Using PCMS to Control AIX Software Development

By Sohail Haque

One of the challenges in managing medium to large software projects is the inability to retrieve real-time, "up to the minute" information on the current status of the project. Use of team meetings, paper-based status reports and E-mail are typical methods that project managers use to gather new information. This article shows how SQL Software's Process Configuration Management Systems (PCMS) can be used in software development on IBM's RISC System/6000 using AIX.

Software engineering is constantly being challenged to improve itself. Several panaceas have been proposed over the last ten years. First was the development of rigid requirements, design, and coding phases, combined with upper-Computer-Aided Software Engineering (CASE) technology. Now, object-oriented analysis, design, and programming languages, plus the concepts of rapid application development and prototyping are currently in vogue to improve software productivity.

Throughout these methodology wars, one very basic problem has not been addressed: the lack of centralized information. Software engineering organizations lack centralized data that provides both the engineering team and management with the status of all changes and activities related to software.

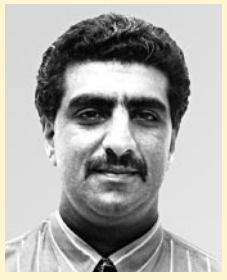
This real-world engineering information contains the status of all defects, enhancement requests, and customer calls, incorporated with the physical changes of various design models, requirements documents, and physical source code. It is gathered via the approaches found within Process Configuration Management Systems (PCMS). PCMS collects real-world engineering information by combining process workflow behavior; change initiators such as defects, enhancement requests, and new requirements; and physical change activity on the corresponding application objects with help desks and customer support databases.

What is PCMS?

PCMS is a set of integrated products developed by SQL Software to enable enterprises to leverage their financial investments in engineering technology and customer service. Although SQL Software offers a scalable set of products that are fully integrated within themselves, they are also open and can be woven into customer environments and frameworks. They are built using industry-standard technologies, such as relational databases, to protect customer investments.

PCMS is an active system that combines a process workflow engine with comprehensive change and problem management, version management, build automation, and help desk products. These components enable PCMS to support the entire life cycle of both software and system development including software production and manufacturing, documentation, and hardware. It enables the resultant systems to be enhanced and maintained, while providing comprehensive customer support services.

PCMS represents a new breed of industrial-strength configuration management systems that provide concurrent engineering and build


Sohail Haque

support for the developer as well as important status, cycle time, and customer service reporting for management. Unlike traditional tools that passively log events without providing any real control or anticipation of problems, PCMS actively supports project cycles including development, production, maintenance, and customer support.

PCMS enables managers, developers, and support staff to achieve higher quality, and reduced development time and costs.

PCMS Process Engine

An advanced process engine, common to all PCMS products, enables project processes and their interrelationships to be modeled. Because of these relationships, objects (such as source files) and composite objects (such as executables and documents) can be managed, cross-related, audited, and reported on. PCMS adjusts effortlessly to different project life cycles and new methods and is readily adaptable to corporate processes. PCMS process models can be used and replicated across a large enterprise, dispersed projects, and between contractors.

Figure 1 shows the components of the PCMS product set.

PCMS*VMB:	Comprehensive version management, software build, release and distribution product to support developers and management
PCMS*CTS:	Enterprise-wide change and problem management product
PCMS*Helpbench:	Solution-based help desk
PCMS*PCwin:	Seamless access to PCMS (with security controls) from a PC Windows and Windows NT environment
PCMS*ART:	Project-level archive, retrieval, and transfer system
PCMS*NLS:	Network library support for mixed UNIX, VMS®, Microsoft, and Windows NT solutions; provides networking to the process engine
PCMS*SII:	System integration interface including API and pre- and post-event trigger callouts

Figure 1. PCMS product set

Optimizing PCMS Performance

A PCMS network can be divided into library nodes (those on which PCMS item libraries reside) and non-library nodes (the remainder of the nodes). The network can be configured to take advantage of the available computing resources. PCMS*NLS provides networking facilities to permit operations across both homogeneous and heterogeneous environments. It can also be used to spread the processing load.

Using Oracle® as an example, Oracle processes should execute on the fastest node in the network, and if possible, have no PCMS logins on it. Operating system parameters should be optimized with as much RAM as possible for each library node in the network. If a single-user workstation is used on the network, the working set sizes can be significantly increased to reduce paging.

Creating UNIX Accounts for PCMS and Oracle

Certain accounts must be created before installing any PCMS and Oracle products.

Two special accounts, pcms and oracle, should be created—each in a group by itself with no other accounts in the same group. These two accounts will own the pcms and oracle files. Apart from this, they should only be used for Oracle and PCMS administration functions as described in *PCMS Database Administration Guide*, which comes with the system. An additional directory may be required to hold the Oracle database files. Before creating a PCMS base database, the correct Oracle runtime must be installed.

After the installation is complete, the next step is to load the Flight Simulator (FS) demo, which has the process model for software development and change. This process model will be used in the rest of this article.

Configuring PCMS for Projects

PCMS has a screen-based fully configurable process model that is known as a *control plan* when it is completed. This demo product's control plan includes documentation. This article will discuss a portion of the full control plan.

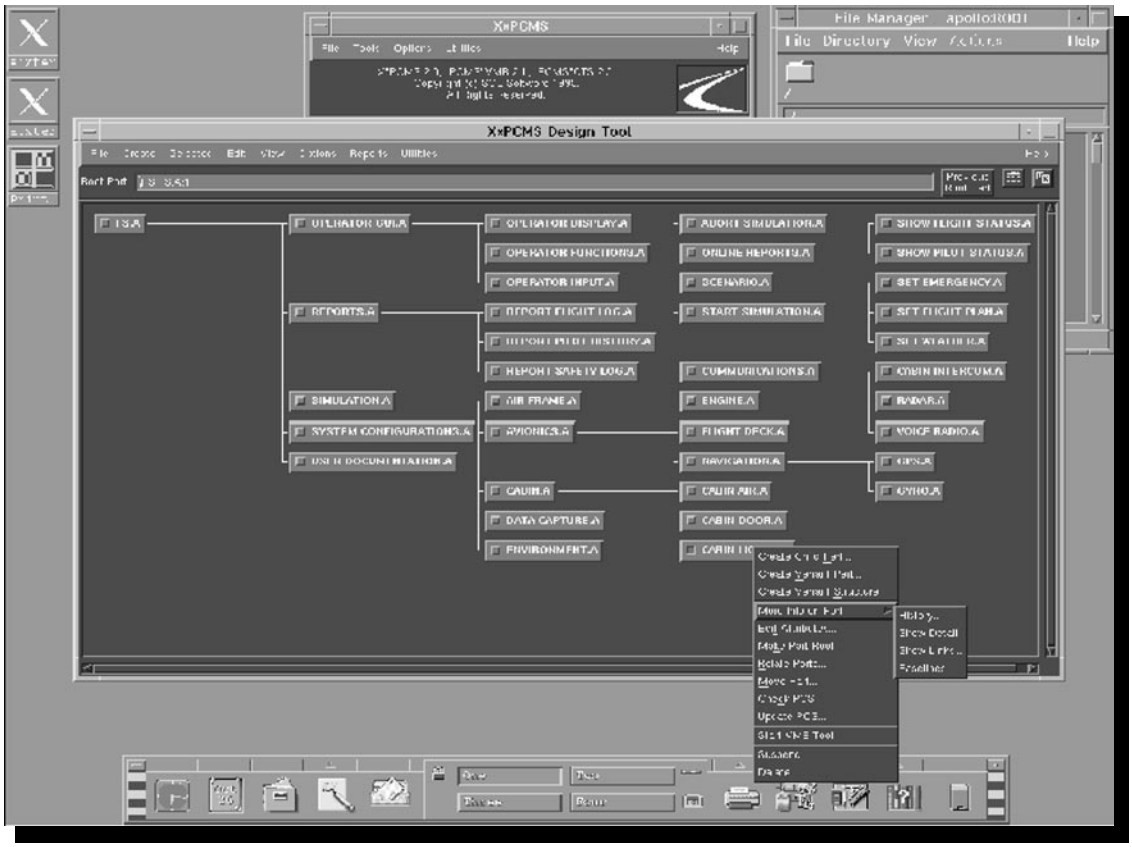


Figure 2. Product structure for Flight Simulator

Product Breakdown Structure

Product is the root node at the top of the PCMS product structure and the highest-level design component. The product is usually a family of products, a single product, a project, a system, or an application. A system can be structured into subsystems; systems and subsystems can be managed in a similar way.

In the Flight Simulator product, each subsystem consists of *modules*, the smallest units to be programmed or engineered.

Figure 2 shows the breakdown of the product structure for a Flight Simulator.

Design Parts and Product Items

Within the project, many product items will be the same type. For example, a project will have many source files and module specifications that are classified for control purposes into item types. For example, source files could be defined as type SRC; design specifications could be defined as type DS. Physical items could have life cycle states such as created, reviewed, audited; they could also be modified during product development.

In the product breakdown structure, design parts can have classifications similar to items. A documentation plan can be constructed using the design part categories and item types. The mandatory and optional item types for each category of design part can be defined. For example, a module must have a module specification and source code. The documentation plan provides a mechanism to ensure that a product represented by a product baseline is complete, as shown in Figure 3.

Activities and Responsibilities

When a set of development activities is completed, the result is a deliverable item or a modified product item. An activity may not result only in a new or modified product item, but it could also be a change in its life cycle state.

The life cycle of a product item is a set of state transitions. For example, a source file has the following states:

State 1: Under work. The product item is planned or it is being worked on under the control of the engineer. The engineer completes the work and moves the item revision to the next state.

State 2: In review. From the engineer's point of view, the product item is completed and

ready for review. The item revision is mailed to the lead engineer, where it goes into the lead engineer's "To Do List" or pending tray. The lead engineer looks at the source and related documentation, then moves the item revision to the next state.

State 3: Approved. The product item has been checked and can be used or released.

The person who performs an activity on the life cycle has a role on the project. Development work is performed by those who have a role within the product sub-tree being developed; for example, engineer for "Operator GUI."

The role is the part you play within the development process. For example, you may be the engineer for your own Show Flight Status module and a tester for the PC Controls subsystem. Your user role is the "hat you are wearing" when you perform an activity on a physical item.

Since personnel and circumstances may change, PCMS provides the necessary administrative functions to assign and maintain users' roles within the design part structure, and to delegate responsibilities for functions such as role assignment and design part creation.

The development process includes the following:

- Analysis and design until the product is ready to be built
- Change
- Build and test loop
- Component assembly into product release sets

Product Item Type	Description	Comments
DOC	Any project documents that are not defined explicitly in the control plan, such as progress reports, reference information, interface definitions	
MIN	Minutes from project meetings, such as technical, review, or quality meetings	Major actions from these meetings may be held as change requests or internal change documents to track progress
RS	Requirements specification, usually associated with the product or subsystem	For interface to requirement tools define item types `RQT_RDR, RQT_RSD`
UG	User Guide(s)	
FS	Functional specification	
DS	Design specification	
MS	Module specification to describe the module being developed	Could be generated from CASE tool
SRC	Source files, C++, ADA, BASIC	Define formats for source items in the CM plan
OBJ	Object code files compiled from SRC	By keeping this item under control the build process can do minimal rebuilds by recompiling only those items that have changed
TPL	Test plan	
TPR	Test procedures	
PBL	Product baseline item	Used to track the details and handover of a baseline during the product development, such as a system baseline from development to systems test to customer acceptance testing; it can exist at any level and will reside at the top-level design part of the product baseline
DI	Development item	For engineers and developers to keep notes and supporting project documentation

Figure 3. Product items with a workset for Flight Simulator

The supporting processes may be used for activities that are repeatedly used during the development activities.

Role names that describe those involved in the processes are generic. In smaller or less formal organizations, one individual may cover more than one role as defined in this section. The development process breaks down into phases, each with its own set of deliverables: some to the end customer and some to the next stage in the development process. Deliverables from each stage in the process are the main objects to be controlled and managed by PCMS as product items. After each phase, a product baseline should be taken to capture a snapshot of the documents and project files. This baseline can serve as a reference point for the next phase, and management reports can show the progress.

Figure 4 shows the roles involved in the development processes.

Figure 5 lists the tasks in the development process and the roles that are either involved or responsible.

Development Process Roles

TM	Test Manager
DM	Development Manager
LE	Lead Engineer
LA	Lead Analyst
RM	Release Manager
ENG	Engineer
BUI	Builder
AN	Analyst
DE	Design Engineer
AUT	Author
TE	Tester
CM	Configuration Manager
ITM	Integration Test Manager
STM	System Test Manager

Figure 4. Roles in the development process

DEVELOPMENT TASK	RESPONSIBILITY*							
	AN	LA	ENG	LE	TM	DM	RM	CM
Analysis								
Prepare Analysis docs	R	I						
Review Analysis docs	I	R		I				
Prepare model	I	R		I				
Review analysis deliverables	I	I		I				
Make analysis Baseline	I	R				I		
Verify Baseline contents				I				
Develop product acceptance criteria	I	I		I	R	I		
Review product acceptance criteria	I	I		I	I		R	I
Design								
Prepare Design docs	I	I		I				
Prepare Int and System Test docs		I		I	R	I		
Prepare Design model	I	I		R				
Review Design model	I	I		I				
Review Design docs				R	I	I		
Review Int and System Test docs		I		I	I	I	R	
Extend PCMS design structure				R				I
Prepare data model						R		
Make functional Baseline	I		I	R				
Verify Baseline contents			I	I		R		

*I=Involved, R=Responsible

Figure 5. Development process tasks

The product life cycle represents the state of a product at any point in time. After a subsystem or the whole product (for example, Flight Simulator) is completed, a product baseline file is created and included in the Product Baseline (PBL). The item representing the PBL documents the transition from the development team to the release manager and the test teams.

The Product Baseline uses the life cycle shown in Figure 6. As the baseline is released into the various environments, the PBL file moves through the life cycle. The product baseline should be created at the product or subsystem level of the design part structure.

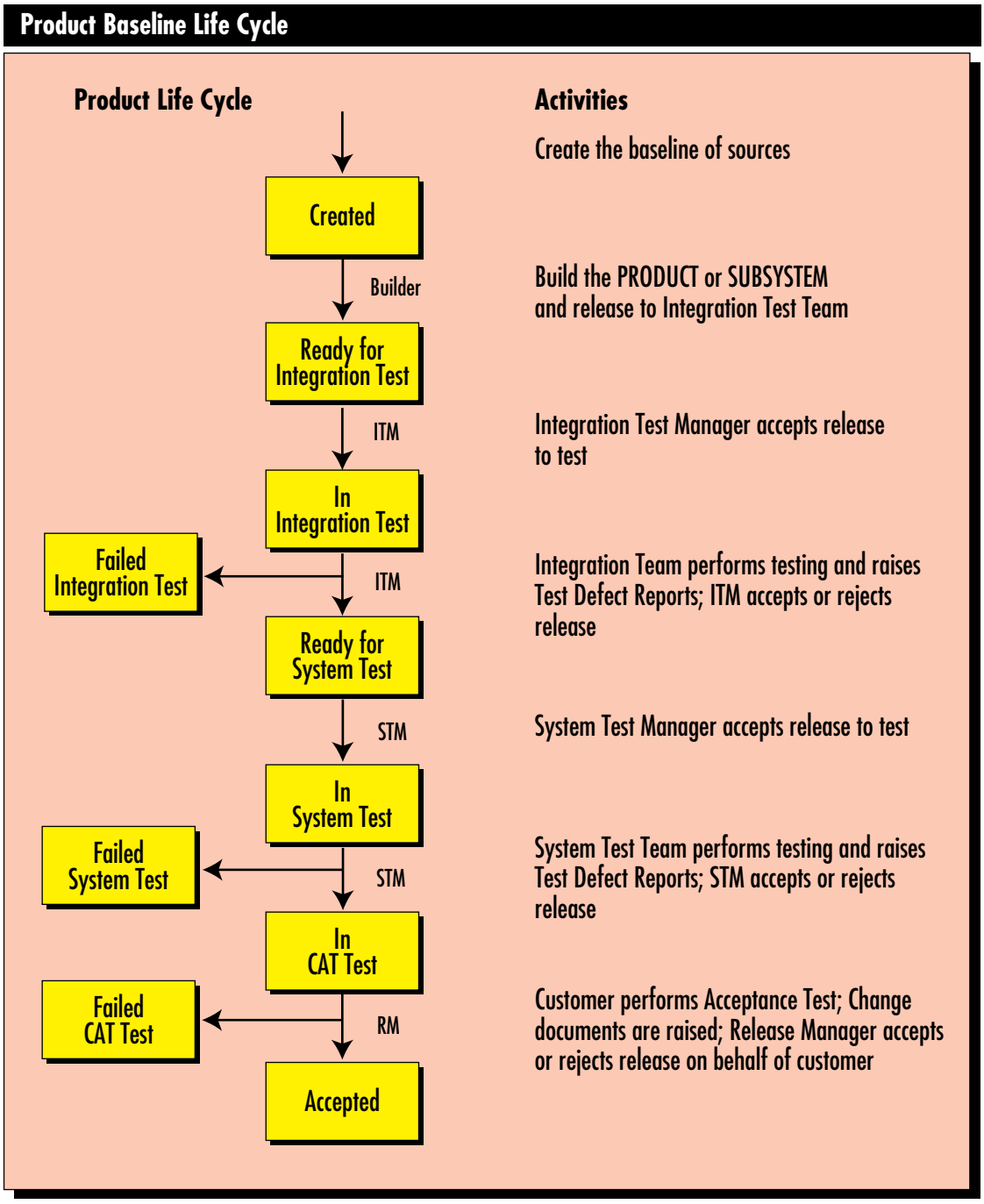


Figure 6. Product baseline

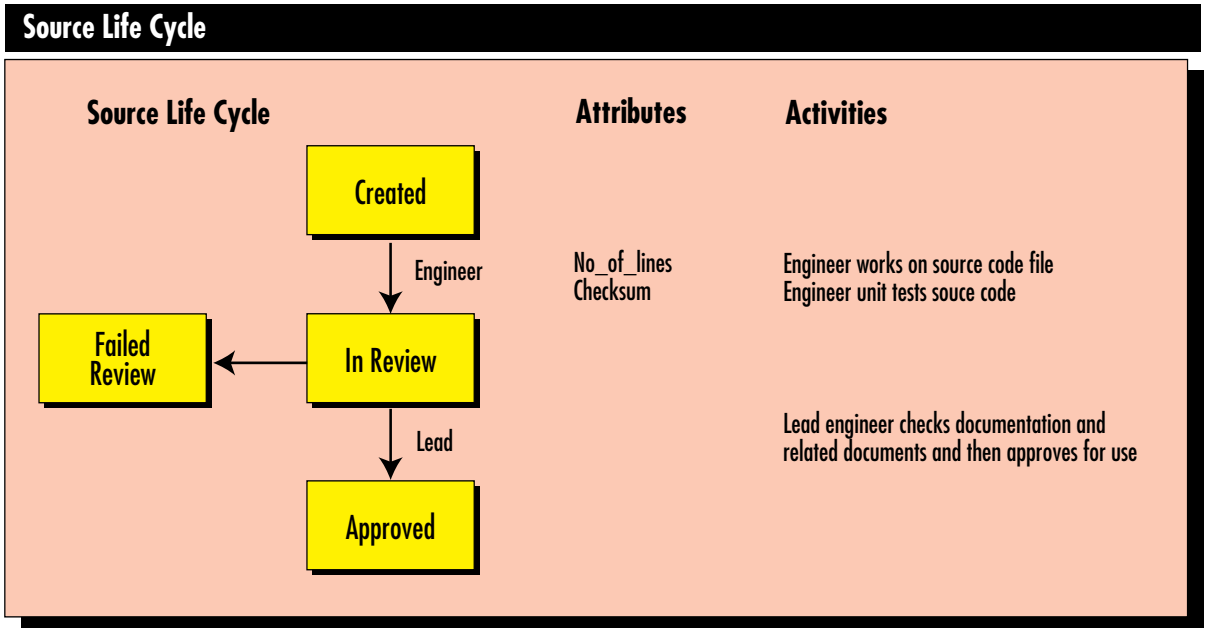


Figure 7. Source life cycle

Source Life Cycle

Source files are used for building software products. For PCMS build purposes, a single PCMS item type may differ in item formats. For example, one SRC item may have a C format, another C++, and another ADA. Using the PCMS browse and edit scripts, specific editors can be associated with PCMS items, based upon their item type and format. The following are associated item types:

SRC	Source files
INP	Source screen files
DAT	Data files
DBS	Database script files

A product item will have a life cycle that is a set of state transitions, such as a source file (SRC), shown in Figure 7.

A PCMS*API event may be used to derive the attribute values `no_of_lines` and `checksum`. Other metrics attributes can be added.

The Maintenance Process

Typically, maintenance is concerned with maintaining development releases and performing bug fix and patch releases. In PCMS, maintenance can be effectively modeled by using the release baseline and capturing all amendments within the change documents. Then amendments can be applied to the original release baseline via the set of change documents using Create Revised Baseline (CRB).

Figure 6 shows this scenario within the development process. Maintenance, however, must address issues arising from the help desk and support environment. Collating changes into appropriate work packages provides a convenient mechanism for planning maintenance activities. It enables both problems and requests to be processed and placed in a work package that represents the work to be done and the time frame.

The change management process and the use of work packages is important to maintenance. An overview of the change management process and an example of a PCMS life cycle is shown in Figure 8. The roles in the change control process are as follows:

CRV	Change Reviewer
CA	Change Assessor
CL	Change Lead Engineer
CE	Change Engineer
CB	Change Board
CM	Change Manager

Reporting

SQL Reporting generates comprehensive standard reports. These reports allow you to track

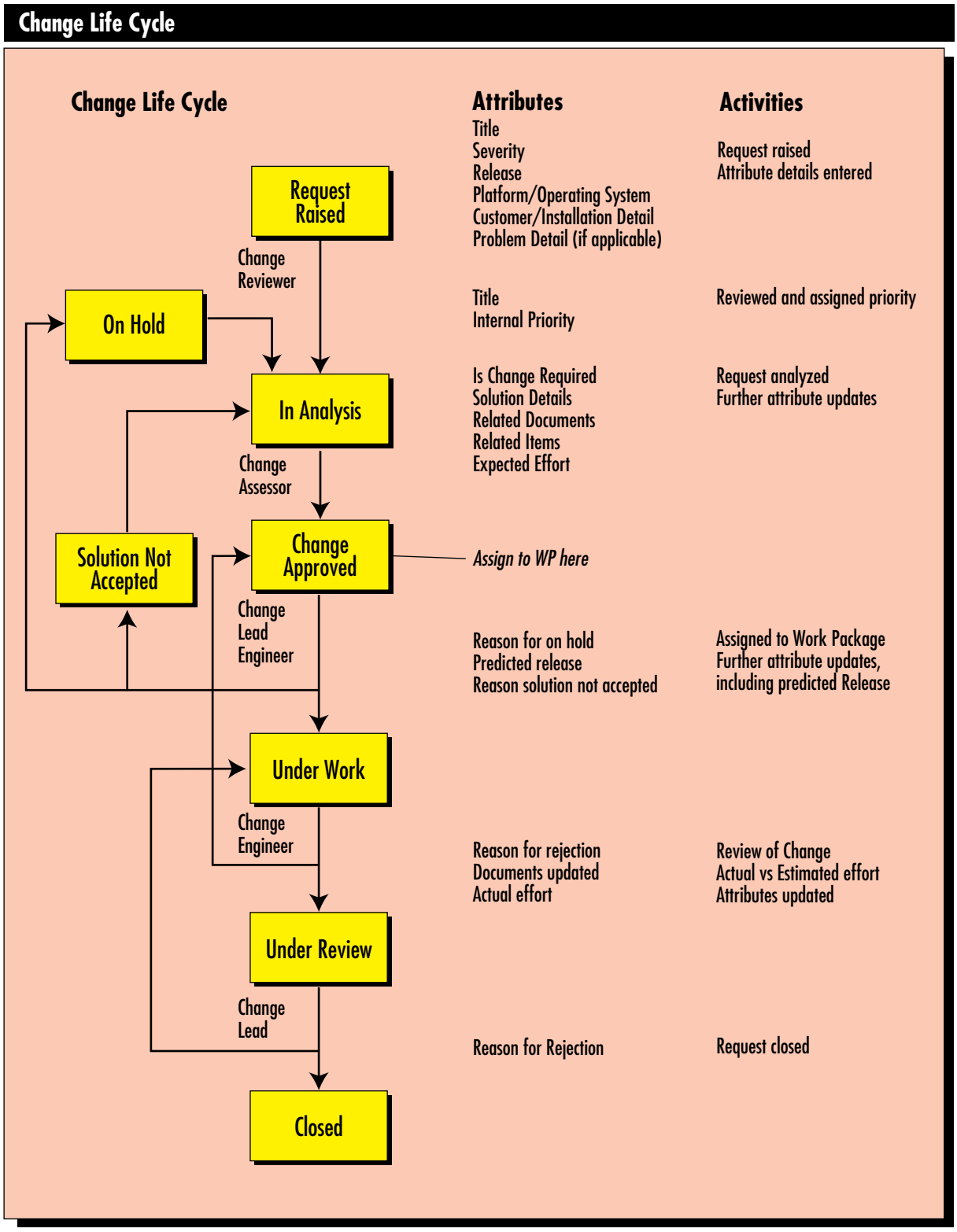


Figure 8. The change management process and PCMS life cycle

the history of development objects and change documents, such as defect reports and other statistics from the relational database used by PCMS. Since PCMS is an active system, it provides a dynamic rather than a static view of your change process. Therefore, managers can review the status of any part of the project at any time in order to assess impact, take actions, and plan ahead.

Benefits

Usually when a change request reaches a particular state, change activity begins against various types of information to correct that change request. This could be a combination of design documentation, source code, user documentation, test plans, and test suites, as shown in Figure 9.

Connecting and packaging this changed information against the change controls automates the correction process. Creating a change package, grouping together multiple physical changes as a change package, and tying this change to a particular state in the change life cycle, can offer important benefits including the following:

- Automation of changes from development to test to release
- Faster testing turnaround times
- Reduced compilation and linkage cycles
- Improved quality over the contents of a release
- Faster generation of release notes and product errata

Migration often fails because of poor change packaging. Testing builds fail because the correct set of changed objects cannot be created. Changes are left behind in development directories, and engineering hours are lost trying to diagnose the reason that test builds fail. This increases the impact of the compile on the machine and adds effort to each testing cycle. Much wasted time is eliminated by repository tracking and controlling these packaged changes.

If the dependencies between the programming, quality assurance, and documentation are tracked and packaged, overall quality improves with each change activity. In order to create the test suite, the test group must have a solid understanding of the changes to the design and the code. Likewise, the documentation group must know what design changes will occur so that the documentation matches the operation of the product.

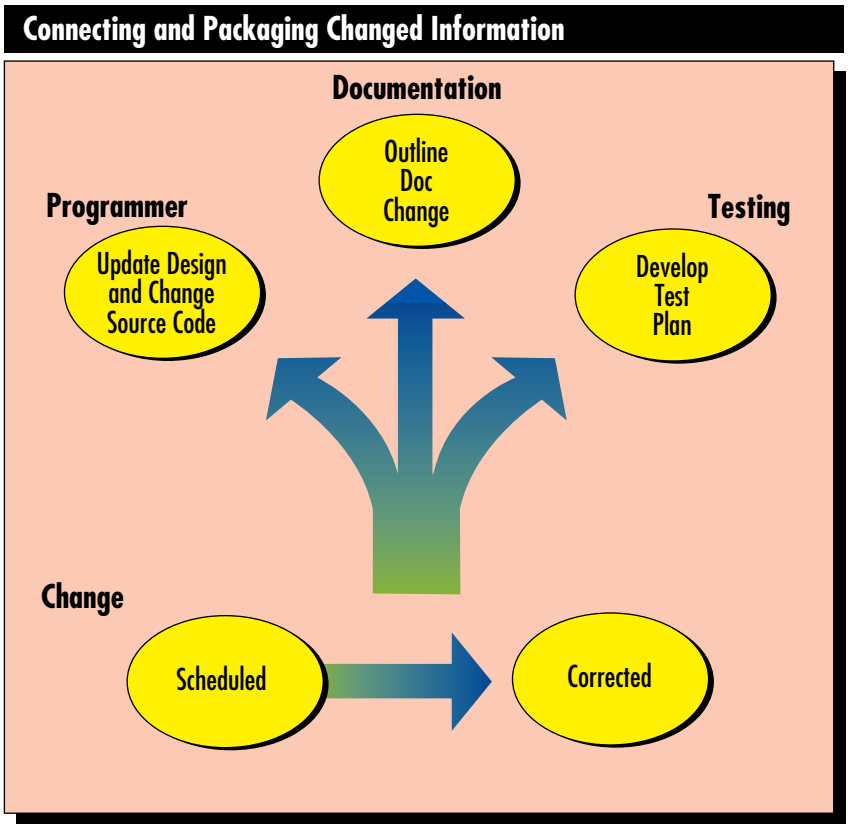


Figure 9. Changed information

As changes are packaged and rolled out into the next release, summarizing the physical changes included in that release accelerates release note documentation and provides valuable knowledge transfer to customer support/help desk personnel who must support the application. If the application is a commercial product, this transfer of knowledge can extend to the field sales force and technical consulting staff—providing great benefits to the organization.

Sohail Haque, SQL Software, 8500 Leesburg Pike, Vienna, VA 22181. 703-760-0448. Internet: info@sql.com. Mr. Haque is director of Technical Services. He has over nine years of experience in Process Configuration Management.