

TABLE OF CONTENTS

1. PREFACE	2
2. PURPOSE OF INTRODUCTORY GUIDE.....	2
3. GENERAL OVERVIEW OF MODEL.....	2
4. MODEL STRUCTURE	4
4.1 PHYSICAL ORGANIZATION.....	4
4.2 LOGICAL MODEL VIEWS	5
4.3 THE RELATIONSHIP BETWEEN THE SOFTWARE LIFE CYCLE AND SELECTED PRACTICES.....	6
5. PRACTICE OVERVIEW	6
5.1 SOFTWARE PROJECT MANAGEMENT	7
5.2 SOFTWARE TESTING	10
5.3 SOFTWARE OUTSOURCING	14
5.4 SOFTWARE QUALITY ASSURANCE	17
5.5 USER REQUIREMENTS MANAGEMENT	21
5.6 POST IMPLEMENTATION SUPPORT	24
5.7 SOFTWARE CHANGE CONTROL.....	26
6. GLOSSARY OF TERMS	30

1. PREFACE

The HK Software Quality Assurance Model provides the standard for local software organizations (independent or internal; large or small) to:

- Meet basic software quality requirements;
- Improve on software quality practices;
- Use as a bridge to achieve other international standards;
- Assess and certify them to a specific level of software quality conformance.

2. PURPOSE OF INTRODUCTORY GUIDE

This *introductory guide* provides an overview of the model by introducing the reader to its various components.

As part of the overall package, the introductory guide should be read by persons who have no knowledge of the model and:

- are interested in to know about the model;
- who are deciding on whether to adopt the HK Software Quality Assurance Model as the standard to improve its organization's software practices;
- require a starting point towards understanding the rest of the model.

3. GENERAL OVERVIEW OF MODEL

The HK Software Quality Assurance Model consists of:

- a *description* of seven of the most essential practices in a software organization included in:
 - a *Procedure Handbook* that details the procedures that are required to be followed for each of the seven practices;
 - a *Guidebook* that provides guidelines, templates and tools that should be used in adhering to procedures.

As part of the overall model package, the following supporting pieces of documentation are also provided:

- an *Introductory guide* to serve as an introduction to the model, the seven practices and a reference to the other supporting pieces of documentation in the overall package.
- an *Implementation & Improvement Guide* describing the steps that should be followed to
 - install and use these practices in a software organization;
 - enable a software organization to reach a higher degree of software quality conformance in adherence to the model.
- an *Assessment Tool* to enable a software organization to effectively evaluate its degree of software quality conformance in adherence to the model.

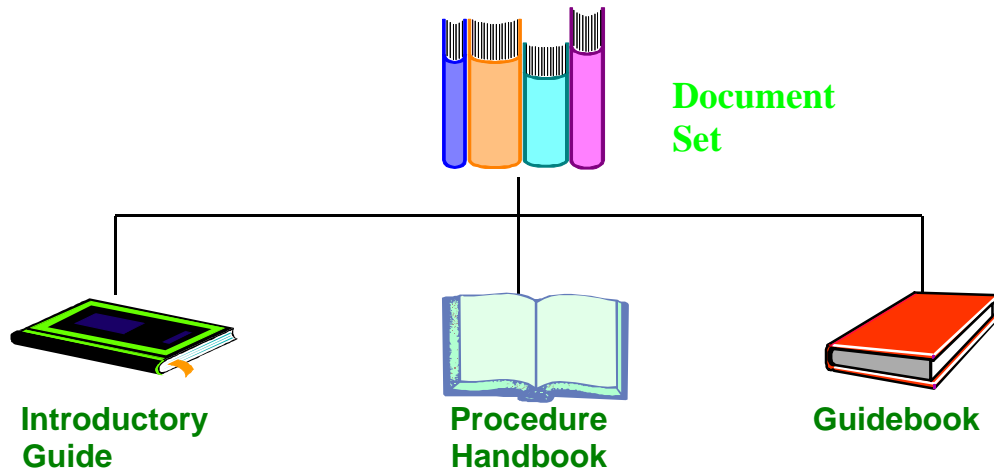
The seven practices that form the basis of the HK Software Quality Assurance Model are:

- a) *Software Project Management*-the process of planning, organizing, staffing, monitoring, controlling and leading a software project.
- b) *Software Testing*-the process of evaluating a system (where the software resides to:
 - confirm that the system satisfies specified requirements;
 - identify and correct defects in the system before implementation.
- c) *Software Outsourcing*-the process that involves:
 - Establishing a software outsourcing contract (SOC);
 - Selecting contractor(s) to fulfill the terms of the SOC;
 - Managing contractor(s) in accordance to the terms of the SOC;
 - Reviewing and auditing contractor performance based on results achieved;
 - Accepting the software product and/or service into production when it has been fully tested.
- d) *Software Quality Assurance* – a planned and systematic pattern of all actions necessary to provide adequate confidence that the item, product or service, and conform to established customer and technical requirements.
- e) *User Requirements Management* – the process of discovering, understanding, negotiating, documenting, validating and managing a set of requirements for a computer-based system.
- f) *Post Implementation Support* – the process of providing operations and maintenance activities needed to use the software effectively after it has been delivered.
- g) *Change Control*-the process of evaluating proposed changes to software configuration items and coordinating the implementation of approved changes to ensure the integrity of the software remains intact and uncompromised.

An overview of each of these practices is provided in Section 5 below.

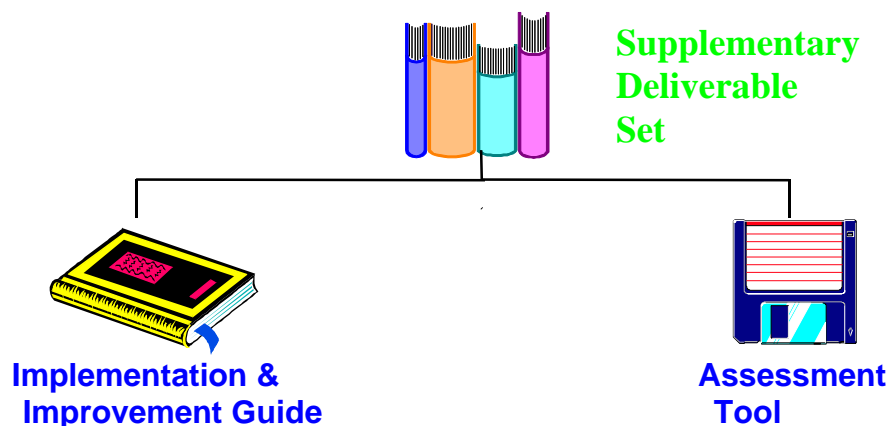
4. MODEL STRUCTURE

4.1 PHYSICAL ORGANIZATION



Physically, the model is presented as a set of documents, the first set of which is illustrated in the above diagram and includes:

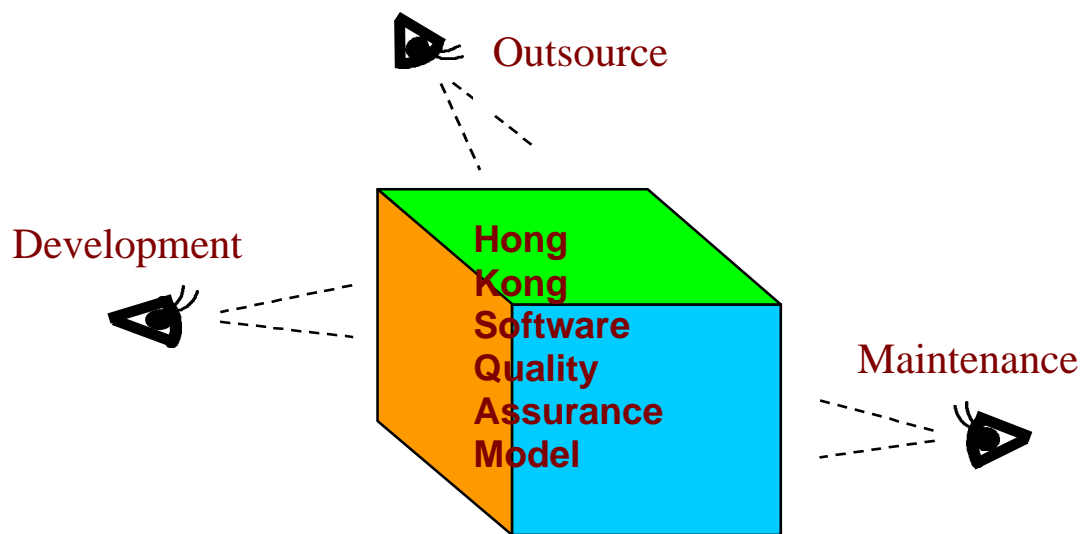
- 4.1.1 an *introductory guide* to serve as an introduction to the model, the seven practices and a reference to the other supporting pieces of documentation in the overall package.
- 4.1.2 a *Procedure Handbook* that details the procedures that are required to be followed for each of the seven practices;
- 4.1.3 a *Guidebook* that provides guidelines, templates and tools that should be used in adhering to procedures.



A supplementary set of deliverables that comes with the model is illustrated in the above diagram and consists of:

- 4.1.4 an *implementation and improvement guide* describing the steps that should be followed to install and use these practices to reach a higher degree of software quality conformance in a software organization.
- 4.1.5 an *assessment tool* to enable a software organization to effectively evaluate its degree of software quality conformance in adherence to the model.

4.2 LOGICAL MODEL VIEWS



The seven practices that are described in the model were selected on the basis that all local software organizations can logically be viewed as being classified into three broad categories:

- 4.2.1 Development practices;
- 4.2.2 Maintenance practices;
- 4.2.3 Outsourcing practices.

The above diagram illustrates this fact.

4.3 THE RELATIONSHIP BETWEEN THE SOFTWARE LIFE CYCLE AND SELECTED PRACTICES



The above diagram illustrates how each of the selected practices are related to the software life cycle. Essentially, User Requirements Management, Software Testing and Post Implementation Support are specific phases of the Software Life Cycle whereas Software Project Management, Software Outsourcing, Software Quality Assurance and Change Control provide support to the activities carried during the software life cycle.

5. PRACTICE OVERVIEW

For consistency, each practice is described under the heading of Definition, Objectives, Governing Policy, Primary Responsibility and Process Overview Diagram.

Details of procedures for performing each of the practices are provided in the *Procedure Handbook*.

5.1 SOFTWARE PROJECT MANAGEMENT

5.1.1 DEFINITION

Software Project Management is the process of planning, organizing, staffing, monitoring, controlling and leading a project. In addition, the management of a software project has to consider the human, system and hardware aspects as well since one has to live with the other.

5.1.2 OBJECTIVES

The major objectives of software project management are:

5.1.2.1 Effective Project planning, scheduling and budgeting.

5.1.2.2 To minimize project risks

5.1.2.3 To measure project processes and products for improvement to current and future projects.

5.1.2.4 To provide for effective project tracking and progress reporting that ensures adequate visibility throughout the project life cycle and for corrective actions to be taken as necessary.

5.1.2.5 To ensure that the project deliverables are delivered on time under project constraints.

5.1.2.6 To meet business objectives and objectives for the project.

5.1.3 GOVERNING POLICY

Every software project shall have a Project Manager from the software organization.

5.1.4 PRIMARY RESPONSIBILITY

5.1.4.1 Software Project Manager (SPM)

The SPM plays a crucial role on a software project. He or she is expected:

- a) To assume overall responsibilities for delivering the software product on time, within budget and with the required quality, ie, conformance to requirements.
- b) To define organizational roles and allocate staff to serve in these roles.

- c) To provide guidance and direction to the project team in a timely manner.
- d) To serve as the interface to people and groups within and outside of the parent organization who have a stake, interest or involvement on the project.
- e) To understand and make major decisions on the technical and non-technical aspects of the project
- f) To monitor the project by measuring progress.
- g) To report progress to initiators and senior management.

To fulfill these responsibilities, the SPM should possess adequate project specific and people related skills, knowledge and experience in a number of major fields including:

- Supervision of a project team;
- Problem-solving;
- Decision-making;
- Project control.

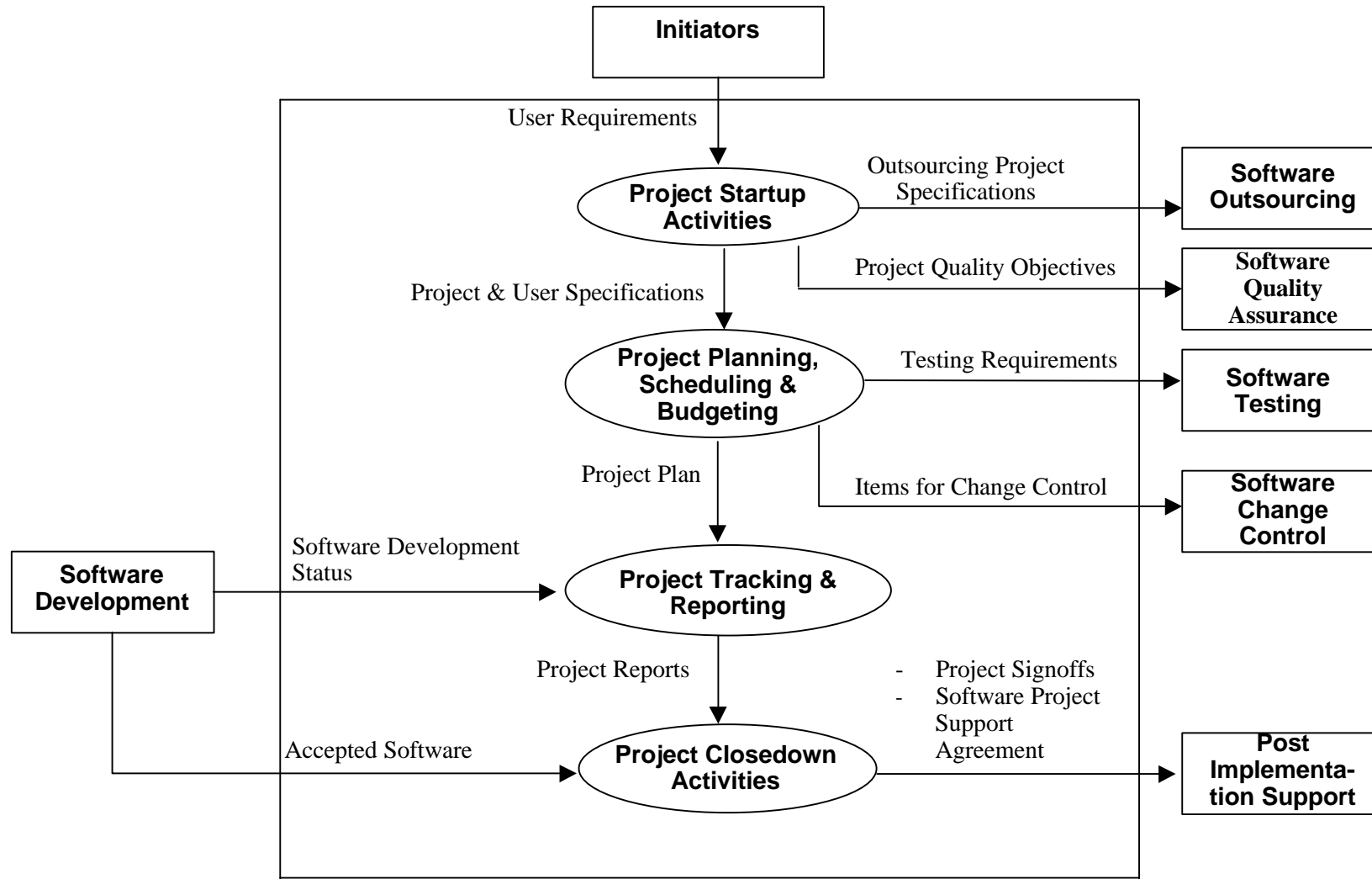
5.1.4.2 Project Steering Committee & Project Assurance

For large projects, other roles are required to be set up including:

- a) A Project Steering Committee (PSC) that is responsible for the overall project guidance and decision making. The PSC normally would comprise representatives from executive, user and development management.
- b) A Software Quality Assurance (SQA) team that is responsible for assuring project as well as product quality (refer SQA practice of the model).
- c) For additional guidance on the above, refer also to:
 - Section 6.1 of Practitioner Guide on Project Management;
 - Points 1, 2 and 3 in Sections 6.1.3 and 6.1.4 of Guidelines on Flexibility in Adopting Methodologies.

These guidelines are available in the Information Technology Services Department (ITSD) of the government.

5.1.5 Project Management Process Overview



5.2 SOFTWARE TESTING

5.2.1 DEFINITION

Software Testing is the process of exercising or evaluating a system or component, (where the software resides and under specified conditions) by manual or automated means to:

- confirm that it satisfies specified requirements;
- identifies differences between expected and actual results.

Compared with other verification techniques, testing is the most direct because it executes the software.

Software Testing is a major component of software life cycle activities of which there are three main kinds namely:

5.2.1.1 Unit Test

Unit testing verifies the design and implementation of all the components from the lowest level including:

- a) Verifying that the module is doing what it is supposed to do i.e. 'black box testing' and it is doing it in the way it was intended i.e. 'white box' testing.
- b) Identifying the most probable paths through a module and designing the tests to ensure these paths are executed.
- c) Ensuring that every module statement shall be successfully executed at least once.

5.2.1.2 System Test

The process of system testing is best described under two main areas:

- Integration and integration testing;
 - System testing.
- a) Integration is the process of building a software system by combining components into a working entity as specified during the Design (DG) phase of the Software Life Cycle (SLC). Integration testing is done on major components of the system after they have been assembled. Integration testing should verify that these major components interface correctly including data exchange and control flow portions of the interface.
 - b) System testing is the process of testing an integrated software system. System

testing can be carried out in the development or target environment or a combination.

System testing shall verify compliance with system objectives to include:

- End-to-end system tests i.e. passing data into the system, correctly processing and outputting it.
- Practice for user acceptance tests i.e. verification of functionality and that user requirements will be met.
- Stress tests i.e. measurement of performance limits.
- Preliminary estimation of reliability and maintainability.

5.2.1.3 User Acceptance Test

User acceptance tests validate the software that it demonstrates the capabilities (functional and operational) of the software in its operational environment.

User acceptance tests should be based on the user requirements as stated in the User Requirements Document (URD).

User acceptance tests verify the Software User Manual (SUM) and do not replace the need for system training.

5.2.2 OBJECTIVES

The primary objectives of software testing are to:

- 4.2.3.1 To uncover defects and ensure that the defects are corrected before going live with the software. Defects are any anomalies or deviations from specified user requirements.
- 4.2.3.2 To ensure that the software goes through an effective process for reducing the number of defects as each testing phase is completed, leading to removal of all known defects before going into production.
- 4.2.3.3 To provide confidence to developers and users that the developed software will run in the test or operating environment as specified in user requirements.
- 4.2.3.4 To provide a first look of and training on the system for developers, support staff and users.

5.2.3 GOVERNING POLICY

5.2.3.1 All developed software will be tested to be free from all known defects before release to the production environment.

5.2.3.2 Testing shall involve users of the system or component and will be in the target or simulated environment.

5.2.4 PRIMARY RESPONSIBILITY

The responsibilities for testing vary in accordance to the type of testing as follows:

5.2.4.1 Unit Test

a) Software developers or developing teams that produced the modules, are primarily responsible for unit testing including taking any corrective action on defects uncovered.

5.2.4.2 System Test

a) Software developers, mainly project managers and systems analysts, are primarily responsible for Integration, Integration Testing and Systems Testing including taking any corrective action on defects uncovered.

b) Software developers clarify issues with users.

c) Users are involved in obtaining 'first look and feel' of the system.

d) Operations and maintenance staff may be called upon to observe the behaviour of the system in a target or simulated environment and in fine-tuning activities.

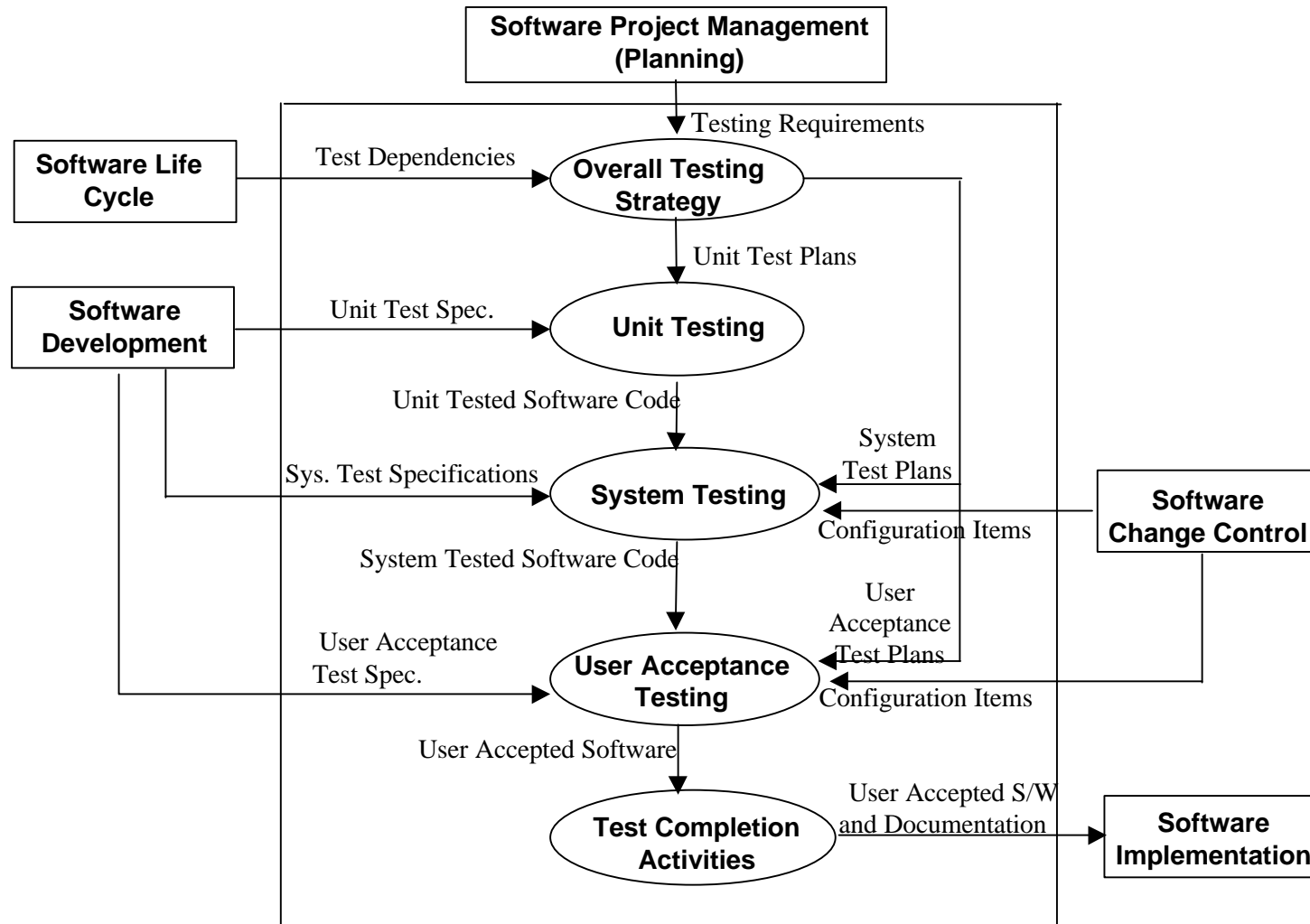
5.2.4.3 User Acceptance Test

a) Users, mainly represented by end-users, user management or system owner, are primarily responsible for User Acceptance Testing including signing off to signify acceptance of tested software, training in the use of the new system and preparation for the use of the new system in the live business environment.

b) Developers, mainly Project Managers or Systems Analysts, provide support in addressing systems related issues and taking corrective action on defects uncovered.

c) Operations and maintenance staff are involved in observing the behaviour of the system in a targeted or simulated environment. They are also involved in any necessary activities on fine-tuning the software leading to confirmation to accept the system in a production environment.

5.2.5 SOFTWARE TESTING PROCESS OVERVIEW



5.3 SOFTWARE OUTSOURCING

5.3.1 DEFINITION

Software Outsourcing (SO) is the process that involves:

- 5.3.1.1 Establishing a software outsourcing contract (SOC);
- 5.3.1.2 Selecting contractor(s) to fulfill the terms of the SOC;
- 5.3.1.3 Managing contractor(s) in accordance to the terms of the SOC;
- 5.3.1.4 Reviewing and auditing contractor performance based on results achieved;
- 5.3.1.5 Accepting the software product and/or service into the production or operating environment when it has been fully tested.

The process allows for the flexibility of maintenance activities being carried out in house to the client or within the supplier's organization.

5.3.2 OBJECTIVES

The major objectives of the software outsourcing practice are:

- 5.3.2.1 To enable effective SO contract planning, scheduling and budgeting;
- 5.3.2.2 To minimize SO contract risks;
- 5.3.2.3 To provide for effective SO contract tracking and progress reporting that ensures adequate visibility throughout the SO contract life cycle and for corrective actions to be taken as necessary;
- 5.3.2.4 To ensure that the software product and/or service is fully verified and validated before acceptance.

5.3.3 GOVERNING POLICY

Every SO contract shall have a Software Contract Manager from the software organization.

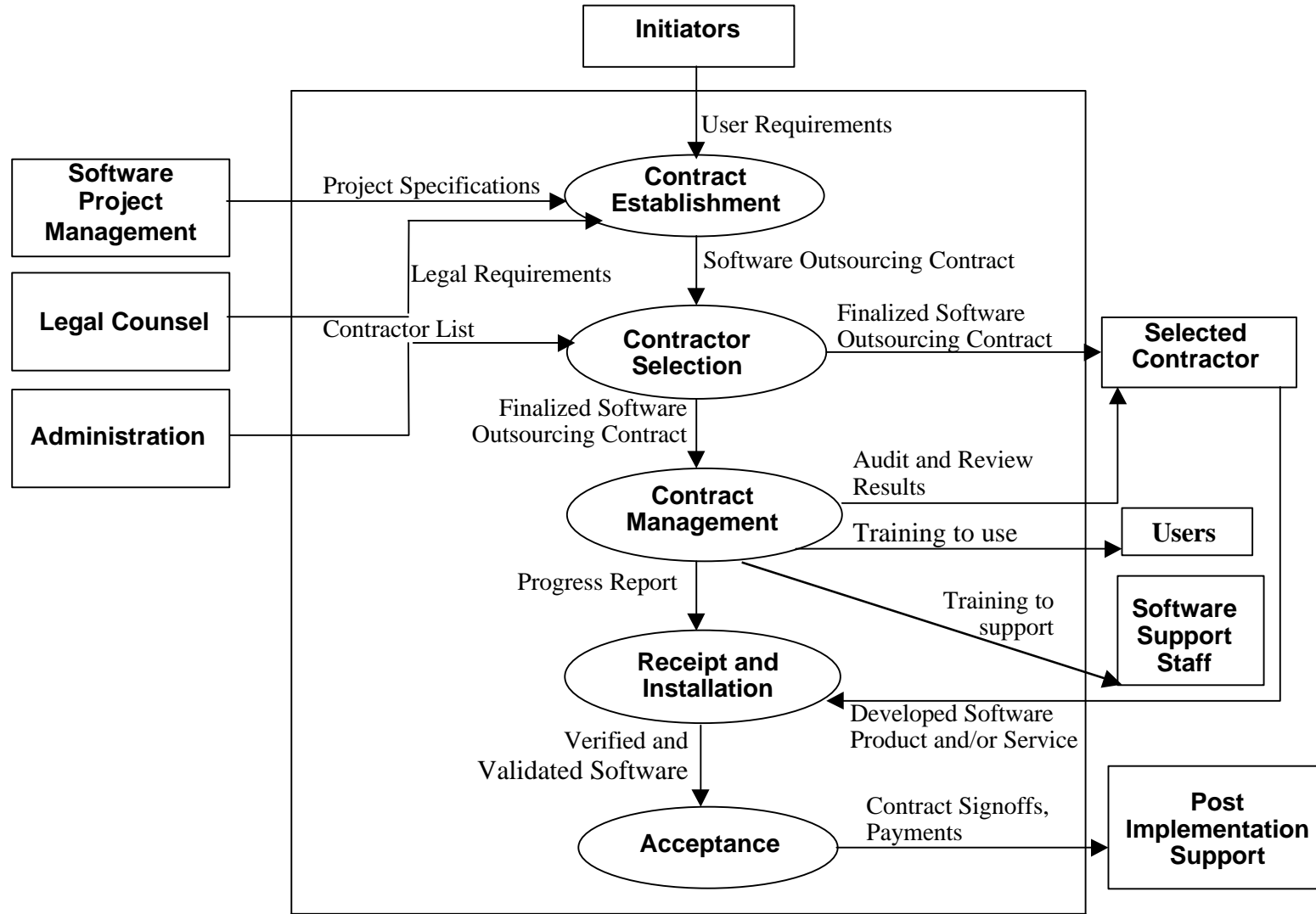
5.3.4 PRIMARY RESPONSIBILITY

5.3.4.1 Software Contract Manager (SCM)

The SCM plays a crucial role in a SO contract. He or she is expected:

- a) To facilitate the establishment of the SOC;
- b) To facilitate the selection of the contractor to fulfill the terms and conditions of the SOC;
- c) To assume overall responsibilities for ensuring that the software product and/or service is delivered by the contractor on time, within budget and with the required quality;
- d) To serve as the interface to people and groups between the parent and contractor organizations within and outside of the parent organization who have a stake, interest or involvement on the SOC;
- e) To monitor progress of work carried out by the contractor and in preparation to receive the software product and/or service, reporting progress to initiators and senior management;
- f) To facilitate verification and validation activities before acceptance of the software product and/or service into the production or operating environment.
- g) To fulfill these responsibilities, the SCM should possess adequate contract management and people skills, knowledge and experience in a number of major fields including:
 - Negotiation;
 - Interpersonal;
 - Problem-solving;
 - Decision-making;
 - Project control.

5.3.5 SOFTWARE OUTSOURCING PROCESS OVERVIEW



5.4 SOFTWARE QUALITY ASSURANCE

5.4.1 DEFINITION

Software Quality Assurance (SQA) is a planned and systematic pattern of all actions necessary to provide adequate confidence that the item, product or service conforms to established customer and technical requirements.

The SQA function provides management with a degree of confidence that an independent, technically trained group is monitoring the goals, methods and performance of applications from the beginning of the software project, relieving software management of personally performing this function.

5.4.2 OBJECTIVES

5.4.2.1 The major objectives of an SQA function are to develop the minimum requirements and procedures to ensure:

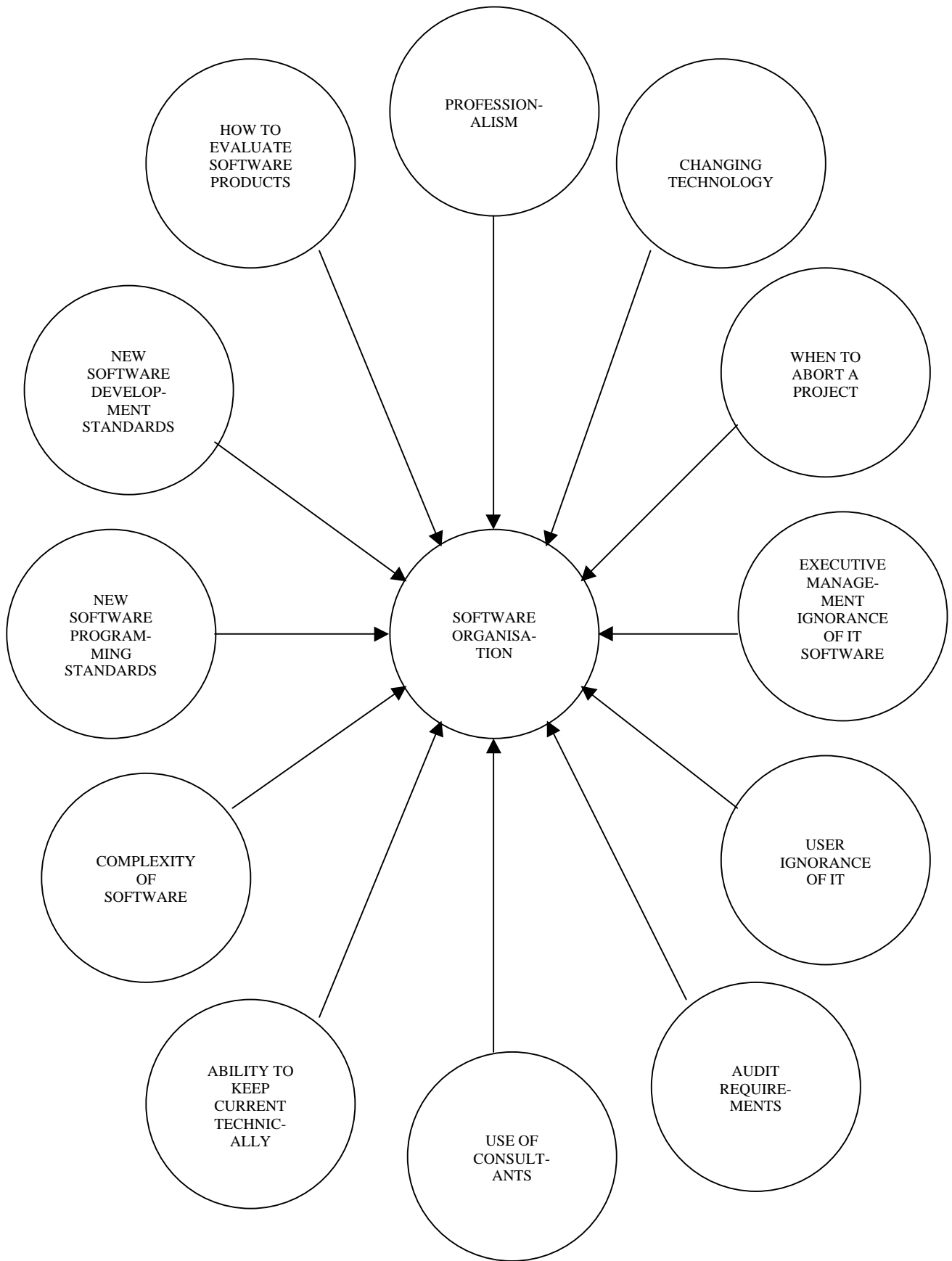
- a) The system meets the needs of the user departments and other users.
- b) The system is consistent with the needs of other users i.e. one system does not infringe on the rights of other system users.
- c) User objectives are consistent with the objectives of the software organization. In all cases, the software organization's objectives should have a higher priority than the objectives of any single user.
- d) System objectives meet the objectives of the software organization.

5.4.2.2 Other objectives of the SQA function could include the need to ensure that:

- a) System objectives are consistent with industry and government requirements.
- b) The system complies with the intent of the software organization for adequate controls and is audible.

Figure 1 below illustrates the challenges faced by a software organization which an SQA function could help solve:

Figure 1 Challenges of the Software Organization



5.4.3 GOVERNING POLICY

The SQA function should be staffed with individuals who are as knowledgeable as senior staff of the software organization. In addition, SQA personnel could be full-time, part-time or seconded for training from other functions within the software organization.

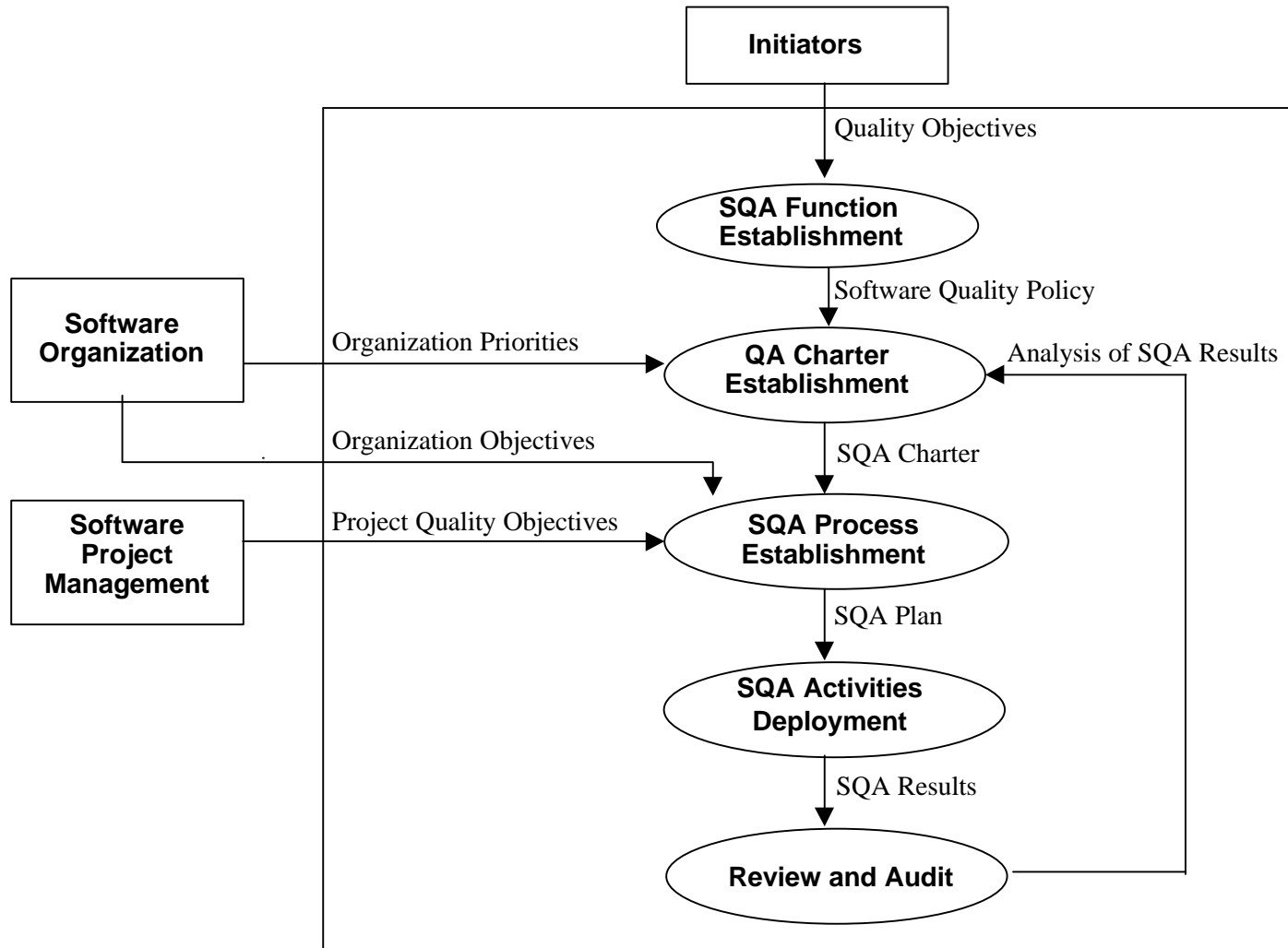
5.4.4 PRIMARY RESPONSIBILITY

Members of the SQA group (SQA practitioners) are responsible for providing SQA support to the software organization function. This support can be viewed as an extension to the responsibilities of the manager of the software organization.

The responsibilities of SQA group could be those of an independent function usually found in larger software organizations, or merged with another function that reports directly to the software organization manager in smaller organizations.

In practice, the SQA group could also serve as the user/customer's advocate, management's 'eyes and ears', and take the leading role in metrics development and process improvement.

5.4.5 SOFTWARE QUALITY ASSURANCE PROCESS OVERVIEW



5.5 USER REQUIREMENTS MANAGEMENT

5.5.1 DEFINITION

The following provide a definition of frequently used terms in describing the User Requirements Management practice:

5.5.1.1 User Requirements Management

It represents the process of discovering, understanding, negotiating, documenting, validating and managing a set of requirements for a computer-based system.

5.5.1.2 Requirements

It contains a mixture of problem information, statements of system behavior/characteristics and design and manufacturing constraints, and can basically be broken down into two basic types namely:

- high-level and abstract called *Stakeholder or User* requirements;
- detailed specifications of requirements called System requirements.

5.5.1.3 Requirements Document

It is an official statement of the systems requirements for customers, end-users and software developers. Other names for the requirements document include the 'functional specification', the 'requirements definition' and the 'software requirements specification' etc.

5.5.1.4 Requirements Engineering

It covers all the activities involved in discovering, documenting and maintaining a set of requirements for a computer-based system.

5.5.1.5 Requirements Engineering Process Maturity

It is the extent to which an organization has a *defined* requirements engineering process based on good requirements engineering practices.

5.5.2. OBJECTIVES

The objectives of user requirements management are to produce a set of system requirements which, as far as possible, are complete, consistent, relevant and reflects what the customer actually wants.

5.5.3. GOVERNING POLICY

There are no specific policies that govern the use of this practice.

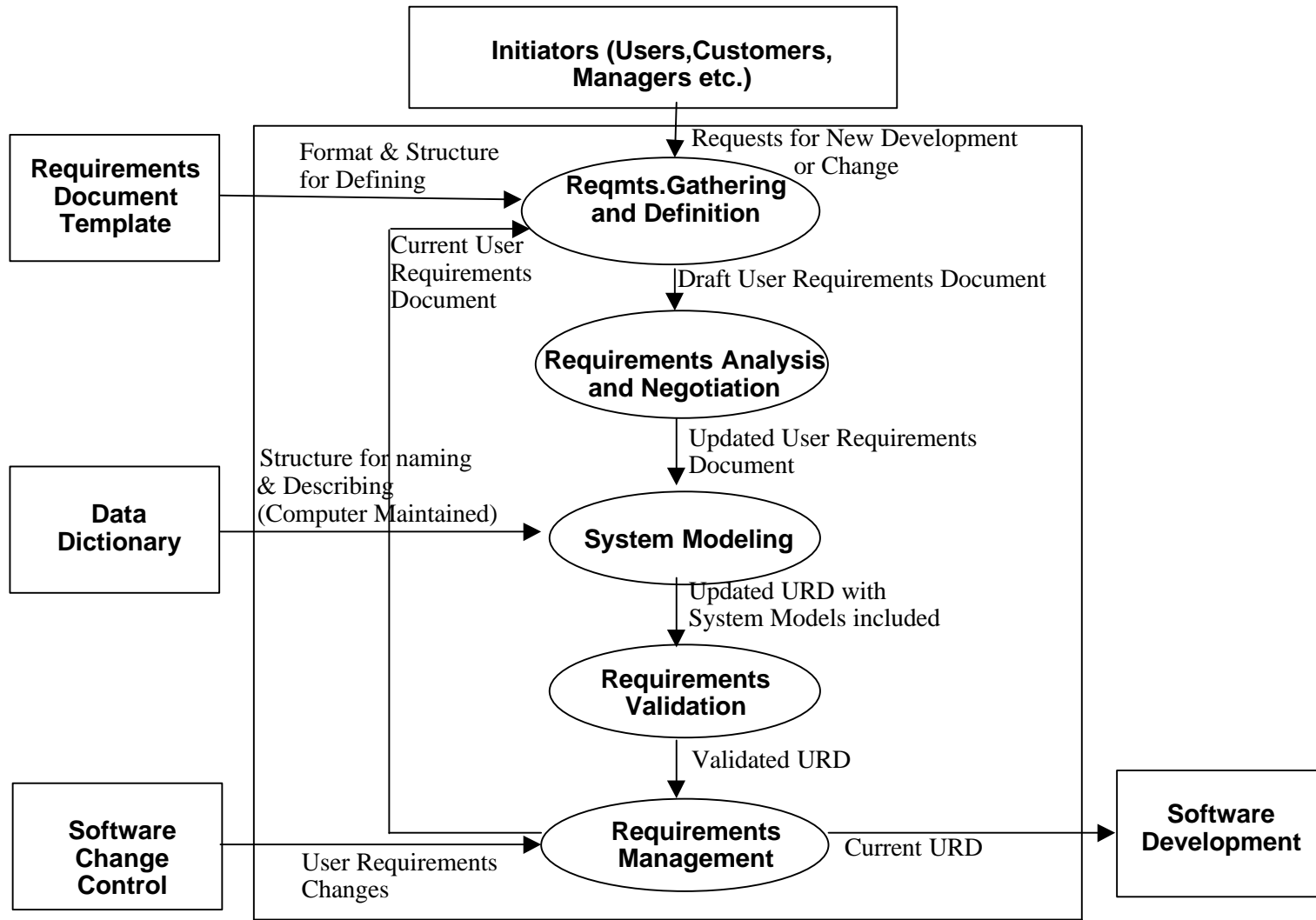
5.5.4. PRIMARY RESPONSIBILITY

The *manager of the software organization* or his delegate is primarily responsible for carrying out User Requirements Management activities defined below.

End-users and end-user management of software products and services are responsible for providing the support needed for carrying out User Requirements Management activities defined below.

Software developers are responsible for understanding, documenting, adhering and translating specified requirements into workable software solutions during development.

5.5.5 USER REQUIREMENTS MANAGEMENT PROCESS OVERVIEW



5.6 POST IMPLEMENTATION SUPPORT

5.6.1 DEFINITION

Software post implementation support is the process of providing operations and maintenance activities needed to use the software efficiently and effectively. This process begins after the software has been delivered into an 'operational' or 'live' mode in which users operate the software and utilize the end products and services it provides.

The major activities in software operations are:

- User support which provides training, direct assistance and other support in the use of the software;
- Problem reporting to enable problems to be resolved in an effective and timely manner.

Software maintenance activities can be classified as:

- Corrective which removes software faults;
- Perfective which improves the system without changing its functionality;
- Adaptive which modifies the software to keep it up to date with its environment.

5.6.2 OBJECTIVES

The objectives of post implementation support are to provide an effective product or service to the end-users while correcting faults, improve performance or other attributes or adapt to a changed environment to keep the software usable and useful after the software has been delivered into an 'operational' or 'live' mode.

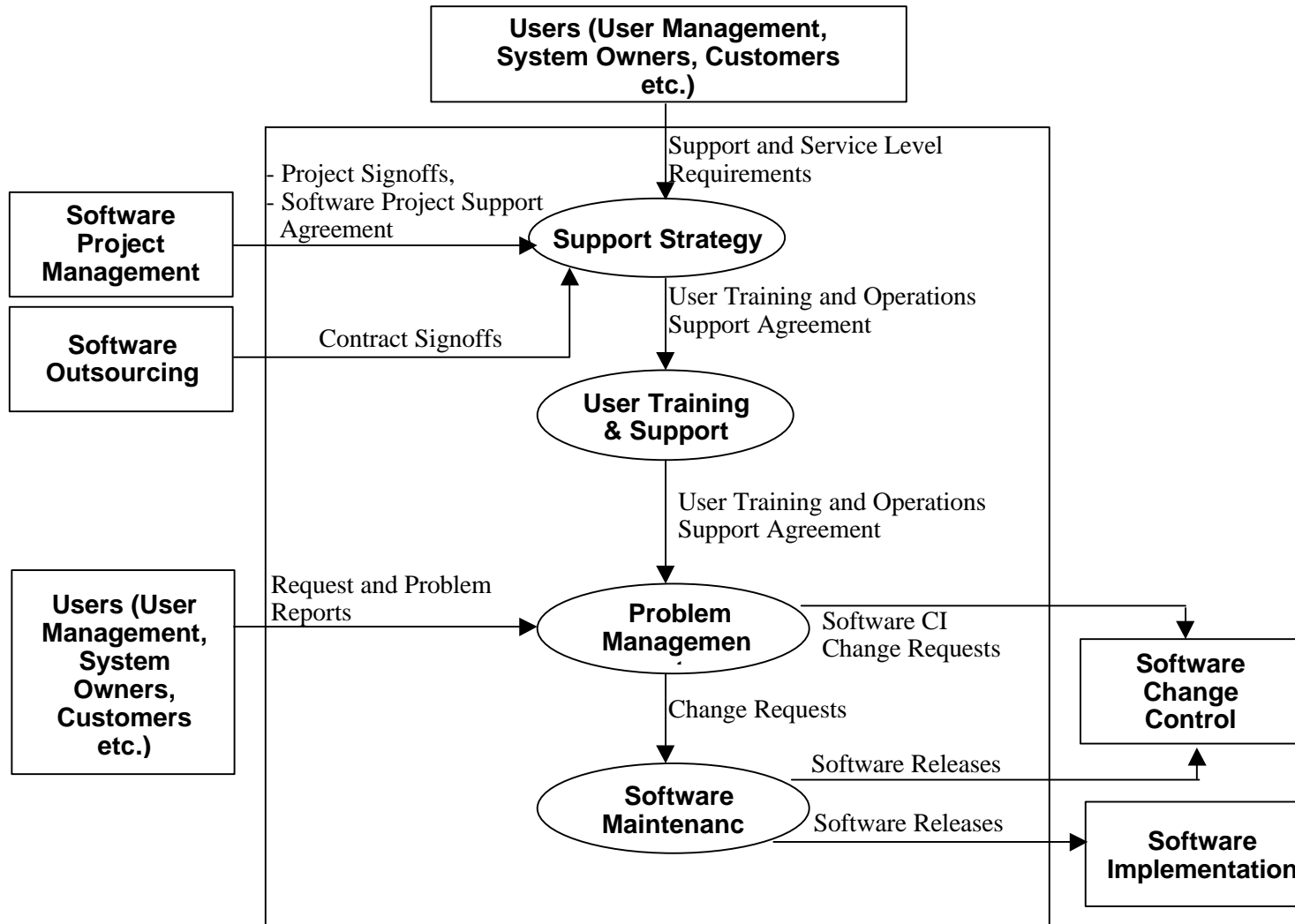
5.6.3 GOVERNING POLICY

There shall be a maintenance organization for every software product in operational use.

5.6.4 PRIMARY RESPONSIBILITY

The software maintenance manager (SMM) and operations manager (SOM) are responsible for providing leadership to the maintenance and operations organizations respectively. The user Help Desk, a function commonly found in larger organizations that provide first level assistance and response to users, is also the responsibility of either the maintenance or operations organizations.

5.6.5 POST IMPLEMENTATION SUPPORT PROCESS OVERVIEW



5.7 SOFTWARE CHANGE CONTROL

5.7.1. DEFINITION

Software Change Control refers to the process of evaluating proposed changes to software configuration items and coordinating the implementation of approved changes to ensure the integrity of the software remains intact and uncompromised.

A configuration item (CI) is an aggregation of hardware and software, that is designated for change control and is treated as a single entity in the change control process. Examples of configuration items are:

- *Software component*, such as a version of a source module, object module, executable program or data file;
- A version of a compiler or operating system;
- *Baseline*, such as a software system under development;
- *Release*, such as a software system in operational use;
- *Document*, such as a requirements document.

Software change control and related activities can best be described in the context of software configuration management which is formally defined as the discipline of applying technical and administrative direction and surveillance to:

- Identify and document the functional and physical characteristics of a configuration item;
- Control changes to the configuration item;
- Record and report change processing and implementation status;
- Verify compliance with specified requirements.

5.7.2. OBJECTIVES

The objectives of software change control are to ensure that:

- Software components can be identified;
- Software is built from a consistent set of components;
- Software components are available and accessible;
- Software components never get lost (e.g. after media failure or operator error);
- Every change to software is approved and documented;
- Changes do not get lost (e.g. through simultaneous updates);
- It is always possible to go back to the previous version;
- A history of changes is kept, so that it is always possible to discover who did what and when.

5.7.3. GOVERNING POLICY

5.7.3.1 Change Control

All software items e.g. documentation, source code, executable code, files tools, test software and data shall be subjected to change control procedures.

5.7.3.2 System Libraries

As a minimum requirement, in order to ensure security and control of software, the following system libraries *shall* be implemented for storing configuration items (CIs):

- Development (or dynamic) library, where code development and unit testing takes place;
- Test (or master) library, where integration and systems testing are carried out;
- Production (or archive) library, where acceptance testing is carried out.

In addition, the following change control policies will apply:

All development *shall only* be carried out in the development library.;

- Production libraries *shall not* be modified (instead when there is a new release, the previous version is being backed up) ;
- Procedures for backing up all libraries shall be established.

Figure 1 illustrates the relationship between units of module, baselines and releases and software libraries.

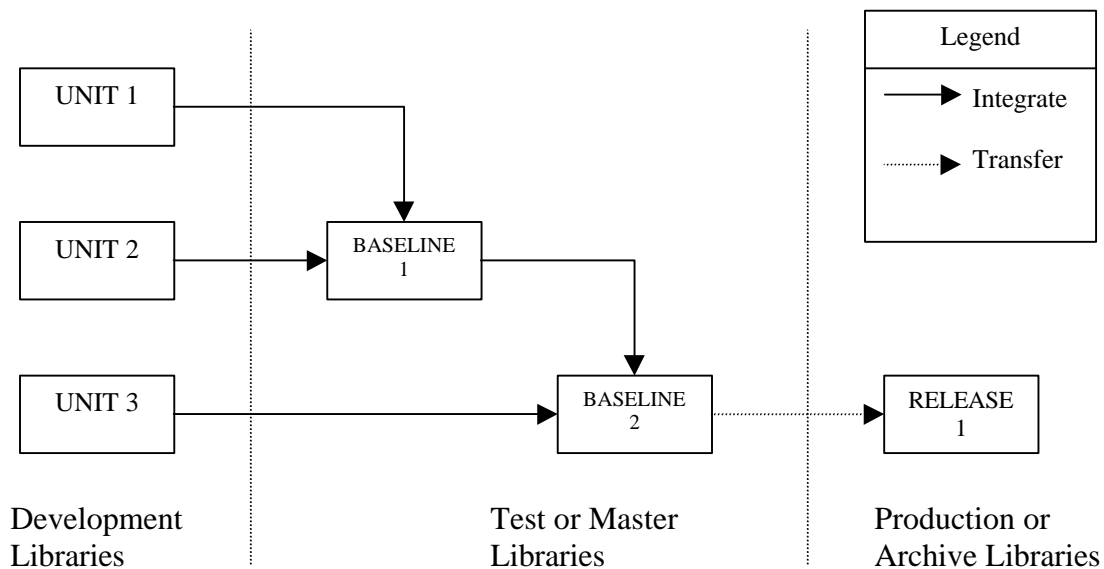


Figure 1: Units, Baselines, Releases and Libraries

5.7.4. PRIMARY RESPONSIBILITY

5.7.4.1 Change Management

Software project management is responsible for organizing software change control activities and defining associated roles during the development phases of the software life cycle.

During the operations and maintenance phases of the software life cycle, responsibility for the software change control lies with the software maintenance manager (SMM).

5.7.4.2 System Librarian

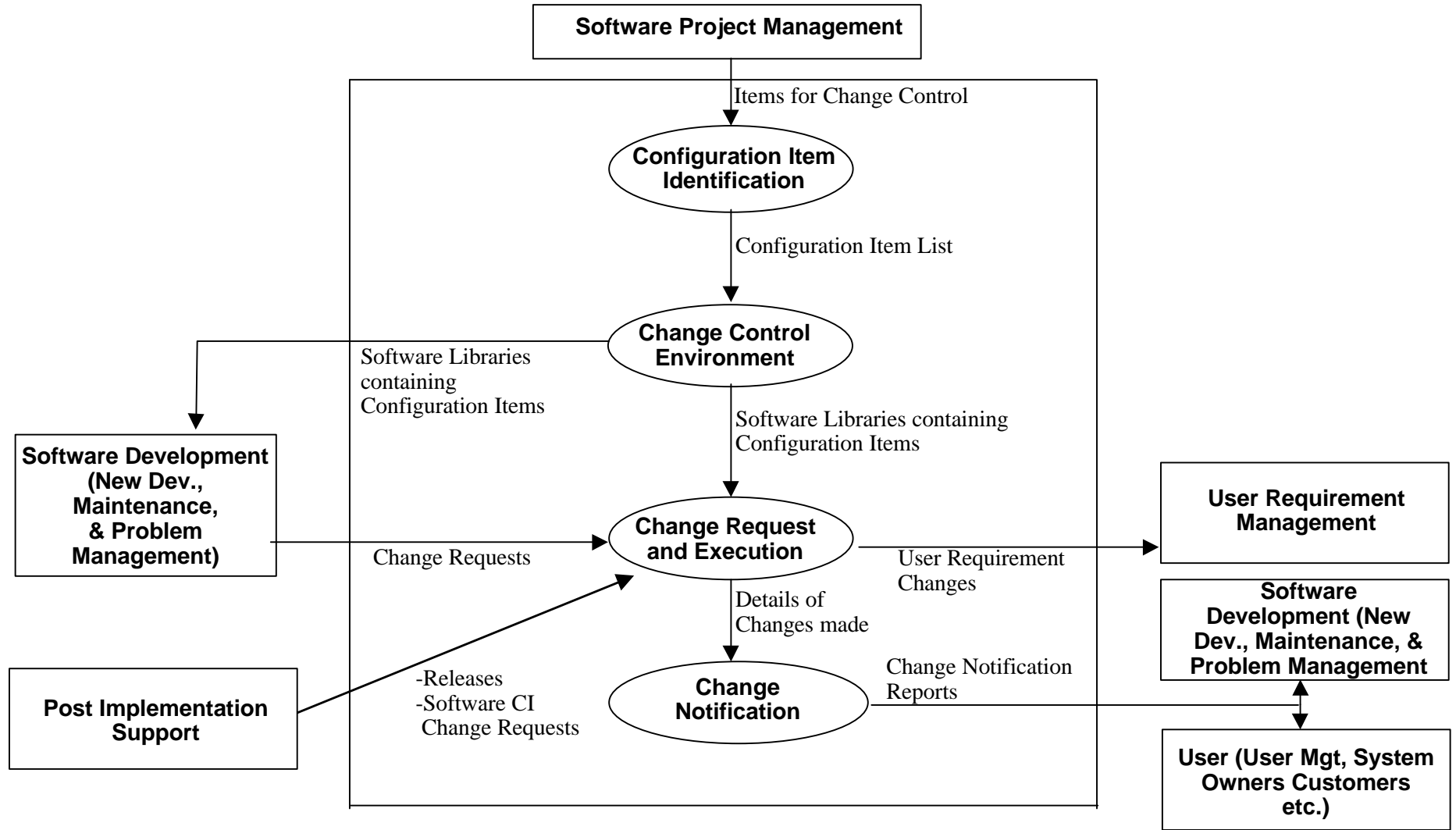
The Change Manager should be supported by a System Librarian who reports to him and who is responsible for:

- establishing new test libraries;
- updating test libraries;
- backup of test libraries to production libraries;
- control of access to test and production libraries.

5.7.4.3 Change Auditor

The Change Auditor should be a part of an independent group in the software organization e.g. the Software Quality Assurance (SQA) group. He/she is responsible for verifying whether the planned software change control activities are properly implemented, at regular intervals or at the end of the development project, through the use of physical inspections of the Configuration Items (CI) or other quality assurance procedures.

5.7.5 SOFTWARE CHANGE CONTROL PROCESS OVERVIEW



6. Glossary of Terms

1. **Change Control**

Through the exercise of relevant controls, this aspect of maintenance & support focuses on all the work required to ensure the integrity of hardware and software that are being changed in the course of development and when systems are moved into production. Parts of Change Control impact or are related to Configuration Control and Problem Tracking or vice versa.

2. **Change Management**

Through communication and regular interaction with users, this aspect of IT management focuses on the integrity of hardware and software that are being changed in the course of development and when systems are moved into production. Parts of Change Management impact or are related to Configuration Management or vice versa.

3. **Configuration Control**

Through the exercise of relevant controls, this aspect of maintenance & support focuses on all the work required to ensure the integrity of hardware and software that are being used during development and when systems are in production mode. Parts of Configuration Control impact or are related to Problem Tracking or vice versa.

4. **Configuration Management**

Through communication and regular interaction with users, this aspect of IT management focuses on the integrity of hardware and software that are being used during development and when systems are in production mode.

5. **Development Documentation**

This represents the documentation that is required to be developed as software/systems are being developed to ensure continuity and auditability of development effort, and to provide the means for support and operations of the software/systems as they are moved into production.

6. **Documentation Control**

This represents the area of maintenance & support where documentation that is required to be updated as software/systems are being changed to ensure continuity and auditability of development effort, and to ensure their reliability as software/systems are moved into production.

7. **Guideline**

Rules and instructions on how work should be done or tasks should be carried out.

8. **Help Desk**

To satisfy user requirements, this area represents all the activities that are required to exist to operate the Help Desk which serve as the first point of contact by users for IT services related to problems encountered or change requests.

9. **Implementation**

This area represents the work needed to execute jobs to move the already tested and checked software/system(s) into the production environment.

10. **Implementation Checklist, Review & Audit**

Before implementation takes place, this area represents the work needed to conduct a final check that everything has been completed successfully before moving the software/system(s) into the production environment. The areas that need to be checked for completeness include

- *maintenance, operations, user documentation,*
- *tested software results,*
- *signoffs to signify approval of key systems development work,*
- *user and business preparedness in using the new or changed system(s),*
- *IT operations preparedness to run and support the system in production mode &*
- *the readiness of the business and users to accept the new system including any required marketing and communication preparations.*

11. IT Management

That aspect of the organization which is responsible for the operations of the IT function including people, projects, other resources, financial and coordination with users.

12. IT Operations

That function of the IT organization that is responsible for executing jobs to operate software/system(s) in the production environment.

13. Maintenance & Support

That function of the IT organization that is responsible for maintaining and supporting existing software/system(s).

14. Method

A way of achieving an objective.

15. Metric

A metric is a quantitative measure of the degree to which a system, component or process possesses a given attribute.

16. Operations Procedures

This area represents the procedures needed to execute jobs, load/unload disk storage devices, tape drives, operate telecommunication devices, respond to console messages and error conditions, track operational problems, maintain operating room environment etc.

17. Policy

A statement of top management directives to decision-makers at lower levels to guide and determine present and future actions within an organization. (QAI)

18. Post Implementation Review

This area represent the work needed to assess a recently installed or existing system against their intended objectives and how well they are performing in meeting customer requirements.

19. Practice

A method of doing work which through common usage has become an accepted way used in producing products and providing services in an organization. (HKJC).

20. Problem Management

Through communication and regular interaction with users, this aspect of IT management focuses on problems identified during development and/or in

production mode from their identification through to their final resolution. Parts of Change Management impact or are related to Configuration and Change Management or vice versa.

21. Problem Tracking

Through the exercise of relevant controls, this aspect of maintenance & support focuses on monitoring software problems identified during development and/or in production mode from their identification through to their final resolution.

22. Procedure

A set of manual steps to be followed to accomplish a task each time the task is done. (IEEE)

23. Process

A formal method for doing work which allows the same quality to be replicated from product/service to product/service; implies the use of standards and procedures. (QAI)

24. Project Management

Mainly to ensure the success of an IT project, that aspect of IT management that deals with the management of an IT project (s) with responsibility over planning, organizing, staffing, monitoring, controlling and leading the IT project (s).

Delineation between software and system project management must be made since the former is by definition only a part of the latter. It should also be clear that the management, development, support and operations of software also involves the hardware aspects as one has to live with the other.

25. Quality Control (QC)

QC represents the techniques and activities that are used during software development that serve to detect and correct defects surfaced during inspections, reviews, walkthroughs and testing.

26. Quality Management

Mainly to satisfy user requirements, that aspect of IT management that focuses on the quality of products and services provided by IT by such means as quality planning, quality control, quality assurance and quality improvement.

27. Request Management

To ensure continuing integrity of software/systems in use, this area of maintenance and support covers the time when a request for software change (due to a problem or new product feature) is generated from users to the time when the change is implemented.

28. Service Level Monitoring

This area represent the work needed to measure the effectiveness of IT operations in meeting predefined service level agreements (SLAs) and steps taken to improve service performance in satisfying customer service requirements.

29. Software Development Process (SDP)

Mainly to minimize the risk of software defects and for consistency purposes, the SDP represents a formal method used for developing software.

30. Software Distribution

Mainly through the exercise of relevant controls and the proper use of specific techniques, this aspect of IT management focuses on the work required to

ensure that the software/systems continue to perform at status quo as they are being moved to a different location/site and/or platform.

31. Software Purchase

Mainly through the exercise of relevant controls and the proper use of specific techniques, this aspect of IT management focuses on the work required to ensure that the software products/packages acquired serve the objectives and purpose they were purchased for.

32. Software Tools Utilization

Mainly to ensure that software tools serve the purpose they were acquired for, this aspect of development represents techniques used in selecting, installing and applying software tools used during the development of software.

33. Software/System Installation

That function of the IT organization that is responsible for implementing software/system(s) into the production/operating environment.

34. Software/Systems Development

That function of the IT organization that is responsible for developing new software/system(s).

35. Standard

Mandatory requirements employed and enforced to prescribe a disciplined and uniform approach to software development. (CMM)

36. Standards Management

Mainly to ensure consistency of practices employed in IT, that aspect of IT management that focuses on managing the standards that are used in IT including policies, processes, procedures, technical & product standards, and guidelines.

37. Sub-contractor Management

Mainly to ensure that user requirements are satisfied, that aspect of IT management that deals with the management of contractors or outsourcing organizations who have been recruited to provide a particular IT service or deliver a particular IT product.

38. Systems Maintenance Process (SMP)

Mainly to minimize the risk of software defects and for consistency purposes, the SMP represents a formal method used for developing changes to software. Parts of the SMP impact or are related to the Systems Development Process (SDP) B1 and Quality Control (QC) B3 above and vice versa.

39. Task

A sequence of instructions treated as a basic unit of work. (IEEE)

40. User Management

Mainly to ensure regular communication with users, that aspect of IT management that focuses on involving the users of systems or projects that are being supported or developed by IT including the communication and participation of the users in resolving management, quality, resources and project issues.