要求仕様記述手法「USDM」ってどんなの？

～明日から使えるUSDMのエッセンス～

USDM (Universal Specification Describing Manner)

派生開発推進協議会（AFFORDD）
USDM勉強会資料 ／短編 1

派生開発推進協議会
代表　清 水　吉 男
URL=http://affordd.jp

株式会社 システムクリエイツ 代表取締役
URL=http://homepage3.nifty.com/koha_hp
http://kohablog.cocolog-nifty.com/
shimz@nifty.com

agenda

- 如何なる要件分析技術も、「仕様」として適切に表現する技術がなければ、効果を発揮しない.
- 本書は「**USDM**」に則った要求の仕様化技術を紹介するものです。

1. 仕様の問題って？
2. USDMの特徴
3. まず要求を表現しよう
4. 次に要求を仕様化しましょう
5. 画面仕様も同じように書けるよ
6. 品質要求を表現するコツ
7. 演習ーちょっとUSDMで書いてみよう
8. ところで、派生開発で威力を発揮するUSDMって？

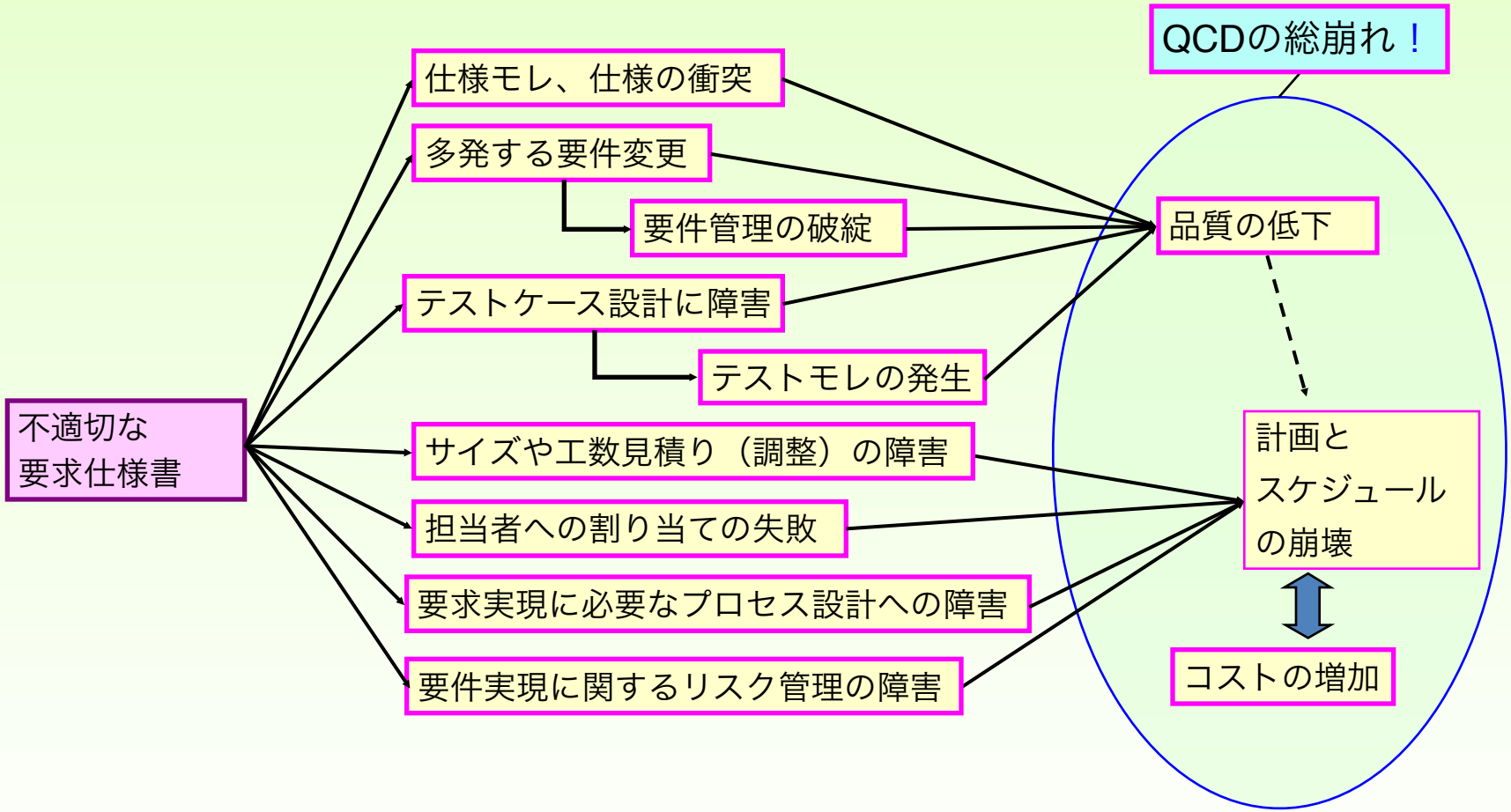
＊:「派生開発 (XDDP)」･･･「保守」の一種で、機能追加を含むすべての変更を効果的に扱う開発アプローチ
（XDDP = eXtreme Derivative Development Process の略）

1. 仕様の問題って？

- 不適切な仕様化によってもたらされるトラブルは、現実のプロジェクトの中で、いろいろな表情を見せる
- ソフトウェア開発の歴史は “仕様モレ” “仕様トラブル” との戦いでもある

大部分は仕様の問題ですね

- トラブルの多くは、不適切な要求仕様書を根源に持つ



仕様の問題には２つのタイプがある

- それぞれ原因が異なるので、対処方法も異なる

タイプ	状況	原因	対応
仕様モレ	必要な仕様が書かれてない	構造の問題	USDMで解決できる
曖昧表現	表現が悪く、読み手に正しく伝わらない	適切な表現方法を知らない	文章の表現力を習得する必要

- USDMの階層表現によって、助詞や接続詞が使われる機会が減ることで、曖昧さの問題はいくらか緩和される
- でも曖昧表現はUSDMでは根本的には解決されない

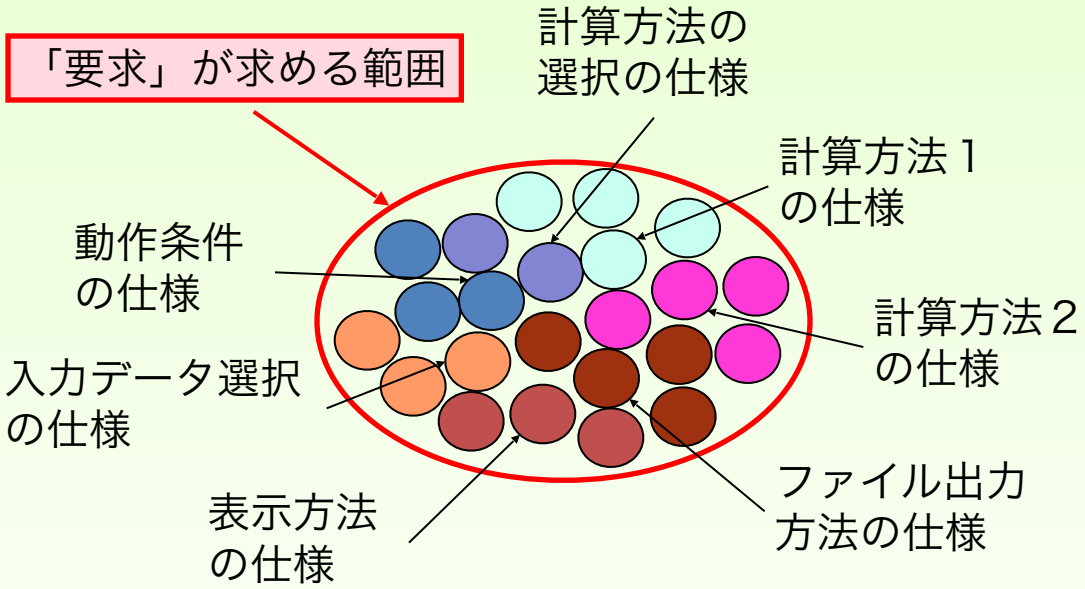
できて欲しいことの「範囲」が見えない

- その機能の中心的な仕様は誰でも拾えるが、“それも必要だったの？”という類いの仕様が漏れることが多い

機能名

仕様

- タイマー機能
  - このボタンを押すとタイマーが起動する
  - １回押ごとに１分加算する
  - 分の表示は５９までとする
  - ただし、蓋が開いているときは押せない



“要求”が表現されていないために、その機能を満たすべく“仕様”が拾いきれない

品質要求が課されていますか？

- 「機能」を満たしていても・・・・・

操作に対する応答が遅い

入力BOXの位置が悪く操作が疲れる

エラー時の応答に統一性がなく混乱する

- 品質要求を実現するための**スキルが必要**
  - 機能を実現する設計スキルとは必ずしも同じではない

その結果、5000行や２万行の関数作ってしまう

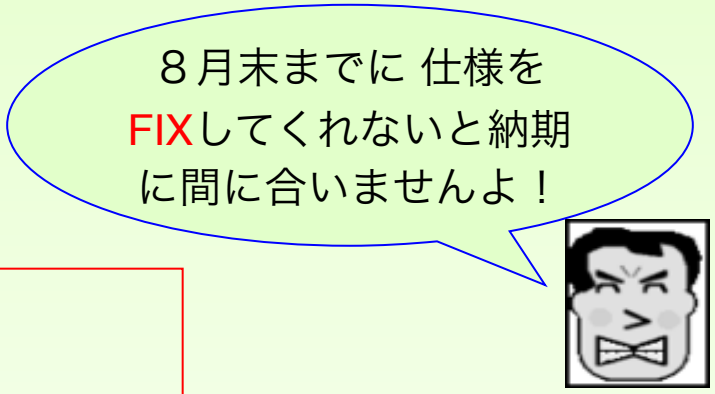
- 機能要求を実現することしか**求めてこなかった**ために、「保守性」などの品質を実現するためのスキルが身に付いていない

その後の「派生開発」で混乱する

“FIX”を強要していませんか？

- “FIX”を要求する背景には、もともと好ましくない状況がある
  - そこに書かれている要求仕様書では“合意”が形成できる状況にはない

“仕様”がよく見えない状態 → 要求仕様書



- 何を確認すれば良いのかわからない
- 何が不足しているのか分からない
- レビューを繰り返してもほとんど進展しない

「要求仕様書」の構成が邪魔をしていることが多い

↓

後で（動くようになってから）必ず仕様変更が発せられる

“FIX”は仕様変更の予約語である

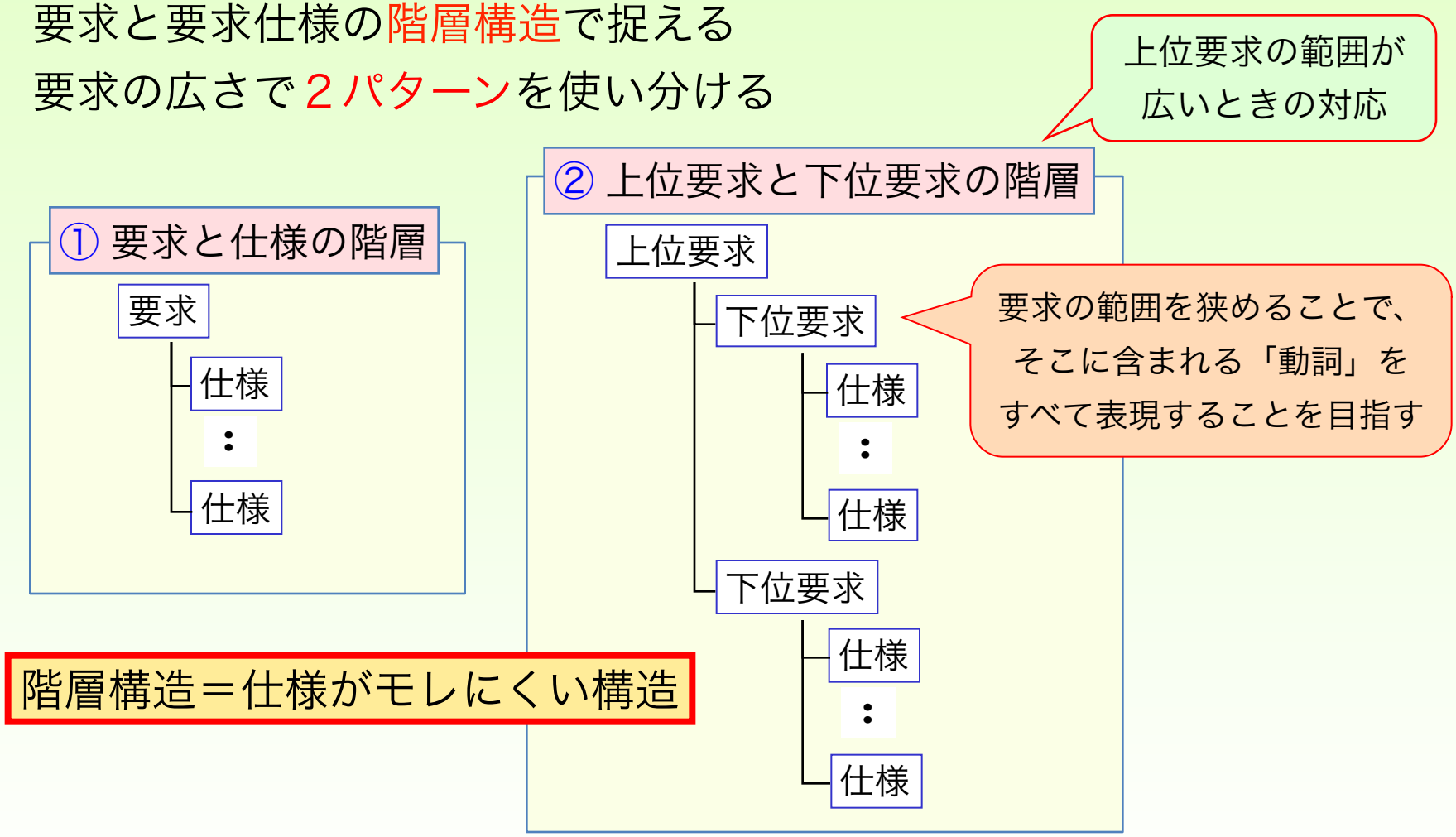
設計者自身が仕様変更を呼び込んでいることも少なくない

2. USDMの特徴

- ここでは、USDM（派生開発の変更を含む要求仕様の表現方法）の特徴について説明します

要求と仕様を階層構造で捉える

- USDMでは、「範囲」や「階層化」という考え方を持ち込む
- 要求と要求仕様の階層構造で捉える
- 要求の広さで２パターンを使い分ける



「仕様」は要求の中の「動詞」にあるという考え

“仕様”は、要求の中の「動詞」および「目的語」に存在する

USDMの基本的な考え方

- 「動詞」に対して＜仕様グループ＞を立てる
- 仕様は、動詞を「プログラムコード」に変換するための記述

この枠が要求の「範囲」

要求

振る舞い＝
イベントに始まる
一連の動き

イベント

・を・・して（動詞） → ・を・・して（動詞） → ・を・・して（動詞） → ・を・・して（動詞） → ・を・・する（動詞）

＜仕様グループ＞
・・・・・
・・・・・
・・・・・
仕様

＜仕様グループ＞
・・・・・
・・・・・
・・・・・
仕様

個々の「する（動詞）」に対して具体的にどのような処理を(する)べきかを記述する

USDMのこの構造が仕様モレを軽減する

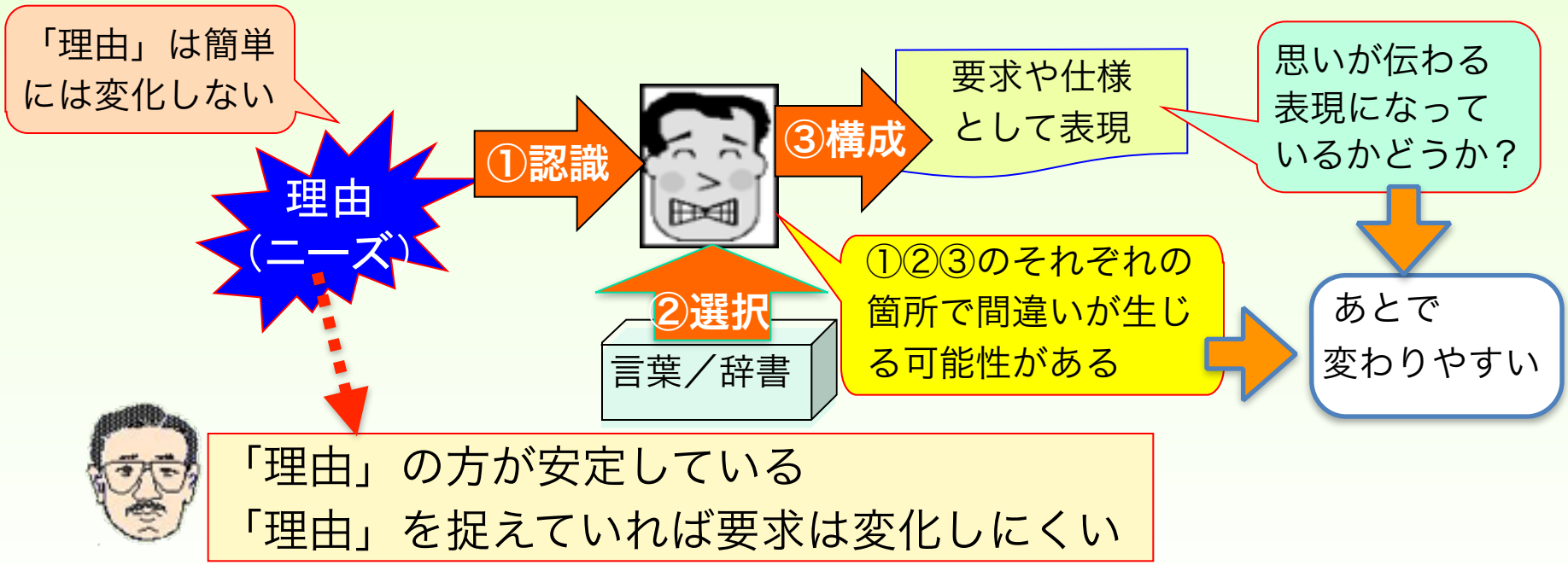
ソースコード
仕様をコンピュータに処理させるためにプログラムコードに変換したもの

「理由」を付けて要求を支える

- 人は、自分の思いを相手に伝えることができるとは限らない

この状態を解消する方法として、USDMでは要求には必ず「理由」を付ける

- 「理由」があることで、「要求」の意味を理解しやすい
- 「理由」を書くことで、その要求に根拠がないことに気付くこともある
- 「理由」によって仕様の引き出しや設計方法にも配慮できる



USDMにおける要求仕様書の定義

- 「USDM」では要求仕様書を“作るための文書”と考える
  - 作業者に対して「作って欲しいこと」が書かれた文書
  - 一般の「要件定義書」に近い

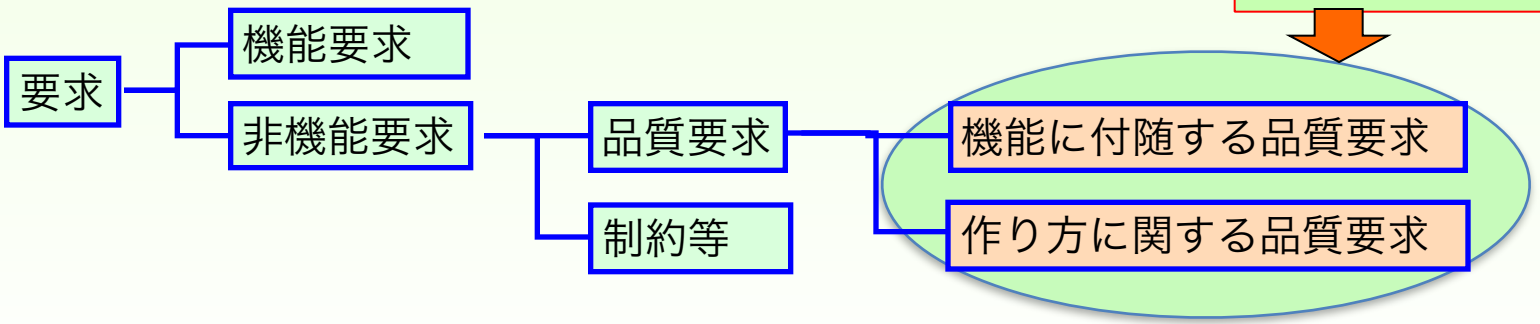
USDMにおける定義

「USDM」では、要求仕様書とは、今回のプロジェクトで実現して欲しいこと（Requirements）について、“作ることの関係者”が実現内容についての認識を特定（Specify）できている文書

「USDM」では「Specification」とは関係者が同じものをイメージできる状態のことと捉える

- 要求仕様書では機能要求と非機能要求を扱う

USDMでの分類



要求仕様書と機能仕様書の使い分け

- 「USDM」では、「要求仕様書」と「機能仕様書」を使い分ける
  - 要求仕様書・・・今回“作るため”の文書
  - 機能仕様書・・・現状の製品やシステムの機能を説明している文書

現実には「機能仕様書」の名称にこだわらない

	要求仕様書	機能仕様書（等）
目的	作るためのもの	機能を説明するもの
関係者	計画書で特定されている	特定されていない
表現	関係者がSpecifyできる状態	一般的な記述
納期やコスト	背後に背負っている	意識されない
バージョン管理	今回だけの文書	ソースコードと対でバージョンアップする

3. まず要求を表現しよう

- 多くの問題は、「要求」が適切に表現されていないことに起因している
  - 「機能名」の下に、いきなり「仕様」が書かれている

ここでは、「ＵＳＤM」の表記を Excel を使って説明しますが、要求や要求仕様の表記そのものは Word でもできます。
ただし、要求仕様に対して設計時にTM(Traceability Matrix）などの情報を付加するには Word では制約が多い。

要求とは何か？

- システムに求められる機能や性能、作り方に対する品質など、実現したいことを「要求」として抽象的に表現したもの

作業者に対する要求

- システムや製品に求められる振る舞いとしての機能
- パフォーマンスなどの機能に付随する品質
- 保守性などの作り方に対する品質

機能仕様の表現でも、作業者の方で、“・ように作って欲しい”と「要求」の形に読み替えてくれる

実現したいゴール

要求	MAL01	受信した電子メールをキーワードで検索してメーラーで再利用する
要求	BYY03	ネット販売の購入の最終確認操作では誤操作が入る余地を排除する
要求	FLE10	今日の売れ筋商品のトップ１０について１週間の販売動向を同時に表示する

要求の引き出しから仕様化へのステップ

- 要求を発見する研究は世界中で進められている
  - 顧客の求めるものを整理し、そこから「要求」の形にまとめる
  - 『実践ソフトウェアエンジニアリング』（ロジャー・プレスマン著）では、要求エンジニアリングを以下のステップで説明している

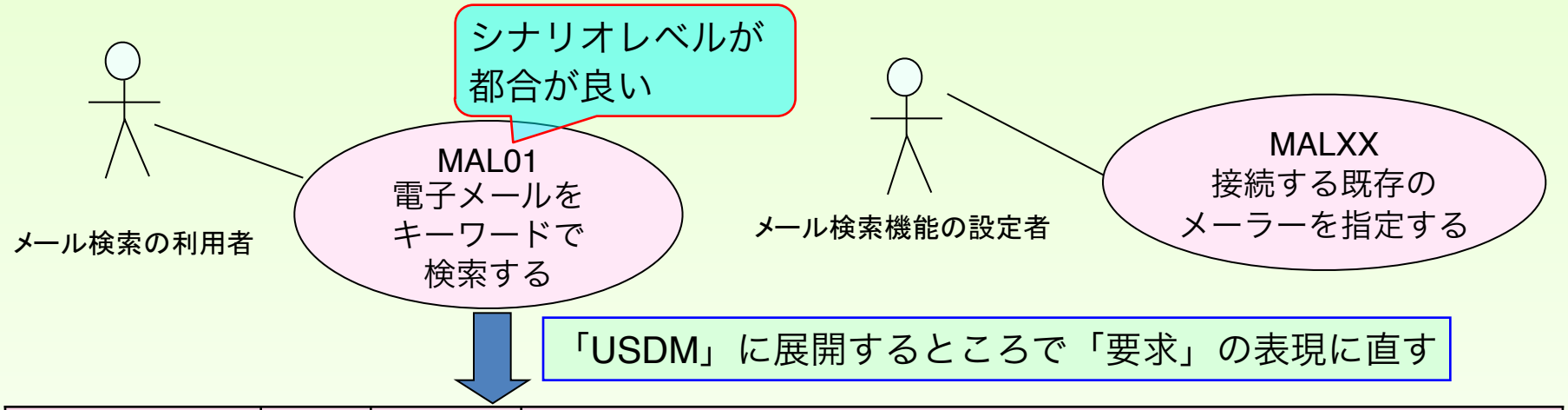
①	開始	ニーズや新しいサービスの必要性が生じたとき
②	要求獲得	ここで要求を発見する
③	推敲	獲得した要求を表現しながら調整し洗練する
④	交渉	要求間の調整、組織内部での要求の調整など
⑤	仕様化	要求が安定したところで一気に仕様化する
⑥	検証	レビューを繰り返して精度を高める → ベースライン設定へ
⑦	要求マネージメン	その後の変更管理

繰り返す

- 上記の②～④の中で要求が引き出される
- 一般には、⑤の段階で別の文書が作られるが、「USDM」では一体で扱う

ユースケースから要求を引き出す

- ユースケースと対応させる
  - ある程度のところまでユースケース（＝要求）で抽出を進めていき、途中でUSDMの表記に切り替える.
  - すべての要求をユースケースで拾いきれない



「USDM」に展開するところで「要求」の表現に直す

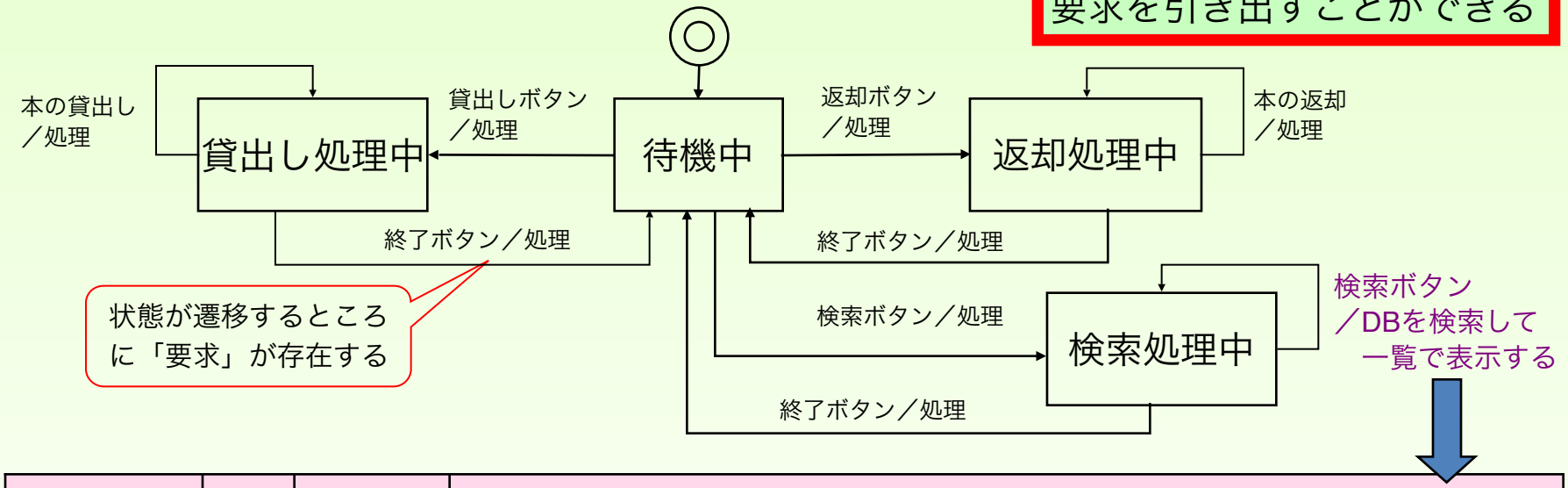
メール検索機能	要求	MAL01	受信および送信した電子メールをキーワードで検索して再利用したい
		理由	メールが多くて関連するメールを探せない
		説明	

要求の下に仕様を引き出す

“状態遷移”から要求を引き出す

- 遷移する際に「振る舞い」を持つので、「要求」を引き出しやすい
  - “イベント”を捉えて“振る舞い”を表現する

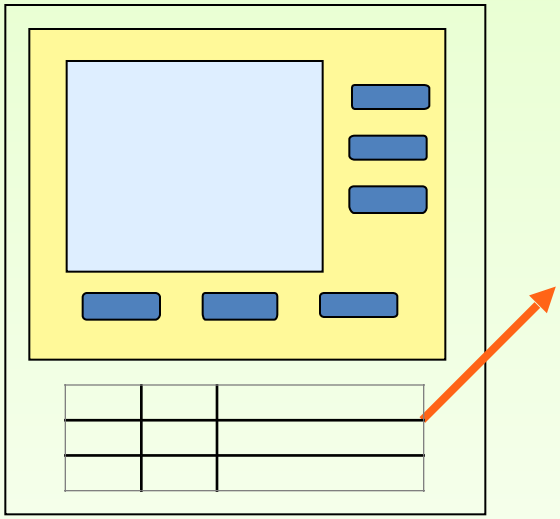
データの状態遷移からも
要求を引き出すことができる



検索ボタン	要求	BOK01	入力された書籍情報を元に書籍DBを検索し、発見した書籍を一覧で表示する
	要求	BOK02	新たな書籍情報を使って、一覧で表示された中から表示を絞り込む
貸出しボタン	要求	BOK03	入力した書籍情報と個人情報から当該書籍を当該個人に対する「貸出し中」の状態にす
返却ボタン	要求	BOK04	入力した書籍情報から書籍DB内の当該書籍を返却済みとし、個人データから抹消する

「操作画面の要素」から要求を引き出す

- 画面に配置された表示領域やボタンなどの要素も「振る舞い」を持つ



商品データ表示域	要求	DS.A.01	売れ筋商品のトップ２０をリアルタイムで表示する
		理由	タイムセールなど機敏な販売戦略を考えるため
天気予報ボタン	要求	DS.A.05	天気予報をネットから入手して、最初の２日分は時間単位で表示し、その後の５日分はそれぞれ１日単位で予報を表示する
		理由	商品の仕入れや陳列方法の判断に使う

一般の要求仕様書と同じパターンで表現できる

- 画面も「状態」の一種
- 状態遷移の考え方で「画面遷移」を作り、そこから要求を引き出すことができる

「範囲」の見せ方で見えるものが変わる

- 「範囲」の表現が不明確だと、その後の仕様の抽出でモレやすくなる

「A」と「B」を比べて、「範囲」の違いを感じてください

要求	PRT11	A	１週間の天気予報を表示する
		B	天気予報をネットから入手して、最初の２日分は時間単位で表示し、その後の５日分はそれぞれ１日単位で予報を表示する
要求	ALM03	A	予想との乖離が一定以上開いたときは通知する
		B	商品毎に設定されている当日の売り上げ数量の予測に対して、実売データとの間に大きな開きがあるときは警告メッセージをマネージャーのPC画面に出す

- 「動詞」が見えることで仕様を抽出しやすくなる

要求を洗練する

- 振る舞いの「範囲」をイメージできるように要求の表現を調整する
  - 関係者が同じ「範囲」をイメージできることが大事

要求	MAL01	受信および送信した電子メールをキーワードで検索して再利用したい

- 電子メールのファイルのすべてを対象にしない方が良さそうだ
- 再利用する時は、電子メールのソフトに直接接続できた方が良いだろう

- 実現したい「範囲」を表現することで、関係者の認識のズレを矯正する

要求	MAL01	事前に指定された受信および送信した電子メールをキーワードで検索し、選択した電子メールをメーラーに繋いで再利用したい

なるほど！
こうして表現されると、
どのような「仕様」が
必要か見えるね！

「理由」とセットで表現する

- 要求には、それが必要な「理由」（存在理由）や「背景」がある
  - その「要求」がないと困ることは？
  - その「要求」があることで恩恵を受けることは？

- 「目的」は扱わない
- 「理由」と重なるケースは「理由」として記述する

- 「理由」は「要求」の一部
  - USDMでは、意図的に欄を分けて表現する

要求	SAL01	商品毎に設定されている当日の売り上げ数量の予測に対して、実売データとの間に大きな開きがあるときは警告メッセージをマネージャーのPC画面に出す
	理由	すぐに原因を調べて、陳列方法の変更など適切な対応策を講じる必要がある

  - 「理由」によって要求に対する関係者の認識のズレを調整できる
  - 要求全体に対する理由の他に、特定の「動詞」に理由が存在することもある
  - 必要かつ有効な仕様を引き出せる
  - 設計上の工夫を引き出すこともある

範囲が広い要求は２層に分割・階層化する

- 一般には、上位要求の下に「時系列」で配置する
  - 隠れていた「動詞」を下位層の要求に表現する
  - 下位層の要求にも「理由」が必要

最上位の要求の表現が変わっていることに注意

要求	MAL01	電子メールのグループを指定して、そこからキーワードで検索してメーラー上で再利用した	
	理由	メールが多くて、関連するメールを 探すのに手間取る	
	要求	MAL01-01	表示された検索グループの中から一つを指定する
		理由	検索グループが複数あるから
	要求	MAL01-02	いくつかのキーワードの入力を受付け、それらを組み合わせて検索する
		理由	可能性のあるキーワード（複数）で探したい
	要求	MAL01-03	検索結果を表示し、見つかったときは「subject」などの情報を一覧で表示す
		理由	Subjectと日付などから目的のメールを探すことになる
	要求	MAL01-04	一覧の中から選択されたメールを開く
		理由	中身を見ないと分からないから
	要求	MAL01-05	一つのメールを開いた状態でメーラーに繋いで編集できるようにする
		理由	コピーの手間を省いて編集の操作に入りたい

動詞から仕様のグループを設定する

- 範囲が狭められた要求の動詞の単位で仕様の＜グループ＞を設定
  - 「目的語」も仕様を持つことがある
  - 要求の中で「動詞」に表現されていないケースもある
    - ＜表示領域の初期表示＞　など

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 その際、異常値が検出されたときは警告を表示する
	理由	異常が検出されたときの平均値を確認したい
	＜受信方法＞	
	＜平均値の算出方法＞	
	＜計測データの表示＞	
	＜異常値の判断＞	
	＜警告の表示＞	

要求にすべての「動詞」が表現されていれば、あとはこれらの＜仕様グループ＞の下に仕様を展開するだけ

4. 次に要求を仕様化しましょう

- 仕様化が曖昧なことが、多くの問題を引き起こしている
- 要求が、適切に表現されなければ仕様は引き出されない
- ソースコードに変換されるのは「仕様」である

仕様の条件

- 要求に含まれる“具体的”な処理や振る舞いを表現したもの
  - 仕様は「要求」から導出される
    - 全ての仕様は、いずれかの要求に属する
  - 実現可能性が見える（設計者の立場から）
    - 前提条件や処理や振る舞いのための「制約」なども含むこと
    - 具体的であるために“仕様間の矛盾”が見える
    - 設計の様子がイメージできることがある
  - 検証可能性が見える（評価担当者の立場から）
    - その仕様が実現していることを検証する方法が見える
    - 具体的であることで、関係者の間で違った意味に解釈されない
  - 品質要求も例外ではない
    - 品質要求にも、実現可能性と検証可能性は求められる

事前に設定されたグループの仕様を埋める

- 動詞の単位に設定された＜仕様グループ＞の下に仕様を埋める
  - 仕様の範囲が狭められているので、容易に仕様を抽出できる
  - 「生産性データ」が使える

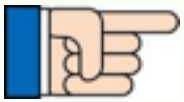
	要求	MAL01-03	検索結果とメールのSubjectを表示し、選択された内容を表示する
		理由	目的のメールが一つとは限らないので、絞り込めるような操作がしたい
		＜検索結果の表示＞	
	□ □ □	MAL01-03-1	検索されたメールの件数を一覧の上に表示する
	□ □ □	MAL01-03-2	該当するメールが存在しないときは「該当無し」を表示する
		＜検索メールの表示＞	
	□ □ □	MAL01-03-5	検索されたメールの「Subject」を一覧で見せる
	□ □ □	MAL01-03-6	検索されたメールに連続番号をつけて表示する
	□ □ □	MAL01-03-7	検索されたメールの件数が１０件を超えるときはスクロールバーを見せる
		＜メールの中身の表示＞	
	□ □ □	MAL01-03-10	一覧から１つのメールを選んで、その内容を見ることができる
	□ □ □	MAL01-03-11	開示されたメールの中で検索キーワードと一致している文字列を赤色で表示する

USDMでは「Specify」という考え方を重視する

- 要求仕様書では「Specify」という考え方が役に立つ
- 事例
  - A社への発注➡途中で質問事項のやり取りあり➡予定通りに完成
  - B社への発注➡途中で質問事項の交換なし➡途中で頓挫➡PJを中止
- 原因
  - A社とB社に開発能力の差があることは事前に把握していた（コスト差）
  - B社の担当者全員が今回の製品のドメイン知識が不十分で、A社と同じ書き方では「Specify」できない状態だった

タイトルは「要求仕様書」でも、
中身は「機能仕様書」だった

要求仕様書は今回の関係者が「Specify」できたことが問われる文書



関係者の能力の違いによって書き方を変えるべきという認識がなかった

[認定仕様]の考え方

- USDMでは、特殊なケースとして「要求」の状態であっても、その状態で実装可能とするケースを「認定仕様」として容認する
  - 「要求」や<仕様グループ>単位で対応する
  - 「Specify」の考えを活用したもので、「要求仕様書」だけに適用できる

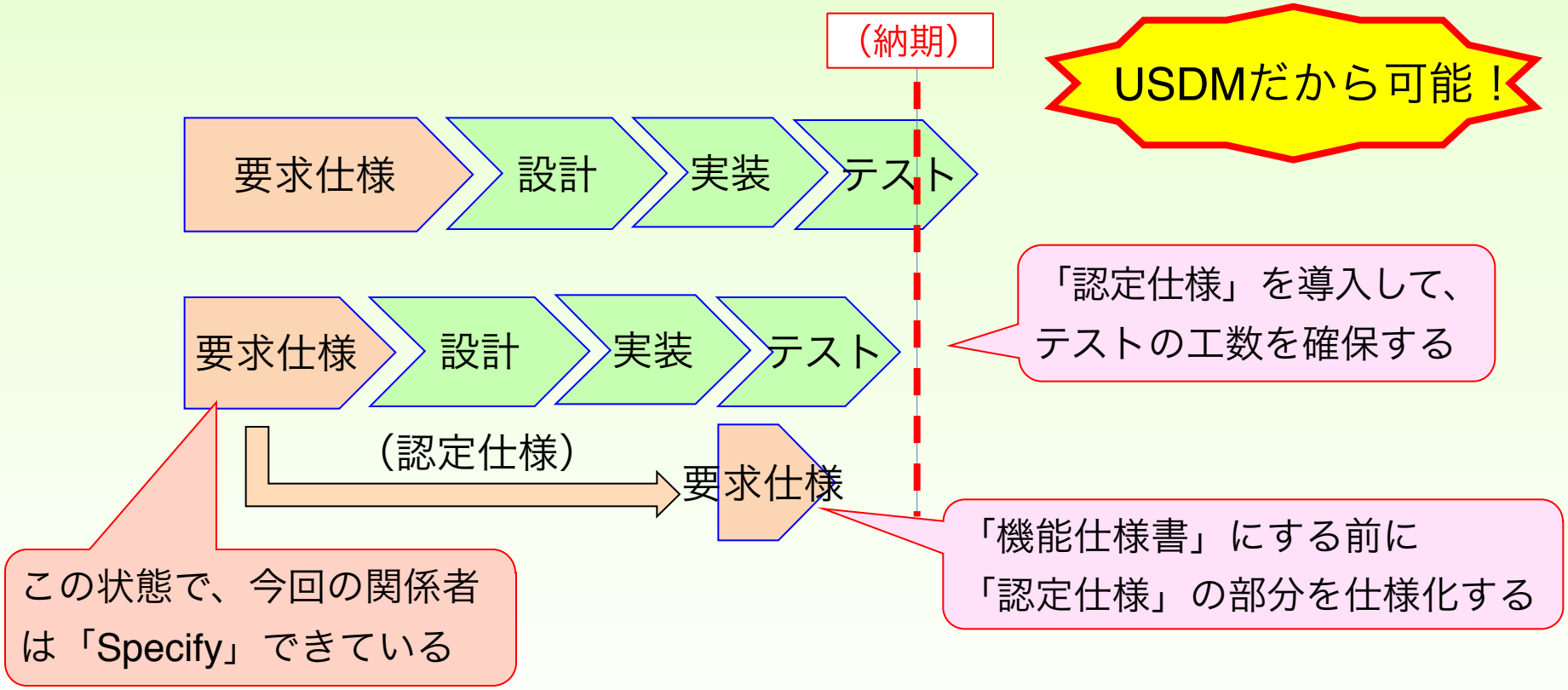
「仕様」として認めたことを示す

「MAL01-01」は、この下に仕様を展開しない

要求	MAL01	事前に指定された受信および送信した電子メールをキーワードで検索してメーラー上で再利用したい	
	理由	メールが多くて,関連するメールを 探せない	
□□□	要求	MAL01-01	検索グループを指定する
		理由	検索グループが複数あるから
	要求	MAL01-02	いくつかのキーワードを組み合わせて検索できる
		理由	可能性のあるキーワードで探したい
	□□□	MAL01-02-1	検索したいキーワードを入力できる
	□□□	MAL01-02-2	複数のキーワードを「AND」と「OR」で繋ぐことができる
	□□□	MAL01-02-3	キーワードは最大８個まで指定できる

「認定仕様」の応用

- 要求仕様書には納期やコストを背負っている
  - テスト工程を確保するために「Specify」の考え方を活用
  - テスト工程の裏を活用して仕様を補充する



仕様から要求を立てる -1-

USDMの得意技

- “みにくいアヒルの子”現象
  - 仕様が、その「要求」と「理由」を共有していない状態
  - “みにくいアヒルの子”現象が起きる原因
    - ① 読み返している中で処理の“流れ”がイメージされ、その先の仕様が浮かんだ
    - ② 関連する仕様を展開している時に“ふっと”必要に気付いて追加した

この仕様は、要求と理由を共有しない

要求	MAL01-02	いくつかのキーワードを組み合わせて検索できる
	理由	可能性のあるキーワードを簡単に探したい
□ □ □	MAL01-02-1	検索したいキーワードを入力できる
□ □ □	MAL01-02-2	複数のキーワードを「AND」と「OR」で繋ぐことができる
□ □ □	MAL01-02-3	キーワードは最大８個まで指定できる
□ □ □	MAL01-02-4	検索されたメールの「Subject」を一覧で見せる

- “みにくいアヒルの子”現象の対応は3種類ある

① この仕様が収まるべき要求を探して移動する
② 新たに適切な要求（と理由）を設定し、不整合を起こしている仕様をそこに移す
③ 「理由」と「要求」の表現を変えて問題の仕様を受け入れる

仕様から要求を立てる -2-

USDMの得意技

- 要求を設定したことで、新たに仕様の欠如に気付く可能性がある

要求	MAL01-03	検索結果を扱いやすく表示し、そこから選択したい
	理由	目的のメールが一つとは限らないので、絞り込めるような操作がしたい
□ □ □	＜検索結果の表示＞	
□ □ □	MAL01-03-1	検索されたメールの件数を一覧の上に表示する
□ □ □	MAL01-03-2	該当するメールが存在しないときは「該当無し」を表示する
	＜検索メールの表示＞	
□ □ □	MAL01-03-5	検索されたメールの「Subject」を一覧で見せる
□ □ □	MAL01-03-6	検索されたメールに連続番号をつけて表示する
□ □ □	MAL01-03-7	検索されたメールの件数が１０件を超えるときはスクロールバーを見せる
	＜メールの中身の表示＞	
□ □ □	MAL01-03-10	一覧から１つのメールを選んで、その内容を見ることができる
□ □ □	MAL01-03-11	開示されたメールの中で検索キーワードと一致している文字列を赤色で表示する
	＜メールの選別＞	
□ □ □	MAL01-03-15	不要なメールを選んで一覧から消すことができる
□ □ □	MAL01-03-16	一度不要として消されたメールを復活することができる

一般の要求仕様書の構成では、このテクニックは使えない

必要なら仕様にも「理由」を付ける

- 仕様にも、必要なら【理由】や【説明】をつけて補う
  - なぜ、この仕様が必要なのか？
  - 【理由】によっては、設計上で配慮されることがある
  - 【説明】は、仕様の動きなどを補足するものであって、ソースコードにはならない

要求	CH-02	印刷開始時にインクの残量を調べて、最後まで印刷できない可能性があれば印刷依頼者のＰＣ上に知らせて欲しい
	理由	印刷内容を知っている人に、最初に危険性を知らせたい
	＜チェックのタイミング＞	
□ □ □	CH-02-1	印刷開始の操作を完了した時点でプリンターと交信してインクの残量（全カートリッジ分）を知る 【理由】印刷ボリュームと比べて判断する必要があるから
	＜確認操作＞	
□ □ □	CH-02-2	「注意」または「警告」のアナウンスを出したときは、「確認」の操作によって印刷処理を続ける 【説明】インクの交換はこの間に行い、交換後に「確認」操作によって印刷を続ける

ペースト作文を避ける -1-

- 「ペースト作文」は仕様モレを招く
  - “コピー＆ペースト”によって、一部を変更して文章を完成させる方法
  - 条件の組み合わせに関連した仕様では、「ペースト作文」が頻繁に行われる

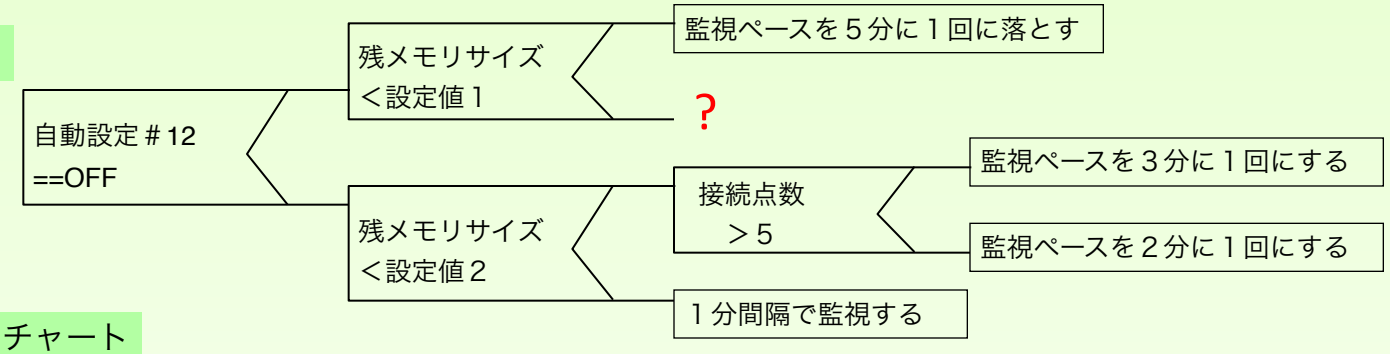
□□□	THM01-1	前回温度が基準値１以下で、かつ、今回温度が基準値１以下の場合は、「A」
□□□	THM01-2	前回温度が基準値１以下で、かつ、今回温度が基準値１を越える場合は、「B」
□□□	THM01-3	前回温度が基準値２以下で、かつ、今回温度が基準値１以下の場合は、「B」
□□□	THM01-4	前回温度が基準値２以下で、かつ、今回温度が基準値１を越える場合は、「C」
□□□	THM01-5	前回温度が基準値２を越え、かつ、今回温度が基準値１以下で、自動設定ONの場合は、「D」
□□□	THM01-6	前回温度が基準値２を越え、かつ、今回温度が基準値２以下で、自動設定OFFの場合は、「E」
□□□	THM01-7	前回温度が基準値２を越え、かつ、今回温度が基準値２を越える場合は、「F」

- 「ペースト作文」は、読み手にとっては苦痛きわまりないもの
  - 条件の組み合わせや必要な変数を見落とす
  - せめて「々（おなじ）」の文字を使うべし
  - ただし、行数の制限も必要

ペースト作文を避ける -2-

- 違いが分かるような表現に変える
  - 「PADフロー」や「デシジョンチャート」などの図やマトリクスを使う方が、
  - 読み手に優しく伝わり、条件のモレや矛盾などを発見しやすくなる

PADフロー



デシジョンチャート

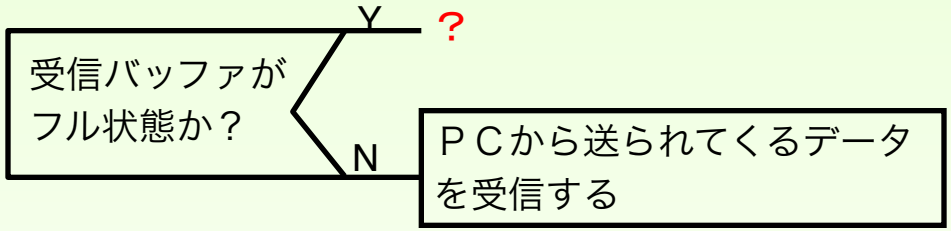
前回温度	今回温度	自動設定		対応
≦ 基準値１	≦ 基準値１	―		A
	＞ 基準値１	―		B
≦ 基準値２	≦ 基準値１	―		B
	＞ 基準値１	―		C
＞ 基準値２	≦ 基準値１	ＯＮ		D
	≦ 基準値２	ＯＦＦ		E
	＞ 基準値２	－		F

否定表現の問題

- “否定表現”とは？

  受信バッファがフル状態の時は、データを受信しない

  - ヒアリングの場や文章での表現の中では自然に使われることがあり、その場は、それで「分かった」と認識される
  - コーディング時に「IF」の片側が定義されていないことに気付くが、それでは遅い！



- 「否定表現」が早い段階で認識されることで適切な対応方法が可能になる

- “肯定表現”でも「IF」の片側の定義が欠けるケースがある
  - 「在庫があることを確認できたときは、速やかに出庫の手配をする」

仕様変更率 “５％以下” を目指す

- 仕様としての表現が適切かどうかを測る方法は？

① 要求仕様のレビューの指摘件数（指摘率）
② 仕様関係のバグの発生率（KLOC単位）
③ 仕様化作業の生産性データ
④ 要求仕様のベースライン設定後の仕様変更率
⑤ 仕様の問合せ件数（率）　など

ベースライン設定後の仕様変更率

$$\left(\frac{\text{変更仕様数}}{\text{総仕様数}} \times 100\right) \leqq 5\%$$

CMMIに取り組む際には早々に確保すべき値

５％以下になることの効果
- 要件管理での対応が容易になる
- 追加機能の受け入れのための変更が混乱しない
- 次回以降にこの機能の仕様変更が混乱しない

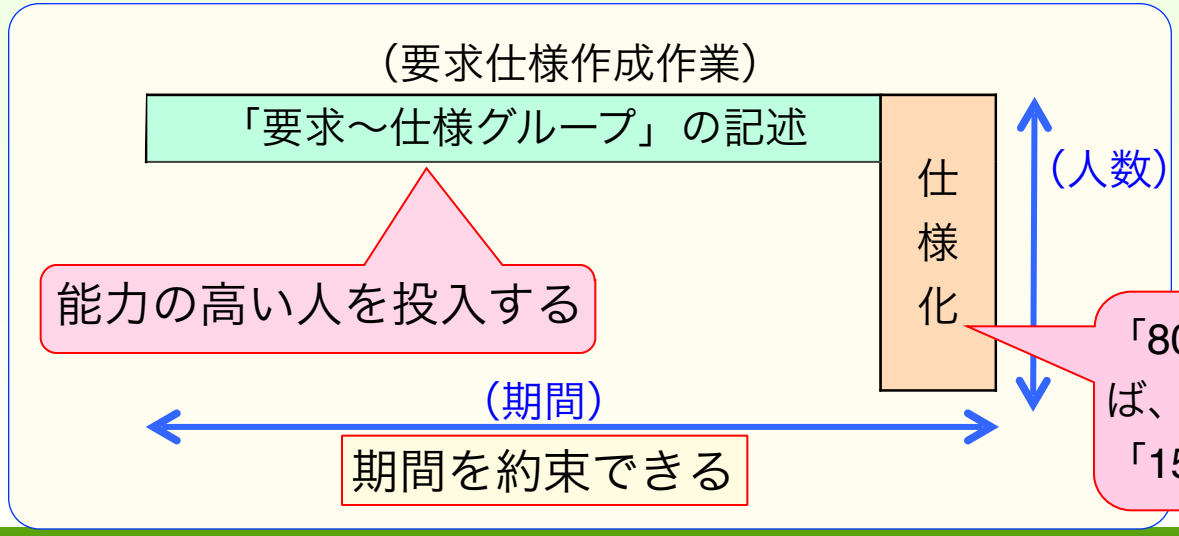
仕様化の作業には「生産性データ」が使える

- ＜仕様グループ＞の下に仕様を抽出する作業は、要求の内容や特徴に左右されにくいので「生産性データ」が使える

① 仕様化作業に新たに人を投入することが可能
② 仕様数の見積りデータと合わせて仕様化作業の工数を算出する
③ 生産性データから成果物（入力）の出来映えを判断できる

条件が整えば、「６０仕様～１００仕様／時間」は可能

納期を約束する方法



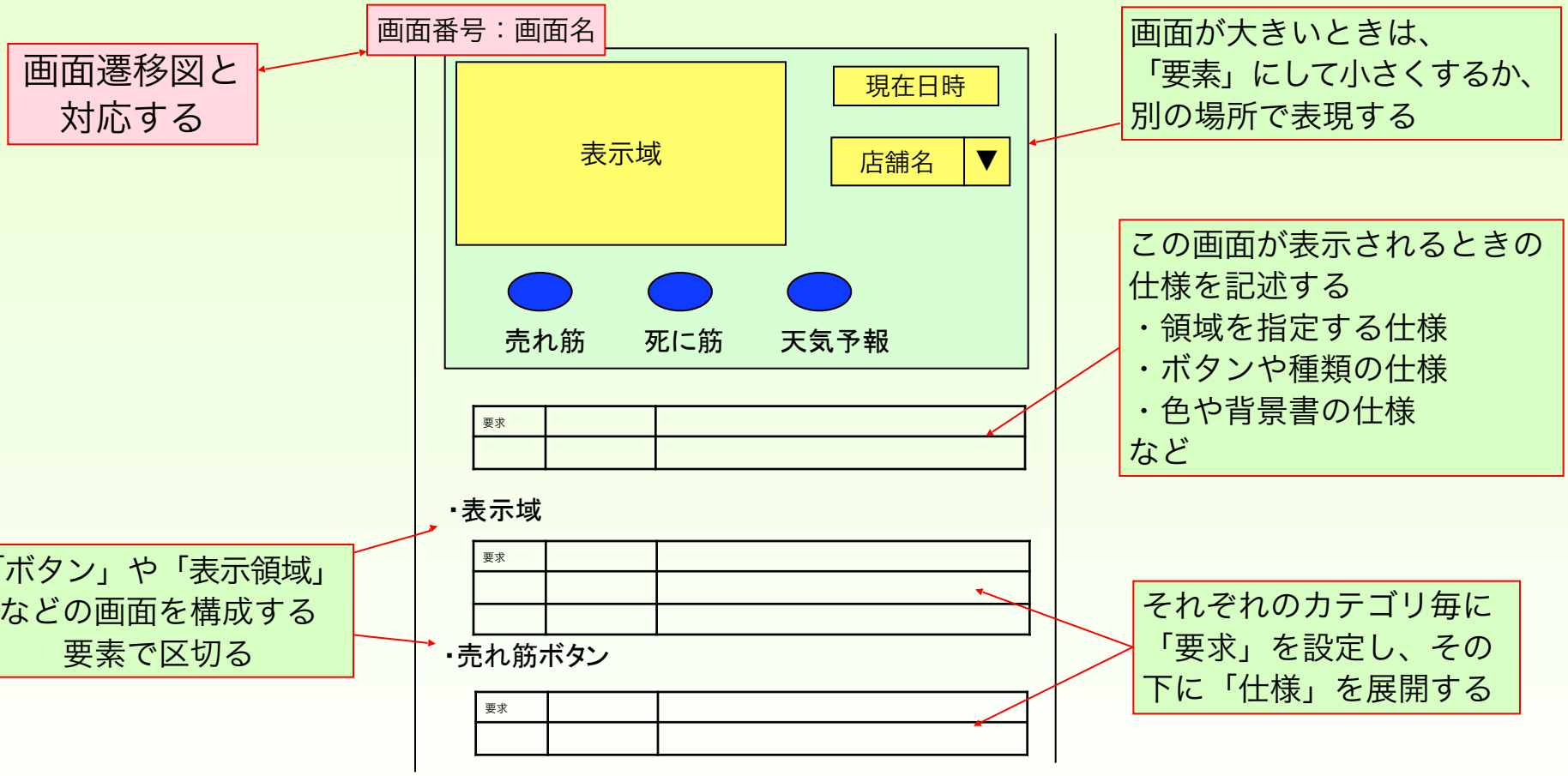
「80仕様／時間」の人が10人集まれば、「10000仕様」の仕様化作業は「15時間」で書けることになる

5. 画面仕様も同じように書けるよ

- 最近では設定や動作などの多くの操作が「画面」を通じて実現できる
- 画面仕様書にも、「要求」と「仕様」の構成を取り入れることで、精度の高い画面仕様書となる

画面の配置図の下に仕様化する

- 個々の画面の中のそれぞれの構成要素ごとに「要求」を表現する
  - 通常の要求仕様の表記法をそのまま適用できる



画面要素ごとに要求と仕様を表現する

- 個々の画面構成のすべての「要素」に対して「要求」を表現し、「仕様」を展開する
- 一般に「動詞」が少なく、ほとんど「１階層」で対応できる

GUIツールで確認する項目

GUI

天気予報ボタン	要求	WES05	本日と明日の時間単位の天気予報と、その後の６日間の１日単位の天気予報をネットから入手して表示する	
		理由	商品の仕入れや陳列方法の判断に使う	
		＜表示フォーマット＞		
	□□□	WES05.1	本日と明日の分は２４時間分を４時間毎に区切って６マスで表示する	●
	□□□	WES05.2	その後の６日分は１日単位で６マスで表示する	●
		＜各マスの表示データ＞		
	□□□	WES05.5	本日と明日の一つのマスには以下のデータを表示する ・時間（４時間単位の最初の時間） ・天気記号 ・予想降雨量、ただし本日で過ぎた時間のマスには降雨実績を表示する ・予想気温	●
	□□□	WES05.6	その後の６日分の一つのマスには以下のデータを表示する	●
		＜ネット経由でのデータの獲得＞		
	□□□	WES05.1	天気ボタンの押下によって、ネットに接続し○○から天気データを入手する	
	□□□	WES05.1	本日の２４時間分のデータから表示対象のデータを表示する	

仕様の＜グループ＞がパターン化する傾向

- ボタン類の要求は、 仕様の＜グループ＞を“パターン化”できることが多い
  - ＜グループ＞の一部は、当該ボタン独自の仕様を扱うことになるが、そのほかの＜グループ＞は共通になることが多い

  - ＜このボタンが押せる条件＞
  - ＜押した時のファイル操作＞
  - ＜計測を停止させる方法＞　・・・・・・当該ボタン特有のもの
  - ＜計測が停止したことの確認方法＞・・・当該ボタン特有のもの
  - ＜次の画面への切り替え＞

- ＜グループ＞のパターン化によって、仕様の抽出がはかどる

ボタンの性質によって複数パターン存在する

画面操作の品質要求も忘れずに

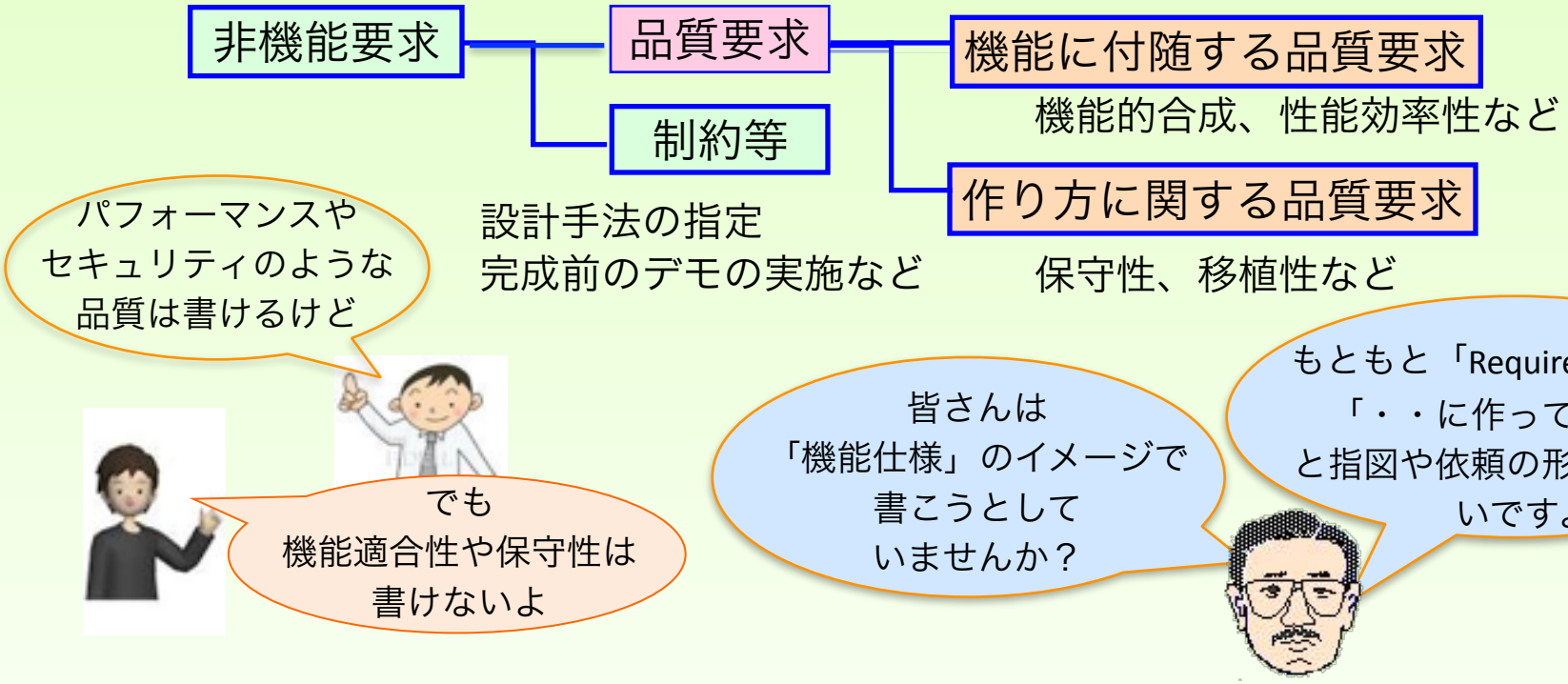
- 画面操作によっては、その画面特有の品質要求が存在することがある
  - 応答性
    - 操作に対するスムースな反応
    - 入力データの表示の速やかさ
  - 配色や操作性
    - 操作画面として目が疲れないこと
    - 画面内の項目の見易さや選択しやすさ
    - 適切に状況を伝えることで誤操作を防ぐ

6. 品質要求を表現するコツ

- 品質も「要求」と「仕様」を表現する
  - 個々の品質特性に対して実現したいレベルや範囲を要求として表現する
  - 品質要求に対して、それを実現するための仕様を展開する
- 「仕様」であることの条件は、通常の仕様と同じ
  - ただし、保守性などの品質特性については、プログラムを実行して確認することはできない

非機能要求とは

- 現実には、品質要求は明文化されていないことが多い

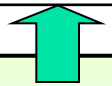


- 品質要求の内容によっては設計に大きな影響を与える
  - 複数のモジュールが連携して実現することが多い
  - モジュール構造やデータ構造の設計を間違えれば要求は実現しない

品質特性の例（製品の品質）

- それぞれのシステムに応じて想定される「品質要求」を把握しておく

(副)品質特性	認識	要求表現事例	仕様表現事例
モジュール性	変更が他のコンポーネントへの最小の影響ですむように構成されている度合い	処理やデータ構造を変更した時に、影響が拡散しないように作って欲しい	
再利用性			



品質特性								
副品質特性	機能適合性 (Functional Suitability)	性能効率性 (Performance efficiency)	互換性 (Compatibility)	使用性 (Usability)	信頼性 (Reliability)	セキュリティ (Security)	保守性 (Maintenability)	移植性 (portability)
	機能完全性 (Functional completeness)	時間効率性 (Time behaviour)	共存性 (Co-existence)	適切度認識性 (Appropriateness recognizability)	成熟性 (Maturity)	機密性 (Confidentiality)	モジュール性 (Modularity)	適応性 (Adaptability)
	機能正確性 (Functional correctness)	資源利用性 (Resource utilization)	相互運用性 (Interoperability)	習得性 (Learnability)	可用性 (Availability)	インテグリティ (Integrity)	再利用性 (Reusability)	設置性 (Installability)
	機能適切性 (Functional appropriateness)	容量満足性 (Capacity)		運用操作性 (Operability)	障害許容性 (Fault Tolerance)	否認防止性 (Non-repudiation)	解析性 (Analysability)	置換性 (Replaceability)
				ユーザーエラー防止性 (User error Protection)	回復性 (Recoverrability)	責任追跡性 (Accountability)	修正性 (Modifiability)	
				ユーザーインターフェース快美性 (User interface aesthetics)		真正性 (Authenticity)	試験性 (Testability)	
				アクセシビリティ (Accessibility)				

（JIS X 25010、品質特性・副品質特性より引用）

品質の定義は要求仕様の表現になっていない

- ISO-25010の「定義」は評価の視点

品質特性	定義	
時間効率性（time behaviour）	製品又はシステムの機能を実行するとき，製品又はシステムの応答時間及び処理時間，並びにスループット速度が要求事項を満足する度合い	• どの機能に適用するのか？ • 目標とする応答時間は？ • その時間を満たせない時の代案は？
修正性（modifiability）	欠陥の取込みも既存の製品品質の低下もなく，有効的に，かつ，効率的に製品又はシステムを修正することができる度合い	• 変更（修正）時に何を満たしていれば良いのか？ • どのように作ればこの定義を満たすの？

作ったあとの評価尺度であったり、後でわかることだったりする

個々の品質特性に対して、何ができればよいのか分かりにくい

「USDM」の様式で「要求」と「仕様」に表現することで、設計に織り込めるようにする

品質要求の表現

「実現可能性」のレベルで表現する

- 品質要求も「要求」として表現する
  - 要求として表現することで、品質要求を仕様化できる
  - 対象の品質特性を明記することで、関係者が同じイメージでレビューできる
  - 「パフォーマンス」などでは、「目標」が数値で表現されることがある

品質特性

要求	DSP01-02	【時間効率性】「表示」ボタンを押してから２秒以内に画面の表示が完了すること
	理由	ここで表示するデータの処理量が多く、表示が遅いとハングアップしたと勘違いする可能性があるため
要求	QUA.02	【変更性】ソフトウェアの保守性を確保して変更に簡単に崩れないシステムを作って欲しい
	理由	この後５年以上（３０回以上）、ここから製品展開したいから デリバリを的確に実現することでシェアをひっくり返したい
要求	QUA.03	【障害許容性】接続する機器が故障しても、全体の動きを止めないで欲しい
	理由	工場のラインを制御している装置だから

品質要求を仕様化する

- 品質要求も具体的に仕様として定義されないものは実現しない
  - 必要な品質特性に応じて達成すべき要求と
  - それが達成したといえるための仕様を記述する

- 「２秒」は目標（=要求）であって仕様ではない
- プログラムコード上に現れることはない

要求	DSP01-02	【時間効率性】　「表示」ボタンを押してから２秒以内に画面の表示が完了すること
	理由	ここで表示するデータの処理量が多く、表示が遅いとハングアップしたと勘違いする可能性があるため
□□□	DSP01-02-1	画像処理するデータ量が「2MB」以内の時は、通常通りに処理する
□□□	DSP01-02-2	画像処理するデータ量が「2MB」を越えるときは、処理中は「プログレスバー」を表示する

- この仕様を満たせば、「２秒以内に表示が完了した」と“みなす”

「要求」と「仕様」の階層構造が「みなす」という概念を成立させる

保守性も仕様化しないと実現しない

- 保守性や移植性の品質要求も、仕様化しないと実現しない
  - ただし、“品質を織り込む”設計技術が必要
  - 通常のテストで確認できないが“仕様としての条件”は同じで、実現していることの検証は「静的テスト」で行う

要求	QUA.02	【変更性】 ソフトウェアの変更に対する耐久性（崩れにくさ）を向上させて欲しい
	理由	この後５年以上最低３０回のバージョンアップをここから展開したいから
□□□	QUA.02-1	一つのクラスに含まれるメソッドは １０個以下とし、越える場合は事前に了解をとる (*2)
□□□	QUA.02-2	モジュールの複雑度(*1)は１７以下を原則とし、越える場合は事前に了解をとる (*2)
□□□	QUA.02-3	モジュールの凝集度も“手順的凝集度”以下になる場合は事前に検討する (*2)
□□□	QUA.02-4	処理と管理は明確に分離し、関数の呼び出しの深さが５を越えるときは事前に了解をとる (*2)
□□□	QUA.02-5	タスク間でアクセスし合うグローバル・データを作らない。アクセス時間等による制限から、グローバル・データとなることが避けられない時は、品質検討会議で承認されること (*2) 【説明】割り込み処理との間でのみ共有するケースはこの制限を受けない

- この仕様を満たせば、「保守性」を満たしていると“みなす”

(*1)＝マッケイブ（McCabe) の循環的複雑度（Cycromatic Complexity)
(*2)＝PLが対応したり、品質検討会議のような組織で対応する

品質を織り込む技術が必要

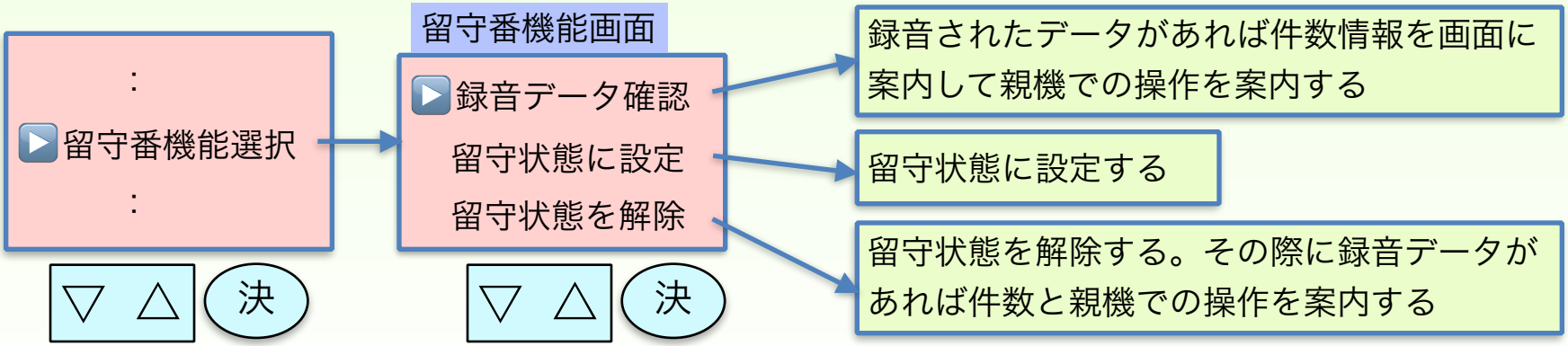
- 品質要求は、機能を実現する設計技術とは別の設計技術が必要
  - 品質要求が課されなかったことで、これらの知識や技術の習得を怠った

品質を織り込むための知識や技術	有効な品質要求
• データ構造 • アルゴリズム	• 性能効率性などの性能に関する品質要求
• モジュールの分割基準 • モジュールの尺度（凝集度、結合度、複雑度など） • クラスの分割基準（デザインパターンなど） • クラスの尺度	• 保守性や移植性などの品質要求
• マルチタスクシステムに於いて、タスクの外にグローバルデータを出さない設計技術 • 「Switch文」を使わない状態遷移の設計技術 • 複数の状態遷移をコラボレーションさせる設計技術 • タスク単位でリエントラントに設計する技術	• 保守性、移植性、互換性、性能効率性に関する品質要求

7. 演習ーちょっとUSDMで書いてみよう

FAX電話（子機）の留守番機能を仕様化してみよう

- 子機の操作
  - 「留守番機能選択ボタン」の押下でこの機能の操作が始まる
  - カーソルの移動は「▽」「△」のボタンで操作し、「決定」ボタンで選択する
  - 留守番機能の画面が表示された時、カーソルは一番上にある
  - 留守番機能の画面に表示されるメニューは「再生」「留守設定」「留守解除」の３種類
  - 状況を判断して使いやすい操作（使用性）を考えてください
- データ
  - 留守設定の状態（未設定状態／設定中状態）
  - 留守時に録音されたデータ（件数［再生未／済｜録音内容］）
  - 留守時にかかってきた電話に流す応答メッセージ（「ただいま電話に・・・」）は、別の設定操作で登録されるので、ここでは意識しないでよい

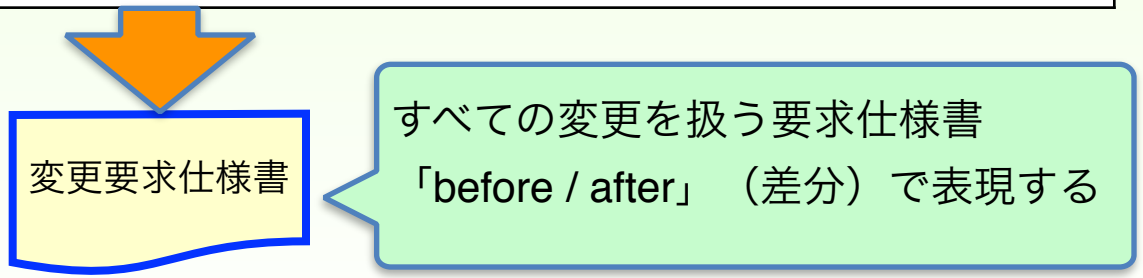


8. ところで、派生開発で威力を発揮するUSDMって？

派生開発でもUSDMの特徴が活きる

- USDMの特徴を活かすことで、派生開発に効果的を発揮する要求仕様書（変更要求仕様書）となる

USDMの特徴	派生開発での効果
要求と仕様の階層表現	変更の狙いや意図を「変更要求」で表現し、具体的な仕様レベルの変更を「変更仕様」として階層で表現する
	当初の変更に対して影響を受けて変更する箇所も、その変更要求の下位層に表現することで、関係がわかりやすくなる
要求と仕様を分けて表現	変更を仕様レベルで表現することで、影響箇所に気付きやすくなる
要求に理由をつける	変更には必ず「理由」があり、それを明示することで変更の意図を正確に把握することができる
Specifyを前提とする	変更には設計プロセスがないため、「before / after」で変更要求と変更仕様を表現することで変更の様子をイメージさせる



（参考文献）

- 文献（単行本）
  - ① 【改訂第２版】要求を仕様化する技術・表現する技術（技術評論社）
  - ② 『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）

USDM

XDDP