the administrative computing department that is responsible for running batch portions of production systems.)

- Systems Manual

  The Systems Manual is for use by those responsible for on-going maintenance of the system.

- User Documentation

  The User Documentation contains instructions for the functional office(s) on how to use the system, and may be combined, if appropriate, with the functional office's internal procedures manuals. The User documentation is completed by the functional office(s) or the administrative computing department, or some combination of staff from these departments, and may consist of online help or a hardcopy manual, or a combination of the two. For a vendor package (or a vendor-developed application), the User Documentation may be completed by the vendor, or the vendor in combination with UC staff.

b. Training

If training of staff within the functional office(s) is necessary for the newly implemented system or system modification, the training may be performed by

the functional office or by the administrative computing staff, or by some combination of these departments, depending on the nature of training required. Planning for any training required must occur well before implementation of the system, so that staff in the functional office can be trained and ready to use the system at implementation. For a vendor package (or a vendor-developed application), training may be performed by the vendor, or the vendor in combination with UC staff.

Training, such as introductory overviews of the system or system modification, may also be held as needed for maintenance staff and for the production control staff. The Project Leader is responsible for coordinating any necessary training within the administrative computing department.

c. Operations Manual

Purpose of the Operations Manual. The Operations Manual contains information required for the production control group to run batch portions of an application. If the application does not include any batch processing, or if the functional office is responsible for the batch processing, then there is no need for an Operations Manual for the application.

The primary purpose of the Operations Manual is to document how and when to run the batch processes, describe any tasks that must be performed before or after batch jobs are run, and provide instructions on what to do when a job does not complete successfully. In addition, the Manual should give enough of a background on the application to provide some context for the batch jobs. The Manual also includes information useful for reference, such as lists of datasets and reports, and a description of all interfaces.

Although the Manual is intended for use by the production control group, this Manual may also be useful to those responsible for maintaining the system, and should be readily available to maintenance staff as well as the production control staff.

Contents of the Operations Manual. The Operations Manual is divided into two major sections:

> Part 1 - General Information. Part 1 of the Operations Manual includes overall information about the application.
>
> Part 2 - Running the Jobs. Part 2 of the Operations Manual documents the individual job procedures within each Operations Stream.

The contents for both parts are described below.

Part 1 - General Information
The first part of the Operations Manual provides overall information about the application. The following sections should be included:

1. System Description

   The System Description section should contain a brief overview of the application. This overview can be taken from other documents, such as the Requirements Definition or System Definition document, or the Systems or Users Manual.

2. Application Control Information - Project Contacts

   A list of the contacts for the project should be listed, including contacts for:

   - production control
   - maintenance programming
   - functional offices within the campus or OP (if the application serves more than one department, then a contact for each major department should be listed)
   - other UC site contacts, if any interfaces with the other UC sites exist (this may include: campuses, OP, laboratories, Hastings, or ASUCLA)
   - Outside or vendor contacts (if data is collected from or sent to outside services, e.g., fiche vendors, a list of contacts must be included)

3. Processing Schedule

   Depending on the application, scheduling of jobs may be straightforward (e.g., “run monthly as soon as input data is available”), or quite complicated. For example, scheduling jobs for the Payroll system is complex, both because of the number of different jobs and their inter-relationships, and because of the interaction between the jobs and the processing in the functional office.

   This section of the Operations Manual should give an overview of how scheduling of jobs is accomplished. For applications where scheduling is complex, details for how to determine the scheduling for jobs must be described, and who is responsible for producing the schedule must be stated (e.g., the functional office, or production control in conjunction with the functional office).

4. Interfaces TO the Application

All interfaces must be documented. The following information should be included for each interface file sent TO the system:

- a brief description of the interface;
- what location or locations the interface file is from (e.g., campuses and/or laboratories and/or OP, another application run locally, or an outside vendor) and, for interface files from applications run locally, what application the file is from;
- which job or jobs use the interface file;
- The schedule for receipt of files (for files sent on a regular basis, this schedule is usually expressed in terms of number of work days from the beginning or end of a period of time. For example, for a monthly file, the file may be due the 5th working day after the end of each month.);
- the method for transmitting the file (e.g., the file will be transmitted electronically via ftp to a specified target machine, or the file will be transmitted on tape medium, etc.);
- The name of the dataset, for whatever target medium on which the file will be sent (e.g., for files transmitted to a local MVS machine, the MVS dataset name (DSN) is required);
- Acceptable values for any portion of the dataset name which is variable, e.g., acceptable format for cycle parameters such as mmmyy (e.g., FEB98);
- Any file characteristics that must be known in order to process the file (e.g., for files transmitted to a local MVS machine, the record length (lrecl) and record format (recfm, i.e. FB (fixed blocked) or VB (variable blocked)) are required);
- The retention period to be assigned to the file; and
- A list of other information to be provided by the sending location when the file is sent, such as record counts for the file.

5. Interfaces FROM the application

All interfaces must be documented. The following information should be included for each interface file created by the application:

- a brief description of the interface;

- what location or locations the interface file is for (e.g., campuses and/or laboratories and/or OP, another application run locally, or an outside vendor) and, for interface files created for applications run locally, what application is receiving the file;

- which job or jobs produce the interface file;

- what the target schedule is for production of the interface file (e.g., by the 10th working day of the month);

- the method for transmitting the file to the target location (e.g., the file will be created on a local machine, for locations to “get” electronically using ftp, or the file will be transmitted on tape medium, etc.);

- The name of the dataset, for whatever target medium on which the file will be sent (e.g., for files created on a local MVS machine for locations to “get” using ftp (file transfer protocol), the MVS dataset name (DSN) is required);

- Acceptable values for any portion of the dataset name which is variable, e.g., acceptable format for cycle parameters such as mmmyy (e.g., FEB98);

- Any file characteristics that must be kept constant for other locations to be able to process the file (e.g., for files created on a local MVS machine, the record length (lrecl) and record format (recfm, i.e. FB (fixed blocked) or VB (variable blocked)) must be kept the same for other locations to “get” the files for processing using ftp);

- The retention period to be assigned to the file; and

- Information regarding how notification to the recipient of the file is handled, including any information to be provided, such as record counts for the file.

6. Account Information

   The account(s) used by the production control group for this application should be described.

7. DataSets

   A list of datasets used by the application should be included. Depending on the application, the list may be split into the following categories:

- libraries (e.g., on MVS, the libraries used for the application-specific software);
  disk datasets (include dataset name and description; this list may be split by machine, for applications where multiple machines are used);
- onsite tape datasets (include dataset name and description); and
- offsite tape datasets (include dataset name and description).

8. Reports

   A list of reports should be included, with the Report ID, the Job that creates the report, and a brief description. The list should be in order by Report ID.

9. Security

   Security within applications can be quite complex, and a complete explanation of all security used within an application is outside the scope of this Manual. However, if there are overall rules for the application related to security that might affect the production control group, then these may be stated in this section. For example, for a Cobol/DB2 application running on MVS, the production control group may have permission to run specific update programs (or DB2 “plans”), but may not have update access directly to the DB2 database. This may be important to know if problems arise or in the case where a one-time process is required.

10. Console Logs

    The retention period for the hardcopy or online console logs should be stated. Instructions for accessing console logs should be included. (Note: unless otherwise stated, the retention period for all hardcopy and online console logs is 13 months.)

11. How to Update this Manual

    Information must be provided on where the machine-readable version of the manual resides, and how to update it and print copies. People who should be given a copy of the manual after revisions are made should be listed. Generally, the list should include: the production control contact and the maintenance contact. In some cases, the list may include one or more functional office contacts as well.

12. Glossary of Terms Used (optional)

A glossary of commonly-used terms that are application-specific may be included. For example, the term “monthly maintenance” within the Title Code System (TCS) refers to the online updates performed by the users on a daily (or as needed) basis -- having a definition of this term in a glossary could be useful for the production control group.

13. Transmission

    Any files or reports that are transmitted to other UC sites (or external vendors) or made available for pick-up electronically by other UC sites (that are not already described under Interfaces) must be listed, including:
    - the job that transmits the file or report or makes it available for pickup; and
    - the dataset name of the file or report.

Part 2 - Running the Jobs

The second portion of the Operations Manual includes instructions for running each batch job within the application. The batch procedures to be run for the application must be organized by Operations Streams. Each Operation Stream is a logical set of batch procedures that are normally run together. A smaller system may have only a single Operation Stream, while larger applications with extensive batch processing may have a number of Operation Streams. An Operation Stream may consist of only a single job, or many inter-related jobs. Examples of Operation Streams include: Payroll Monthly compute, and General Ledger monthly processing.

A list of all Operation Streams in the application should be included at the beginning of Part 2 of the Manual. Also, a brief description of the Operation Stream should be included at the beginning of each Operation Stream.

Within each Operation Stream, there should be a separate section on each batch procedure or job. The following sections should be included, as needed, for each process:

1. Description

   A brief description of what functions the job performs (e.g., updates the primary database, produces reports, creates interface files, etc.)

2. Frequency

   The frequency of the job (e.g., monthly, upon request, etc.).

3. Programs Executed

A list of the programs executed in the job, with a brief description. For MVS jobs, the JCL Procedure and Job step should be included.

4. Files

The files used in the job, including Input Files (i.e. read-only files), Created Files, and Updated Files.

Note: This list of files should include databases that are read or updated by the job, even in cases where the use of the database may not be obvious from looking at the job itself. For example, DB2 tables that are read or updated from an MVS batch job do not appear in the JCL, but should be included in the list of files.

5. Resource Requirements

Resource requirements might include, for example, execution time (particularly if the job is long-running), tape drives, and lines of printed output.

6. Pre-Processing Instructions

- Dependencies

  Any dependencies for this job must be listed. Examples are: confirmation that another job has successfully completed (where confirmation may be performed by the production control group or by functional office contacts), receipt of data from other systems, or status of a CICS region (e.g., a specific CICS region must be down before the job can run).

- Prompted Variables

  All variables that are prompted or required as part of submission of the job must be listed and described. For example, if a cycle date is required, the Manual should show the input format (e.g., mmmyy where mmm is JAN, FEB, etc.), and describe how the value of each variable is derived (e.g., use the current month, use the prior month, the value must be supplied by the functional office contact, etc.).

7. Post-Processing Instructions

Instructions for tasks to be performed after completion of the job. These tasks may include, for example:

- verification procedures (checking of return codes, record counts, totals, etc.)
- log entries (record counts, volume serial number of tape to be sent to an outside vendor, etc.)
- fiche report handling
- interface file handling (for files created by the application and sent to another system)

8. Report Distribution

   Instructions for distributing reports. All reports to be distributed must be listed (report id, and dataset name (if the report is retained on disk)), with the number of copies and the person to whom the report is sent.

9. Rerun Instructions

   Rerun and restart instructions. The instructions should include information on what to do when the job ends abnormally in any given step, and what to do if a complete rerun is required even when the job has successfully completed. Instructions may include, for example, information on restoring databases or files (this may refer to a separate job); deletion of created files before rerunning; or contact the maintenance programmer under certain circumstances.

d.Systems Manual

Purpose*.* The intended audience for the Systems Manual is the programmer(s) responsible for maintenance of the application. The Systems Manual must provide a technical overview of the system, and must provide details on technical approaches taken in the application that are not readily apparent from the code.

Code as Documentation*.* One of the primary sources of information for any maintenance programmer is the source code itself. Application developers are responsible for writing clear code, and for including comments in the code. To the extent possible, every program should have comments at the beginning describing the functions the program performs, and giving other overall information. Throughout the code, comments should be included to assist in following the flow of logic, and to describe any particularly complicated logic.

The Systems Manual is not intended to repeat information that can be found easily in individual programs. Instead, the Systems Manual should provide overall information

and technical details regarding the application, that would be difficult or impossible to discern by looking at individual programs.

Contents of the Systems Manual*.* Since applications vary widely in technical approaches, it is the Application Developers' and Project Leader's responsibility to ensure that the Systems Manual contains information that will be useful to the maintenance programmers and that the Systems Manual is complete. The following sections are proposed as guidelines.

e. Application Description

This section provides a brief overview of the application. (This overview may be taken from the Requirements Definition or System Definition document.)

f. Technical Environment

This section provides a description of the technical environment, including:

- the platform or platforms upon which the system runs (i.e. Operating System, and, if the application is dependent on specific hardware, the hardware required);
- The application software used for developing the system on each platform, including programming languages and/or application development tools; and
- any utility products required for each platform, if these utility products are specific to the application (utilities that can reasonably be assumed to be available given the platform need not be listed; for example, there is no need to list a "sort utility" on MVS, since one can assume such a utility is available).

g. Databases

The databases must be identified, and the database structure described. An overview of each database and/or file should be provided, and the relationship between databases or files should be described. For relational databases, the Create Table SQL code may be used to provide the details of each table. In addition, the referential integrity must be described, and the primary keys identified (either in the SQL code, or outside of it, depending on the code).

h. System Flowcharts

If available, system flowcharts showing the flow of data and/or processes should be included. (These flowcharts may be included as an Appendix.)

i. Glossary of Terms

Any application-specific terms that are commonly used should be defined. For example, the term "primary appointment" in the Corporate Personnel System (CPS) might be defined in the glossary.

j. Application Interfaces

All application interfaces, both FROM this application to other applications, and TO this application, must be identified and described. For interface files produced by the application, the job(s) producing the interface file should be identified. For interface files received from other applications, the dependencies on each interface file should be described.

k. Security and Authorization

The methods used for providing security and authorization for the application should be described. This may include, for example, database security (e.g., DB2 or Sybase database permissions), dataset security (e.g., RACF protection of non-DB2 files in MVS), or process security (e.g., RACF protection of CICS resources, or internal application-specific security to allow only select users to access certain processes, such as update access to the database).

l. General Technical Approach

Information regarding the general approach taken for coding the application should be described. For example, an MVS/CICS system may have a uniform structure for each of the online programs, and a uniform approach for the relationship between programs and CICS transaction id's. In this case, this uniform generic structure of the programs should be described.

Naming conventions should also be described. Often there are application-specific naming conventions that, once explained, can help during maintenance.

Any other overall technical approaches or uses of technology that would be helpful to know when maintaining the system should be included. (Note: this section is the most difficult to define, since it varies dramatically for each application.This section is also one of the most potentially useful and important sections. It is the responsibility of the Application Developer(s) and the Project Leader to ensure that this section provides reasonable and useful information.)

m. Ad-Hoc Access to Data

If end-users will be provided ad-hoc access to data within the application, then the method for providing this access should be described. Also, any restrictions imposed on the application due to the fact that end-users have direct access to the data should be described. For example, if modifications to specific portions of the database could affect the end-users' view of the data, then this should be noted, so that any database changes made due to system maintenance will be coordinated with end-users.

n. User Documentation

Purpose. The purpose of user documentation is to provide the functional office(s) that use the application with information on how to effectively use the application, or use the data provided by the application. The content, style, and extensiveness of user documentation will vary greatly from application to application, depending on the size, complexity, and user interaction for each application. The sections described below are guidelines for the types of information that may be contained within the user documentation. It is the responsibility of those who are writing the documentation to ensure that the information provided is useful and relevant.

Some applications include varying levels of access for different groups of users. If the difference in access is extreme, then it is probably appropriate to create separate user documentation for the different groups of users. For example, if an application allows online update for a small number of users in a single department, and allows online inquiry for a large number of users across many departments, then it would be best to have separate documentation for the update portion of the application and for the inquiry portion of the documentation. With separate documentation, each user group could be provided with only the documentation applicable to their needs.

For operational systems it may be appropriate, in some cases, to combine the user documentation for an application with the functional office's internal procedure manual. In this case, the functional office will need to either take the lead or be heavily involved in writing the user documentation, so that it will be correctly integrated with the documentation for their internal procedures.

User documentation may be provided using different formats, or a combination of formats. For example, for online update or inquiry, online help may provide sufficient information for user documentation. For other applications, a hardcopy manual may be most appropriate. For many applications, a combination of the two (online help plus a hardcopy manual) works best.

Contents of the User Documentation. The following sections include topics that would normally be found in user documentation for an application.

o. Overview of the Application
An overview of the application should be provided. This can often be taken from the Requirements Definition or System Definition document. The overview should include a description of the purpose of the application, and some information regarding the functional office or offices using the application.

This overview should also review the major processes of the system, and describe how they interrelate.

p. General Technical Environment
While functional office users should not need to know any details about the technical environment in which an application runs, it is often helpful to know a little about the technical environment. For example, for an application with a client/server architecture

where the client side operates under PC/Windows and the server side provides the database, it is useful for users to know that the client portion runs within the familiar Windows environment, but also to understand that updates to the database occur in a central, shared database, and any changes made to the data will be available immediately to other users.

q. Online Update

Users with online update capabilities in a system must have sufficient information to correctly update the database(s). Often the information required for using a particular update screen may be incorporated into text on the screen itself, or in online help available from the screen. The information available online may include, for example, valid code values for individual fields; information on which fields are required; and other edit information, such as edits involving more than one field.

In addition to information required for individual screens, some overall information on updating the database may be required, depending on the complexity of the update process. For some applications, the user may need to understand some of the interrelationships of the data in order to update the database. For example, it may be that a department must exist as an entry in the database before an employee in that department is entered into the database.

Any features that are available across screens may be described in the user documentation. For example, if there is an application-specific “copy” feature, that allows the user to copy data from an existing source before modifying the data and updating the database, then this could be described in the user documentation.

r. Batch Processing

While functional office users may not need to understand the details of any batch processing included in the system, any interdependencies between online update, data retrieval, and batch processing must be described. For example, if a financial system requires that data entered online must balance to zero before a batch process can proceed, then this needs to be described. Another example is when users are responsible for reviewing data provided on a standard report to confirm its accuracy before batch processing can proceed. Any such control points, where there are dependencies between actions taken by the functional office (whether it is online update, review of data, or manual procedures) and batch processing must be clearly documented.
In addition, any post-batch processing that is the responsibility of the functional office must be documented. For example, if the database is updated through a batch process, and an edit report with warning-level errors is produced for review by the functional office and possible subsequent data corrections, then this must be documented.

In some cases, the functional office is responsible or shares responsibility for scheduling processing for an application. This is particularly critical for operational systems where internal procedures within the functional office affect the timing of

processing within the application. If scheduling is either partly or totally the responsibility of the functional office, than this should be included in the user documentation.

s. Data Integrity

Any system controls built into the application to ensure data integrity is preserved should be described in the user documentation. Depending on the nature of the system controls, this topic may be incorporated into the sections on online update or batch processing. An example of one type of system control mechanism is a regular edit report, run to check certain aspects of the database. The user documentation should explain the purpose of such an edit report, and describe how to recognize if there may be potential data problems based on the information in the report.

t. Retrieval of Data

Retrieval of data from the system for use by the functional office(s) may be through a variety of methods, including: routine production of standard reports, online inquiry to specific data, or ad-hoc inquiry or reporting.

For standard, production reports and standard online inquiry screens, the users must be provided with information on data included in the report or on the screen. This can often be provided through the Data Element Dictionary. If some of the data displayed on a report or an inquiry screen is derived (rather than simply displaying data available in the database), and the derivation is not obvious, then these derived fields must be documented.

If there are any timing issues which may affect the data provided to users on standard reports or inquiry screens, than these issues must be explained. For example, if data in the database comes from multiple sources, which may sometimes result in data within the database having somewhat different effective dates, then this should be explained.

If functional office users have the capability to perform ad-hoc inquiry and reporting using some or all of the data within the application, then the users must have an understanding of the structure of the database and interrelationships of data as well as definitions for individual data fields. The user documentation must provide sufficient information regarding the data available for ad-hoc inquiry and reporting for the functional office users to make effective and accurate use of the data. If users have access to only one simple file (or table), then this may be straightforward to document. If users have access to multiple, complex databases, then this documentation must be far more extensive.

u. Data Element Dictionary

The data element definitions established during system development must be made available to the functional office users. This dictionary may be incorporated into a user manual, or may be made available separately.

v. Authorization Levels
If the application includes different levels of authorization for users, then each level and the capabilities available for each level must be documented. For example, if most users of an application have update access to the main database, but another level of authorization allows update access to specific tables used within the application, than these different update capabilities must be described.

w. Problem Resolution
Information on who to contact in the case of a problem arising should be available as part of the user documentation. This contact person could be someone within the functional office responsible for overall coordination for the application, or may be someone within the administrative computing department.

15. POST-IMPLEMENTATION REVIEW

a. Purpose
The purpose of the Post-Implementation review phase is to:

- Evaluate the system in terms of its success at satisfying the functional needs described in the System Definition or Requirements Definition document;
- Ensure that production processes are going smoothly;
- Ensure that procedures are in place to handle on-going maintenance of the system; and
- Evaluate the system for inclusion in any local Contingency/backup Plan.

This phase also provides an opportunity for reviewing the systems development process, and identifying possible improvements for future system development projects.

b. Description
The method for performing the post-implementation review is at the discretion of the Project Leader and the Director of the administrative computing department. The review may be informal, or may be conducted more formally, depending in part on the scope of the system.

C. DATA RETENTION AND PRIVACY

1. DATA RETENTION

a. Overview

Definition of Data Retention. Data retention refers to the length of time machine-readable files are kept within applications, either on disk or on tape medium. On most platforms, the data retention period is set when a file is created, and deletion of the file at the end of the time is handled by the operating system. In some cases, deletion of the file requires some manual intervention by the production control group or the data center operations group.

Reasons for Keeping Data. There are two different areas which affect decisions regarding how long to keep data within an application:

- retaining data for purposes directly related to running the application; and
- retaining data for functional reasons, which may include legal requirements, fiscal requirements, administrative requirements, or requirements for historical data.

The first area listed above is the responsibility of the administrative computing department, and the second is the responsibility of the functional office in compliance with University policies including those pertaining to records retention.

Description of Offsite Storage. Each of the UC campuses and OP contract with a vendor which provides storage for machine-readable data on tape medium. This storage allows the campus or OP to retain copies of selected data files in a place physically separate from the local data center. The intent is to have access to these copies of data files in the event of a disaster affecting the campus or OP data center, rendering it inoperable for a significant amount of time.

b. Data Retention for Application Processing and Recovery Purposes

Onsite Data Storage. Data retention periods established for data files within an application for the purposes of application processing and recovery must provide for the following:

- The use of the data as it is needed within the application (i.e., keep each data file for a sufficient length of time so that it is available for all production processing requiring the data, including, for example, annual as well as daily or monthly processing);

- The use of the data in interface files as it is needed within the target application (i.e., keep each interface file for a sufficient length of time so that it is available for all production processing requiring the data within the application for which the file was created); and

- The recovery of an application in the event that application-specific problems occur requiring restoration of all or part of the data and the restart of processing.

Data files residing onsite (that is, on disk or tape medium stored onsite at the local data center) must have retention periods established to satisfy the above requirements. Sufficient backup files (generally tape copies of critical data files) must exist with appropriate data retention periods assigned to allow for recovery in the event of problems occurring within the application. The Project Leader is responsible for ensuring that adequate backup files are produced as part of regular production processing, and that the data retention periods assigned to data files satisfy the above requirements.

Offsite Data Storage for Disaster Recovery. Depending on the application, it may also be necessary to provide for start-up of the application at an alternate site in the case of a disaster which renders the local data center inoperable for a significant amount of time. For applications where this is required, offsite storage of critical files is required. For these applications, the Project Leader is responsible for ensuring that offsite backup copies of critical files (i.e., all data files required to start up the system at an alternate site) are produced at reasonable points as part of regular production processing and that these backup files are assigned adequate retention periods. (In general, only one cycle of each critical file needs to be at offsite storage at any one time for disaster recovery purposes.)

The complete plan for bringing up applications at alternate sites in the event of a disaster affecting the local data center is outlined in the campus or OP contingency plan document. After any new application is put into production, the application must be evaluated, and a decision made as to whether or not it should be included in the contingency plan.

c. Data Retention for Functional Purposes

Onsite Data Storage. There are two UC policies that are directly related to the retention of records within the University:

- The UC Records Disposition policy (Business & Finance Bulletin RMP-2)

This policy defines the different purposes for retaining records (including legal, fiscal and administrative requirements), and provides guidelines for determining how long to retain records (balancing the requirement to keep records as long as they may be

needed against the requirement to avoid keeping unnecessary records and wasting resources).

- Vital Records Protection policy (Business & Finance Bulletin RMP-4)

This policy defines vital records as those records that, if lost or destroyed, would be difficult or impossible to reconstruct and the loss of which would affect a significant right of an individual, a significant right or asset of the University, or the performance of an essential function of the University. The policy also provides guidelines for the protection of vital records.

In addition, the UC Records Disposition Schedules list retention periods for specific data (generally reports or forms). For further information regarding the policies related to records retention, refer to the above Business & Finance Bulletins and the UC Records Disposition Schedules.

When establishing data retention periods for purposes other than application recovery, the requirements to be satisfied include:

- Provision for retention of historical data, in the case where ad-hoc requests (originating either from within UC or from external constituents such as the California State Legislature) for historical data are considered likely to occur;

- Compliance with legal requirements for retention of files (e.g., the Federal government requires that tax records for employees must be kept for a specified number of years); and

- Fulfillment of all legal, fiscal and administrative requirements of the University and interested external agencies (see the above policies regarding records retention for further information on these requirements).

The above requirements must be taken into consideration when establishing data retention periods for datasets within an application. While data retention periods must be sufficient to satisfy the above requirements, excessively long data retention periods must also be avoided, to avoid unnecessary use of machine and staff time resources.

Data retention periods must be established for every application dataset when the system is implemented in production. The Project Leader is responsible for ensuring that data retention periods have been assigned for all data files within an application. However, the primary responsibility for determining the length of time that machine readable files must be retained for functional, historical or legal purposes (i.e., for purposes other than application processing and recovery) rests with the functional office that owns the data, or is the office of record.

Offsite Data Storage for Data Preservation. There may be cases where offsite storage of copies of specific data files is appropriate, even where they are not needed for disaster recovery. For example, if a machine readable file is deemed a vital record for the University in accordance with the Vital Records Protection policy in Business & Finance Bulletin RMP-4, then it may be appropriate to periodically copy the file and send the copy to offsite storage. Another example is where retaining historical data is considered critical, in which case select historical files may be sent offsite. The functional office has primary responsibility for determining the need for offsite storage for data for purposes other than disaster recovery.

d. Summary

A table summarizing the reasons for retaining data within applications, both for data stored onsite in the local data center and data stored offsite in case of disaster, is presented below.

	Application Processing and Recovery	**Functional Requirements**
Onsite Storage	Keep data long enough for normal processing, and for recovery from application-specific problems	Keep data long enough to satisfy requirements for access to historical data, and legal, fiscal and administrative requirements
Offsite Storage	Keep copies of files offsite which are needed for starting up the application at an alternate site, in case a disaster renders the local data center inoperable for a significant amount of time (see local contingency plan)	Keep copies of files offsite which are considered critical to the University, in case a disaster damages data files stored onsite within the local data center

2. PRIVACY

a. Introduction

Privacy, in this context, is defined as the right of individuals to know the personal information that is maintained about them and to control the ways in which that information is collected, disseminated, maintained and used. In response to widespread automation in the 1970's, and the accompanying perception of potential risks to individual privacy, regulations were enacted at the federal and state levels to protect individuals' right to privacy. In turn, the University issued several policies which are designed to ensure compliance with these privacy laws.

This section outlines the six privacy requirements that must be met in the development and operation of computer systems that collect and maintain personal information on individuals:

1. The CART requirement;
2. Disclosure;
3. Access;
4. Amendment;
5. Identification of Authorized Users; and
6. Notices.

Some examples of personal data likely to be encountered in system design are: Social Security number; birth date; citizenship; ethnic, veteran or disability status; home address or telephone number; and income tax withholding. This section is primarily concerned with what analysts and programmers must do in designing systems to comply with the legislation and the ensuing University policies on privacy. It does not, therefore, quote nor refer to the actual laws and policies. Business and Finance Bulletin (BFB) RMP-8, *Legal Requirements on Privacy of and Access to Information*, is the University policy document which cites and sets forth the implementation of applicable laws concerning privacy.

b. Privacy Requirements

The CART requirement. Privacy legislation requires that personal data maintained on an individual be **C**omplete, **A**ccurate, **R**elevant, and **T**imely. The Project Leader, in consultation with the campus or OP data administration group, should consider the following in the design of the system:

- **Completeness** Personal data elements must be complete, as omissions or incomplete data may be harmful to the individual. For example, a record of a downward classification of employment status, to be complete, should contain an explanation of the reclassification.

- **Accuracy** Personal data elements must be correct because, again, errors may be harmful to the individual. For example, an algorithm for determining ethnicity from other data elements should not be used since the resulting data may not be accurate.

- **Relevance** Personal data elements must be relevant to the functions of the system. Thus, the personal data elements collected should be necessary to the current information needs of the appropriate functional office.

- **Timeliness** Personal data elements must be kept up-to-date, that is, changed as the information pertaining to the individual changes. In addition, data elements should not be retained beyond their useful life nor beyond the retention period required by law or by University policy. [see also section 3.1 Data Retention]

Disclosure. Privacy legislation and University policy specify several regulations concerning the disclosure of personal information. ("Disclosure" is defined here as the dissemination of personal information in any form to anyone other than the person to whom it applies.) Appropriate measures should be taken to prevent unauthorized use or disclosure of personal information. This can be satisfied through an appropriate combination of password protection, control over the issuance of computing accounts, and maintenance of logs of disclosure of personally-identifiable information. If a log is used, it must provide the following information:

- The name, title and business address of the person to whom the information is disclosed.

- The date of disclosure.

- The data elements disclosed.

- The purpose of the disclosure.

Access. Privacy legislation gives individuals the right to review the personal information maintained on them. Thus, systems containing personal information must include a method for extracting an individual's record and all information relating to that individual. This means the extraction method must also provide a mechanism to convert (to a readable form) any information that is in coded form.

Amendment. Privacy legislation also gives individuals the right to request changes to the personal information maintained on them. Thus, in the operation of the system, there must be a means for changing any or all data elements linked to an individual.

Identification of Authorized Users. To comply with privacy regulations, personal data should only be disclosed to, or amended by, authorized users. Therefore, there must be a procedure or mechanism for identifying authorized users --thus preventing disclosure of personal information to anyone else.

Notices*.* To comply with legislative requirements, two types of notices must be prepared: system notices and individual notices. Each of these is described below.

c. System Notice

A system notice must be prepared for each system that maintains personal information. Its purpose is threefold:

1. To provide management and users with a complete but concise picture of the system's effect on privacy;

2. To provide management and users with the information needed to compile the individual notice(s); and

3. To inform users of their obligation not to make unauthorized disclosures.

The system notice must contain:

- The name of the system.
- The name and address of the unit(s), department(s) or division(s) maintaining the records.
- The categories and approximate numbers of individuals on whom personal information will be maintained.
- A statement on whether or not the Social Security number will be collected and, if so, why.
- The types and the actual data elements of personal information collected.
- The categories of individuals or agencies who will be using the personal information.
- The purpose and use of the personal information and, if different, the function(s) of the system.
- The source(s) of the information.
- If the information is collected from the individual, the consequences of a nonresponse to each data element of personal information requested.
- The title, business address, and business telephone number of the University officer who is responsible for the information contents of the system.
- The title(s) of the authorized users.

- The procedures for handling access, amendment, disclosure, storage, retention, and disposal of the personal information or the chapter and names of the systems documentation where such procedures are described.

- The authority for collection of the personal information (e.g.,University policy, Federal or State regulations, etc.)

The Project Leader is responsible for preparation of the system notice, with input and review by the data administration group.

The system notice will be retained in the system documentation with a copy to the data administration group.

d. Individual Notice

Any form that is used to collect personal information from an individual must contain or be accompanied by a notice which 1) describes how and by whom the information will be used and 2) explains the individual's right to review his or her record and to request changes in the data maintained.

The collection of personal information from students is governed by different laws and, in some cases, may not require the individual notice that must be provided to non-students. However, the distinguishing of student from non-student, at different points in the academic career, is sometimes a difficult task. In the development of systems involving students or potential students, where personal information is being collected, the Project Leader should seek guidance from the functional office as to whether or not an individual notice is required.

The notice to individuals must contain:

- The name of the administrative unit that is requesting the information.

- The title, address, and telephone number of the functional University officer who is responsible for the information content of the system of records.

- The authority, whether granted by statute, regulation, or University policy, that sanctions the collection and maintenance of the information.

- For each item of information, whether submission of such information is mandatory or voluntary.

- The consequences, if any, of not providing all or any part of the information requested.

- The principal purpose(s) for which the information is to be used.

- Any known or foreseeable transfers outside the University.

- A statement of the individual's right of access to those records containing personal information on him or her.

Example A which follows this section provides an example of a privacy notification meeting the above requirements. Please note that it contains references, in italics, to the specific system for which it was created. If this notification is used as a model, the italicized phrases would need to be tailored to the specific system for which it is intended.

In addition, if the Social Security number is being collected from anyone, including students, federal law requires that the collection form be accompanied by a second notice, which identifies the authority for collection and use of the Social Security number, and states whether disclosure of one's Social Security number is mandatory or voluntary. The actual wording of the social security number notice is specified by University policy. Non-system specific examples of Federal privacy notices can be found in the University policy document, Business and Finance Bulletin (BFB) RMP-8, *Legal Requirements on Privacy of and Access to Information.*

The responsibility for preparation of individual notices rests with the Project Leader with input and review by the data administration group.

EXAMPLE A

State Privacy Notice

> "The State of California Information Practices Act of 1977 (effective July 1, 1978) requires the University to provide the following information to individuals who are asked to supply personal information about themselves.
>
> The principal purpose for requesting the information on this form is *to report payment for income tax purposes to Federal and State government* as applicable. *University policy and State and Federal statutes* authorize the maintenance of this information.
>
> Furnishing all information requested on this form is mandatory- failure to provide such information will delay or may even prevent *the payment* for which the form is being filled out. Information furnished on this form is used by University departments for *non-payroll payments* and may be transmitted to the *State and Federal* governments as required by law.
>
> Individuals have the right to access to this record as it pertains to themselves. *Campus Accounting Officers* are responsible for maintaining the information contained on this form."

(For Federal Privacy Notice examples, see BFB RMP-8)

D. CHANGE MANAGEMENT AND MAINTENANCE STANDARDS

1. INTRODUCTION

Campuses shall establish standards and procedures for changes to application software in conformance with these standards. Campus standards should be developed in consideration of current best practices in information technology and provide for the items identified below. Campuses may default to use of these standards, if so desired.

Procedures for handling changes to application software should reflect the size, complexity and risk of the application and change, as outlined below.

2. MAINTENANCE STANDARDS FOR SMALL MODIFICATIONS TO LOW RISK APPLICATIONS

For changes to Small/Simple, Low Risk systems and application software changes that do not exceed 1 month of effort (IT staff time) to complete or more than $15,000 in total cost, only the following apply:

- The request for change must be logged and retained. E mail can serve these purposes;
- Program changes must be documented and dated; and
- Additional local campus maintenance standards for small projects, if any, must be applied.

3. MAINTENANCE STANDARDS FOR MODIFICATIONS REQUIRING MORE THAN ONE MONTH OR $15,000 TO COMPLETE OR NON-LOW RISK APPLICATIONS

a. Source Control

When changes are made to existing system and application software, control of source code requires:

- Procedures for segregating production and test code;
- Procedures for checking source code in and out;
- Procedures for testing the changes in stages of increasing impact;

- Standards for identifying the changes that were made (including the date of the change and the name of programmer);
- Procedures for notifying others of the changes;
- Procedures for user review of functionality and presentation (see 4.3.2);
- Criteria for installing the modified source code in production mode.

b. User Participation

Campus maintenance standards should provide for the participation of users at appropriate stages of the change process. For both enhancements and program fixes, user participation should be documented so it is clear that changes were not made at the discretion of the technical staff alone. Such documentation may take the form of a log or other suitable format, including electronic mail. When changes do not affect the functionality of an application (e.g., a calculation) and are done for performance, security, reliability or performance reasons, user participation is not necessary.

Standards for documentation of user participation should incorporate procedures for the following items:

- Receipt of requests for enhancements or identification of system problems;
- Identification of issues associated with the proposed changes, and discussion and resolution of those issues;
- Priorities for proposed changes, and discussion and resolution of differences regarding priorities;
- If appropriate, cost and time estimates for proposed changes;
- User review and acceptance of testing;
- User review and acceptance of completion of the changes and readiness for production.

4. MAINTENANCE STANDARDS FOR LARGE MODIFICATIONS

For changes to Large/Complex, High Risk systems and for application software changes that exceed 12 months of effort (staff time) to complete or more than $150,000 in cost, the standards in Section 2, Phases of Systems Development apply.

IV. COMPLIANCE/RESPONSIBILITIES

A. Functional Office and Administrative Computing Department Responsibilities
One of the first tasks of any system development project is to identify the participating departments, and clarify responsibilities. The two obvious participants are the functional office identified as the key user of the application (assuming there is an identifiable primary user), and the administrative computing department, if the campus or OP administrative computing department is going to develop or install the system. The possible split of responsibilities between the functional office (or offices) and the administrative computing department varies along a continuum from the administrative computing department taking primary responsibility throughout the project development effort, to the functional office taking primary responsibility with the administrative computing department serving in the role of a consultant. While many different models are viable, it is critical that the split of responsibilities is clearly understood from the beginning of the project. Assignment of responsibility for systems development must be accompanied by a corresponding commitment of staff time from each participating office.

As responsibilities are being defined, it is strongly recommended that a single entity (e.g., one manager or executive in the functional office, or possibly a task group or steering committee), is assigned clear ownership or sponsorship of the project. This person or group will be responsible for making key decisions, such as determining whether to increase the scope or budget of the project during development if changes arise, or to stay within the originally-allocated budget and schedule and forego the proposed changes.

It is also recommended that procedures for handling changes that may arise during the systems development process be agreed upon as part of the determination of responsibilities.

The level of formality by which responsibilities and the method for handling changes are delineated may vary widely, in some cases consisting of an informal agreement while in other cases the decisions may be codified in a formal “governance contract” that all parties adhere to throughout the development process.

In the case where the administrative computing department will lead the project through most or all of the development phases, it remains critical that the functional office is heavily involved, particularly during the earlier stages of the process (i.e., Project Proposal, System Definition or Requirements Definition, and Requests for Information and Proposals for Package purchases), for the project to be successful. It is also critical that senior-level management within the functional office(s) are involved at pivotal points in the project development process to ensure that the objectives of the project have been accurately identified and are being fulfilled.

The general question of who will be responsible for the on-going support for a system should be addressed along with the determination of responsibilities during the development process. Responsibility for on-going support of required hardware (including servers, workstations, printers, and networking) and maintenance of the software should be addressed.

Clarifying the roles and responsibilities for a project is particularly important if the project may involve the purchase and installation of a vendor package. At the point when a vendor is selected, there is yet another group to coordinate with, making project management even more complex. It is also critical that the role and responsibilities of the vendor, both during system installation and for on-going support, are clearly delineated in the contract (see section 2.8 Vendor Contract and Installation Plan).

B. Steering Committee and Project Team

The organizational structure for handling projects can vary from campus to campus and from project to project. As noted earlier, ownership of a project may be assigned to a Steering Committee. For larger projects, it is often effective to have a high-level steering committee composed of senior-level management from functional offices, the administrative computing department and, if appropriate (e.g., for financial systems), internal audit. The steering committee provides overall strategic direction and decisions for the project, including establishing the business case for system development and the planning principles, defining the overall scope of the project, communicating with all stakeholders in the system, developing a decision framework, and providing management oversight throughout development and implementation. A Project Leader is appointed who will be responsible for the day-to-day management and oversight of the project. For larger projects, the Project Leader will lead a project team, made up of representatives from the functional offices and the administrative computing department. (This project team is in addition to the technical design and programming team assigned to the project within the administrative computing department.) The project team is responsible for performing much of the work during the analysis phase (Project Proposal, System Definition or Requirements Definition, and Request for Information, Request for Proposal, and Feasibility Study for vendor package solutions), and continued project management throughout the prototyping, design or installation phases and final implementation. In some cases, the project team may continue after implementation to assist with on-going maintenance issues.

C. Internal Audit

Internal Audit's involvement during high risk systems development is important. Internal Audit can provide advice and assistance in the identification and development of adequate system and process controls. Accordingly, Internal Audit must be notified of all systems development projects early in the development process (normally at the Project Proposal phase). Based upon the risk and scope of the project, Internal Audit

will establish a plan with the Project Leader that details Internal Audit's involvement during each phase of the project.

Internal Audit may elect to participate in the development process for new systems or major enhancements to existing systems in the case where the project is not considered high risk. The level of involvement for non-high risk applications will be determined by Internal Audit.

D. Office of the President Review

1. Campus Responsibility

For major, campus-wide applications the Project Plan highlighting the major milestones established for the project must be forwarded to the Associate Vice President for Information Resources and Communications (IR&C) at the Office of the President, for information and review. Upon reaching major milestones in the project, an updated Project Plan showing progress measured against major milestones should be forwarded to the Associate Vice President for IR&C. Office of the President review is only necessary for major campus-wide applications, such as a new, campus-wide financial system**.** The review by Office of the President will help assure the uninterrupted flow of corporate data from campus systems, and will help assure continued smooth University operations.

2. Implementation Procedures

Applications meeting the criterion of "major, campus-wide" are those which supply information to the corporate financial (e.g., campus budget, general ledger/financial, chart of accounts, etc.), the corporate contracts and grants, the corporate student or the corporate personnel (except base Payroll/Personnel) systems.

The Project Plan should be presented in a form which adheres to the steps in these Systems Development Standards and which indicates major milestones. Preferably the plan should be the same plan used at the campus level. Additionally, the Associate Vice President should be invited to participate in major campus project milestone reviews. It should be noted that the Associate Vice President will coordinate reviews of major campus applications with UCOP functional officers (in Business and Finance, Student Services, etc., as appropriate) and with the University Auditor.

E. Hardware and Software Acquisition by Functional Offices

Some sites may have specific policies regarding acquisition of hardware and software. For example, at Office of the President, the "Policies and Procedures for Acquisition by Systemwide Administration of Computers, Related Equipment, and Software" (dated January 18, 1981) must be followed and appropriate approvals obtained.

V. PROCEDURES

None

VI. RELATED INFORMATION

None

VII. FREQUENTLY ASKED QUESTIONS

None

VIII. REVISION HISTORY

This policy was last updated on May 18, 2001, and reformatted into the standard University of California policy template on June 15, 2012.