【SWEST13】S45-d
「新しい開発文書の時代を迎えて」

システム開発文書品質研究会
(ASDoQ)

2011 年 9 月 2 日

「新しい開発文書の時代を迎えて」構成と対象

【構成】

チュートリアル

＋パネルディスカッション

【対象】

若い開発者

（もちろん熟練した開発者の方にも…）

「新しい開発文書の時代を迎えて」時間割

◆13:00 ～ 13:40

【チュートリアル】「誰がために開発文書を書く」

講師: 山本 雅基（名古屋大学）

◆13:50 ～ 15:50

【パネルディスカッション】「開発文書と私」

パネラ:

坂本 佳史（日本 IBM）
塩谷 敦子（イオタクラフト）
清水 吉男（システムクリエイツ）
杉本 明加（富士設備工業）
藤田 悠（長野高専）
森川 聡久（ヴィッツ）
山本 雅基（名古屋大学）

【チュートリアル】
「誰がために開発文書を書く」

◆講師: 山本 雅基（名古屋大学）

◇コーディネータ: 栗田 太郎（フェリカネットワークス）

◆概要:

技術者が開発文書を書く目的をご存じですか? 誰のために何を書いているのでしょうか? (あるいは書いていないのですか?) 誰のために何を書くべきですか? それは, この課長やあの部長のためではありません. 品質保証部やプロセス改善グループのためでもありません. 品質や機能安全やセキュリティや開発プロセスの標準に定められているからでもありません. 発注した企業, お客さまから要求されているから? いえ, 違います. 誰のために何を書くの? 開発文書に関する経験が豊富な組込みの専門家による, 若い開発者に (もちろん熟練した開発者の方にも) 向けた, 笑顔があふれる, ためになるチュートリアルです. 是非ご参加ください.

【パネルディスカッション】
「開発文書と私」

◆パネラ:

坂本 佳史（日本 IBM），塩谷 敦子（イオタクラフト），清水 吉男（システムクリエイツ），杉本 明加（富士設備工業），藤田 悠（長野高専），森川 聡久（ヴィッツ），山本 雅基（名古屋大学）

◇コーディネータ: 栗田 太郎（フェリカネットワークス）

◆概要:

2011 年の 7 月に立ち上げたシステム開発文書品質研究会 (ASDoQ) では，開発に関わる文書品質の分析，測定，その改善を目指して活動を行います．本パネルディスカッションでは，会の研究員が持ち寄る文書および文章に対する意見，実例の交換や，文書の課題に関する会場を交えた議論を通して，開発文書の重要性，文書作成の課題等を探っていきます．多くの方々のご参加とご発言をお待ち申しあげております．

◆各パネラの発表内容

- 私にとっての開発文書
- 開発文書にまつわる思い出
- 問題のある文書，文章の実例とその弊害，作成の背景
- 私が書いた，私が感動した良い文書，文章の実例とその意義，書き方，工夫

パネラの自己紹介

（氏名の五十音順）

坂本 佳史（日本IBM）

自己紹介

坂本 佳史 Yoshifumi Sakamoto

sakay@jp.ibm.com

日本アイ・ビー・エム株式会社

グローバルビジネスサービス / スマーターエンジニアリング事業開発

PMP®, IBM Certified Professional – Executive Project Manager

[プロフィール]

1985 年日本IBM 入社. 論理回路設計、システム設計・開発などに従事.

その後, プロジェクトマネージャとして主に組み込み機器の開発プロジェクト、SoC (System on Chip)の開発プロジェクトに従事.

現在は研究開発をテーマとしたプロジェクト群のプログラム・マネージャ、

組み込みソフトウェア開発におけるプログラム・マネジメントに従事.

私にとっての開発文書

製品開発プロジェクトにおける最も重要なコミュニケーション手段の1つ

しかし現実は，

- 文書の作成 - エンジニアが最もやりたくない“作業”
- 作成された文書 - 文書化が目的なので“取説”と同じ扱い
- 文書作成の作業 – 真っ先に削減の槍玉にあげられる

コミュニケーションや開発プロセスで活用されていない場合も多い

塩谷 敦子（イオタクラフト）

塩谷敦子 Shioya Atsuko の自己紹介

■ソフトウェア開発文書の改善活動家

◆教育と技術相談

- 名古屋大学NEP*)公開講座「ドキュメントレビュー」 (2010～)
  http://www.nces.is.nagoya-u.ac.jp/NEP/　NEP *) : 社会人組込み技術者向け教育プログラム
- 組込みシステム産業振興機構による組込み適塾(先進的組込みソフト産学官連携プログラム)内「ソフトウェア開発ドキュメンテーション」講座(2008～)
  http://www.kansai-kumikomi.net/ptraining/img/kumikomi_panf2.pdf
- その他、富士通ラーニングメディア, CQ出版などで公開講座のほか、
  個別企業向け教育(開発の現場での実文書を使ったカスタマイズ研修など)
- 開発の現場でのソフトウェア開発文書診断

◆研究

- 長野高専の寄付研究部門にて文書診断に関する研究 (2009～)
  http://mdes.nagano-nct.ac.jp/
- システム開発文書品質研究会(ASDoQ)
  http://asdoq.jp/, www.facebook.com/ASDoQ

◆広報

- ドキュメンテーション技術の連載記事「理系のための文書作成術」
  CQ出版 電子・組み込み技術の総合サイト「組み込みネット」の技術解説(2010/8～2011/6)
  http://www.kumikomi.net/archives/kumikomi/technology/

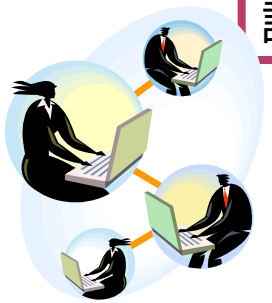
■自動車メーカ系列の研究所時代: 上司からの“赤ペン入れられまくり経験”が今に活きる

＜私にとっての開発文書＞

開発者のみなさんと共有する“よすが”

よすがとは：（「寄す処(か)」の意．1) 身や心のよりどころとすること．頼りとすること．
2) 手がかり．手だて．方法．（提供元「デジタル大辞泉」）

講習会



ソフトウェア開発者たち

開発文書

診断

受診

ソフトウェア開発者

治療・予防・
生活改善

・ライティング力の向上へ
・開発力の向上へ
・自身のマネージメント力の
向上へ

・開発文書の品質向上へ
・工程作業の品質向上へ
・開発プロセスの品質向上へ
・製品の品質向上へ

・組織力の向上へ

清水 吉男（システムクリエイツ）

System Creates

自己紹介

- （株）システムクリエイツ　代表取締役　（ただし社員はいない）
- 1968年にソフトウェアの世界に参入．但し失敗が多く3年で退散.
- １年後に復帰、その後、仕様を漏らさない表記法を考案
  - （→後にUSDMとなる）
- 1977年に組み込みシステムに転向、
- 翌年、派生開発の依頼を受け、１週間で「XDDP」のプロセスを考案
- 以後、1995年までのすべての開発案件で納期遅れ、仕様トラブルなし
- 1996年からプロセス改善のコンサルティングに転向
- 2010年、派生開発推進協議会を設立し、「XDDP」などの普及を図る
- 著書
  - 「ＳＥの仕事を楽しくしよう」（ＳＲＣ）
  - 「要求を仕様化する技術・表現する技術」（技術評論社）
  - 「『派生開発』を成功させるプロセス改善の技術と極意」（技術評論社）
  - 「わがSE人生に一片の悔いなし」（技術評論社）

System Creates

私にとっての開発文書

- 最初の3年・・・要求仕様書で破綻→撤退
  - 当時の要求仕様書・・・顧客から「FIX」の言質を取って来る！
  - テスト終盤に顧客からクレーム・・・納品拒否！
  - 設計技術、設計文書には特に問題はなかったが、顧客の求めるものを実現できていなかった

- 復帰後、仕様漏れのない要求仕様書の書き方を模索
  - IBM/360のOSを勉強しているなかで、ヒアリング時に「仕様」の他に「要求」と「背景」を聞き出すことを考えついた
  - 顧客が気付いていない仕様に気付く効果も
  - その後の「USDM」に繋がる

これ以降、仕様に関するトラブル・・・なし

杉本 明加（富士設備工業）

自己紹介

- 職歴
  - 2002年～ 某T社系列会社で計測器の組込みアプリ開発に従事
  - 2006年～ 某ソフトウェア開発会社で組込みRTOS開発に従事
  - 2010年～ NCES(名古屋大学大学院情報科学研究科附属組込みシステム研究センター)で研究業務に従事
  - 2011年～ 富士設備工業株式会社でツールサポートに従事
- OSS開発歴 *← 今日はここの立場でディスカッション*
  - 2006年～ 某ソフトウェア会社でTOPPERSプロジェクトに参画
  - 2010年～ 個人としてTOPPERSプロジェクトに参画
    TOPPERS/ASP V850依存部担当
    TOPPERS/SSP ターゲット非依存部，Cortex-M3担当
- 興味
  - 楽しくソフトウェアを開発すること
  - みんなの役に立つソフトウェアを開発すること

私にとっての開発文書

- 昔
  - 大嫌い，今も好きではない…
  - 動くもの（ソースコード）が正義，他の成果物は付属品
- 今
  - 要件（＝実現したいこと）を動作するソースコードに変換する思考過程を表現したもの

    ソフトウェアの骨格を表現するには自然言語のほうが整理，理解しやすい

    → よりよい成果物の提供のために重要
  - 不特定多数が関わるオープンソースプロジェクトではドキュメントが重要

    ソースコードだけでは抽象度が低すぎる

    → 多くの人との情報の共有のために重要
- 経験を積むことで文書の重要性に目覚めた

藤田 悠（長野高専）

藤田 悠の自己紹介

- 2009年から長野工業高等専門学校　寄附研究部門制御システム開発研究部門(ミマキエンジニアリング）にて活動しています.
- 組込みソフトウェア開発文書に潜んでいる問題の「診察」と「診断」と「治療」と「予防」をする「文書診断法」の体系化を行っています.
- 文書診断で明らかになった問題を「治療」するために,企業の技術者を対象にした研修を行っています.研修の内容には，診断した文書に含まれる問題を題材にして，その問題を解消する演習などがあります.

私にとっての開発文書

- 書き手の考えがつまびらかになります

  考えが尽くされていないことを文書にしようとしても，
  あいまいにしか書けません．書いて見直すことで，
  考えを深めなければならないことに気づきます．

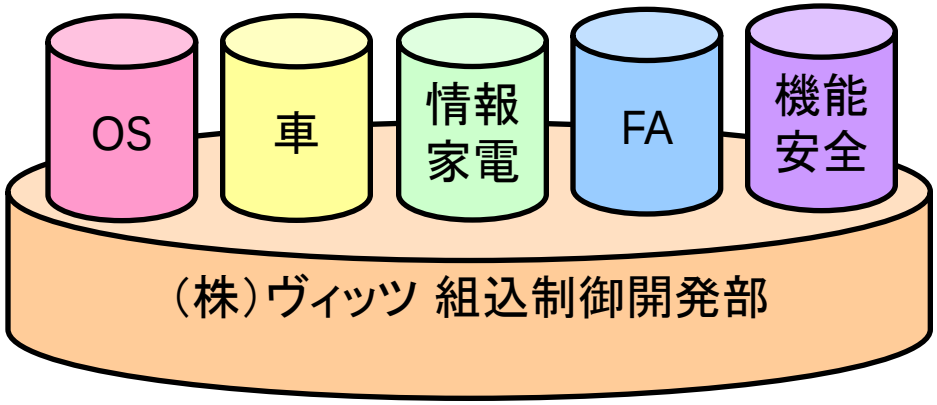
- 正解は一つではありません

  「問題である」という指摘が正しいとは限りません．
  しかし，指摘を受け止めて，指摘の余地を与えない
  程に熟考と推敲を繰り返すことで品質が高まります．

森川 聡久（ヴィッツ）
自己紹介と機能安全の概要

森川@ヴィッツ の自己紹介

- 業務経験
  - 工作機のソフト開発
  - デジタルTV受信機のソフト開発
    - BSデジタル、CS110度、地上波デジタル、北米受信機
    - DVD/HDDデコーダのソフト開発
  - 車載用プラットフォームの研究開発
    - OSEK/VDX仕様OS、保護機能OS、タイムトリガOS、AUTOSAR OS
    - FlexRay通信ミドルウェア
  - **機能安全対応開発（2006年～）**
  - 品質活動
    - 品質管理規定の策定
    - 品質保証室
  - 組込み教材開発
  - 社内教育
- **現在の業務**
  - **機能安全対応の支援**
- 社外活動
  - SWEST13ローカルアレンジメント委員長
  - イーエスピー企画 土日システム開発部



機能安全が何故求められているか？

①安全なシステムを開発するため

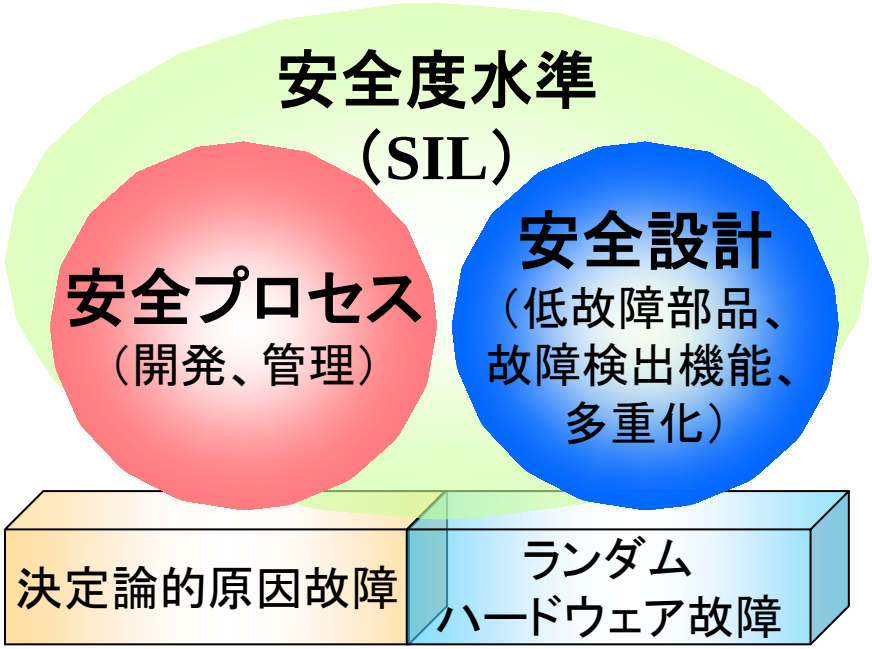
従来：不具合が無ければOKだった。
今後：故障への対策が必須。
　故障しても危険にならないように、予め対応しておく必要がある。

②安全性の説明が必要になりつつある

きちんと説明できるだけのエビデンスを残すことが必要。
独立した立場の人による、安全性の確認（検証、監査）が可能であること。

③非関税障壁への対応

機能安全対応しないと、海外へ輸出できなくなる可能性がある。



私にとっての開発文書(1/2)
~機能安全観点から~

- 機能安全の必要事項

満たせば → **安全性を説明可能**

  - **完全性**
    - 必要な情報が全て記載/記録されていること
    - 過不足・矛盾がない
    - あらゆる観点での設計検証を実施して確認

  - **再現性 ＋ 客観性**
    - 再現/再検証できること
      (他者でも、数年後でも)

⇒ 再現できない内容だと・・・
・安全であることを、客観的に確認できない
・後の不具合発生時に、問題原因を追究できない

  - **可読性 ＋ 客観性**
    - 同意の理解ができること
      (他者が見ても、数年後に見ても)

⇒ 誤解するような内容だと・・・
・関連モジュールに不具合混入の恐れ
・後のメンテで誤修正の恐れ

- **備考:米国IV&V 対応でも同様に必要!!**
  - IV&V:独立性検証と妥当性確認(Independent Verification&Validation)
    - 例)自動車がアクセルを踏まないのに急加速した問題で、米国NASAが検証

私にとっての開発文書(2/2)
~機能安全観点から~

- 課題:
  - 国際規格は抽象的
    - “明確な基準”は存在しない
      → どの程度文書化すればよいか不明確
- 現状の評価基準
  - 認証機関の独自判断
    - 認証機関によってばらつきあり。
    - 但し、肝となる考え方はほぼ共通か。(森川の感覚)
  - 自社の安全基準
    ＋ 各ステークホルダーのスキル基準

“基準を満たしているか”
よりも、
“肝を理解して対策しているか”
が重要!!

これがわかっていないと規格に振り回されてしまう。

「開発文書と私」

- 各パネラの発表
  - 「開発文書にまつわる思い出」
  - 「問題のある文書」
  - 「良い文書と工夫」
  - 「まとめ」
- 質疑応答

藤田 悠（長野高専）

ー文書診断の現場から (1/2).
設計行為のための文書の記述

開発文書にまつわる思い出

文書化にあらがう理由（わけ）

- ドキュメントを書かない理由
  - 誰も読んでくれない.
    何にも使われていない.

- ドキュメントを読まない理由
  - ソースコードを直接見て得られる以上の
    有用な情報が得られない.

問題のある文書

＜Aインク切替処理STEP1起動条件＞

01-01　何れかの　でもセットされたときに起動する。

のタイミングで全色の波形をAに切り替えている。

01-02　Aインクで初期処理した場合は起動しない。

【理由】最初から　が選択されるので、メッセージ表示する意味がないから。

Aカートリッジが1本でもセットされた状態で初期処理を実施した場合、自動的にインクセットはAになる。

【理由】最初からAパターンが選択されるので、Bインクセットにしておく意味がないから。

01-04　この後、　えば、次回からは起動しな

【説明】調整処理パターンプリントしたら、着弾を確認したとみなし、調整処理実　定する。

01-05　調整処理未実行では、以下のタイミングでSTEP1を再起動する。

リモート移行時
装置起動後のローカル移行時
リモートからキー操作によるローカル移行時

構成毎に何を記述するかが不明確
主語、述語、目的語が不明確
主語、述語、目的語が不明確
開発文書としての図表の基本的な表記ルール違反
表現の不統一
開発上の表現が不明確
助詞の誤り
用語定義なし
表現が不明確
表現が不明確
主語、述語、目的語が不明確
主語、述語、目的語が不明確
用語定義なし
表現が不明確
用語定義なし
表現が不明確
表現が不明確
表現が不明確
引用・参照の不備
主語、述語、目的語が不明確
意味的な誤用
開発上の表現が不明確
主語、述語、目的語が不明確

「起動条件」と宣言してあるのに・・・

3件の指摘

良い文書と工夫（1）

①表や箇条書きで整理したもの

起動条件
何れかの色のAインクカートリッジが1本でもセットされたとき。
位置調整が実行されていない状態でモードが変わったとき。
非起動条件
Aインクで初期充填した場合
Aカートリッジが1本でもセットされた状態で初期充填を実施したあと位置調整を行った場合

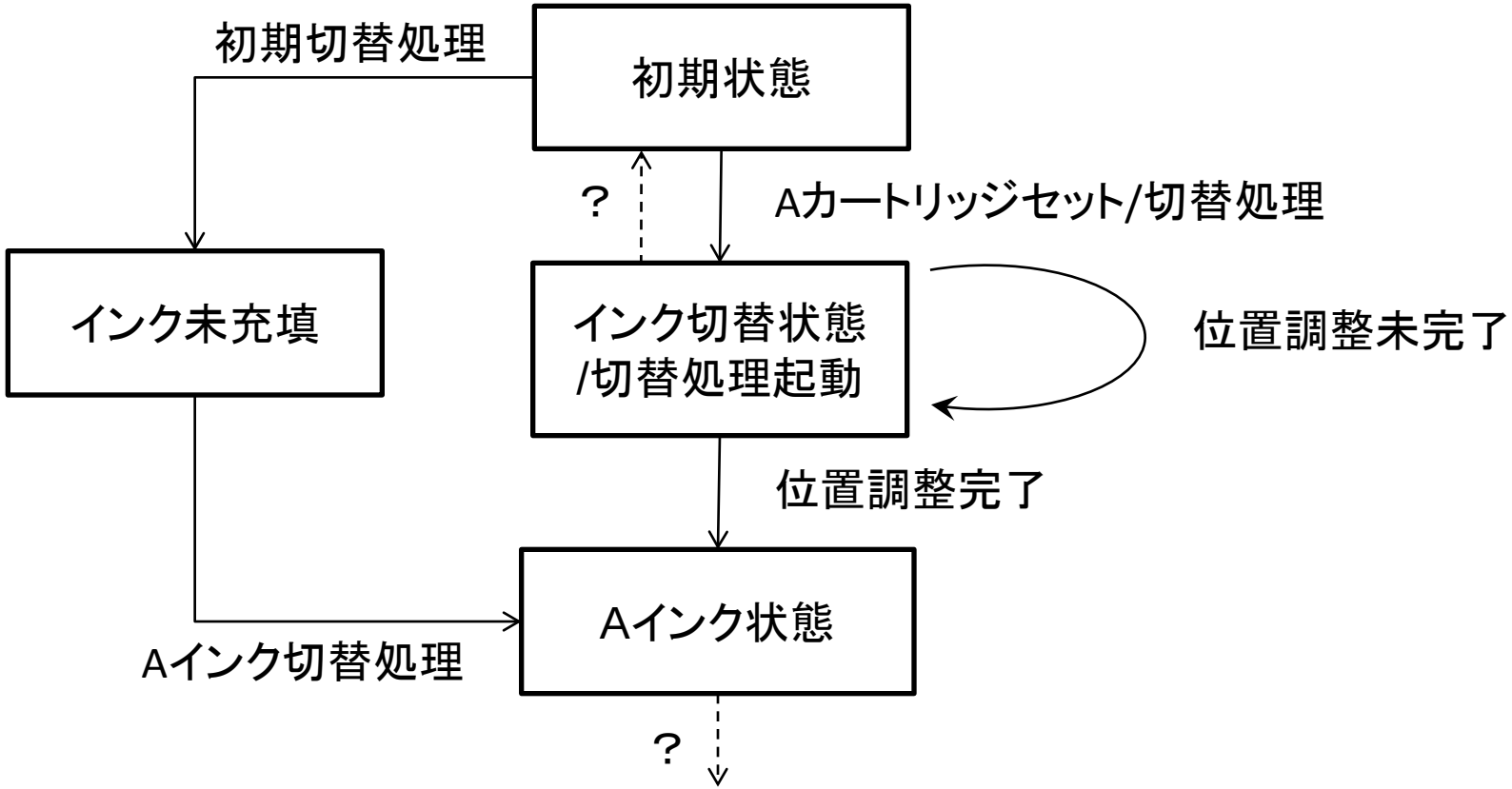
②段落の構成を考えて書き表したもの

Aインクカートリッジがセットされたときにインク切り替え処理を起動する。ただし、Aインクで初期充填した場合と、Aインクをセットされた状態で初期充填時に位置調整が行われている場合には、起動しない。

位置調整が行われていないときは、モード変更時にインク切り替え処理を起動する。

良い文書と工夫（2）

③図を用いて分析をおこなったもの



設計の妥当性や条件が網羅されているか
について踏み込んで考えている

まとめ

- 脳内推敲や脳内設計をする超人的な能力がない場合には，文書を使って推敲や設計をしましょう.
- 文書診断によって洗い出された問題に対処するための取組みは，文書の書き表し方を直すだけでなく，構造や設計を深く考えることにつながります.
- 文書の品質を高めるために，客観的な視点からの意見を積極的に取り込みましょう.

参考文献

- 三島 浩，『技術者・学生のためのテクニカルライティング 第2版』，共立出版，2001
- 森田 良行，『基礎日本語辞典』，角川書店，1986
- 風間 力三，『文章ドクター　悪文の診断と治療』，東京堂，1962
- 岩淵 悦太郎，『新版 悪文』，日本評論社，1961

塩谷 敦子（イオタクラフト）
ー文書診断の現場から (2/2).
「開発文書の心得 10 箇条」

＜開発文書にまつわる思い出＞

赤ペン先生が赤ひげ先生になるとき

この部分があいまいで，理解できません

診断

受診

ここは〇〇の設計の内容を書いたつもりでした．しかし，自分でも設計の根拠があいまいで，設計結果の検討そのものが不十分だったと気づきました

日本語表現としての分かりにくさをストレートに指摘したことが，開発上の不備や不足の指摘につながることがある．

例1：「水漏れ検出プログラム要求定義書」の目次

- 処理方法の記載は，要求定義ではない
- 「要求定義書」は「要求を定義するもの」目次からも，それが分かるように．
- 要求定義を明示しないまま，後工程へと進んでしまうことに．

例2：あるソフトウェア設計書の記述

3. 入力インタフェース
3.1 モータ電流センサ
3.1.1 目的
モータ電流センサは、モータ回転数のフィードバッグ制御
目的とした電流値を算出するために、使用する。

3.1.2 入力データ定義
電流センサとして、以下の2つを必要とする。
- 左モータ電流センサ
- 右モータ電流センサ

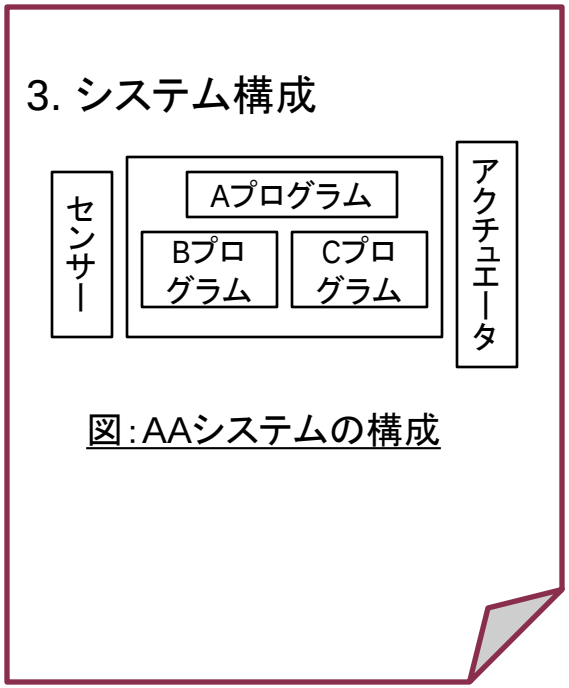
:

- システム側の「センサ」とソフトウェア側で定義すべきデータを混同している．設計対象は何かを明示しなければならない．
- e.g.「モータ電流センサに対して，ソフトウェアが取り扱うデータを各モータ電流値と定義する」

赤ひげ先生とは：山本周五郎の小説『赤ひげ診療譚』を原作として，黒澤明監督が映画化しタイトルを「赤ひげ」とした．
人情味あふれる名医として描かれている主人公の医師が赤ひげ先生と呼ばれていることから，そのような医師を例えて．こう呼ぶ．

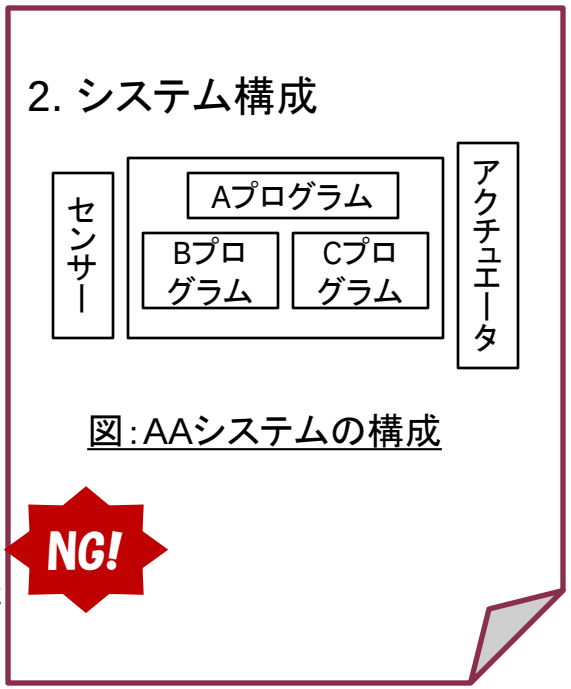
＜問題を含む文書＞
同じことを伝えてもソフトウェアは出来上がらない！

要求仕様書内の記述例



上位文書

アーキテクチャ設計書内の記述例



下位文書

- 上位文書（前工程作業）からのコピーは意味がない．
- 下位文書では，入力情報から導き出す何かを出力すること．
- やむを得ずコピーする場合は，なぜコピーするかを明記する．

＜良い開発文書と工夫＞
ソフトウェアドキュメンテーションの心得 10箇条

第1条. 開発プロセス構造は、ドキュメント体系で決める

第2条. 開発作業の構成は、ドキュメント構造で決める

第3条. 目次作りを開発作業計画の始点にする

第4条. 見出しと本文で、「言行一致」を実現する

第5条. アブストラクトは、読み手のためならず

第6条. 結果だけでなく根拠を明示する

第7条. 能動態で、作業に自信と責任を持つ

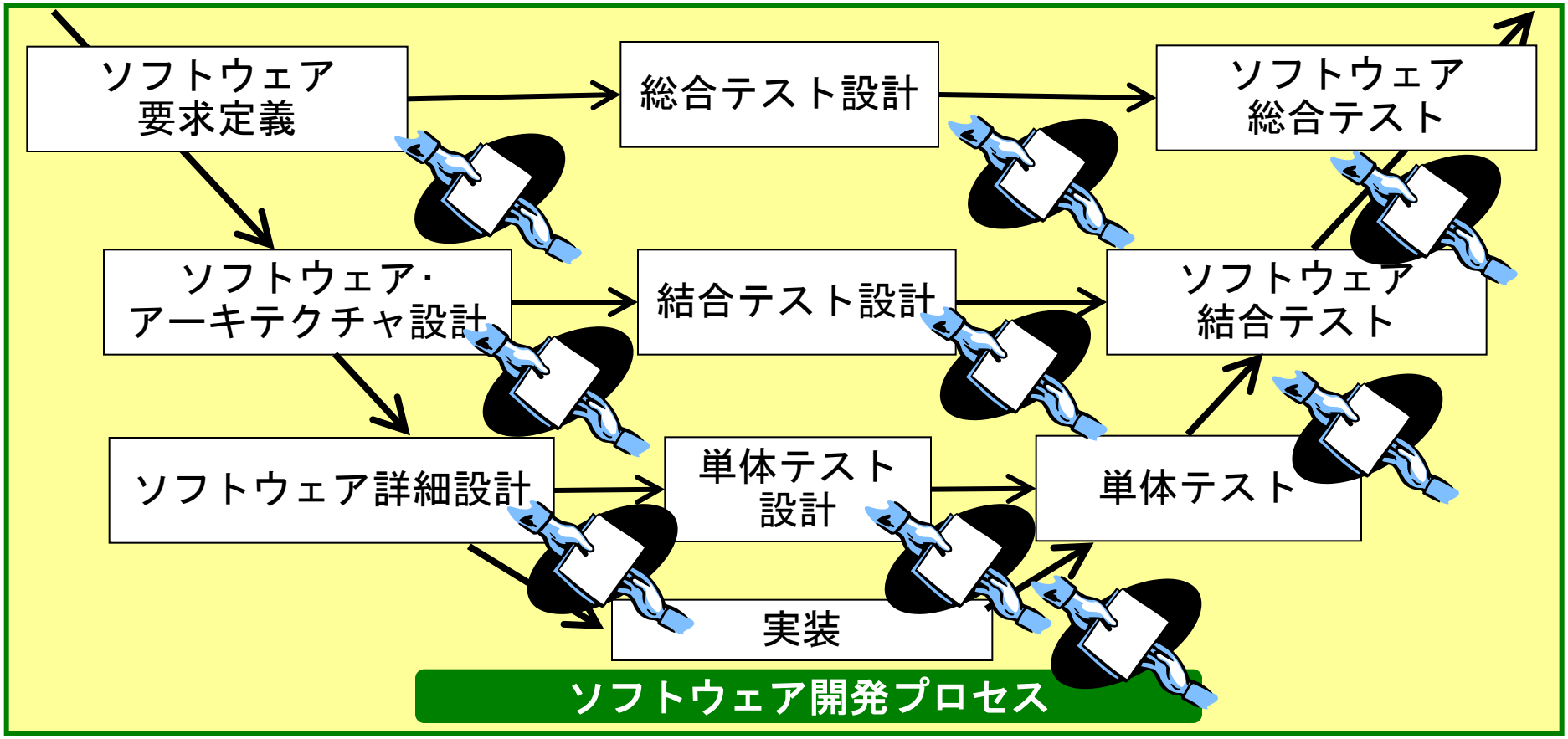
第8条. 用語定義は、読み手のためならず

第9条. 図やモデル表現を過信しない

第10条. 記述の「入力」と「出力」を明確に分ける

（同）イオタクラフト「ソフトウェアドキュメンテーション」研修教材から

＜良い開発文書と工夫＞
ソフトウェア開発は伝言ゲームじゃない



同じことを伝えない

◆開発は文書を順次詳細化していく作業

◆記述の入力（前提）と出力（結論）を区別する

＜おわりに＞
「書くこと」を自分の味方につけましょう

- 苦手意識から，「書くこと」を敵に回さないで.
- 「書くこと」は，必ず自分を守ってくれる.
- 自作業の品質を示すのは「書くこと」.
- 「書くこと」は「考えること」だから.
- 「考えること」も「書くこと」に同期しようじゃないか.
- 「ソフト開発」は「ドキュメンテーション」.

＜参考文献＞
実践的なテクニカルライティング参考書

[1]阿部圭一「明文術 —伝わる日本語の書きかた」NTT出版株式会社, 2006.

演習付

情報処理学会誌の書評にて紹介された．著者の研究分野は情報教育ほか（日本社会情報学会(JSIS)会長）．日本語文章の書き方に関する本であるが，本書内容にはソフトウェアドキュメンテーションの考え方に一致する点が多い．

[お薦めの理由]日本語ライティングのポイントと，それがソフトウェア開発にも共通していることを理解する上で参考となる．文書の組み立てや論理展開にも触れ，演習問題もあるので，総合的な参考書となる．

[2]一般財団法人テクニカルコミュニケーター協会 編著「日本語スタイルガイド」一般財団法人テクニカルコミュニケーター協会, 2009.

TC技術検定例題付

テクニカルコミュニケーター（TC）協会は，主にマニュアル作成を仕事とする人たちが集まり技術交流を行う団体である．テクニカルコミュニケーターの知識や技術の到達度を全国共通の基準で検定する「テクニカルコミュニケーション（TC）技術検定試験」を実施している．本書は，協会が扱う知識や技術を基に，日本語作文技術の要点としてまとめたもの．

[お薦めの理由]日本語ライティングの基本を押さえるためのガイドブックとして参考になる．

[3]日本情報システム・ユーザー協会/編, 福田 修/著.「SEを極める 仕事に役立つ文章作成技術」日経BP社, 2005.

演習付

日本情報システム・ユーザー協会(JUAS)内のソフトウエア文章化プロジェクトの2005年活動成果を基に，まとめられたものである．ソフトウェア開発上の文章の重要性や必要性などソフトウェア開発と文章作成との関連性を，開発者ではない人にも理解できるように説いている．内閣訓令の原則に従って編集された，付録の「現代仮名遣い」と「送り仮名」は参考になる．

[お薦めの理由]ソフトウェア開発と文章作成の関係を，一般的な概要として簡単に理解するために参考になる．

＜参考文献＞
開発者のみなさんと共有する“よすが”記事

[4]ドキュメンテーション技術の連載記事「理系のための文書作成術」
CQ出版 電子・組み込み技術の総合サイト「組み込みネット」の技術解説
（2010/8～2011/6）
http://www.kumikomi.net/archives/kumikomi/technology/

第1回　開発文書を分かりやすく記述する
第2回　図表を表現手段として活用する
第3回　開発文書の書き方はしごとのやり方を示す
第4回　自分の「赤ペン先生」を持とう
第5回　設計レビューでするべきこと，してはいけないこと
第6回　仕事で文書を「書かされている」あなたへのメッセージ

書くことは考えること

- 毎回，頭の中が整理され，あらたな根拠や拠りどころを発見しました．
- 仕事の仲間に赤ペンを入れてもらうことで，自分の考えのあいまいな点を発見しました．

坂本 佳史（日本IBM）
―開発の現場から.
それから自然言語処理技術の活用

開発文書にまつわる思い出

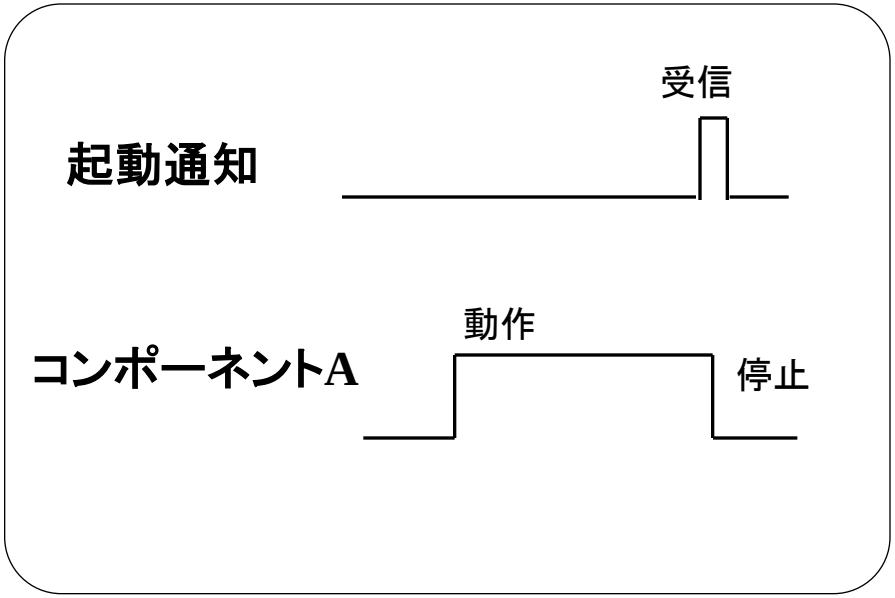
システムLSIの要求仕様書に,

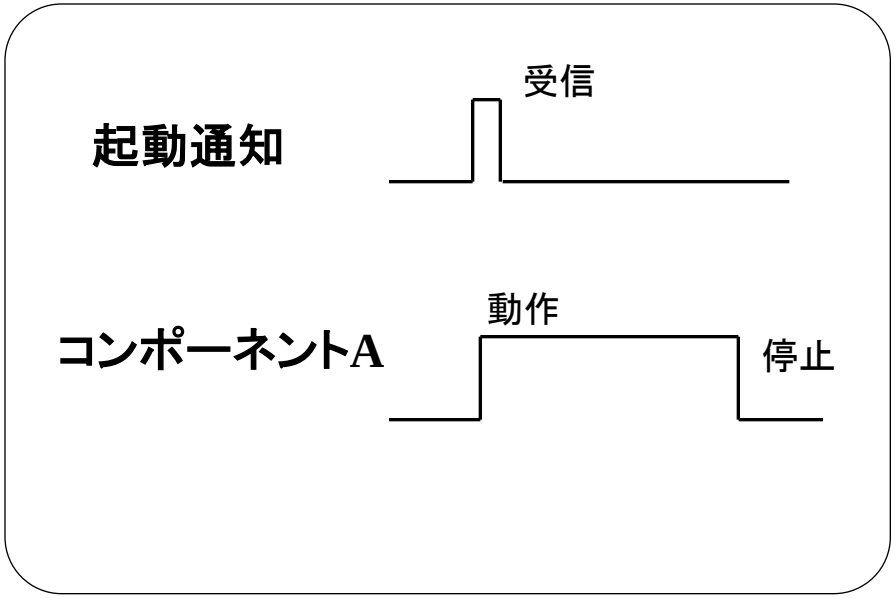
「その他のエラーは全てハードウェアでハンドリングすること」

の記載が有った.
この一文においてお客様と開発エンジニアの理解が異なったことから大きな問題に.

問題のある文書

「コンポーネントAは、
起動通知を受信するまで動作しない」





一意に判断できない

良い文書と工夫

良い文書

“開発文書を作成する開発メンバーが、

読み手に誤解を与えず正確に情報を伝えられる”

工夫

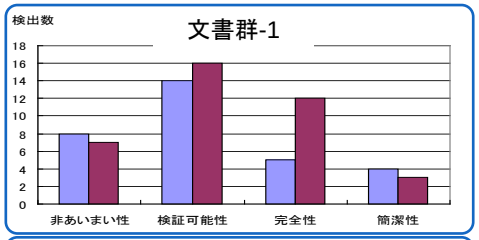
1.あいまいさが無く、一意の表現で記述する
2.開発文書に求められる情報をもれなく記述する
3.正しい文法で記述する
4.簡潔な表現で記述する

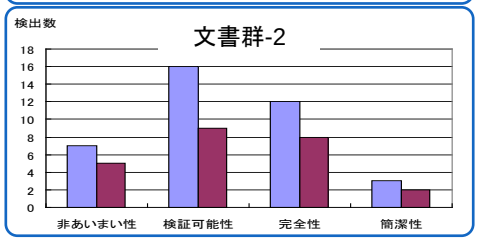
自然言語処理技術と文書品質の確認

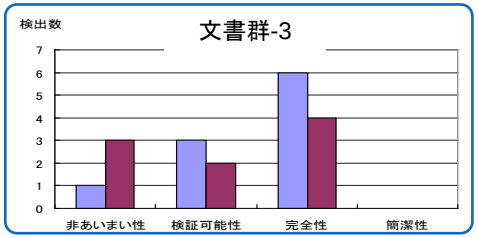
オフィスPCで高度な形態素解析や文型解析が可能

非あいまい性 – 内容が一意に解釈できること
検証可能性 – “うまく機能する”などの検証できない記述がないこと
完全性 – 正しい文法を用いて必要な情報を記述していること
追跡可能性 – 文書内外の必要な関連情報を参照できること

文書品質の定量化を実現







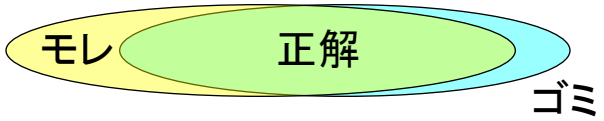
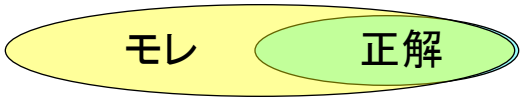
自然言語処理と目視の検出傾向に強い相関

自然言語処理による検出
目視 による検出

相関係数 = 0.8

自然言語処理で悪い文書に分類されると
目視検査でも悪い文書と判断する傾向が強い

自然言語処理 vs. 目視検査

手段 / 指標	本手法(DCT)	目視
検出結果		
1. ゴミの少なさの指標 適合率 *p*	0.93	1
2. モレの少なさの指標 再現率 *r*	0.84	0.42
3. 総合的な指標 F値 *f*	0.88	0.59
4. 有効検出数 総検出数 * *p*	908	380
5. 作業時間	8分	480分

自然言語処理技術は実用段階

まとめ

開発文書の品質を向上することにより製品開発の品質を高めるためには、正しく相互理解された用語を、正しい日本語表現で使用することが大切です

読み手に誤解を与えない文章を記述することが重要

参考文献

•明文術	阿部 圭一
•要求を仕様化する技術　表現する技術	清水吉男
•ソフトウェアドキュメンテーション	デンソークリエイト
•要求仕様定義ガイドライン	JUAS
•要求工学	大西淳・郷健太郎
•ソフトウェアの仕様化と設計	花田 収悦
•SEのための「構造化」文書作成の技術	佐藤健
•エンジニアマインド Vol.9	技術評論社

杉本 明加（富士設備工業）
ープログラマの視点から

問題のある文書

- 中身のない文書
- 文書を書くことに追われて書いた文書
  - 例）機能安全プロセスの研究プロジェクトにて
    - とりあえずテンプレートを埋める
    - 中身は最初にあまり考えずに書いた人のコピペ
  - 認証が通ったとして，それでよいのか？
- きちんと書く気がない，必要ないなら書かなくていい！
  - 乱暴かもしれないが，中途半端に書かれているくらいなら無いほうがよい
    - 無いならソースコードが正しいと分かる
    - 中途半端だと何が正しい情報か判断できない

自分にとっての良い文書

- μITRON4.0仕様書
- TOPPERS統合仕様書
  - 正確に，厳密に，丁寧な構成で記述されている
    - 背景，ポリシー，仕様本体，補足や参考情報が整理されていてどこに何が書いてあるかわかりやすい
    - 細かい部分もほぼ漏れがなく，未定義や実装依存もはっきりと書かれている
  - 右にごく一部を抜粋
  - 継続的にメンテナンスされている（μITRONの歴史から考えると10年以上になる）
  - 一方，いくつか難点もある
    - TOPPERS統合仕様書はプレーンテキストなので人に依っては読みづらい
    - 正確で厳密な反面，記述量が多いため全体を把握するのは大変

<TOPPERS統合仕様書>
1.1 TOPPERS新世代カーネル仕様の位置付け

TOPPERSプロジェクトでは，2000年に公開したTOPPERS/JSPカーネルを始めとして，μITRON4.0仕様およびその保護機能拡張（μITRON4.0/PX仕様）に準拠したリアルタイムカーネルを開発してきた．

μITRON4.0仕様は1999年に，μITRON4.0/PX仕様は2002年に公表されたが，それ以降現在までの間に，大きな仕様改訂は実施されていない．その間に，組込みシステムおよびソフトウェアのますますの大規模化・複雑化，これまで以上に高い信頼性・安全性に対する要求，小さい消費エネルギー下での高い性能要求など，組込みシステム開発を取り巻く状況は刻々変化している．リアルタイムカーネルに対しても，マルチプロセッサへの対応，発展的な保護機能のサポート，機能安全対応，省エネルギー制御機能のサポートなど，新しい要求が生じている．

TOPPERSプロジェクトでは，リアルタイムカーネルに対するこのような新しい要求に対応するために，μITRON4.0仕様を発展させる形で，TOPPERS新世代カーネル仕様を策定することになった．

ただし，ITRON仕様が，各社が開発するリアルタイムカーネルを標準化することを目的に，リアルタイムカーネルの「標準仕様」を規定することを目指しているのに対して，TOPPERS新世代カーネル仕様は，TOPPERSプロジェクトにおいて開発している一連のリアルタイムカーネルの「実装仕様」を記述するものであり，ITRON仕様とは異なる目的・位置付けを持つものである．

まとめ

- 文書を書くことを目的としてはいけない
- あくまで良い製品，良いサービスを提供するために開発文書を作成することが重要
- 自分が作っているものを使う人のために，
  - 私は文書を書きます
  - 良い文書とは何かを考え続けます

森川 聡久（ヴィッツ）
ー機能安全と第三者評価

開発文書にまつわる思い出

~機能安全観点から~

- 場面：機能安全認証機関に、プロセス規定の審査を受けた時
- 指摘事項：「・・など」という曖昧な表現はNG
- 理由：曖昧な表現があると、正しい判断ができなくなる。
  → ここにリスクがある!!

事例）必要とするスキル定義の抜粋

第一階層		第二階層		説明
1	プラットフォーム	1	MPU	I/O 制御、AD 入力、DA 出力、タイマ制御、PWM 制御、割り込み制御、プロセッサモード、システムタイマなど
		2	ソフトウェア	リアルタイム処理、リアルタイムカーネル、システムコール、割り込み処理、デバイスドライバ、ミドルウェア、マルチタスク処理、例外処理など

細かなことに気を配ることは、安全対応上必須

問題のある文書
~機能安全観点から~

問題文書（例：RTOS安全マニュアル）

5.1 タスクが実行すべき時に実行されない

When it is necessary to execute the task, it is not executed.

タスク実行すべきタイミングで実行されない場合の故障検出について。

対応方法：

故障検出ライブラリの実行シーケンスモニタ機能により、実行すべきタスクが実行されずに他のステップが実行されることをチェックすることで故障を検出する。

改善

改善後の文書（例：RTOS安全マニュアル）

5.1 タスクが実行すべき時に実行されない

When it is necessary to execute the task, it is not executed.

タスク実行すべきタイミングで実行されない場合の故障検出について。

対応方法：

故障検出ライブラリの実行シーケンスモニタ機能により、実行すべきタスクが実行されずに他のステップが実行されることをチェックすることで故障を検出する。

対応例：

タスク A から act_tsk によりタスク B を起動させようとした時に、タスク B が実行されない場合について、下記図 4.1 のように実行する API の実施前、実施後に実行される順番にチェックポイントを連番で設定する。シーケンス番号については、チェックする一連の固まりでまとめるために設定する。

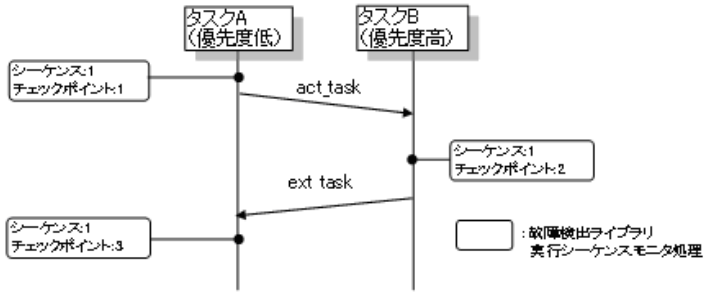


図 5.1 タスクが実行すべき時に実行されない場合

実行時にチェックポイント 1→2 の順で処理される場合は正常だが、何らかの要因によりタスク B が実行されない時にチェックポイント 1→3 の順で実行される場合に故障として検出する。

そのため、チェックポイント 3 が実行される時、その一つ前の実行チェックポイントがチェックポイント 2 ではない場合にチェックポイント 3 処理の戻り値として通知される。

改善内容
- 具体例 を追加
- 明確な図を追加

第3者評価しやすい文書

人によって解釈が異なる。

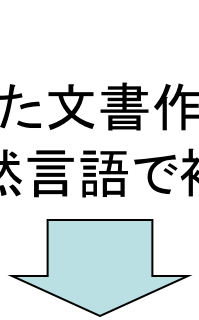
⇒ 問題：
- 開発時に不具合を混入する可能性あり。
- 将来の保守が困難。

⇒ 対策：「具体例」、視覚的な「図」を併用

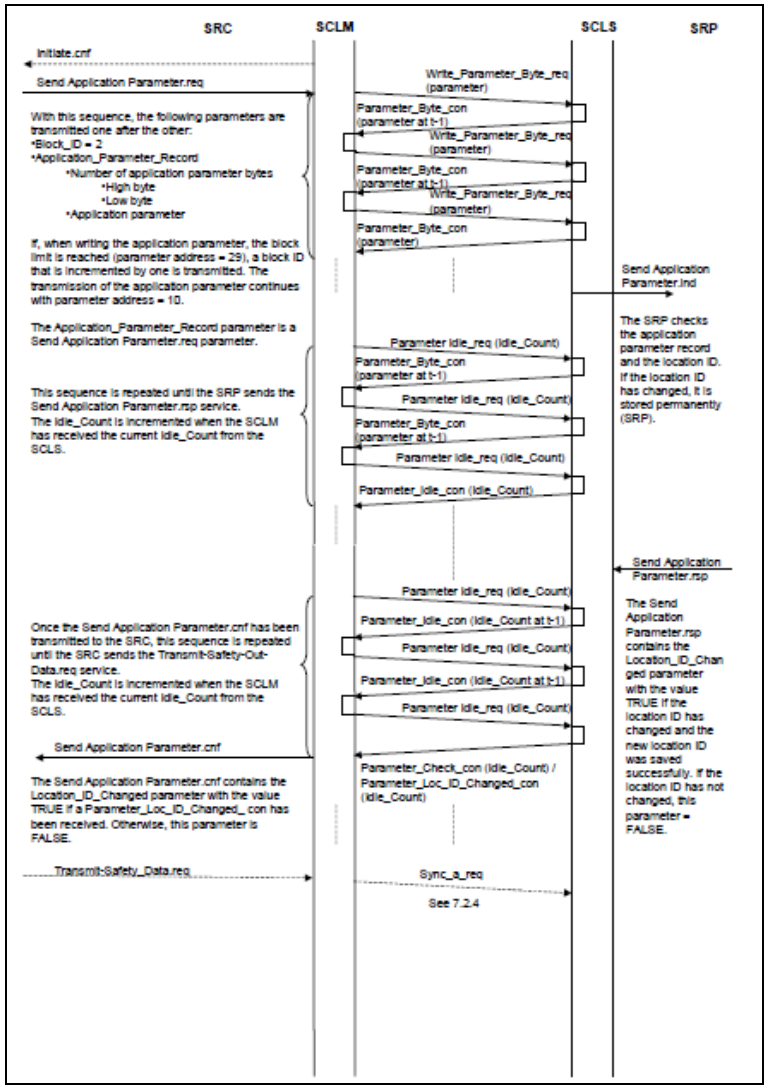
＜森川の考え＞
- 日本語を改善することも重要だけど・・・
- 例や図を追加するほうが、手っ取り早い改善策

良い文書と工夫例1
~機能安全観点から~

- 図・例の追記策の課題：
  - 文書量の増加
    - 書く方も読む方も大変!!
  - 図と説明文の対応を取るのが大変
- 対策：
  - 図を中心とした文書作成。
  - 図の中に自然言語で補足説明を記載



- 新たな課題：
  - 図の構成管理をどうするか。
  - 文法を本当に一意に定義できるか。
  - 過不足、矛盾が無いことをどう確認できるか。

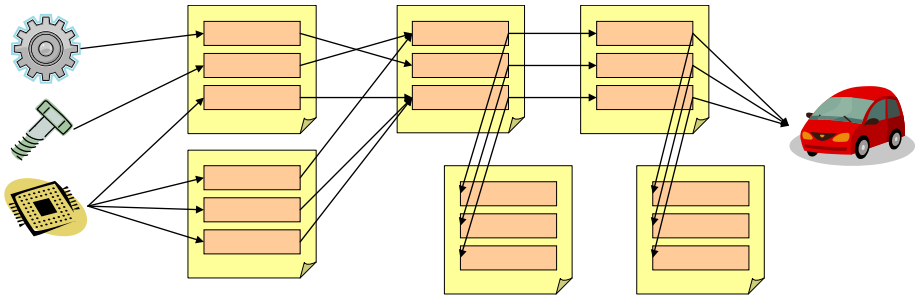


IEC 61784-3-6 より抜粋
アプリ間の通信プロトコルを定義

良い文書と工夫例2(1/2)

~機能安全観点から~

- 機能安全要件:
  - 要求事項のトレーサビリティ管理
    - 各要求項目、設計項目、テスト項目、関数などの相互の関連を管理する。
- 目的:
  - **各項目の過不足・矛盾が無いことの確認**
  - 変更時の影響分析
- 対応方法:
  - 「要求事項」と「補足説明」を区別し、要求事項のみをトレーサビリティ管理する
    - 開発文書には「要求事項」と「補足説明」が混在している
- 課題:
  - 何を「要求事項」として管理するかが難しい。
  - 混在していると、要求事項の完全性が確認しづらい。



TOPPERS「次世代車載システム向けRTOS仕様案」より抜粋

2.2 保護機能

2.2.1 【REQ004】保護機能における機能レベルの設定

AUTOSAR OS 仕様の保護機能に対して機能レベルを設定することで，オーバヘッドの小さな実装を可能にすること．

要求事項

要求の理由

AUTOSAR OS 仕様の各種の保護機能関して，オーバヘッドの大幅な増加が見込まれるという懸念がある．アプリケーションプログラムからの機能要求とハードウェア性能との兼ね合いから，規定の保護機能をサブセット化してオーバヘッドを軽減するような実装を可能とする．

補足説明

2.2.2 【REQ005】メモリ保護機能における実装依存規定の排除

AUTOSAR OS 仕様のメモリ保護機能に関する実装依存規定を減らすこと．

要求事項

要求の理由

メモリ保護を実現するハードウェアの持つ機能が多種多様であることから，AUTOSAR OS 仕様ではメモリ保護機能として満たすべき機能規定の多くがオプショナル（“The Operating System may …”という記述）となっている．これらを OS に搭載すべきか否かを明確に規定することで異なる OS 実装の間での搭載機能の違いが生ずることを防ぐ．

補足説明

良い文書と工夫例2（2/2）

~機能安全観点から~

「要求事項」と「補足説明」の判断の難しい例
（OSの仕様書より）

1. カーネルに登録したタスクは、実行できる状態、休止状態、待ち状態のいずれかの状態を取る。 ⇒ **3状態が機能的に必要**
   ⇒ **要求事項**

2. また、実行できる状態と待ち状態を総称して、起動された状態と呼ぶ。 ⇒ **状態の名称の説明**
   ⇒ **補足説明**

3. タスクをカーネルに登録していない仮想的な状態を、未登録状態と呼ぶが、 ⇒ **状態の名称の説明**
   ⇒ **補足説明**

4. 本OSでは、タスクの未登録状態は存在しない。 ⇒ **存在しないという要求と捉えるか、存在しないので対策不要（項目1のみでOK）か？**
   ⇒ **？？？**

厳密な判断をしようとすると
結構大変!!

まとめ

- 機能安全／IV&V対応でも、開発文書品質は重要!!
  - 完全性、可読性、再現性、客観性

- 特別な要求はない。
- しかし、突き詰めると意外と難しい・・・。
  - 開発文書の書き方に工夫が必要
  - 課題も多い

- 開発文書の書き方の“決定版”を確立していきたい。（ASDoQの活動にて）

清水 吉男（システムクリエイツ）
―「派生開発」と USDM

XDDP: eXtreme Derivative Development Process
USDM: Universal Specification Describing Manner
PFD: Process Flow Diagram

System Creates

開発文書にまつわる思い出

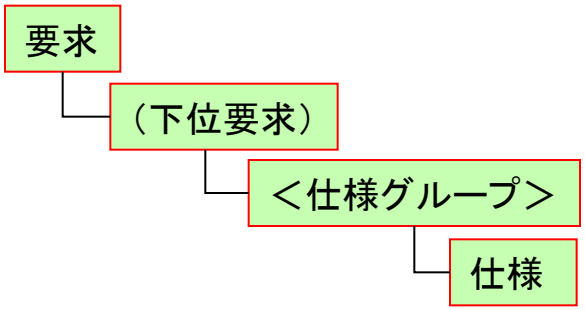
- 最後の開発（集大成）＝ 1994年～95年：２つの新規開発を重ねた
  - A） 初代カラリオの開発
  - B） 内線電話のデジタル化転送装置
- 開発方法
  - A）＝構造化手法
    - 構造化分析（ハトレー／ピルバイ法の改良）～構造化設計のシームレス接続
    - 当初・・・仮想の製品仕様で着手．途中で正式の製品仕様に切り換える
  - B）＝文書を階層化する方法
    1. システム設計書
       2. データ構造設計書
    3. タスク仕様書＋タスク設計書＋タスク内の構造図（SC）＋関数仕様／設計書
    - 顧客側で新規開発の設計書レビューができない
      - ②の文書をタスクごとに「１つ」にまとめた
      - 「章」や「節」ごとに並行して記述し、調整に伴う手戻りを最小化
      - 要求仕様と設計に関するミス・・・なし

良い文書と工夫(1)

「仕様」は「要求」に含まれる「動詞」およびその「目的語」にある

- 「要求」の役割・・・すべての「動詞」を表現することで仕様の抽出を促す

- 階層表現で仕様を漏れにくくする（「USDM」の特徴）



要求	DA07	計測データを受信し、平均値を算出しながらリアルタイムに表示する. ただし異常値が検出されたときは警告を表示する

- この要求の「赤字」の部分について仕様を展開すればよい

- コンサルティングでの事例
  - 約2600仕様に対して、ベースライン設定後の仕様変更＝13項目（0.5%）
    - すべて記述されている項目に対する変更 ＝ 実質的に仕様の抽出漏れなし

良い文書と工夫(2)

- 「要求仕様書」と「機能仕様書」の違いを明確化

① 保守性の仕様の表現がスムースに

機能仕様の姿勢では
書けない

要求	QUA20	［保守性］以後のバージョンアップの変更に対して崩れにくいように作って欲しい	
	理由	１０年間はベースのソースを劣化させることなく維持したい	
	□ □ □	QUA.02-1	モジュールの複雑度は１７以下を原則とし、越える場合はPLの承認をとること
	□ □ □	QUA.02-2	“手順的凝集度”以下になる場合は事前にPLの承認をとること
	□ □ □	QUA.02-3	処理と管理は明確に分離し、起点関数からの呼び出しの深さは５以内とする．部分的にこれを越える場合は事前にPLの承認をとること
	□ □ □	QUA.02-4	タスク間でアクセスし合うグルーバル・データを作らない．グローバル・データとなることが避けられない時は、事前にPLの承認をとること 【説明】割り込み処理との間で共有するケースはこの制限外とする.

② 「認定仕様」の導入へ

- 仕様化の必要性が低い要求に対して、「要求」の横に「仕様マーク」を付けることで仕様化作業をパスして設計にとりかかることを認める.
- ただし、機能仕様書にはこの記述は認められない

まとめ

- 「USDM」によって仕様漏れは大幅に改善できる・・・確認済み

しかしながら

「要求」の表現によっては・・・

→ 動詞が隠れてしまって、仕様を抽出できないことがある

「仕様」は抽出できているが・・・

→ その表現によっては読み手に真意が伝わらないことがある

- 「USDM」で表現しても・・・・その表現のしかたに課題が残っている

以上

ありがとうございました!
ASDoQで再会しましょう!!