

WESCO HELPCycle Document

TABLE OF CONTENTS

1 EXECUTIVE SUMMARY	2
1.1 Purpose.....	2
1.2 Scope.....	2
1.3 HELPcycle Objectives	2
1.4 Summary.....	2
1.5 Terminology.....	3
2 HELPCYCLE METHODOLOGY	5
2.1 HELPcycle Manufacturing Processes	5
2.2 HELPcycle Phases.....	8
2.2.1 Analysis and Planning Phase	9
2.2.2 Design Phase	12
2.2.3 Construction Phase.....	15
2.2.4 Training Phase.....	18
2.2.5 Implementation Phase.....	19
2.2.6 Acceptance Phase.....	20
2.2.7 Maintenance Phase	21
3 HELPCYCLE BASIS.....	23
3.1 Capability Maturity Model.....	23

Executive Summary

Purpose

This document provides a high-level description of WESCO's project lifecycle: **HELP**cycle, which is a combination of project management and quality management activities. It defines the activities performed during each of the seven project lifecycle phases, and explains associated quality activities that are performed outside of the project lifecycle, laterally across the organization.

Scope

This document refers to the *WESCO **HELP**cycle Standard* that contains the complete set of best practices that collectively encompass WESCO's project lifecycle methodology. Project lifecycle is applied to all software development and system deployment activities performed by WESCO.

HELPcycle Objectives

Project lifecycle provides the foundation for the predictable creation and implementation of defect-free software that fulfills client business requirements.

Summary

The **HELP**cycle methodology is the set of best practices that facilitate timely completion of a project. The premise of the WESCO methodology is that software creation is a science rather than an art-type activity. This science relies on the steady performance of a team applying standard processes to produce predictable results.

Today many organizations survive on the completion of high-risk projects. Industry-wide reliance on personal heroics, outdated management practices, and outdated software creation practices consistently cause such projects to be completed late or not to be completed at all. **HELP**cycle is sufficiently comprehensive to ensure

timely completion of these large, complex, high-risk projects that frequently involve:

- Multiple delivery units, subcontractors, vendors, customer personnel, and interfacing entities, all of whose activities must be effectively organized and coordinated.
- Multiple technologies such as LAN, WAN, Client/Server architecture, imaging, and multiple-platform production environments.
- Multiple contracts for software, hardware, tools, and services such as business analysis, software development, training, communications, and installation, all of which must be synchronized within a vision of the project that meets customer requirements and is acceptable to all participating vendors and customers.
- Geographically distributed project teams where critical project activities are performed in multiple locations, often by multiple companies.
- Multiple user communities in geographically distributed locations.

The **HELP**cycle methodology is scalable, and consistent application ensures completion of all projects regardless of size or complexity.

Terminology

Bug – Minor technical or functional error or inadequacy not affecting system effectiveness.

Client – Individual or group responsible for accepting and using project deliverables. See Also: Performing Organization

Terminal Error – Any error that does not allow completion of all required system tasks or quality activities.

Quality – Set of activities that ensure delivery of sufficient services and defect-free products.

Defect – Technical or functional inadequacy effecting system effectiveness.

Methodology – Set of best practices, rules, and postulates employed by a discipline.

Application – Named set of computer processes that satisfy some functional requirement.

Performing Organization – Entity that provides charter for a project. The complete set of individuals or groups responsible for accepting and using project deliverables.

Project – Set of activities performed over a finite time culminating in delivery of a unique service or product.

System – Collective set of all technologies and processes required to perform a task.

Quality Assurance – Set of quality activities that ensure continued effectiveness of a quality program.

Quality Control – Set of quality activities that ensure products are delivered defect free.

Project Deliverables – All tangible, verifiable products transferred from project team to client.

System Deliverables – Subset of project deliverables that includes all deliverables directly required for efficient operation of a system. i.e. programs, documentation, production data.

HELPcycle Methodology

This section explains the standard processes, project phases, and standard development practices that make up the **HELP**cycle methodology. A discussion of critical software creation processes is in *HELPcycle Software Development Processes*. The section on *Software Development Phases* contains a description of each project phase and the associated application of the processes described in the previous section. Phases described in *Software Development Phases* are broken into subsections describing tasks, deliverables, and quality. For a more in-depth look at WESCO Quality practices please refer to the document *WESCO Quality Standard*.

HELPcycle Software Development Processes

HELPcycle is dependent on the consistent application of logical processes involving team effort rather than discrete individual efforts. We rely on the steady performance of a team working within a standardized customer-oriented process framework to produce predictable results in project execution. The standard processes we use to ensure on-time, defect-free delivery include:

1. Software Metrics
 - Establishing a Baseline
 - Collecting Metrics
 - Computing Metrics
 - Evaluating Metrics
2. Risk Analysis
 - Risk Identification
 - Risk Projection
 - Risk Assessment
 - Risk Management and Monitoring

3. Project Scheduling

- People-Work Relationship
- Task Definition & Parallelism
- Effort Distribution
- Scheduling and Resource Allocation

4. Project Tracking and Control

- Peer Reviews and Review Reporting
- Progress Reporting
- Status Meetings and Reporting
- Subjective Assessment
- Version Control
- Configuration Management
- Defect Tracking

5. Technical Assessments and Reviews

- All activities have been performed as planned
- All deliverables are reviewed and accepted
- Problems and issues are being recorded, tracked, and resolved
- Verify that all risks are addressed
- Required involvement from all parties is being obtained
- Feasibility of future plans

6. Project Monitoring

- Planned and actual achievement dates of milestones
- Outstanding issues and concerns
- Upcoming milestone completion dates

- Plans for the next period
- Monitoring of project scope and error statistics
- Summary of defect analysis findings and corrective actions taken

7. Quality Assurance

- Process Metric Evaluation
- Quality Assurance Group Process Audits

HELPcycle Phases

Projects are usually divided into project phases to provide better management control and appropriate links to the ongoing operations of the performing organization. The phase sequence defined by most project lifecycles generally involves some form of technology transfer or handoff, such as requirements to design, or software to implement. Deliverables from the preceding phase are usually approved before work starts on the next phase. A subsequent phase is sometimes begun before approval of the previous phase deliverables, however, when the risk involved is deemed acceptable. A project lifecycle is iterative in nature, so activities are often performed at multiple points during a project. For example, a general implementation plan is developed during the Analysis and Planning Phase that takes into account all system requirements. However, the implementation planning process is iterated during the Implementation Phase and specific dates and resources are outlined.

Each project phase is marked by completion of one or more deliverables. A deliverable is a tangible, verifiable work product such as a feasibility study, a detail design, or a working prototype. The deliverables, and hence the phases, are part of a generally sequential logic designed to ensure proper project definition and execution. The conclusion of a project phase requires review of both key deliverables and project performance in order to (a) determine if the project should continue into its next phase and (b) detect and correct errors cost effectively. Each project phase normally includes a set of defined work products designed to establish the desired level of management control.

As with most project lifecycle phases, **HELP**cycle phases provide management control and the appropriate links back to the performing organization. **HELP**cycle phases are the vehicles that ensure consistent, logical application of the standard **HELP**cycle software creation processes to a project. Application of the conceptual methodology is consistent for all projects, but are not identical. Every project has unique aspects, and uses a different subset of the total set of best practices described by the methodology, to provide maximum benefit within the confines of these unique challenges and constraints. A large or high-risk project requires more stringent application of the standard

software creation processes than does a smaller, less critical project, where applying the same processes designed to provide management control on large projects could greatly affect the total cost of a project. The following sections provide a brief description of each phase.

Analysis and Planning Phase

Analysis and planning is the first project phase. It begins with development of an initial project plan that will both guide early analysis and planning activities, and plan for future project phases. The initial project plan is the first iteration of the deliverable Project Plan that is used to guide project execution and control. Deliverables from this phase define all future project activities, therefore they are critical to the successful development and delivery of a defect-free product.

Quality issues are dealt with during the analysis and planning phase. This includes quality control planning and the assignment of quality responsibilities. Quality assurance activities are defined for all project phases. These activities are defined as standard WESCO activities and are not as project-specific as are quality control activities. The scope of quality assurance activities is defined during the analysis and planning phase. This definition is critical to on-time, on-budget project completion. By establishing the scope for quality assurance early in the planning process, we ensure that quality activities do not dictate the project scope, and thus potentially endanger project on-time delivery.

Requirements Engineering

Requirement engineering is begun upon acceptance of the initial project plan. In general terms, requirements engineering is the cooperative effort to define requirements, analyzing them for feasibility, and then formally documenting these requirements to guide future project activities. In this context, requirements fall into two categories: functional requirements and general system constraints requirements. This requirements engineering effort is critical to project success. The requirements that are outlined in the system requirements specifications document phase of this process define the project scope and are a baseline for all ensuing project activities.

The deliverable for this portion of the analysis and planning phase is the **Systems Requirements Specifications** document. This document is a comprehensive set of all requirements elicited during the systems requirements analysis.

WESCO requirements engineering activities focus only on user-functional system requirements and business-driven constraint requirements. A common, and often damaging practice is to perform system requirements with an eye toward specific design issues. WESCO believes in performing its requirements engineering tasks without reference to any possible ensuing design activities. This allows for a completely unbiased requirements analysis. We believe that insufficient requirements definition cripples a project before it ever begins, because any project activities that follow, including design, are performed assuming that adequate requirements engineering was performed. If this assumption is not true, then the incomplete requirements engineering presents a direct risk to project completion, and an indirect risk to all performing organizations.

The deliverables for this portion of the analysis and planning phase are the **Implementation Plan** and **External Interfaces Document**. At a minimum, this document summarizes system hardware needs, system software needs, database organization, training requirements, security measures, and all required external interfaces. This document is regarded as a living document, and details are added throughout the life of the project.

Analysis and Planning Phase Tasks

1. Develop an initial project plan
2. Perform requirement engineering
3. Compile project standards and practices
4. Develop a final project plan
5. Develop an implementation plan and an external Interfaces document

Analysis and Planning Phase Deliverables

System Requirements Specification (SRS) Document

Comprehensive set of all requirements elicited during system requirements analysis. Project success depends upon this document accurately defining project scope.

Implementation Plan and External Interfaces Document

Implementation requirements are planned with respect to system requirements and dependent client business processes. At minimum it summarizes system hardware needs, system software needs, database organization, training requirements, security measures, and all required external interfaces. This is a living document and detail will added throughout the life of the project.

Project Standards and Practices Document

All common standards, development tools, metrics, and practices are contained in this document. Common practices included in this document are essential Quality activity descriptions including versioning, unit testing standards, and a configuration management plan. All members of the project team are provided with the final copy of this document.

Project Plan

A consistent, coherent document used to guide both project execution and project control. The project plan may be phase-delivered, with greater detail included with each release. For example, the first release might contain generic resources and undated durations while the final release will contain specific resources and explicit dates. The project plan is used throughout the project to:

- Guide project execution
- Document project planning assumptions
- Document project planning decisions
- Facilitate communication
- Define management reviews and status reporting
- Provide a baseline for progress measurement and project control

Phase 0 Analysis Overview Document

The Phase 0 Analysis Overview Document is a compilation of all other documents described above.

Analysis and Planning Phase Quality

Quality issues dealt with during the analysis and planning phase of **HELP**cycle include quality control planning, and the assignment of quality responsibilities. Quality assurance activities are defined for all phases by standard WESCO activities and are not as project-specific as are quality control activities.

Scope is defined during the Analysis and Planning phase. Requirement definition is critical to on-time, on-budget project completion. Quality activities can only ensure that defined requirements are met and standards adhered to. If, at any point in a project lifecycle, quality activities begin to dictate scope, the quality activities lose their integrity and on-time delivery quickly becomes unattainable.

Design Phase

The project design phase is dependent upon adequate completion of requirements engineering. The focus of this phase is to develop a best-fit, cost-effective design that incorporates all defined functionality.

The high-level design document is developed first to lay out system concepts, for example:

- the number and type of user
- processing load, system genre (stand-alone, client/server ext.)
- communications medium
- high-level system architecture

As the process progresses, detail is added to the design document. The final iteration of the complete system design includes a database design, program specifications (often including pseudo code logical algorithms), and an external interface design.

Acceptance of these design documents is a prerequisite for development of the functional system prototype.

A prototype includes screen layouts that contain adequate detail to permit the user to review core system functionality, and to provide an overall look and feel. Feedback from testing is incorporated into the system requirements specifications document if the risk of doing so is deemed acceptable. Risk analysis must confirm that the project scope does not expand with the addition of user feedback. If user feedback causes the scope to be modified, the project is redefined and the requirements engineering effort repeated.

Design Phase Tasks

Design Phase Deliverables

High-Level Design (HLD) Document

System requirements elicited during the analysis and planning phase and presented in the system requirement specifications document are used to design an optimum system.

Detail Design Document

System detail is included in this document, specifically: database design, program specifications, and an external interface design. The detail design document is not necessarily separate from the high-level design, thus the documents may be presented together for smaller projects.

Unit Test Plan

A unit test plan is developed per the program specifications that are developed during detail design. All software deliverables are tested in accordance with this plan.

Functional System Prototype

User feedback requesting functionality modification is carefully examined to discern the effect of the modification on project scope and risk. It is essential that the project scope does not expand with the inclusion of user feedback. If user feedback indicates that project scope is inadequate, then the project plan is void and analysis must again be undertaken and the project redefined.

Design Phase Quality

System design integrity is verified. Design walkthroughs are performed to ensure that system development sufficiently meets all requirements and that the design can accommodate future enhancement.

The unit test plan is finalized during the design phase. System integrated test scripts and the data sources for these scripts are defined. These test plans are modified as user feedback from the prototyping process and other user testing is incorporated into the system design.

Deliverables

High-Level Design Document

The system requirements elicited during the analysis and planning phase and presented in the system requirements specifications document are used to design an optimum system that is presented in this document.

Detail Design Document

System detail is included in this document, specifically: database design, program specifications, and an external interface design. The detail design document is not necessarily separate from the high-level design. The documents may be presented together for small projects.

Unit Test Plan

A unit test plan is developed in compliance with program specifications developed during the detail design phase. All software deliverables are tested in accordance with this plan.

Functional System Prototype

During the time this document is being written, user feedback is examined for requests to modify scope and risk to the project. Those requests that are found to be within project scope are implemented, tested, and incorporated into the program. Those that are out of scope are evaluated, and if found to be valid are submitted as a change order request.

Software Development Phase

The software development phase contains most of the actual “building” activities within the project lifecycle. Modules are coded, tested, and delivered to a central location for system integrated testing.

During the software development phase of the project life cycle, attention must be taken that programmers adhere to coding standards, system design is interpreted correctly, and that the entire development group is coordinated. Every developer must be aware of coding standards and practices. Periodic structured code walkthroughs by members of both the development and the quality assurance groups ensure that coding standards are adhered to. Adherence to coding standards and practices during development dramatically improves documentation and system maintainability. Many system documentation materials are generated automatically from program source code. Throughout the Software Development Phase, reusable software assets are identified to minimize development and system maintenance costs. Development group meetings are held regularly, even if modules are felt to be independent. Module dependencies are often impossible to identify before they are built, so coordination must take place during coding.

System, operation, and user documentation are developed during the construction phase of the system lifecycle. User documentation will often include on-line documentation. The contents of both online and paper user documents are almost identical and are developed simultaneously.

Comprehensive documentation is critical for efficient system maintenance and operation. System documentation must contain detail sufficient to enable an analyst to develop a complete understanding of the system.

Standard system documentation content:

- Functional hierarchy diagrams
- Entity relationship diagrams
- Detailed program pseudo code

- Database design
- Testing plan
- Test data details
- Standard operations documentation:
- System maintenance procedures
- Batch process logic and information
- Sequencing of batch processes
- Back up procedures
- Recovery procedures

Software Development Phase Tasks

- Software creation
- Defect tracking
- Version control
- Configuration management
- Documentation development
- System integrated test plan development
- Unit testing
- System integrated testing

Software Development Phase Deliverables

System Deliverables

System deliverables include all programs, modules, interfaces, and supporting software assets, and corresponding system and user documentation.

Integrated Test Plan

System integrated testing is completed before implementation. Integrated test plan must ensure the following:

- All functional requirements have been met.
- Performance testing based on the Performance criteria.
- Recovery procedures are in place.
- Stress testing confirms adequate system capacity.
- Security controls are adequate.

Software Development Quality

Unit testing is performed continually throughout software development. Effectiveness of unit testing is maximized as turn-around time for an individual unit is minimized. The development group performs unit testing per the unit test plan. Code that meets all requirements in the unit test plan is marked as unit test certified and is moved to a common library for integrated systems testing.

Common test data developed and compiled during the construction phase is stored in the integrated testing library. This carefully developed data is used for system integrated testing. Computer-aided testing is often employed during system integrated testing. Specific scripts are written that increase data volume to test system performance on a large database. In addition, computer-aided testing may be used to test the capacity of individual system components, such as networking or database access, and to highlight bottlenecks that can be alleviated.

System integrated testing takes place during this phase. Both analyst and user testing is performed as directed by the test plan that is developed during the design phase. All modification requests are analyzed for impact on the system and project scope. If risk is deemed acceptable, then the modification is designed, coded, unit tested, and released for system testing.

Project team members, who are not working closely with the overall development effort, perform integrated system testing. A separate testing team brings a fresh perspective to the project. Since testing team members have not been working on the system for days or even months, they can devise innovate testing scenarios and data sets. All information gathered during testing, including testing comments, errors, inconsistent user interfaces, or other defects, are entered into the defect tracking system and reported to the project manager. Modifications resulting from testing will again be unit test certified, and moved to the common system testing library.

Five specialized activities performed as part of testing.

1. Stress testing
2. Storage testing
3. Performance and response time testing
4. Recovery testing
5. Human factors testing

Training Phase

Training is key to maximizing return on investment for a system development project. WESCO believes that varying training to meet specific needs ensures long-term system viability. When a client assumes complete control of system maintenance, WESCO's role shifts to that of train-the-trainer. A long-term relationship can occur when WESCO provides direct training to end users and system maintenance people, and also provides training material to minimize lost productivity following implementation of a new system.

Training Phase Tasks

- User Training
- System Support Personnel Training

Training Phase Deliverables

Training Materials and Services

All training materials and services are defined and delivered according to an agreed upon schedule.

User Documentation

User documentation is identified in this phase, and is produced and delivered in compliance with the schedule.

Training Phase Quality

To be effective, training must address all client-critical functions. Training materials are developed using many of the same quality processes that are used for software deliverables. User documentation that is delivered during the training phase is versioned concurrently with all other system deliverables. Before the beginning of training, WESCO performs a training walk-through to ensure that the training program addresses all critical system functions.

Implementation Phase

Implementation is performed in compliance with the plan developed in the requirement analysis and planning phase. Exhaustive production environment testing may be required. Configuration items are maintained as outlined by the implementation plan and maintained by the project librarian.

Quality activities such as versioning and configuration management are continued to ensure a robust and well-documented implementation.

Implementation Phase Tasks

There are many activities performed in support of the core implementation tasks. These support activities are defined in the Implementation and External Interfaces document and also in the Project Plan. The core tasks of the implementation phase include:

- Installation
- Production data conversion
- Parallel run (if applicable)
- Live cut run

Implementation is performed in compliance with the plan developed during the Design Phase. Exhaustive testing of the production environment is required. Minor system adjustments and refinements are made during this phase.

Implementation Phase Deliverables

Integrated System (with necessary data)

Post Implementation Review Report

Implementation Phase Quality

Implementation is performed per the plan developed in the design phase. Exhaustive testing of the production environment may be required. Minor system adjustments and refinements are made during the implementation phase. Configuration management is maintained as outlined by the implementation plan and maintained by the project librarian.

Acceptance Phase

Acceptance phase activities and timing are dependent on the nature of a given project. On many projects, the client performs acceptance phase activities parallel with other project activities. In general, the acceptance phase activities include final user testing and signoff on all major system deliverables. At the same time, WESCO often incorporates user testing comments into the system during a software creation phase that occurs parallel to the acceptance phase. This is done only after an analysis determines that an acceptable risk level exists.

Acceptance Phase Tasks

- Client acceptance testing
- Iteration of project construction phase and associated quality activities.

Acceptance Phase Deliverables

Typically, there are no new deliverables associated with the acceptance phase. Client must signoff on all major deliverables in a timely fashion.

Acceptance Phase Quality

Continually updating software versions as deliverables are accepted allows for efficient maintenance activities. In addition, during this phase quality audits are often performed to ensure the robustness of the project lifecycle. As part of these process audits, complex metrics are calculated and evaluated to determine trends that are adverse to quality. These metrics provide quantifiable evidence of HELPcycle strengths and weaknesses.

Maintenance Phase

Maintenance phase activities fall into three distinct types:

- scheduled system maintenance
- bug fixes
- system enhancement

Scheduled system maintenance is included in the project lifecycle for a time defined in the contract. After this time expires, system maintenance becomes the customer's responsibility. The time during which WESCO is responsible for system maintenance allows for a complete knowledge transfer from the project team to the customer's maintenance personnel.

During the maintenance phase, as bugs are identified, a modification schedule is created and executed. All bugs are tracked and progress toward fixing them is reported regularly. Once resolved, the bug report is removed from the list and archived as part of the project documentation.

System enhancements are defined as those system change that fall outside of the scope of the original project. These enhancements can be as simple as a new report, or as complicated as creating a new subsystem. System enhancements must be well defined. They then fall under the HELPcycle rules until they are completed. Enhancement requirements engineering, design, coding, and testing are managed and executed by applying the same standard processes that are used during the initial development effort.

Maintenance Phase Tasks

Maintenance phase tasks are defined on a per project basis.

Maintenance Phase Deliverables

Post implementation deliverables can include any type of deliverable from the project lifecycle.

Maintenance Phase Quality

Tasks from all other phases can occur during the maintenance phase. Often, less stringent application of quality tasks is required during the maintenance phase because of the less complex nature of most phase objectives and deliverables. It is critical that the following quality activities are performed to ensure ongoing system viability:

- risk assessment
- versioning
- configuration management
- exhaustive testing
- deliverable standard audits

HELPcycle Basis

Capability Maturity Model

The Software Engineering Institute (SEI) of Carnegie-Mellon University has developed a method of assessing an organization's software engineering proficiency called the Capability Maturity Model or CMM. This vehicle provides a scale for evaluating the ability of an organization to manage its software development process. Industry studies show that progression through the CMM dramatically increases productivity and lowers defect rates. Through the development of CMM-standard methods and practices, WESCO consistently delivers defined projects on time and within budget.

The model states that software processes is predictable, controllable, and measurable. The use of a software development process allows an organization to normalize the tools, methods, and work patterns used to develop software. CMM provides a way to measure an organization's progress through industry-recognized stages of maturity. Every organization operates at one of the following levels of maturity:

Initial– Level One

Although Level One organizations may have a few formal procedures for planning and tracking their work processes, they rarely enforce them. Organizations at this stage frequently operate without project plans and can rarely provide accurate cost estimates. Change control is imprecise. Software maintenance is often difficult because problems are not recorded during development.

Repeatable– Level Two

In Level Two, organizations gain control of plans and commitments. Although reaching this level indicates that an organization has made some progress, the journey toward quality control has barely begun. The organization's control to this point stems from individual experience, and thus is relevant only to similar work done by the same people. Organizations at Level Two risk losing ground when faced with new challenges, such as organizational changes

or new tools and products.

Defined– Level Three

An organization has successfully built the foundation from which all major progress can be achieved when it reaches Level Three. At this point, the organization has a consistent process to which all people in the organization adhere, and from this foundation the organization can examine its process and determine how to improve it. For the first time, the process– not the individual– defines the work. Although the organization can provide improved quality and more reliable cost estimates and schedules, it cannot fully quantify the value of the process.

Managed– Level Four

Advancing to Level Four provides organizations with major improvements. Although metrics are used in lower levels, organizations begin to use metrics consistently to examine and improve processes in Level Four. It is important that all groups within the organization gather identical data and use identical definitions so results can be compared. These data are not used to compare projects or individuals. Such an approach undermines the validity of the data and corrupts the team focus of the process.

Optimizing– Level Five

To some degree, optimizing occurs at all levels of the capability maturity model. However, until an organization reaches Level Five, the emphasis is usually on optimizing the products that are being developed. At Level Five, the focus shifts to the process itself. Metrics are in place that help find and fix errors much faster than at other levels. The data gathered at this level help organizations reduce the cost of correcting errors by identifying the weakest areas of the process.

CMM provides a yardstick for identifying how close an organization is to optimizing its software development process. WESCO standard methodologies and processes are based upon our commitment to providing top quality products and services. Our steady implementation of methodologies that will bring us to the Level 5– Optimizing are a major assets to furthering our ability to provide world-class products and services.