

PS.1 Support Strategy

1 D O M A PS.1.1 Agree on hand-over

1 D O M A PS.1.1.1

After the software has been accepted by the users and released (for development as well as outsourced) into an 'operational' or 'live' mode, developers i.e. project manager, systems analyst, programmers, should provide software user support and maintenance *for a user-required period of time and level of support.*

1 D O M L PS.1.1.2

When the software is finally accepted, the responsibility for these activities should be handed over to a maintenance team led by the software maintenance manager (SMM). (Note: For small organization, the SPM should play the maintenance role.) Major handover deliverables include all documentation, programs, macros, security procedures etc.

1 D O M L PS.1.1.3

The SPM should ensure that all documentation is complete and contents consistent at handover.

2 D O M A PS.1.2 Establish service levels

2 D O M A PS.1.2.1

The SMM should establish service level agreements with users including the availability, delivery and support requirements of the software system, eg, in a Maintenance Plan which covers procedures of maintenance activities, resources and facilities etc.

2 D O M A PS.1.2.2

The SMM shall report to users on performance in meeting the service level agreements, taking the necessary steps to improve or correct as required.

2 D O M A PS.1.2.3

The SMM should establish regular (normally quarterly)

reviews with users of existing software systems to determine what services should be continued, updated or eliminated. The reviews should also include a determination of what software systems should be maintained and what could be retired.

PS.2 User Training and Support

Developers provide users and operators with the following support activities:

- Training the users and operators to operate the software and understand the products and services;
- Providing direct assistance during operations;
- Providing set-up expertise to configure the software system;
- Providing data management expertise to handle data.

1 D O M A PS.2.1 Train Users/Operators

1 D O M A PS.2.1.1

With the support of user management, developers (the project manager or analyst) should establish a training plan for users/operators, in the project planning stage of the software life cycle which could range from training to familiarize with the software user operation manual to a course on the use and operations of the software system.

1 D O M A PS.2.1.2

Developers should conduct this formal training to the users/operators in accordance with the training plan and in the presence of maintenance staff and any stakeholders who would benefit from the training.

1 D O M A PS.2.2 Provide Direct Assistance to Users/Operators

1 D O M S PS.2.2.1

Developers should supplement the formal training with direct assistance to users/operators in the use and operations of the software.

1 D O M L PS.2.2.2

Developers should supplement the formal training with help desk support especially when the number of users become too large to:

- a) provide users/operators with advice, news and other

information about the software;

- b) receive problem reports and decide how they should be handled.

1 | D O M A | PS.2.3 Provide Set-up Service

1 | D O M A | PS.2.3.1

Developers should provide users/operators with expertise and assistance in setting up the software system in the case where:

- a) the software set-up is complex;
- b) the software system is shared by multiple users;
- c) software expertise is needed to minimize the risk of errors.

1 | D O M A | PS.2.4 Manage Data

1 | D O M A | PS.2.4.1

Developers should provide assistance to users/operators in managing data to include the following activities:

- a) configuring data files for users/operators;
- b) managing disk storage devices;
- c) providing backup and archiving expertise and support.

PS.3 Problem Management

1 | D O M A | PS.3.1 Report Problems

1 | D O M A | PS.3.1.1

In using the software system, users should document any problems encountered in Software Problem Reports (refer PS.A for a form template of a Software Problem Report). Problems could also be reported by a variety of sources including maintenance team, operators etc.

1 | D O M A | PS.3.1.2

The user reporting the problem should decide and indicate whether the problem is:

- a) Critical or non-critical; a problem is critical if the software or an essential feature of the software is unavailable;
- b) Urgent or routine; a problem is urgent if the solution is needed as soon as possible.

1 D O M A PS.3.1.3

The user reporting the problem should:

- a) Describe the problem and the operating environment as accurately as possible so that the problem can be reproduced as required;
- b) Attach any printouts and log files with the software problem report;
- c) Document omissions to specific capabilities or compliance with some constraints.

1 D O M A PS.3.2 Address Problem

Figure 1 below shows the life-cycle of a software problem.

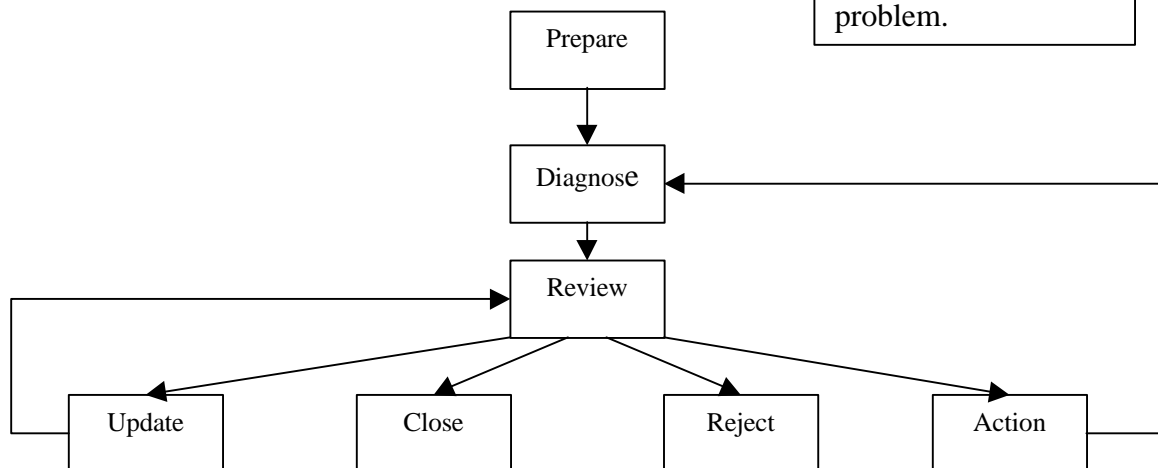


Figure 1: Life Cycle of Software Problem

1 D O M A PS.3.2.1

The user reporting the problem should prepare and submit the completed software problem report and relevant attachments to the software maintenance team for diagnosis.

1 D O M A PS.3.2.2

In the software maintenance organization, a staff (normally an analyst or a programmer) is assigned by the SMM the task of diagnosing the problem.

1 D O M L PS.3.2.3

After a preliminary diagnosis of the problem, the analyst or programmer shares the results of the diagnosis with the SMM to decide upon *one of four* possible outcomes for the problem:

- a) *Reject* the problem due to inadequate or inaccurate information provided; in this case the user reporting the problem will be contacted to supply the necessary information required;
- b) *Complete* a Software Change Request form including documenting affected software CIs changed (refer PS.B) and *update* the software accordingly;
- c) Due to the nature of the problem, *action* the problem to someone else in the software organization to carry out further diagnosis (refer to section PS.4.1 for a further discussion of how problems are addressed through software change);
- d) *Close* the problem because the software update has been completed successfully.

2 D O M A PS.3.3**2 D O M A** PS.3.3.1

The SMM should ensure that data on problem status and performance in addressing and resolving problems are gathered, acting on areas where improvement may be achieved.

2 D O M A PS.3.3.2

The SMM should circulate appropriate report summaries on a regular basis e.g. monthly, to key stakeholders, user and software management to provide awareness and solicit support in managing problems.

PS.4 Software Maintenance

Software maintenance should be a controlled process consisting of the following activities:

- Change software;
- Release software;
- Install release;
- Validate release.

1 D O M A PS.4.1 Change Software

Software change is a four-stage process namely:

- Diagnose problems;
- Review change requests;
- Modify software;
- Verify software.

Diagnose problems (Note: Users will normally be required to input to this process)

1 D O M A PS.4.1.1

In diagnosing problems, the software engineer from the maintenance team collects all software problem reports assigned to him/her.

1 D O M A PS.4.1.2

The software engineer then performs the following steps in problem diagnosis:

- a) Examine the problem report (s);
- b) Reproduce the problem if possible and necessary;
- c) Examine the code and/or software documentation if necessary;
- d) Identify the fault (s) resulting from:
 - a detected error,
 - a deficiency,
 - problem in operating system or software,
 - customer improvement or modification request.
- e) Identify the cause (s);
- f) Write a Software Change Request if necessary.

1 D O M A PS.4.1.3

The software engineer should use debugging tools to help diagnose the problem.

1 D O M A PS.4.1.4

When the cause of the problem has been found, the software engineer should request a change to the software to prevent the problem from recurring.

1 D O M A PS.4.1.5

In some cases, the software engineer may request to implement a temporary 'workaround' solution to a problem pending the implementation of a complete solution.

1 D O M A PS.4.1.6

The software engineer should include the following pieces of information in the change request:

- a) software title or name;
- b) software item version or release number;
- c) changes required;
- d) priority of the request i.e. *criticality*-critical/non-critical and *urgency*-urgent/routine (defined by SMM);
- e) responsible staff assigned (supplied by SMM);
- f) estimated start and end date and manpower effort (defined by SMM).

1 D O M A PS.4.1.7

The software engineer should submit any attachments e.g. marked-up listings of source code and pages of documentation etc., along with the change request to the SMM.

1 D O M A PS.4.1.8

The SMM should decide whether the change request is critical or non-critical. *Note that a change is considered critical if it has a major impact on either the software behaviour or the maintenance budget.*

1 | D O M A PS.4.1.9

The SMM should decide whether the change is urgent or routine.

1 | D O M A PS.4.1.10

If the change is urgent (solution is needed as soon as possible), the SMM has to prepare for it to be released to the user as soon as possible.

1 | D O M A PS.4.1.11

If the change is considered routine, it can be released in convenient groups according to a release schedule discussed in section PS.4.2 below.

Review changes (*applies to large organizations only*)

2 | D O M L PS.4.1.12

A Software Review Group (SRG) should be established to authorize all changes to the software.

2 | D O M L PS.4.1.13

The SRG should consist of the SMM, Software Project Manager, the Software Quality Assurance Engineer and users of the software.

2 | D O M L PS.4.1.14

The SRG should delegate responsibility to the SMM for filtering software problems and the resulting change requests as shown in the following figure.

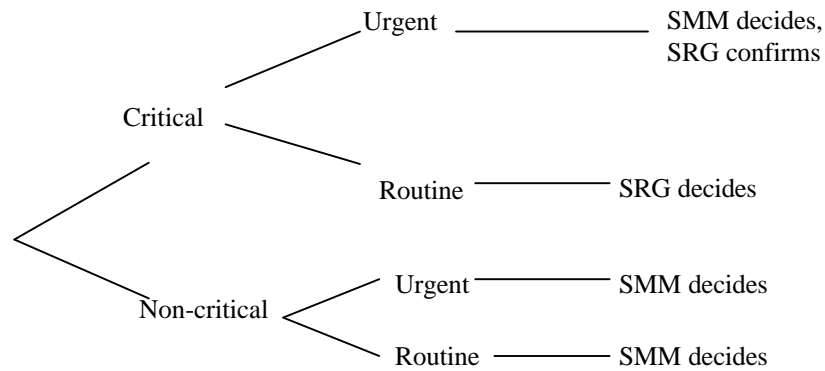


Figure 2: Decision tree for filtering change requests and for actions to be taken

2 | D O M L PS.4.1.15

For critical and urgent change requests, the SMM decides upon the implementation of the change and passes the Change Request and associated attachments to the SRG for confirmation.

2 | D O M L PS.4.1.16

For critical and routine change requests, the SMM passes the Change Request and associated attachments to the SRG for review and decision.

2 | D O M L PS.4.1.17

For non-critical change requests, the SMM decides upon the implementation of the change and passes the Change Request and associated attachments to the SRG for information.

Modify software

1 | D O M A PS.4.1.18

When the software change request has been approved, the maintenance team (normally a software engineer) should implement the change by *modifying the documents and code*.

2 | D O M A PS.4.1.19

In performing the modification (s), the software engineer should adhere to Software Change Control procedures (refer Software Change Control practice). In addition, the

software engineer should *evaluate the effect of the modification* on:

- a) performance;
- b) resource consumption;
- c) cohesion;
- d) coupling;
- e) complexity;
- f) consistency;
- g) portability;
- h) reliability;
- i) maintainability;
- j) safety;
- k) security.

2 D O M A PS.4.1.20

During the evaluation, the software engineer should examine the change options, compare their effects on the software and select the best solution.

PS.C provides guidelines on how to evaluate software based on the above parameters.

2 D O M A PS.4.1.21

The software engineer should ensure that there is consistency between code and documentation achieved by:

- a) thorough analysis of the impact of every change before it is made to ensure that no inconsistencies are introduced;
- b) concurrent update of code and documentation;
- c) verification of changes by peer or independent reviews.

3 D O M A PS.4.1.22

The software engineer should use tools to keep *code and documentation consistent* for example:

- a) tools for deriving detailed design information from source code;
- b) Traceability matrices that check consistencies between the code and design, software requirements and user requirements documents.

2 D O M A PS.4.1.23

The software engineer should ensure that the modifications made (documentation and code) are *reviewed* using the walkthrough or inspection process by someone else other than himself/herself.

1 D O M A PS.4.1.24

The software engineer should ensure that the modified software is re-tested before release, normally carried out by:

- a) unit, integration and system testing each change;
- b) running regression tests at unit, integration and system level to verify that there are no side-effects resulting from the modification.

PS.D provides guidelines on how modified software should be tested.

1 D O M A PS.4.2 Release software**1 D O M A** PS.4.2.1

Apart from changes to fix urgent software problems, the SMM should ensure that modified software should follow a software release process consisting of:

- a) defining the release;
- b) documenting the release;
- c) auditing the release;
- d) delivering the release.

Defining the release

1 D O M A PS.4.2.2

The SMM should define the content and timing of a software release in accordance to the needs of the user that

is:

- a) Solutions to urgent problems are released as soon as possible to the people experiencing the problem or to those likely to experience the problem;
- b) Other changes are released when the users are ready to accommodate them.

1 | D O M L | PS.4.2.3

The SMM should allocate changes to one of three types of releases:

- a) Major release which mainly provide new capabilities where the impacts to the organization is significantly large ;
- b) Minor release which mainly provide corrections to a group of problems;
- c) Emergency release (also called a ‘patch’) which provide a modification to the users who need it as fast as possible.

Documenting the release

1 | D O M A | PS.4.2.4

Having decided on the type of release, the SMM should ensure that it should be accompanied by a relevant and adequate documentation e.g. a Software Release Note (SRN) to include:

- a) The release number;
- b) Changes referencing the problems addressed included in the release;
- c) The software configuration items that will be affected by the release;
- d) The release installation instructions.

PS.E provides a template for the Software Release Note.

Audit release

1 | D O M A | PS.4.2.5

The SMM should ensure audits are performed to verify

that all software configuration items included in the release are present, consistent and correct by:

- a) Checking test reports for the release;
- b) Verifying completeness of operational and support documentation;
- c) Verifying consistency between documentation and code;
- d) Checking availability of all software configuration items specified.

Deliver release

1 | D O M A | PS.4.2.6

After the release has been audited satisfactorily, the SMM (normally through one of his maintenance staff) delivers the release by:

- a) Copying the software into the release media;
- b) Packaging the media with the software documentation;
- c) Delivering the package as required.

1 | D O M A | PS.4.2.7

The SMM should ensure that every release is backed up for reference purposes.

1 | D O M A | PS.4.3 Install Release

1 | D O M A | PS.4.3.1

Upon delivery, the receiver should check the contents of the release package against the Software Release Note for completeness.

1 | D O M A | PS.4.3.2

When all items in the release package have been checked to be complete, the receiver (who may also serve as the installer) then installs the software by following the installation instructions in the Software Release Note.

1 D O M A PS.4.3.3

The installer should:

- a) use tools to support the installation that automate and simplify the installation process as much as possible;
- b) ensure the installation is reversible e.g. by backing up the existing system before starting the installation.
- c) ensure that obsolete software configuration items are removed from software directories after installation.

1 D O M A PS.4.4 Validate Release**1 D O M A** PS.4.4.1

After installation, users should run some or all the acceptance tests to validate the software.

1 D O M A PS.4.4.2

Where required, users could arrange with the installer to go into parallel system operation mode for an agreed period of time before going into a 'live' operation.