



Qualitätsmanagement

SEP: Entwicklung von Projekt „CFX100“

Version 1.0

Qualitätsmanagement der Gruppe „sepis“ zum Software
Entwicklungsprojekt im Sommersemester 2007

Konstantin Kramer, Rolf Schneider

21.04.2007



Inhaltsverzeichnis

1. Zielbestimmung.....	3
2. Projektqualität.....	3
3. Produktqualität.....	3
3.1. Anwendersicht.....	3
3.2. Systemadministratorsicht.....	3
3.3. Entwicklersicht.....	4
3.4. Qualitätskriterien.....	4
3.4.1. Funktionalität.....	4
3.4.2. Zuverlässigkeit.....	4
3.4.3. Software-Ergonomie.....	4
3.4.4. Bedienungsfreundlichkeit.....	5
3.4.5. Änderbarkeit.....	5
3.5. Maßnahmen.....	5

1. Zielbestimmung

Durch den Einsatz von Qualitätsmanagement soll erreicht werden, dass die Kundenanforderungen bestmöglich umgesetzt werden. Das Qualitätsmanagement zielt auf alle Aktivitäten, die bei unserem Projekt stattfinden.

Das Dokument umfasst die Beschreibung der Vorgehensweise, die notwendig ist, um eine möglichst hohe Qualität zu erreichen.

Ziel ist es, dass unser Kunde, die Firma Pepperl+Fuchs, am Ende des Projekts ein qualitativ hochwertiges Produkt bekommt.

2. Projektqualität

Die geschaffene Infrastruktur für einen optimalen Informationsfluss ist zentrales Element, das zur Projektqualität beiträgt.

Die Datenspeicherung aller Dokumente findet im Webdav bzw. SVN nach vorgegebenen Regeln (siehe Projekthandbuch) statt. Diese Speicherung ist allen Teammitgliedern zugänglich.

Das Risikomanagement ist ein weiteres Mittel, das zur Projektqualität beiträgt.

3. Produktqualität

Unsere Planung orientiert sich in der Struktur an ISO/IEC 9126. Im Folgenden werden die verschiedenen Sichten näher beschrieben.

3.1. Anwendersicht

Der P+F Client Konfigurator muss leicht bedienbar sein, sodass ein Nutzer ihn ohne technische Vorkenntnisse intuitiv bedienen kann. Die Software muss also für den Anwender transparent und aufgabenorientiert sein.

3.2. Systemadministratorsicht

Die Software muss einfach zu installieren sein. Dies ist durch Werkzeuge der dafür zuständigen Abteilung bei Pepperl+Fuchs gewährleistet.

Eine einfache Wartung wird durch den Einsatz von XML-Dateien ermöglicht.

3.3. Entwicklersicht

Die Qualität wird durch Einhaltung von Coding-Styles sowie guter Dokumentation angestrebt.

Ein modularer Aufbau mit definierten Schnittstellen dient einer gut verständlichen, wartbaren und erweiterbaren Software.

3.4. Qualitätskriterien

3.4.1. Funktionalität

Die Funktionalitäten sind dem Lastenheft, Pflichtenheft und den Kundengesprächsprotokollen zu entnehmen.

Dabei muss unterschieden werden, ob die Funktionalitäten erreicht werden müssen oder ob sie nur optional sind.

Die Richtigkeit der Funktionalitäten sowie Interoperabilität wird durch das Testen festgestellt und festgehalten. Insofern sind nähere Informationen der Test-Spezifikation bzw. später dem Testkatalog zu entnehmen.

3.4.2. Zuverlässigkeit

Die Zuverlässigkeit soll erreicht werden, indem die Software ausgiebig getestet wird. Die Software ist robust, d.h. Fehlereingaben sollen erkannt bzw. behandelt werden. Die erlaubten Kombinationen bzw. Ausschlusskriterien sollen korrekt umgesetzt werden.

3.4.3. Software-Ergonomie

In erster Linie geht es uns hier um das User Interface. Wir orientieren uns dabei an ISO 9241. Unser Ziel ist es ein System zu entwickeln, das Aufgabenorientierung und Bedienungsfreundlichkeit optimal widerspiegelt. „Ein bedienungsfreundliches System nützt wenig, wenn sich damit die Arbeitsaufgabe nur schlecht lösen lässt. Genauso wenig sinnvoll ist ein effizientes System zur Aufgabenlösung, das niemand bedienen kann.“

Um ein aufgabenorientiertes System zu erreichen, soll möglichst viel Aufgabenwissen erworben werden.

Durch Analyse von Lastenheft, der von Pepperl+Fuchs bereitgestellter Excel-Dateien (Produktzusammenstellung) und Kundengespräche ist das Pflichtenheft entstanden. Zusammen mit den Team-internen Diskussionen ist dies die Basis der Aufgabenorientierung.

3.4.4. Bedienungsfreundlichkeit

Bei der Bedienungsfreundlichkeit spielen folgende Punkte wesentliche Rolle:

- **Transparenz:**
Der Benutzer soll intuitiv erkennen, wie er mit dem System arbeiten kann. Er soll immer wissen, wo er sich in seiner Aufgabe befindet, was er hier tun kann, woher er gekommen ist und wohin er weiter navigieren kann. Hier spielt der Visualisierungsgrad eine wichtige Rolle. Informationen und Funktionen sollten möglichst direkt ersichtlich sein. Eine tiefe Funktionshierarchie, die Funktionen unter vielen Menü-Hierarchien versteckt, soll verhindert werden.
- **Erwartungskonformität:**
Benutzererwartungen an das Aussehen und Verhalten der Software sollen analysiert und umgesetzt werden. Darüber hinaus bringt der Benutzer Erfahrungen aus seinem Aufgabengebiet mit.
- **Konsistenz:**
Das Bedienkonzept der Software ist einheitlich zu gestalten.
- **Unterstützung:**
Der Benutzer soll in seiner Arbeit von der Applikation ideal unterstützt werden. Dies geschieht durch ein Hilfesystem, eine klare Benutzerführung und aussagekräftige Fehler- und Warnmeldungen. Das Hilfesystem ist unter anderem kontextsensitiv, indem es Hinweise auf die Fehlerbehebung aufzeigt.
- **Fehlertoleranz:**
Fehlerhafte Eingaben sind soweit wie möglich abzufangen, um die Robustheit der Software zu erreichen.
- **Individualisierung:**
Der Benutzer soll die Möglichkeit haben eigene Einstellungen und seine Kontaktinformationen zu speichern.

3.4.5. Änderbarkeit

Die Software soll modular aufgebaut sein, um nachträgliche Änderungen und Erweiterungen zu implementieren. So ist der Datenverkehr von Client zu Server eine mögliche Erweiterung, die nach dem Projekt noch hinzukommen könnte. Solche Änderungen sollen durch den modularen Aufbau schnell und effektiv umsetzbar sein.



3.5. Maßnahmen

Im Folgenden werden die Maßnahmen aufgeführt, die nicht Teil der Testspezifikation sind.

Wer	Wann	Beschreibung der Maßnahme
Externe Tester	Testphase	Usability-Test. Anhand eines Fragenkatalogs soll der externe Tester seine Eindrücke über die Software dokumentieren. Durch Auswerten dieser Dokumente sollen Schwachstellen unserer Visualisierung aufgedeckt werden.
Team/Kunde	Anforderungs-analyse	Aufgabenorientiertheit definieren. Näheres siehe unten.
Qualitäts-beauftragter	Design, Implementierung, Testphase	Anhand einer Checkliste soll die Vollständigkeit der Funktionalität überprüft werden.
Entwickler	Design, Implementierung, Testphase	Dokumentation aus Sicht der Entwickler
Entwickler	Implementierung	Coding-Styles