

IEEE Guide for Developing Software Life Cycle Processes

Sponsor

**Software Engineering Standards Committee
of the
IEEE Computer Society**

Approved 21 September 1995

IEEE Standards Board

Abstract: Selected topics covered in IEEE Std 1074-1995, IEEE Standard for Developing Software Life Cycle Processes, are addressed in this guide. The guide provides assistance with Software Life Cycle Model (SLCM) selection, Activity mapping, and management of a software Life Cycle (SLC).

Keywords: software life cycle, processes, software life cycle model, software life cycle process, activities, mapping

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

Copyright © 1996 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1996. Printed in the United States of America.

ISBN 1-55937-589-2

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying all patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; (508) 750-8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Introduction

(This introduction is not part of IEEE Std 1074.1-1995, IEEE Guide for Developing Software Life Cycle Processes.)

A well-defined and carefully followed process is essential to the development and maintenance of high-quality software. IEEE Std 1074-1995 specifies the Activities that constitute a well-defined process. The goals of the Standard and this guide are to provide information and methods that an organization can use to build robust, complete Software Life Cycles for their software development and maintenance. The use of a software life cycle should result in such benefits as reduced cycle time, high quality software, increased customer satisfaction, and a basis for continuous process improvement.

Use of this guide can assist in understanding and applying IEEE Std 1074-1995, IEEE Standard for Developing Software Life Cycle Processes.

Work on this standard began in September 1990. A total of 10 meetings produced the draft submitted for ballot in November 1993. Three ballot resolution meetings were held and the draft was submitted for recirculation in April 1995.

The developers of this document remember with gratitude the contributions of Mr. Joseph Guidos. Joe was a founding member of both this working group and the working group for IEEE Std 1074-1991. He passed away during the development of this Standard.

This standard was developed by a working group consisting of the following members who attended two or more meetings, provided text, or submitted comments on more than two drafts of the standard:

Dennis E. Nickle, *Chair*

David J. Schultz, *Vice Chair*

David W. Burnett and Susan M. Burgess, *Configuration Managers*

Lynn D. Ihlenfeldt and David W. Burnett, *Secretaries*

H. Ronald Berlack
Susan M. Burgess
Larry W. Campbell
Kristin Dittmann
Jean A. Gilmore
Art Godin
Joseph Guidos*

Daniel Gray
Rob Harker
John W. Horch
John O. Jenkins
Linell Jones
Robert Kierzyk
Pat Marcinko

Keith R. Middleton
Noritoshi Murakami
John F. Pellegrin
Robert Shillato
John Swearingen
Diane Switzer
Allan Willey

*Deceased

The following individuals also contributed to the development of the standard by attending one meeting or providing comments on one or two drafts:

Robin Allen
G. Dusty Barlow
Sharon Beckers
John Blyskal
Cornelia Boldyreff
Chris Booth
Hu Cheng
Geoffrey Darnton
Susan M. East
Gabriel Eckert

Mehmet Ficici
Kay Greig
James R. Hughes
Scotty Kilbourne
Sam Godfrey
Thomas M. Kurihara
J. Dennis Lawrence
Renato Luise
Denis C. Meredith
Martin F. Owens
Patricia Remacle
Norman Schneidewind

Tony Sgarlatti
Jim Shimp
Ed Slapsys
David J. Smith
Harry Sneed
Kelley Stalder
Ann Sullivan
Yoshihiko Tokumaru
Haynes Turkle
Janusz Zalewski
Geraldine Zimmerman

The following persons were on the balloting committee:

A. F. Ackerman	Jean Gilmore	Bino Nanni
Oddur Benediktsson	John Garth Glynn	Ann Marie Neufelder
H. Ronald Berlack	Donald Gotterbarn	Dennis E. Nickle
B. P. Bhat	Lawrence M. Gunther	Mike Ottewill
Mark Bilger	David A. Gustafson	Indradeb P. Pal
Ronald Blair	John Harauz	Joseph A. Palermo
William J. Boll	Rob Harker	Patricia Rodriguez
Sandro Bologna	Matthew S. Harriss	Sergiu Samuel
Fletcher Buckley	William Hefley	Stephen R. Schach
Susan M. Burgess	Mark Heinrich	Norman Schneidewind
David W. Burnett	John W. Horch	Wolf A. Schnoege
W. Larry Campbell	Lynn D. Ihlenfeldt	David J. Schultz
Neva M. Carlson	John O. Jenkins	Gregory D. Schumacher
Francois Coalier	David Johnson	Leonard W. Seagren
Geoff Cozens	Robert J. Kierzyk	Tony Sgarlatti
Michael A. Daniels	Scotty Kilbourne	Robert W. Shillato
Geoffrey Darnton	Peter Klopfenstein	David M. Siefert
Taz Daughtrey	Robert Kosinski	Alfred R. Sorkowitz
Neil Davis	Thomas M. Kurihara	Vijaya Srivastava
Raymond Day	Helmut Kurzdorfer	Alfred Strohmeier
Bostjan K. Derganc	Boniface Lau	Robert N. Sulgrove
Leo G. Egan	J. Dennis Lawrence	George D. Tice
Richard L. Evans	Fang Chin Lim	Leonard L. Tripp
Caroline L. Evans	Ben Livson	Margaret C. Updike
John W. Fendrich	Joseph Maayan	Tom Vaiskunas
Jon J. Fineman	Hosakere N. Mahabala	Dolores Wallace
Eitan Froumine	Harold Mains	William M. Walsh
Simon Gabrihelidis	Pat Marcinko	Richard Werling
Juan Garbajosa-Sopena	Robert A. Martin	Natalie C. Yopconka
Yair Gershkovich	Roger Martin	Janusz Zalewski
	Sue McGrath	

When the IEEE Standards Board approved this standard on 21 September 1995, it had the following membership:

E. G. “Al” Kiener, *Chair*

Donald C. Loughry, *Vice Chair*

Andrew G. Salem, *Secretary*

Gilles A. Baril
Clyde R. Camp
Joseph A. Cannatelli
Stephen L. Diamond
Harold E. Epstein
Donald C. Fleckenstein
Jay Forster*
Donald N. Heirman
Richard J. Holleman

Jim Isaak
Ben C. Johnson
Sonny Kasturi
Lorraine C. Kevra
Ivor N. Knight
Joseph L. Koepfinger*
D. N. “Jim” Logothetis
L. Bruce McClung

Marco W. Migliaro
Mary Lou Padgett
John W. Pope
Arthur K. Reilly
Gary S. Robinson
Ingo Rusch
Chee Kiow Tan
Leonard L. Tripp
Howard L. Wolfman

*Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal
Richard B. Engelman
Robert E. Hebner
Chester C. Taylor

Angela M. Girardi
IEEE Standards Project Editor

Contents

CLAUSE	PAGE
1. Overview	1
1.1 Scope.....	1
1.2 Purpose.....	1
1.3 Prerequisites	1
1.4 References.....	1
1.5 Definitions and acronyms	2
1.5.1 Definitions.....	2
1.5.2 Acronyms.....	2
1.6 Organization of this guide.....	2
2. General concepts of the Standard.....	3
2.1 Process Standard.....	3
2.2 Compliance	3
2.2.1 Upward adaptation	4
2.2.2 Downward adaptation.....	4
2.3 Applicability.....	4
2.4 Intended audience.....	5
2.5 How to start using the Standard	5
2.5.1 SLCM	5
2.5.2 Project type	5
2.5.2.1 New projects.....	6
2.5.2.2 Existing projects	6
2.5.2.3 Maintenance projects.....	6
2.6 SLC, SLCM, and methodology.....	6
2.7 Organizational concerns	6
3. Mapping guidelines	7
3.1 An approach to mapping.....	7
3.1.1 Step one—select the SLCM.....	7
3.1.2 Step two—compare Activities to SLCM requirements.....	7
3.1.3 Step three—placing the Activities in time sequence	7
3.1.4 Step four—checking information flow.....	8
3.1.5 Step five—assigning information to documents.....	8
3.1.6 Step six—add actual dates and times	8
3.1.7 Step seven—reconcile the constraints.....	8
3.1.8 Step eight—maintain the schedule and the SLC	8
3.2 SLCM + Activities = SLC	8
3.2.1 SLCM selection.....	9
3.2.1.1 Identification of available SLCMs	9
3.2.1.2 Identification of project attributes	9
3.2.1.3 Identification of project constraints.....	9
3.2.1.4 Evaluation of various SLCMs using lessons learned.....	10
3.2.1.5 Selection of SLCM	10
3.2.2 Instructions for mapping Activities.....	10
3.2.3 Ways to map Activities	10
3.2.3.1 Single instances of Activity	11
3.2.3.2 Multiple instances of an Activity	11
3.2.3.3 Invoked Activity.....	11
3.2.4 Activity time ordering	11

CLAUSE	PAGE
3.2.5 Determining the adequacy of the SLC	12
3.2.6 Creating generic SLCs.....	12
3.2.7 Software as a part of a hybrid (hardware and software) system or a larger software system	13
3.3 Information tracing	13
3.4 Hidden information and tasks	15
3.4.1 Hidden information	15
3.4.2 Hidden tasks	15
3.4.2.1 Invoking Integral Processes.....	15
3.4.2.2 Management Processes	16
3.4.2.3 Project changes and maintenance.....	16
3.5 Information mapping	16
3.6 Mapping constraints.....	16
4. Concepts of the Standard used in mapping.....	17
4.1 Time ordering.....	17
4.2 Iterations and instances.....	17
4.3 Ownership	18
4.4 Integral Processes	18
4.4.1 An example of an Integral Process Invocation	19
4.4.2 Invoked Processes invoking other Processes	19
4.5 Management Processes	20
4.5.1 Advanced planning.....	20
4.5.2 Ongoing management.....	20
4.5.3 Quality management from start to finish.....	21
4.6 Risk management	21
4.7 Maintenance	22
4.8 Retirement.....	22
4.9 Reuse and the SLC	23
ANNEXES AND INDEX	
Annex A (informative) Mapping example.....	25
Annex B (informative) Annotated bibliography	41
Index	46

IEEE Guide for Developing Software Life Cycle Processes

1. Overview

1.1 Scope

This guide addresses selected topics covered in IEEE Std 1074-1995, IEEE Standard for Developing Software Life Cycle Processes. The guide provides guidance in Software Life Cycle Model (SLCM) selection, Activity mapping, and management of a Software Life Cycle (SLC).

1.2 Purpose

The purpose of this guide is to suggest approaches to the implementation of IEEE Std 1074-1995, IEEE Standard for Developing Software Life Cycle Processes (referred to in this guide as “the Standard”). The guide provides clarifications and definitions for various sections of the Standard through examples and tutorials.

This guide is written for any organization or individual responsible for the total or partial implementation of the Standard. This guide offers suggestions for application of the Standard. It is not offered as a detailed procedures manual for developing or maintaining software or software life cycles.

1.3 Prerequisites

Neither IEEE Std 1074-1995 nor this companion document specifies or presupposes a particular SLCM. The Standard does, however, require the user to select a life cycle model to use on any particular software project.

1.4 References

This guide is a companion document to IEEE Std 1074-1995 and is intended to be used in conjunction with that Standard. These are the primary references in this guide:

IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology (ANSI).¹

IEEE Std 1074-1995, IEEE Standard for Developing Software Life Cycle Processes.

A list of other IEEE standards, which may be consulted for additional guidance, is given in Annex B. Although this guide does not require adherence to any other IEEE standard, knowledge of principles and concepts described in the listed standards would be helpful.

¹IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

1.5 Definitions and acronyms

1.5.1 Definitions

The definitions listed here establish meanings within the context of this guide. Definitions of other terms used in this document can be found in [B1]² or [B2]. Terms in the references are not repeated here.

- 1.5.1.1 Constraints:** Restrictions, resources, rules, etc., that limit Software Life Cycle Model selection, project planning, and management.
- 1.5.1.2 Integral Process:** A Process that is needed to complete project Activities successfully , but is outside the management and development processes. The Integral Processes are Verification and Validation, Software Configuration Management, Documentation Development, and Training.
- 1.5.1.3 Invoked Process:** That portion of an Integral Process that is called like a subroutine.
- 1.5.1.4 Process Architect:** The person or group that manages the implementation of the Standard in an organization.
- 1.5.1.5 The Standard:** IEEE Std 1074-1995.

1.5.2 Acronyms

The following acronyms appear within the text of this guide:

IEEE Institute of Electrical and Electronics Engineers
SCM Software Configuration Management
SLC Software Life Cycle
SLCM Software Life Cycle Model
SPMPI Software Project Management Planned Information
SQM Software Quality Management
V&V Verification and Validation

1.6 Organization of this guide

Clause 1 discusses Scope, Purpose, Prerequisites, Reference Documents, Definitions, and Acronyms.

Clause 2 discusses the major concepts used in the Standard. This section provides insight into various managerial and procedural concerns that may arise during the development of a SLC.

Clause 3 provides general instructions for performing the mapping operation required by the Standard. Mapping the Activities presented in the Standard to the user's SLCM, thus producing a time-ordered SLC, is explained. The section provides instructions for mapping, and discusses the related topics of information tracing, information assignment, mapping constraints, and other considerations.

Clause 4 provides additional guidance related to mapping and compliance to the Standard. Frequently asked questions related to the application of the Standard are addressed.

Annex A provides mapping examples.

Annex B lists other IEEE standards in an annotated bibliography that may be consulted for additional guidance.

²The numbers in brackets correspond to those of the bibliographical references in Annex B.

2. General concepts of the Standard

Since this guide is a companion to the Standard, the guide provides clarifications and implementation guidance based on the Standard. A description of the key concepts of the Standard is provided for reference and understanding.

To use the Standard, a Software Life Cycle Model (SLCM) must be available to the organization. The Standard does not dictate which SLCM to use because the user is required to map Activities to whichever SLCM (e.g., waterfall, prototype, etc.) is selected.

The 65 Activities are the heart of the Standard. (Three of these Activities are “If Applicable,” and discussed in 2.2.) These comprise the minimum set of Activities that must be performed to be compliant with the Standard. Activities provide a logical approach to the development, operation, maintenance, and retirement of software. In this context, the Processes serve as convenient groupings of related Activities. The Input Information and Output Information of each Activity are listed in tables that accompany the Activity descriptions.

2.1 Process Standard

The Standard is a Process Standard. All of the Activities are intended to be molded by the Process Architect into a coherent, effective, project-specific development or maintenance process. Information in the Input and Output Information tables is intended to reflect the flow of information, not the documentation set to be produced.

Figure 1 shows the connections among the Software Life Cycle Model Process, Project Management Processes, Development Processes and Integral Processes. Note that the Management and Integral Processes span the development effort, and are connected to all sets of Development Processes. The Software Life Cycle Model Process, at the far left in the figure, precedes all the rest.

2.2 Compliance

In the Standard, there are two types of Activities: “mandatory” and “if applicable.” The Standard specifies the conditions under which an “if applicable” Activity becomes “mandatory.” Compliance with the Standard requires the completion of all mandatory Activities and invocations. This requires that all Input Information to these Activities be processed and that all Output Information be produced.

Adaptation is the addition or deletion of Activities in the use of the Standard. For the purpose of compliance with the Standard, a Software Life Cycle (SLC) may be adapted only by the addition of new Activities. Adaptation would be accomplished during the SLCM mapping Activities.

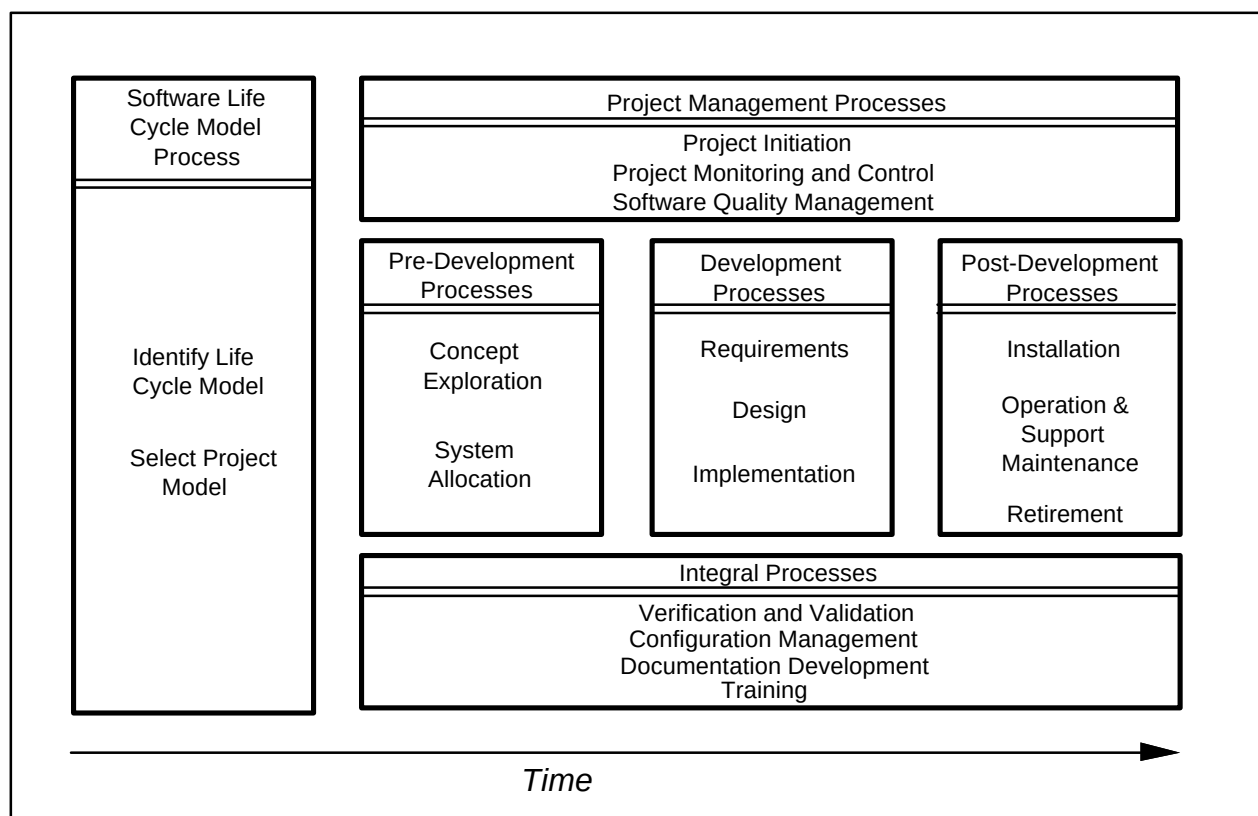


Figure 1—Inter-process connections

2.2.1 Upward adaptation

The Standard contains a minimum set of Activities needed to be used in developing software systems. Users may find that their selected SLCM, organization, or contracting organization requires additional Activities to be performed. Examples of additional Activities include, but are not limited to, safety, security, or subcontractor Activities. These additional Activities are mapped together with the Standard Activities.

Upward adaptation does not affect compliance with the Standard as long as all of the mandatory Activities are fully completed.

2.2.2 Downward adaptation

Downward Adaptation is the intentional exclusion of one or more of the Standard's mandatory Activities. Exclusion of any mandatory Activity creates a software development process that is not compliant with the Standard.

2.3 Applicability

The Standard applies to any software development or software maintenance project. The user of the Standard must understand the applicability of the Standard to the particular project. The user must determine the starting point: is there an existing SLCM or must one be developed? See 2.5 for further discussion.

2.4 Intended audience

The Standard's first audience is the person or group who will determine the implementation of the Standard's Activities. In this guide this audience is called the Process Architect. The selection of the Software Life Cycle Model and the mapping of Activities are the primary responsibilities required of the Process Architect.

The Process Architect manages the implementation of the Standard in the organization. Mapping all the Activities of the Standard to create a workable SLC requires a thorough understanding of software development and maintenance, and all the related support functions. The Process Architect should have the following:

- a) Knowledge of the organization and the way it does software development and maintenance
- b) The authority to effect a software development and maintenance SLC
- c) Knowledge of software development and maintenance methodologies, tools, and standards
- d) Knowledge of IEEE Std 1074-1995

After the Process Architect has mapped the Activities, the Activities' descriptions and requirements can be distributed to their organizational Owners. The second audience for the Standard is these Owners. See 4.3 for a discussion of Owners.

2.5 How to start using the Standard

Application of the Standard depends on two conditions: the presence or absence of an SLCM and the type of the project to which the Standard is to be applied.

2.5.1 SLCM

If an SLCM has not been previously selected by the organization, the Standard is "entered" at Activity 2.3, "Identify Candidate Software Life Cycle Models."

If an SLCM is in place, it should be reviewed for its applicability to the project. If it is applicable, Activities 2.3 and 2.4 should be mapped as completed and the Standard entered at Activity 3.1. If the SLCM is not applicable, a new SLCM may have to be chosen and the Standard entered at Activity 2.3.

2.5.2 Project type

Projects may be new, existing, or maintenance projects.

2.5.2.1 .New projects

For a new project, the Process Architect first selects an appropriate SLCM, as described above. Next, the Activities of the Standard are mapped to the selected SLCM. Some Activities may already have been started or completed. Any such Activities should be mapped and their current status recorded. SLCM selection and Activity mapping are described in detail in clause 3.

2.5.2.2 Existing projects

The use of the Standard for existing projects may require the mapping of previously completed Activities. Those Activities which have been completed will still be mapped and their current status recorded. An existing project for which full IEEE Std 1074-1995 compliance is to be claimed must show all mandatory Activities and evidence of their completion.

In the case where only the applicable, remaining Activities of the Standard are to be used, the starting point will be determined according to the status of the project when the Standard is to be activated. The starting (or entry) point will reflect the remaining Activities of the project.

2.5.2.3 Maintenance projects

The maintenance of software, according to the Standard, is the reapplication of a software development life cycle. As in the case of existing projects discussed above, the reentry point for the Standard will depend on the project's current status.

2.6 SLC, SLCM, and methodology

The terms "Software Life Cycle," "Software Life Cycle Model," and "methodology" are often used interchangeably within the computer industry. This guide utilizes these phrases as defined in IEEE Std 1074-1995.

An SLCM incorporates a methodology for the development and maintenance of software. The SLC is generated from the Standard's Activities mapped to the SLCM.

2.7 Organizational concerns

All of the Activities comprising the Standard are collected for convenience into Processes. Process names like Training, Verification and Validation, and Software Configuration Management could be the names of groups or functions in an organization. There is no intent on the part of the Standard to associate Processes with specific groups or individuals within an organization.

The Activities in each Process are performed by the most appropriate entity within each organization.

For example, Training in the use of development tools might be assigned to the senior member of a development group, whereas user training might be the responsibility of a training group. The Standard does not require any specific, functional Training Group or organization. Another example is the case of Verification and Validation (V&V). The Standard does not require a V&V organization, just that V&V Activities be performed.

3. Mapping guidelines

The underlying concept in this guide is the mapping of the required Activities of the Standard to the selected Software Life Cycle Model (SLCM). Mapping is accomplished by inserting each instance of each Activity into the SLCM (See 4.2). This mapping creates the Software Life Cycle (SLC) to be used for the software development or maintenance project. The following subclauses describe mapping and present the general concerns that the Process Architect should keep in mind while mapping Activities from the Standard to the selected Software Life Cycle Model.

The Standard does not specify any particular SLCM as a starting point for mapping. Likewise, this guide will not specify a particular mapping method, or scheme, other than in general terms. It is intended that organizations select their own SLCM(s) and develop the mapping method which is most appropriate for that organization. If there are requirements specified by an organization, but not specified in the Standard, they are to be included with the Standard's Activities during mapping. Contractual requirements or constraints may also add Activities to be mapped. Annex A gives an example of a mapping exercise using both a specific SLCM and a specific mapping approach. These are examples only, to give the Process Architect a sample of one way to map.

3.1 An approach to mapping

The balance of this clause will suggest a series of steps which an organization may follow while developing its own mapping approach. These steps are suggestions only. The intent of the series of steps below is to present a sequential mapping approach in which each step introduces more information or responds to increasingly critical external constraints. Additional information about mapping is provided in 3.2 through 3.6.

3.1.1 Step one—select the SLCM

Step one, as specified by the Standard, is to identify the SLCM to that the Activities will be mapped. This step may mean that the whole process of locating, evaluating, selecting, and acquiring an SLCM must be performed (see 3.2.1). An organization may already have an SLCM. If this is true, at this time the Process Architect should review the SLCM to ensure it is appropriate for the specific project.

3.1.2 Step two—compare Activities to SLCM requirements

Once the SLCM has been selected, a detailed mapping of the Standard's Activities is performed against the SLCM. Annex A to both the Standard and this guide depict this level of mapping. It is merely the matching of the Activities against the requirements of the SLCM. This step provides a checklist to assure that all Activities are mapped and that all SLCM requirements are covered by Activity(ies). Activities not mapped will be noted in the List of Activities Not Used, as required by the Standard.

Identification of Activity Owners (see 4.3) begins at this time, and is completed prior to the initiation of the owned Activity.

3.1.3 Step three—placing the Activities in time sequence

In this placement, each Activity's starting point is mapped to a general timeline. The timeline need not have actual dates for this mapping. Its primary purpose is to show relative timing and sequencing between Activities. It also serves as a basis for identification of inter-Activity dependencies. The output of this first mapping is time-ordered in the sense of relative starting and ending of Activities with relation to other Activities. This is the definition of an SLC in the Standard. Revisions of this initial sequencing will probably be necessary as more information is considered in succeeding mapping steps.

It is at this point that the first mapping of the Invoked Process Activities will be done. More information is provided for the full mapping of invoked Activities in 4.4.

3.1.4 Step four—checking information flow

The Input and Output Information tables in the Standard specify the information that is to be used and generated by each Activity. This step verifies that the information flow into and out of the Activities will support the relative order into which they have been mapped. While it is unlikely that this will cause a major rearrangement or modification of the mapping from Step 3, it is a necessary check to be sure that all information will be available to the Activities that need it, when they need it.

3.1.5 Step five—assigning information to documents

Each SLCM requires and defines formatted content of its own set of output products. These products are, for the most part, the specific documents that the SLCM delivers. Note that the term “document” does not imply any particular medium. This step compares the Output Information generated by each Activity with the SLCM-required document(s) into which it must go.

Once again, the order of the mapping, this time from Step 4, may have to be modified. If a particular document, specified by the selected SLCM, is to be created at a particular point in the development schedule, all the Activities which contribute information to be recorded in that document must have had an opportunity to generate it.

Identification of external owners (see 4.3), if needed, occurs at this time.

3.1.6 Step six—add actual dates and times

Up to this point, all time ordering could have been accomplished on a merely relative basis, without regard to the actual days, weeks, or months involved. Step 6 now adds the actual date and time specification to the mapping so a project schedule emerges. The outcome of this step is a time-ordered mapping of the Activities to the SLCM, related to real-world time.

3.1.7 Step seven—reconcile the constraints

The realities of external constraints on time available for the project, its personnel resources, interfaces with other portions of the software’s environment, etc. may require still another set of modifications to the mapped Activities. The output from this step is the initial SLC for use on the project and the corresponding project schedule.

3.1.8 Step eight—maintain the schedule and the SLC

Once the SLC and the project schedule have been developed and the project is under way, new factors will come to light which require modifications to the project schedule. These changes may also lead to improvements in the SLC. Improvements to the SLC for this project may in turn lead to better mapping techniques within the organization.

As implied at the beginning of this section, each organization’s mapping method is expected to evolve over time and with experience. This step contributes to that evolution.

3.2 SLCM + Activities = SLC

This section provides additional information on the creation of SLCs from their constituent parts: the selected SLCM and the Activities of the Standard. Detailed guidance is provided in the techniques of SLC creation and the use of SLCs in the special situations: generic (not project specific) SLCs and SLCs that are components of a larger system.

3.2.1 SLCM selection

Each development project is different from those in the past, either in the type of project, availability of tools and hardware, or experience level of the developers and managers involved. There have been many models developed since the waterfall model was proposed, and some or all of them are being used to develop software. One organization may have a “company solution” model that must be followed. In another organization a project manager may have the latitude to modify a particular model, or combine models, to suit the project and its software life cycle management. The following section contains basic evaluation criteria for this selection process.

The process for evaluating and selecting a SLCM has five basic steps, as follows:

- a) Identify all the SLCMs available to the development organization
- b) Identify the attributes that apply to the desired end system and the development environment
- c) Identify any constraints that may be imposed on the selection
- d) Evaluate the various SLCMs using lessons learned in past projects
- e) Select the SLCM that will best satisfy the above steps

3.2.1.1 Identification of available SLCMs

Some of the SLCMs being used for software development have been developed into commercial packages and may be obtained from various vendors. These are usually packaged with one or more development methodology tools.

Many organizations have developed their own internal SLCM(s) based on a standard methodology(ies), modified to fit their way of doing business.

3.2.1.2 Identification of project attributes

The model selection should be based upon project attributes and project-specific evaluation criteria. Examples of project attributes include, but are not limited to, the following:

- a) Prototyping
- b) Rapid development
- c) Purchased application selection
- d) Reverse engineering
- e) Object orientation

3.2.1.3 Identification of project constraints

Project constraints may influence phase relationships and information flow. These might include the following:

- a) Risk impact (life, financial, reputation, etc.) of a faulty product
- b) Contractor and vendor relationships
- c) Internal development organization relationships and formal and informal lines of authority
- d) Hardware and its availability, its own development life cycle, and its dependence on the availability of the software to be developed