**Registration Number:** NIxxxxx

**Engineering Handbook**

|  |  |
| --- | --- |
| Version: | 1.0 |
| Status | Draft |
| Date: | 14th July 2008 |
| Approved: | Paul Moorhead  |

Table of Contents

[1.1 Objectives and Scope 3](#_Toc209262466)

[1.2 Terms and Abbreviations 3](#_Toc209262467)

[2 Introduction 3](#_Toc209262468)

[2.1 Handbook Structure 3](#_Toc209262469)

[2.2 Amendment Record 4](#_Toc209262470)

[3 Planning 4](#_Toc209262471)

[3.1 Estimation 4](#_Toc209262472)

[3.2 Risk Management 4](#_Toc209262473)

[3.3 Scheduling 4](#_Toc209262474)

[3.4 Project Tracking 4](#_Toc209262475)

[3.5 Reporting 5](#_Toc209262476)

[3.6 Review 5](#_Toc209262477)

[3.7 Configuration Management 5](#_Toc209262478)

[4 Requirements Definition 5](#_Toc209262479)

[5 Functional Definition 6](#_Toc209262480)

[6 Design and Software Implementation 6](#_Toc209262481)

[7 System Integration testing 6](#_Toc209262482)

[8 Integration, Validation, Verification & Test (IVV&T) 7](#_Toc209262483)

[8.1 Quality Assurance 7](#_Toc209262484)

[8.2 Interoperability Testing (IOT) 7](#_Toc209262485)

[8.3 Performance and Capacity (PAC) 7](#_Toc209262486)

[9 Release 7](#_Toc209262487)

[10 Configuration Management 8](#_Toc209262488)

[10.1.1 Software Teams: 8](#_Toc209262489)

[10.1.2 Hardware Teams 8](#_Toc209262490)

[11 Project Change Control & Bug Reporting 8](#_Toc209262491)

[12 Reviews & Approval 9](#_Toc209262492)

[12.1 Document Review 9](#_Toc209262493)

[12.2 Code Review and Verification 9](#_Toc209262494)

[13 Software Development Tools 9](#_Toc209262495)

## Objectives and Scope

This handbook forms part of the Feature Creep QMS. The handbook is primarily targeted at people involved in the development of software. It may be considered as a Work Procedure for Software Developers. This handbook must be read in conjunction with the Feature Creep Delivery Framework, which details how the Feature Creep product development lifecycle is monitored and controlled.

## Terms and Abbreviations

FS Functional Specification

IOT Interoperability Testing

MRD Marketing Requirements Document

PAC Performance and Capacity

PDT Product Delivery Team

PPG Product Planning Group

PRD Product Requirements Document

QA Quality Assurance

QMS Quality Management System

SIT System Integration Testing

IVV&T Integration, Validation, Verification & Test

# Introduction

The Software Development Handbook is intended as a reference source that describes all the activities that take place during the software development lifecycle within Feature Creep.

## Handbook Structure

The handbook is structured as follows:

Introduction Briefly describes the handbook structure.

Planning This section focuses on activities involved in project planning specifically for software development projects.

Requirements Definition This section describes how the customer requirements are captured, reviewed and understood in order to produce a Product Requirements Document.

Functional Definition This section focuses on the translation of the Requirements Specification into a more technical representation, which results in a Functional Specification.

Design and Software Implementation This section describes how the functional specification is further translated into code.

Integration Validation, Verification and Test This section describes the activities of the various IVV&T groups within Feature Creep Belfast, including QA, SIT, IOT, FE and PAC.

Release This section describes the deployment of the tested software and the necessary release criteria.

Configuration Management This section defines the Configuration Management procedures to be used during the Software Development Lifecycle.

Project Change Control This section deals with the standard Feature Creep practices for recording changes and defects.

Reviews And Approval This section defines the review and approval practices that are used during the project lifecycle.

Software Development Tools This section describes a minimum set of tools to be used in the Software Development Lifecycle.

## Amendment Record

|  |  |  |
| --- | --- | --- |
| **Date** | **Issue Number** | **Reasons for change** |
| 14th July 2008 | 1.0 | First draft for review |
|  |  |  |

The Quality System is subject to continuous review and improvement. Note that updates to this manual since the previous version are shown by change bars.

# Planning

## Estimation

The purpose of the estimate is to demonstrate that the required functionality can be implemented within a certain cost and timeframe. The Product Requirements Document (PRD) should form the basis of any Engineering Estimates for the Product Development.

If an estimate is prepared incorrectly and agreed with the customer the consequences can be potentially disastrous for Feature Creep. It is therefore essential that any staff producing estimates should closely follow the [Feature Creep Planning Guidelines](file:///D%3A%5CCompanies%5CFeature%20Creep%5CQMS%20Web%5Cproject_control%5Cproj_plan%5Cindex.htm).

## Risk Management

Risk management is an integral part of the Feature Creep Delivery Framework. Risks must be identified and managed by the Product Delivery Team as part of the delivery process. The Product Delivery Manager has responsibility for ensuring this happens.

## Scheduling

A schedule must be produced by the Product Delivery Manager. It shall take into account the estimates and include sufficient contingency for any risks identified. A Schedule plan must be produced for every development and **must** include the following information:

* a plan of the work packages / assignments that the development breaks into
* the planned start and finish dates of each task
* the planned effort associated with each task
* the people assigned to work on each task
* the timing of baselines
* the timing of milestones / gates

All review and inspection activities should also be included in the schedule plan.

The schedule plan must be baselined and approved (being verified at least by informal review). The original baseline schedule should be kept throughout the life of the project. There should also be a running/current schedule so that progress against the baseline can be tracked.

Each new update to the running/current schedule does not need to be explicitly version controlled. Changes to the originally baselined schedule plan do, however, need to be version controlled.

For more information on planning and scheduling, please refer to the Feature Creep Planning Guidelines.

## Project Tracking

Projects are tracked by the Product Delivery Team who will meet at a frequency appropriate to the phase of the project (e.g. weekly during early stages and up to daily as release approaches). Progress against schedule plans and other relevant metrics will be the basis for assessing progress and identifying issues. Relevant metrics will include:

* Revised estimate of work completed and work outstanding
* S-Curve progress on tests executed and tests passed
* Defect open and closure rates

## Reporting

As an overview, team members should report progress to the Product Delivery Manager. The PDM monitors progress against each of the gates or milestones in the project lifecycle. Project affecting issues that could affect the outcome of the project or impact on subsequent tasks (Red Flags) should be raised to the PDM either outside our during the PDT meetings. Progress Reporting normally occurs on a weekly basis although the PDM can request more frequent reports at their discretion.

## Review

The Feature Creep Delivery Framework establishes formal review gates as part of every product delivery. In addition post-release reviews should be held at project completion, with the follow-up actions formally captured and tracked. It is the responsibility of the COO to ensure that this happens.

## Configuration Management

A SharePoint team site will be created for each new product development and used to manage all documentation associated with the project including version control. Source code will be managed within Subversion as described in Section 10.

# Requirements Definition

The Product Requirements Document (PRD) should be written during this phase. It should be primarily based on the Marketing Requirement Document (MRD). The PRD should be written in accordance with the Feature Creep PRD Template. It is essential that the PRD is as unambiguous as possible as it is the baseline for the rest of the software development.

Even though the PRD is 'owned' by the Product Manager, the contents of the PRD should be agreed with the Engineering Group. Requirements listed in the PRD **must** be numbered as the example below shows:

|  |  |
| --- | --- |
| Requirement Number  | Requirement Description  |
| REQ-PROJ-001  | The product shall support a single-node configuration.  |
| REQ-PROJ-002  | The product shall support a multi-node configuration.  |
| REQ-PROJ-003  | The single-node product configuration shall not have a single point of software failure.  |
| REQ-PROJ-004  | Dependency file generation for included files (in particular .h files) will be included as an integral part of the make.  |

The reason that the requirements must be numbered is to facilitate traceability of the system from the Product Requirements Document at all stages of the lifecycle. It is recommended that traceability of the requirements is achieved by the use of a traceability matrix, which would demonstrate how the requirements are incorporated in the design, implementation and test phases. A traceability matrix lists what part of the Design, Implementation and Test Outputs that cover a specific Product Requirement.

# Functional Definition

The main objective of the functional definition phase is to accurately translate the requirements listed in the PRD in order to define the implementation of the product features. This translation should be included in a Functional Specification (FS) document. The FS is normally the lowest level design document used in Feature Creep, although for larger projects more detailed lower level design documents can be written. Additional, non-normative but helpful information will be held in the project wikis.

In most cases though the FS should provide a detailed enough description of the proposed solution to enable coding to be done. It is the responsibility of the engineering team leader to ensure that the FS has covered all requirements listed in the PRD. The FS should be reviewed.

When writing an FS, the Feature Creep FS Template should be used as a guideline. The FS is the main quality record of this phase and should be stored in accordance with the configuration procedures for the project.

#  Design and Software Implementation

A software engineer is assigned to the task of translating the function specification into executable code during this phase. Normally, within Feature Creep projects, the FS is the lowest level design document and all code is written from that. The engineer must therefore read and understand the FS and, where necessary, should meet and discuss any points with the FS designer when clarification is needed. If the FS is not the lowest level design document for the project then this should be specified in the Software Development Plan.

The engineer has a number of responsibilities during this coding phase:

* Present the code at code reviews.
* Build and maintain the Configuration Management Environment.
* Ensure that any flaws, be they design, functional or requirements, are recorded and reported to the PDM.
* Provide support for the code they have handed over to IVV&T.

The main outputs of this phase are described below. All of these components should be stored according to Feature Creep’s configuration management processes.

* Source Code
* Updated Design Documents
* Review Information.

#  System Integration testing

At this stage the Engineering team should perform integration testing of the system components. When the software is ready to undergo IVV&T the Development team have responsibility to produce a formal handoff document, usually in the form of an email. This **must** contain details on the following:

* Bugs contained in this release
* Installation Instructions

All of the agreed functionality **must** be in a working and testable state before the handoff email can sent out. This is the responsibility of the Development Team. It is essential that all general handoff criteria are met, as listed below:

* Software is Installable
* System builds
* All critical code modules have been reviewed

# Integration, Validation, Verification & Test (IVV&T)

## Quality Assurance

The Engineering team will develop a Unit Test Pack in parallel to code development, based largely on the FS and to some extent on the PRD. They have responsibility to design and develop the appropriate test harnesses and to define test cases that will adequately test the software.

Design of the test cases that are included in the Unit Test Pack should take into account the following areas of the software:

|  |  |
| --- | --- |
| Interface  | e.g. Does correct data flow into and out of the unit?  |
| Functional  | e.g. Does the unit conform to the design and requirements that it was implemented from?  |
| Coverage  | e.g. Has the coverage target defined in the project plan been achieved?  |
| Boundary  | e.g. Have all loops and conditions been tested, for example test LOOP 1 to N with values equal to 1, N-1, N+1, N <1 and an interim value?  |
| Repeatability  | e.g. Does repeated testing generate the same results?  |
| Error paths  | e.g. Have all error conditions been exercised?  |
| Data Integrity  | e.g. Does the unit under test perform suitably with valid and invalid supplied data?  |

Test cases may be executed informally during the production of the software. They may also be updated as the software evolves and the testing is further refined. Unit Test results must be recorded and maintained. Where bugs are uncovered as a consequence of the testing, they must be recorded and raised in the Defect Tracking System (SharePoint). When testing is complete the tester(s) should produce a QA Summary Report.

For greater detail on the IVV&T roles and responsibilities refer to the Feature Creep IVV&T Handbook.

## Interoperability Testing (IOT)

The purpose of the Interoperability Test group is to ensure, through systematic testing, that all components of the solutions provided and supported by Feature Creep are fully interoperable with third party software and hardware.

## Performance and Capacity (PAC)

The objective of the PAC testing is to evaluate the product under load, to stress the product and gather performance data. This data is used to verify that the product meets performance and scalability goals, to measure product performance relative to the previous release, and to produce configuration recommendations.

# Release

The software deliverables and associated documentation are delivered out of IVV&T along with a set of test results. A list of outstanding problems may be delivered (where any exist). This list and the results must be used to determine whether the software is in a fit state for delivery through review at the Release Gate

All the various groups involved in Development and IVV&T have to meet a certain criteria before the product can be released. For example, the QA Summary Report can be one of the prime decision enablers as to if a product can be released or not. The Product Planning Group (PPG) should take on board the input provided from Engineering and the IVV&T groups before deciding if a product is ready for release.

The following list of questions should be asked to assist the decision making:

* Have all the VV&T Test Exit criteria been met?
* Are there any critical problems in the list of problems? If so, can the deployment proceed?
* Are there any gaps in the functionality requested by the customer? If so, is the customer aware of these gaps and prepared to accept them?
* Are all software modules and associated documentation available for delivery?
* Are all components of the system available for delivery, that is hardware, software and documentation? If not, when will they be available? Can the deployment proceed?

These questions may be expanded on to suit different situations. If the answers to any of the above questions are negative then a decision is required as to whether or not the deployment should proceed. The decision must be taken by the Product Planning Group.

# Configuration Management

The purpose of Configuration Management is "to establish and maintain the integrity of products of the software project throughout the project's software lifecycle". Configuration Management entails the following:

* Identifying the Software Configuration Items (SCI).
* Uniquely identifying each item.
* Establishing baselines.
* Tracking changes to the items and to the configuration.
* Ensuring change is properly implemented and controlled.

Feature Creep’s product configuration items will be managed in Subversion as follows:

### Software Teams:

* 1. No branches created during development. Branches will be created post-GA for field support.
	2. Frequent check-ins – the load is under constant churn and (usually) improves each day
	3. Breaking build is acceptable as there’s a fast reaction time (also it’s easy to back out disastrous check-ins)
	4. All check-ins must be commented, initialled by developer and tagged with which bugs are fixed

#### Build

* + Automated via a cron-job equivalent,
	+ Build number and build time embedded in binaries.

### Hardware Teams

1. Bug fixes can be created on trunk, or if sufficiently large, and affecting a number of modules, on a branch.
2. New features will normally be added on a branch, created for the purpose, and merged back onto trunk.
3. Changes should be merged back onto trunk as often as possible to reduce the level of major churn.
4. All check-ins to trunk must be commented, initialled by developer and tagged with which bugs are fixed.

#### Build

* + Hardware build process as per software.

# Project Change Control & Bug Reporting

It’s important that Feature Creep responds to externally and internally generated needs for change quickly. To that end the Product Delivery Team will be the vehicle for rapid discussion and agreement on change to any item including specifications, designs, plans, strategies etc. If agreement to change cannot be achieved within a single meeting of the PDT, then an action will be given for discussion outside the meeting and a review of the findings at a future PDT meeting.

Records of agreed changes will be captured in the PDT meetings for which there is a formal template.

Bugs are handled within the Engineering Team and are used continuously to report on defects in test and in the field. SharePoint’s defect tracking web-parts will be used to manage defect workflow.

# Reviews & Approval

## Document Review

Documents such as the PRD, SDP and FS are all key documents in the Delivery Framework and **must** be reviewed. Adequate allowance should be planned into the schedule to allow reviewers time to prepare for and carry out these document reviews.

## Code Review and Verification

Again, adequate time should be planned into the schedule to allow for thorough review of any critical modules of code. The code should be checked for the following:

* Consistency
* Style
* Adherence to Standards
* Errors
* Complexity
* Maintainability

It is recommended that the code, along with any accompanying documents, be presented to the designated reviewers at least three working days before the review meeting.

More informal walkthroughs of code can also be carried out.

# Software Development Tools

It is not possible to produce a concise list of the available development tools available in this document.

As a minimum it is recommended that the following tools are used when planning, designing and producing software and associated documentation:

SharePoint - As document management system

SharePoint – As Bug/Issue Tracking System

Microsoft Project - A planning tool

Subversion - A Configuration Management Tool

Microsoft Visual Studio – An integrated software development environment

Purify - Used to find memory leaks in code