

Excellence im Engineering

durch konsistente Dokumente im V-Modell

Wer bleibt schon von wachsender Komplexität verschont? Müssen Sie sich auch nach einem neuen leistungsfähigeren Mikrocontroller umsehen, weil der derzeitige den Anforderungen nicht mehr gerecht wird? Und wie sieht es aus mit den Anforderungen im Software- und Systems- Engineering? Wie steht es um die Konsistenz der Projekt-dokumentation, den Lasten oder Pflichtenheften zum Beispiel gegenüber dem Code? Wurden sie auf Basis von Word Dokumenten erstellt? Bei steigender Komplexität geraten auch Vorgehen im Software-Engineering an die Grenzen der Effizienz.

WILLERT.
pioneers in embedded software engineering

Im Software Engineering werden immer noch viele Informationen in Form von Office - Dokumenten gemanaged. Dabei entstehen je nach Phase und Technik im Engineering-Prozess separate Dokumente mit redundanten Informationen. Unter dem wachsendem Zeitdruck ist es fast unmöglich diese redundanten Informationen in den verschiedenen Dokumenten konsistent zu halten.

So zeigt eine Umfrage unter unseren Kunden, dass in 95% der Projekte die Dokumentation der Software (egal, ob sie mit MS Word, MS Visio oder anderen dokumentenzentrischen Werkzeugen erstellt wird) nahezu nutzlos ist, da sie nicht konsistent mit dem Source-Code gehalten werden kann und damit nicht verlässlich ist.

In Projekten, in denen auf Grund von Sicherheitsanforderungen hohe Qualität gefordert wird, gehört es schon seit langem zum Stand der Technik, dass alle Dokumente, Modelle, Code und Testfälle untereinander konsistent gehalten werden. Grundsätzlich ist es also möglich und nur eine Frage des Vorgehens. Was mit einer dokumentenzentrischen Arbeitsweise ein fast unmögliches Unterfangen ist wird auf Basis eines Repository auf einmal sehr einfach.

Dieses White Paper zeigt, wie Informationen auf Basis eines Repository effizient genutzt werden können.

Lassen sich redundante Informationen vermeiden?

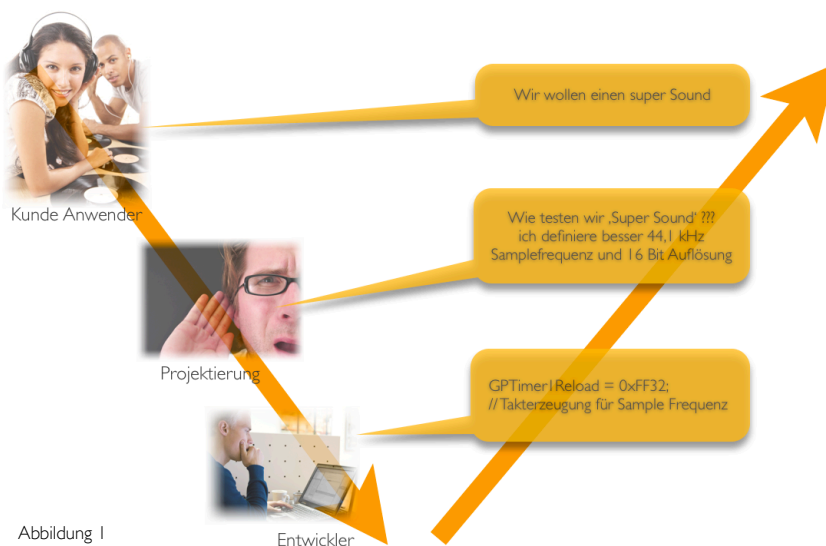
Stellen wir uns einmal die Rollen und Personen in folgendem Projekt vor. Eine Firma möchte die Entwicklung eines Audio DA Konverter in Auftrag geben. Betrachten wir die so genannten Stakeholder in diesem Projekt, hinsichtlich einer Produkteigenschaft, dann könnten in etwa folgende Informationen entstehen.

Der angestrebte Zielmarkt sind Menschen mit gehobenem Musikanspruch, hinsichtlich des Sounds steht im Lastenheft ‚HiFi High End Qualität‘ (Abb. Nr.1)

Der Projektleiter, hat bei dieser Definition berechnete Bedenken bezüglich der Testbarkeit und definiert etwas exakter eine Auflösung von 16 Bit und eine Samplefrequenz von 44,1 kHz entsprechend dem RedBook Standard für Audio CD's.

Der Entwickler programmiert einen Timer, um den Takt für die erforderliche Samplefrequenz zu erzeugen und die Syntax könnte in etwa so aussehen: `GPTimer1Reload = 0xFF32;`

Eigentlich steckt hinter der Syntax aller Stakeholder die gleiche Information. Aber können wir auf die verschiedenen sprachlichen Ausprägungen verzichten? Ich befürchte nicht. Die Aussagen sind domänenübergreifend nicht mehr verständlich oder nicht präzise genug. Wir müssen also mit redundanten Informationen leben, aber wie geht das mit vertretbarem Aufwand?



Im dokumentenzentrischen Vorgehen ergeben sich eine große Anzahl an Dokumenten mit redundanten Informationen und gegenseitigen Abhängigkeiten, wie in Abbildung Nr. 2 zu sehen ist.

Werfen wir einen Blick in die Dokumente, (Abbildung Nr. 3) dann stellt sich heraus, dass es sich in der Regel um eins zu n Beziehungen handelt.

In realen Projekten ist die Größe eines Lasten- oder Pflichtenheftes im Bereich von 100 Seiten nicht selten. Ändert sich eine Anforderung im Lastenheft, dann wird niemand das komplette Pflichtenheft durcharbeiten, um alle Abhängigkeiten herauszufinden. Dieses

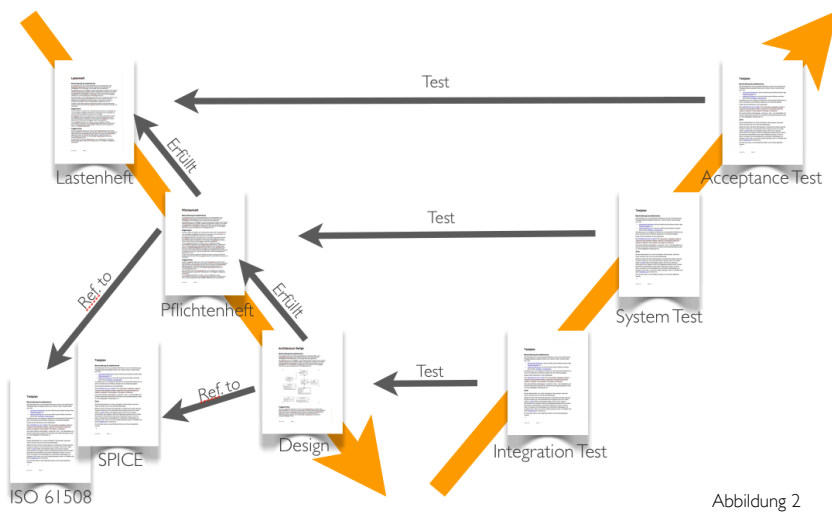


Abbildung 2

ist der erste Schritt zu inkonsistenten Dokumenten.

Einfacher geht es auf Basis eines Repository z.B. mit Hilfe eines Anforderungsmanagement - Werkzeugs. In Abbildung Nr. 4 ist die Gegenüberstellung der Dokumente auf Basis ihrer Beziehungen zu sehen. Hier können Auswirkungen auf einen Blick vollständig erfasst werden. Auf Basis eines Repository lassen sich Abhängigkeiten von Anforderungen eines Projektes in Beziehung setzen.

Sehr häufig werden als Quellen für Anforderungen lediglich Kunden oder Anwender gesehen. In der Praxis können Anforderungen aber von überall her kommen.

Stellen Sie sich vor, die Leistung des eingesetzten Mikrocontrollers ist nicht mehr ausreichend. Es ist ein HW Re-Design geplant und in diesem Zusammenhang muss ein neuer Mikrocontroller ausgewählt werden. Was sind die Anforderungen für den MC? In diesem Fall, in dem die SW wiederverwendet werden soll, kommen die Anforderungen im wesentlichen aus der Implementierung. Wurden im Repository die Abhängigkeiten der Software-Dokumentation zur genutzten Peripherie des MC definiert, dann lässt sich sehr schnell ein Dokument erstellen, in dem alle Peripherie-Elemente mit den Anforderungen aus der Software in Bezug zur MC Peripherie stehen.

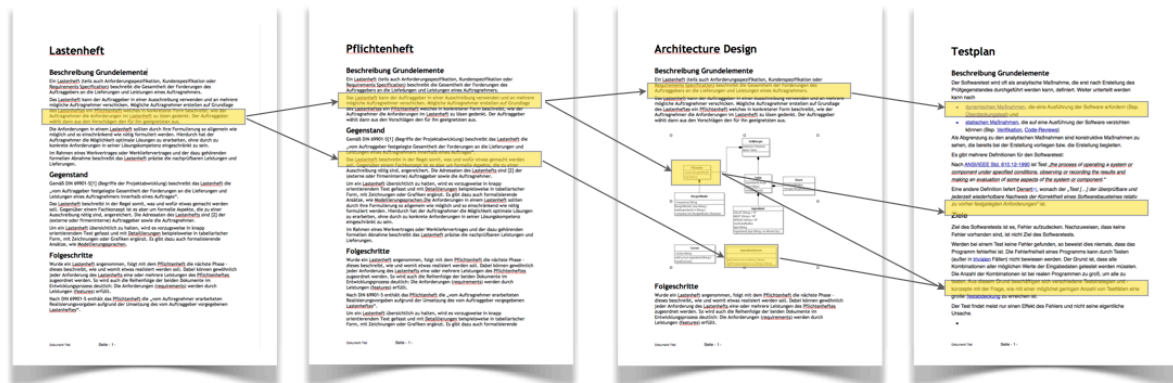


Abbildung 3

CR-5	2 Detailed Description		
CR-6	2.1 Quality of sound		
CR-7	The digital to analogue conversion has to be based on a sample-frequency of 44,1kHz and 16bit resolution.	SS-45 The digital to analogue conversion has to be based on a sample-frequency of 44,1kHz and 16bit resolution.	ASGR-5 The Timing is based on the Samplefrequency 44.1 kHz. ASGR-6 The resolution is based on 16 Bit. ATR-5 Timing of the Timer is based on the Sample Frequency of 44.1 kHz
CR-8	The soundsystem has to be able to produce at least 5Watts with a maximum distortion factor of 1%.	SS-46 The soundsystem has to be able to produce at least 5Watts with a maximum distortion factor of 1%.	
CR-9	2.2 Timbre		
CR-10	The soundsystem has to support only one timbre and should be close to the sound of an Piano	SS-48 The soundsystem has to support only one timbre and should be close to the sound of an Electronic Organ. SS-49 The frequency-response is from 20Hz to 18kHz. SS-50 Polyphony up to 20 simultaneous tones.	NCR-4 Calculates Sine-values for different sound Timbres. At the moment one sound (Electronic Organ) is implemented. ASGR-5 The Timing is based on the Samplefrequency 44.1 kHz. ASGR-3 The AudioSoundGen has to read the valid notes from the Array provided by the NoteController and has to programm the DA Converter. NCR-3 For polyphonic Sound this controller provides an Array of sine-values for the AudioSoundGen.

Abbildung 4

Implementierung auf Basis eines Repository

Nun werden nicht alle Arbeitsschritte auf Basis textueller Informationen durchgeführt. Zum Beispiel wird Software programmiert oder modelliert. An dieser Stelle entstehen redundante Informationen zwischen Source-Code und Dokumentation. Auch diese Zusammenhänge und Abhängigkeiten lassen sich auf Basis eines Repository leichter verwalten.

Modellierungswerkzeuge, auf Basis von UML zum Beispiel, ermöglichen Import und Synchronisation von Anforderungen in das Modell. Innerhalb des Modells können die Anforderungen (z.B. zur Dokumentation des Modells) direkt in einer Darstellung angezeigt werden, sie können aber auch mit Modellelementen verlinkt werden und im Fall der Codegenerierung als Kommentar in den Code übernommen werden. Das gleiche gilt auch für andere Werkzeuge, z.B. zur Erstellung von Regression Tests. Voraussetzung ist immer ein Repository. Auf Basis der Traceability eines Repository's oder mehrerer verlinkter Repositories können folgende Analysen durchgeführt werden.

- Coverage Analyse - sind alle Anforderungen erfüllt
- Impact Analyse - Analyse der Auswirkung möglicher Änderungen
- Trace Analyse - Analyse der Ursache

Automatische Nachricht über Änderungen

Sind die Abhängigkeiten verschiedener Informationen im Repository definiert und hat jeder Stakeholder ein Dokument mit seiner domänenspezifischen Sichtweise auf das System und anschließend ändert sich etwas im System, zu dem sein Dokument einen Bezug hat, dann wird in seinem Dokument ein so genannter Suspect Link angezeigt (Abbildung Nr 5). Mit einem Mausklick darauf kann er die Änderungen in angrenzenden Domänen nachverfolgen.

Auf diese Weise werden alle entstehenden Inkonsistenzen sichtbar und einfach nachverfolgbar gehalten. Eine wesentliche Voraussetzung, um die entsprechend der Domänen und Stakeholder redundanten Informationen untereinander konsistent zu halten.

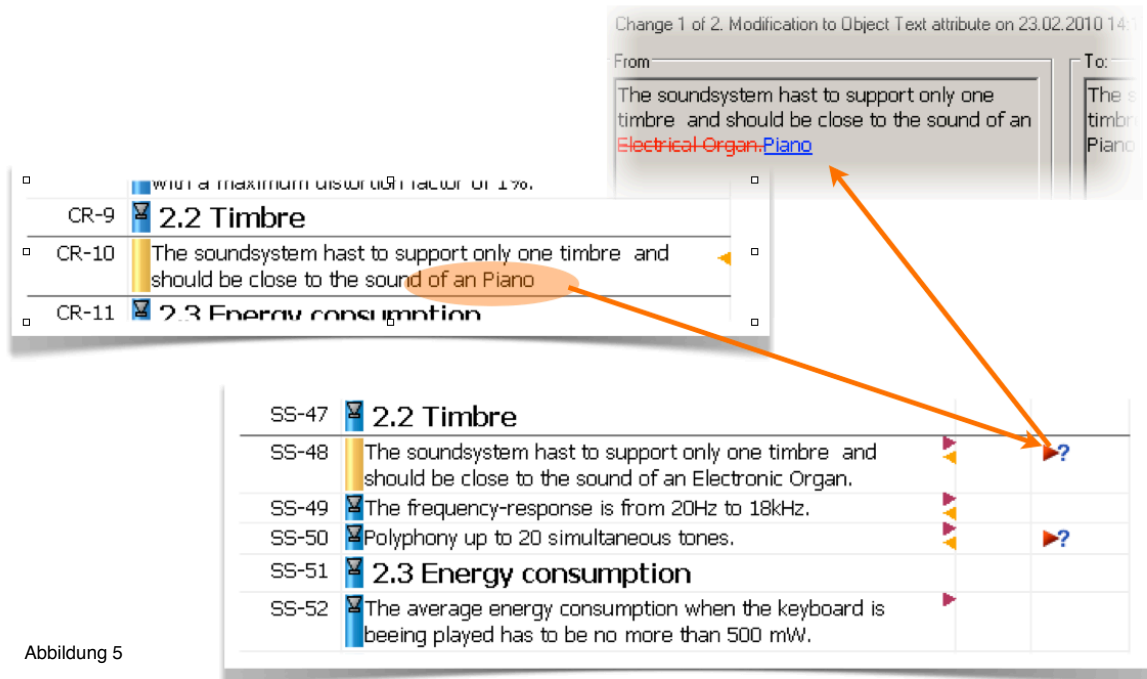


Abbildung 5

Aufwand zur Umstellung eines Dokumenten- zum Repository-zentrierten Vorgehen

Zur Einführung eines Repository zur Speicherung und Strukturierung aller projektrelevanten Informationen sind auf Basis eines geeigneten Anforderungsmanagement-Werkzeuges folgende Schritte erforderlich:

1. Anpassung der Struktur des Repository auf die Vorgehensweise und die Informationen des Unternehmens. (Erstellung eines so genannten Datenbank-Schemas)
2. Definition und Erstellung von Darstellungen und Ansichten der Repository-Inhalte für die verschiedenen Fachdomänen bzw. Stakeholder.

3. Evtl. Import vorhandener Dokumente in das Repository.
4. Definition der Abhängigkeiten.
5. Evtl. Export der Daten in übliche Dokumente für Stakeholder, die nicht mit dem Werkzeug arbeiten.
6. Schulung des Teams im Umgang mit Methode und Werkzeug

Für die Durchführung der obigen Schritte werden folgende Kenntnisse und Aufwand benötigt:

Für die Erstellung eines Datenbank-Schemas muss mit einigen Tagen gerechnet werden. Beim ersten Mal sollte hier auf jeden Fall die Unterstützung eines Experten herangezogen werden. Das Datenbank-Schema ist das Gerüst des Repository's. Bei der Erstellung müssen bereits im Vorfeld mögliche zukünftige Analysen und Darstellungen berücksichtigt werden, die evtl. erst im Verlauf des Projektes interessant werden. Das geht nur mit Erfahrung.

Für die Erstellung der domänenspezifischen Ansichten (Views) müssen ein bis zwei Tage Aufwand gerechnet werden. Auch hierfür ist beim ersten Mal die Unterstützung durch einen Experten hilfreich.

In der Regel liegen bereits Dokumente mit Informationen vor. Gut strukturierte Dokumente lassen sich mit modernen Anforderungsmanagement - Werkzeugen sehr einfach automatisch importieren. Schlecht strukturierte Dokumente müssen vorher strukturiert werden. (GIGO - Garbage in Garbage out). Auf Basis von Styles bzw. Formatvorlagen sind hier pro Seite ca. ein bis drei Minuten zu rechnen, wenn die Inhalte nicht verändert werden müssen.

Die Definition der Abhängigkeiten empfiehlt sich im Verlauf des Projektes nach und nach durchzuführen. Der Vorgang an sich ist ein einfaches Drag und Drop mit der Maus. Innerhalb einiger Monate sind die wichtigsten Abhängigkeiten in der Regel definiert. (Vorausgesetzt das Team hat den Nutzen erkannt, wird in der Praxis fleißig verlinkt)

Inhalte aus dem Repository lassen sich sehr einfach in RTF, MS Word oder HTML Dokumente exportieren. Etwas schwieriger wird es, wenn die Dokumente festgelegte Formate und Inhalte haben müssen. Dann wird auch hier entsprechender Aufwand, z.B. die Erstellung einer MS Word - Vorlage, notwendig sein.

Last but not least bleibt die Schulung des Teams. Grundsätzlich wird zwischen Anwendern (Usern) und so genannten Power Usern unterschieden.

Die Bedienung der meisten Werkzeuge ist heute nicht viel schwieriger, als die Bedienung einer Tabellenkalkulation. Ein Tag Schulung ist hierfür in der Regel ausreichend. Voraussetzung ist, dass das System konfiguriert und eingerichtet ist.

Ein bis zwei Mitarbeiter im Unternehmen sollten auch dafür die Kenntnisse besitzen. Hier sind einige Tage Kombination aus Schulung und Coaching notwendig.

In Summe sollte zur Einführung eines Anforderungsmanagement - Systems mit 10 - 15 Tagen Aufwand, ca. 10T€ für Schulung und Coaching und 2-5 T€ pro Arbeitsplatz für die Lizenz eines Werkzeuges gerechnet werden.

Anforderungen an das Werkzeug

Einer der Hauptvorteile im Anforderungsmanagement liegt in der Traceability - Analyse der Abhängigkeiten und den daraus entstehenden Ansichten. In der Praxis gibt es häufig verschiedene Sichtweisen auf die Traceability (ist erfüllt durch, ist Testfall zu, wurde getestet, entspricht Norm ...). Diese sollten bei einer Analyse auch unterschieden werden können. Dazu müssen verschiedene Linkbeziehungen und deren mögliche Richtungen exakt vorgegeben werden (So genanntes Link Controle). Ist das nicht möglich, so geben die Analysen in der Regel bereits nach kurzer Zeit keine sinnvollen Aussagen mehr.

Die Möglichkeit der Definition verschiedener Link - Arten und die exakte Vorgabe von Regeln zu deren Anwendung ist eines der wichtigsten Eigenschaften. Leider versagen an diesem Punkt schon sehr viele Werkzeuge.

Flexible Darstellung der Repository-Inhalte mit guten Filter-Möglichkeiten ist ein weiteres elementares Kriterium für ein gutes Werkzeug. Was hilft Ihnen das beste Repository, wenn die Informationen und deren Zusammenhänge nicht aussagekräftig dargestellt werden können. Vor allem Filter-Funktionen werden nach einiger Zeit sehr wichtig. Fängt sich Ihr Repository an mit Daten zu füllen, dann wird es für aussagekräftige Darstellungen zunehmend wichtiger zu definieren, was NICHT angezeigt werden soll. Sie hätten ja gerne den Blick auf das Wesentliche. Auch an dieser Stelle versagen bereits viele Werkzeuge.

Natürlich sind gute Import- und Export- Funktionen sehr hilfreich. Vor allem auch zu anderen Werkzeugen. Zu diesem Thema ist IBM® Jazz® eine interessante Entwicklung (<http://jazz.net>). Jazz® hat u.A. zum Ziel Repositories verschiedener Werkzeuge zu vereinheitlichen. Wenn sich Jazz® durchsetzen kann wird Daten-Import und -Export in Zukunft der Vergangenheit angehören.

In diesem Sinn wünsche ich Ihnen viel Erfolg in Ihren Projekten mit und ohne Repository und beantworte auch gerne noch weitergehende Fragen zu diesem Thema.

Willert Software Tools GmbH - Hannoversche Str. 21 - 31675 Bückeburg
Tel.: 05722 - 9678 60 - email: info@willert.de - www.willert.de