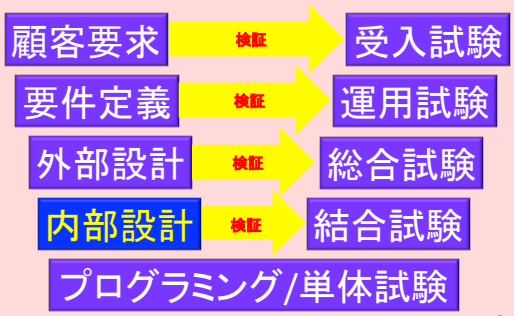
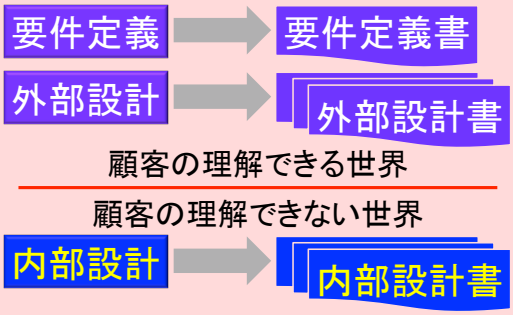
内部（詳細）設計について

ハノイ工科大学　HEDSPI
ＩＴ日本語　講師：　権代　祥一

内部(詳細)設計の位置付け(例)



内部設計(詳細設計)の位置付け



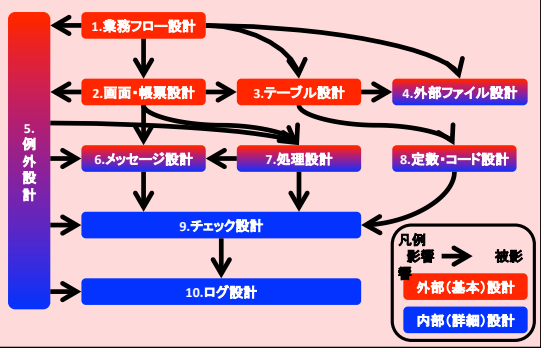
外部(基本)設計と内部(詳細)設計

- 外部(基本)設計
  - 設計者とユーザのコミュニケーションツール
  - ユーザの要望を理解してシステムの仕様にまとめたもの
  - 要望をシステムがどのように実現したかをユーザに理解してもらうための資料

外部(基本)設計と内部(詳細)設計

- 内部(詳細)設計
  - 設計者とプログラマのコミュニケーションツール
  - 設計者からプログラマに向けてプログラミングに必要な技術仕様を伝える資料

外部設計書と内部設計書の体系(例)



内部設計での処理設計（例）

- 顧客にとって意味のある処理やロジックやアルゴリズム以外でプログラマに伝えるべきことを書く。
- 主に内部データやテーブルとの入出力を主体に書く。
- 処理一覧を作成する。

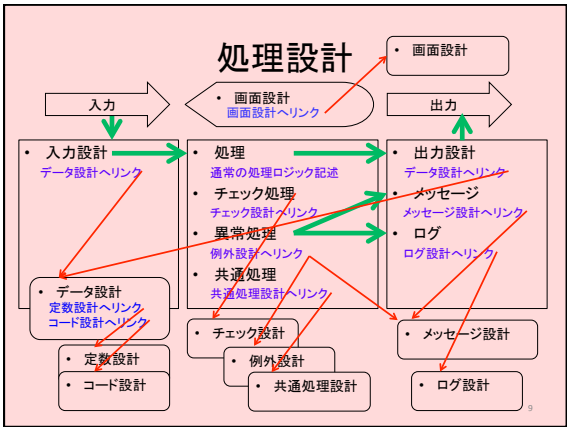
7

内部設計での処理設計（例）

入力＝＝＞処理＝＝＞出力

- **処理**内容の記述
  - 共通**処理**
  - 正常**処理**
  - チェック**処理**・・・チェック設計とリンクする
  - アルゴリズム設計（外部設計のを詳細化）
  - 異常**処理**・・・・例外設計へリンク
- 画面・帳票設計を修飾する形で記述

8



9

リンクするという意味・・・

異常処理・・・・例外設計へ「**リンク**」 ？？？

- 「関連付ける」という意味で使う。
- 異常処理に書くことを例外設計にも書くが、２重管理にならないように１箇所で書くように工夫をしてくださいという意味。

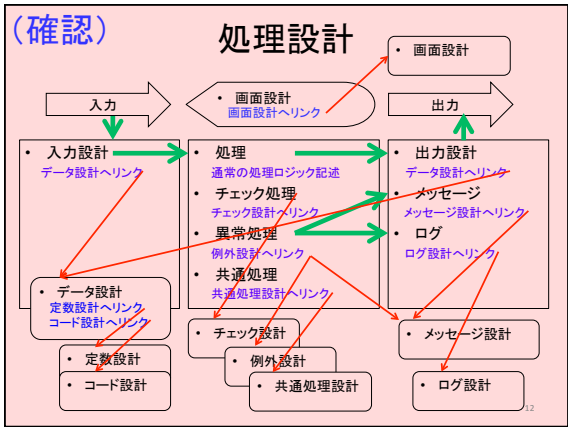
10

リンクするという意味・・・

- 同じことを2回書いた場合、どんな問題が生じるだろうか？

答え：仕様書を修正する場合、同じことを2回修正しなければならない。修正ミスが生じ易くなる。保守性が低下する。

11



12

仕様書は・・・

それだけで独立した

システムである！

13

処理設計での注意事項

- 全てのロジックを細かく詳細に書く必要はない。
- 設計者はプログラマではない。
- 詳細なロジックを考えるのがプログラマの仕事である。
- 設計者はより全体を見るべき。

14

コード設計

- 意味のある文字列データを計算機で処理しやすいように数値化、記号化したもの
- データの識別、データの分類、データの並び替え、データのチェックなどを考えて設計する。

15

内部設計での定数・コード設計

- 顧客ではなくシステムにとって意味のあるコード、定数の決定。
- コード一覧表の作成
- コード定義
  - フラグの意味定義・内容説明
  - ステータスの意味定義・内容説明
  - コードの意味定義・内容説明
- 定数設計とのリンク

16

コード設計について

学校コード、県コード、社員コード、
ISBNコード、・・・・・、バーコード・・・

世間はコードだらけ。

例：ISBNコード（10桁）
(International Standard Book Number)
ISBN G-AAAA-BBBB-C ・・・・・

17

コード設計について

ISBN G-AAAA-BBBB-C

- G: グループ記号　0、1：英語　・・・
  4：日本　・・・（これらは定数）
- AAAA: 出版者記号
- BBBB: 書名記号
- C: チェックデジット

このような取り決めを設計する。

18

コードと定数

コードの中に定数が含まれる場合がある。
- 1:Yes　0:No　・・・・この場合1、0は定数
- 1:Off　0:On　・・・・負論理で設計する場合はこのように逆転することもある。
（新幹線のシステムはこの負論理コード体系で統一されている。）

ISBN番号のグループ記号なども定数
- 2:仏語　3:独語　4:日本　5:露語　7:中国

19

定数設計例

- 定数一覧表の作成
- コード設計とのリンク
- テーブル設計とのリンク
- 外部ファイル設計とのリンク
  - –I／F設計とのリンク
  - –環境ファイル設計とのリンク

20

チェック設計例

- 各入力に対するチェック
  - –記号レベルチェック
  - –形式レベルチェック
  - –意味レベルチェック
  - –記銘レベルチェック

21

チェックレベルについて

- 記号レベルチェック
  - –アルファニューメリックチェック等
- 形式レベルチェック
  - –フォーマットのチェック、文法チェック等
- 意味レベルチェック
  - –意味や論理性のチェック等
- 記銘レベルチェック
  - –過去の記録と比較するようなチェック

22

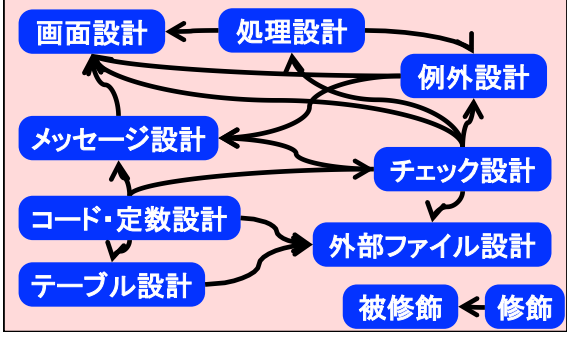
各チェック処理は・・・・

- 処理設計とリンク
- 例外設計とリンク
- メッセージ設計とリンク

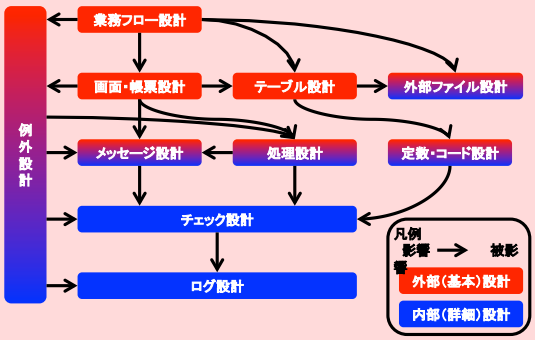
させてください。

23

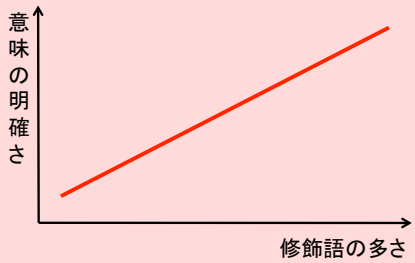
各設計とのリンク（修飾）関係（例）



ちなみに・・・設計体系(例)との違い



(復習)修飾語の量と意味の関係



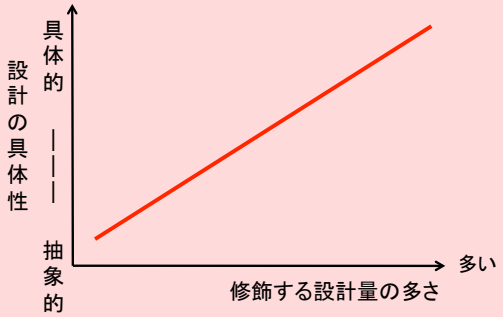
- ちなみに出さない。 何を出さない？
- ちなみに卒業証書を出さない。 誰が出さない？
- ちなみに学校が卒業証書を出さない。 なぜ？
- ちなみに合格していないと学校が卒業証書を出さない。 何に合格していないと？
- ちなみに試験に合格していないと学校が卒業証書を出さない。 何の試験？
- ちなみにFEの試験に合格していないと学校が卒業証書を出さない。 他には？
- ちなみにN3の試験とFEの試験に合格していないと学校が卒業証書を出さない。 これって本当？
- ちなみにN3の試験とFEの試験に合格していないと学校が卒業証書を出さないというのは本当ですか？

内部設計にも同じことが言える

- 外部設計に対して修飾語を追加するようなもの。
- 抽象的な概念が詳細な設計書(修飾語)によって意味が明確に、具体的になっていく。
- 抽象的なままではプログラミングができない。

28

修飾する設計の量と具体化の関係



例外(異常処理、復旧)設計(例)

- チェック処理等で異常を検出した時の例外処理(エラー処理)を設計
- 例外処理でのエラーメッセージ等はメッセージ設計とリンク

30

その他、例外設計とのリンク(例)

- 業務フロー設計とリンク
- 画面・帳票設計とリンク
- 処理設計とリンク
- チェック設計とリンク
- ログ設計とリンク
- 外部設計とリンク

31

内部設計でのメッセージ設計(例)

- 内部ログに書きこむような顧客に見えないメッセージなどの設計を実施(デバグログなど)
- メッセージの一覧表の作成

32

ログ設計

- 問題が発生した場合、必ずログ情報から確認することが多い。
- この設計の良し悪しでシステムの品質が変わってくる。

33

ログの種類

- オペレーションログ(操作ログ)
- イベントログ
- エラーログ
- デバッグログ
- 通信ログ

34

ログ出力のメリット／デメリット

- ログをたくさん出力すると障害解析、調査が容易になり原因把握が簡単になる。
- ログをたくさん出力すると性能が悪くなり、資源をたくさん喰う。解析に時間がかかる。

35

5W3Hによるログ設計のヒント

- Why　なぜ書くのか？目的は？
- What　何を書くか？何に書くのか？
- Who　誰がログを書くか？誰が見るか？
- When　いつ、どのタイミングで書くか？
- Where　どこにログを書くか？
- How　いかに書くか？　方式は？
- How Long　どのくらい保存するのか？
- How Much 開発規模は？

36

設計で注意して欲しいこと

- 全体が見えるようにして欲しい。
- 設計書全体の構造が見えるようにして欲しい。
- 細かく書いても全体が見えなければ意味がない。

37

全体が見えないと・・・・

- 網羅性の検証ができ難くなる。
- 改修時の影響範囲を検証できない。
- 直感性が悪くなる。
- レビューの効率が悪くなる。
- 最適化ができない。

38

SEは・・・・

- 全体が見えていなければならない！
- 「木を見て森を見ず」ではだめである。
- 「木も見て森も見る」でなくてはならない。

39

全体が見えるようにするには・・・

- 一覧表を多用する。
- 仕様書の構造を統一する。
  - ➢記号や章の数字を同じフォントで同じ大きさにする。
  - ➢インデントを統一する。
  - ➢項目の種類の並べ方を揃える。

40

設計は冗長か？

- 設計は冗長である方が良い。
  - –余分な領域を確保する。
  - –余分な項目を作っておく。（重複ではない。）
  - –もしかしたら・・・・・というものに対応しておく。
  - –性能に余裕を持たせておく。
  - –本当の限界値の50～80%を限界値としておく。
- しかし、コストや納期とのバランスが重要である。

41

設計に網羅性（もうらせい）があるか？

- 漏れが無い。重複が無い。(MECE)
- 全てをカバーしている。（網羅している。）
- これらのことに根拠があり、網羅していることを証明できること。

つまり・・・

設計には網羅性が必要だ！

42

網（魚を取るあみ）、羅（鳥を取るあみ）

- 同じことを何度も書いていないか？
- 書き漏らしていることはないか？
- MECE（Mutually Exclusive Collectively Exhaustive）であるように考える。
- 設計書はできる限りリンクさせて、関連づけるような書き方にする。同じことを何度も書かない。

43

設計にあたって

- 誰が何をいつまでにやるのか決めてください。
- 議事録を提出してください。
- 何をやったのか報告書を書いてください。

44

質問をどうぞ!

45