CSC227
Formal Specification of Software(1)

John Fitzgerald
Centre for Software Reliability
University of New Castle

Translated by
Takahiko Ogino
Railway Technical Research Institute

Based upon the book
Modelling Systems: Practical Tools and Techniques in Software Development
(ISBN 0 521 623480 Cambridge University Press 1998)
by John Fitzgerald and Peter Gorm Larsen (IFAD)

凡例

- 本翻訳は、John Fitzgerald博士（Centre for Software Reliability、University of New Castle）の講義用OHPを翻訳したものである。
- 翻訳にあたっては、なるべく忠実な訳をしたつもりである。ただし原稿が、授業の教材であることを勘案し、出来るだけ意味的な注釈を付けたり、空白になっておりクラスで埋められるのであろうと思われる部分は、出来るだけ講義のベースとなっている本、Modelling Systemsからの引用によって中身を埋めた。
- そのような部分をオリジナルと区別するため、フォントの色を変化させることで出来るだけ区別している。
- また、同じ鉄道総研の寺田夏樹君には、日本語と内容のプルーフリーディングをお願いしたことを記載し感謝の意を表します。
- 原OHPの修正も含めて全ての日本語版内容に関する責任は、荻野にあります。
- 誤訳も含めて、内容に関するご指摘をお待ちしています。

ソフトウェア　現在の問題：
なぜシステムをモデルする事が必要なのか

項目

- ソフトウェア開発における挑戦
- 計算機システムのモデリング
- 形式性
- 形式的仕様記述言語
- 講義の構成

ソフトウェアの特徴

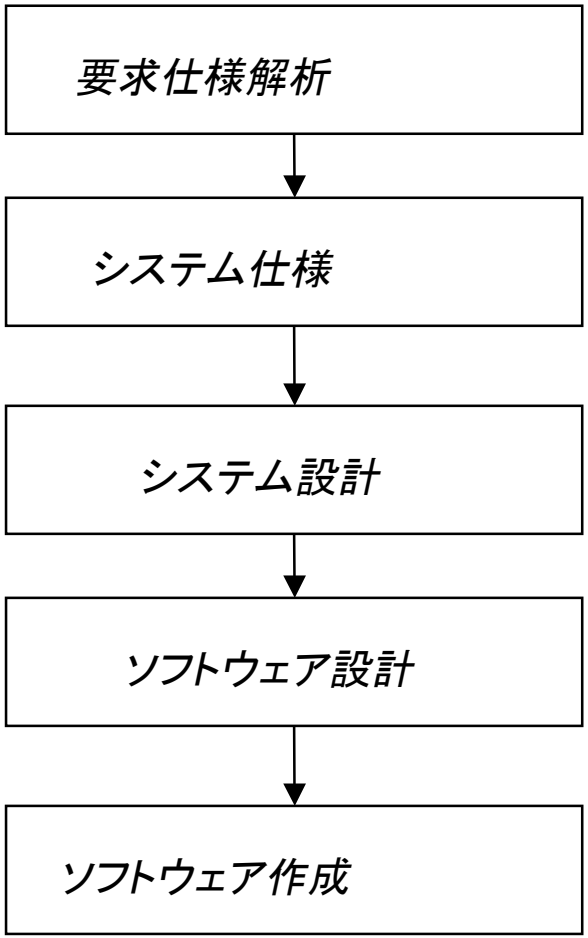
- （生産システムとのアナロジーをとると、次のように言える）
  ソフトウェアを材料にして計算機システムを作成する。
- 他の分野の技術者は、鉄、電気素子、先進原材料（と言った物理的な原材料）を利用している。
- 何がソフトウェアを特別なものとしているのか？
  - 無限に近い柔軟性、物理的特性（質量、体積など）の欠如

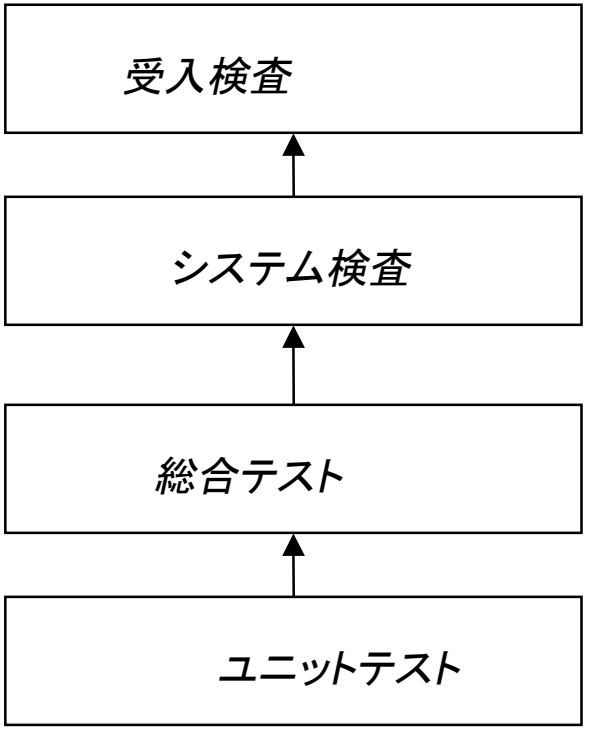
ソフトウェア　現在の問題：　挑戦

- 技術的な観点
  - 以前よりもっとソフトウェアで可能な事が広がっている。
  - ソフトウェアは非常に重大な機能と直結した分野で使用されている。
    - 安全性やセキュリティの分野で使用されているソフトウェアを挙げてみなさい。
      （核関連システムの制御。飛行機や宇宙船の制御。列車やITSなど交通関係システムの制御。…）

- 経済的な挑戦：修正のコスト
  - ソフトウェア開発は、大規模で行われ、しばしば失敗する！
  - どれだけのソフトウェアが引き渡されたときに、すぐさま使用出来る状況になっているだろうか？

ソフトウェア　現在の問題：　挑戦　修正のコスト





ソフトウェア　現在の問題：　挑戦　修正のコスト

- バグを訂正する修正コストは、(仕様で規定されている)やらねばならないことと、見つかったエラーとの距離に関連する。
- 何人かの研究者は、システムテストで発見された設計上の問題点の修正に必要な修正コストの方が、インプリメンテーション・エラーを修正するコストに比べ100倍にまで大きくなると主張している。
- もし、要求仕様や設計を解析し、そこに含まれる未発見のエラーを見つける能力が向上したら、(その中に含まれるであろう)最も高価なエラーのうち、いくつかの修正コストを減らすことが出来るであろう。
- この講義(CS227)では、このような解析を可能にする技術に付いて述べる。

計算機システムのモデリング

- 他のエンジニアリング分野(機械、電気、航空工学など)では、システム・モデルは、要求仕様と設計が同一であることを、確実に確認するための手段として作成されている。
- 例（例は、Modelling Systemsを参照）
- 本講義では、どのようにソフトウェアのモデルを構築し分析することができるか示す。
- モデルの役割
  - 設計のアイデアを実際に試みる、若しくは、プロジェクトのフィージビリティを確かめる。
  - 技術者と顧客とのコミュニケーションの手段。
- 実際に作る前にモデルを作成するのは、ハードであれ、ソフトであれ、最終的に問題が出現する前に、それを発見する事を可能とする唯一の手段である。
- ここに於いて重要なモデルの2つの特性は:
  - 抽象化と厳格

モデリング:　抽象

- 工学モデルは、モデルの目的に関連しない詳細を省略する。
  - 必要な詳細は省略しない
  - モデルは、特定の目的に対して作成される。
  - あるシステムを構築するなかで、様々な目的別に、それぞれを補う形のモデルが複数作成される可能性はある。
- モデルの目的に関連しない詳細の省略を抽象と呼ぶ。
  - 工学モデルが抽象的であるという事は、この意味である。即ち、特定のモデルの目的に対して重要でないと考えるシステムのアスペクトは省略される。
  - 例えば、飛行機の翼の設計時には、コックピットの内部の詳細は省略される。
- どの詳細を省略するべきかの選択はEngineering Skillである。

モデリング：　抽象

- 同じシステムの2つの記述の抜粋を比較してみなさい。
  - FlightFinderシステムは旅行代理店およびその顧客によって使用されることになっている。詳細は、出発地、目的地、希望の日付・時間を含めて、入力される。システムは一連の旅程および料金を関連した制限とともに回答する。

  - 位置はレコードとポインターによって実現されたされた連結グラフ構造中のノードである。各ノードのレコードは、ポインタの配列を持つ。そのポインタは到達可能な目的地とそれぞれのポインタ(目的地)に対して、ハッシュ・テーブルの形で格納されたフライトの時刻表がある。ハッシュ・テーブル中の各レコードはフライトナンバー(8文字のストリング)、出発および到着時刻(標準時フォーマット)、および運行される日付(標準日フォーマット)を持っている。最適のルートを見つけるため、修正された隣接マトリックス上で、ダイクストラの最短のパス・アルゴリズムを使用して、検索することが必要であり、...

モデリング：厳密性

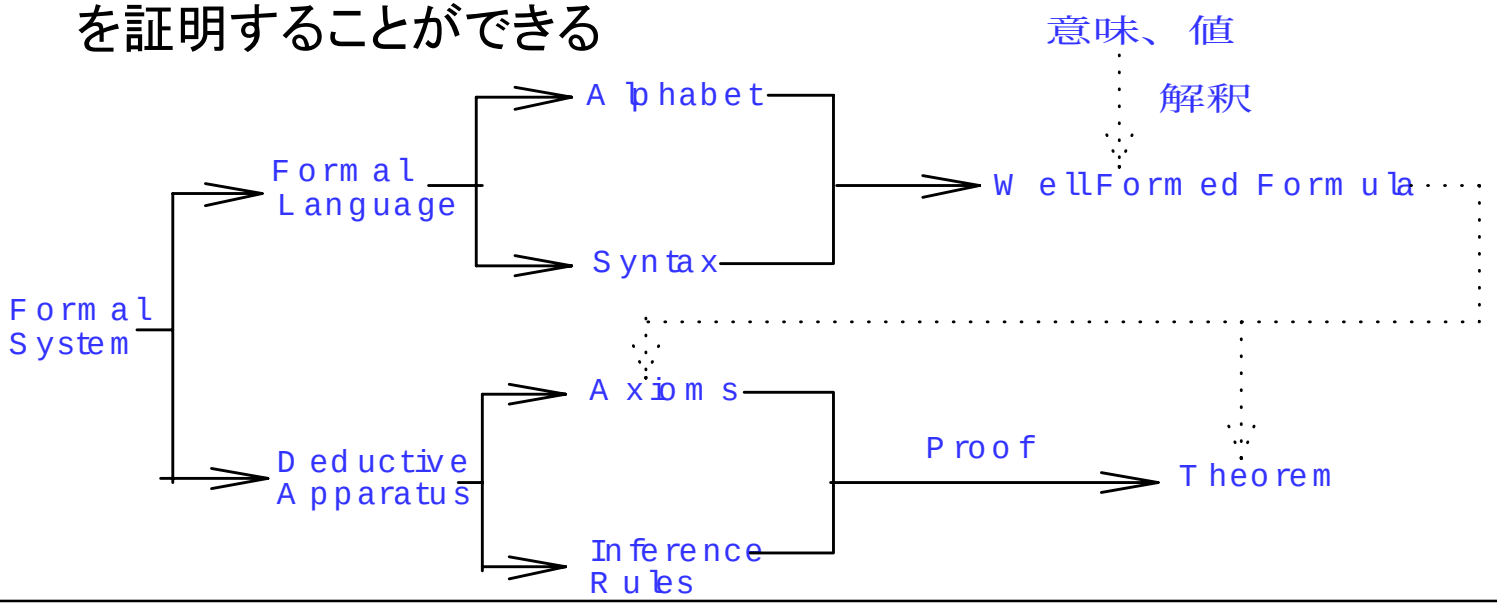
- 計算機システムのモデルの最も重要な特性は分析に対する適応性である。
- 分析は**客観的**でないといけない。(それを行う個々の技術者の意見に従ってはいけない。)
- 分析は**反復可能**であるべきであるし、**機械的な支援**が可能であるべき。
- モデルが表現される言語は**厳格に定義される**べきである:
  - モデルが現実に記述することに関して不一致の余地が無い
  - 分析ツールは、モデルの特性に関して同じ結論に達する。
- プログラミング言語には通常厳密でない定義がある。
  - その結果どうなるか？

計算機システムのモデル化

- システムモデルのこれらの概念は、ソフトウェア開発にどう結びつくか？
- 様々なモデル化技術がソフトウェア開発で使用されている。
  - （擬似コード,自然言語,図的記法,数学的記法,データフローなど）
- 開発の初期で作られたモデルは**仕様**と呼ばれ、開発の後期で作られたモデルは、**設計**と呼ばれる。
- 我々は仕様書に重点をおく。なぜなら、初期の開発でのモデル化が重要であるから。
- **モデル**という言葉を、今後開発するシステムの記述を意味することに使用する。

形式性（Formality)

- 本講義ではモデルを表現するための形式的言語を中心にする。
- 言語がフォーマルであるというのは、そのシンタックス規則、およびその意味(言語の全ての構成要素の意味)が、非常に正確に定義されており、モデルの意味に関する不一致の余地がない場合を言う。
- 形式言語で表現されたモデルは、数学的な証明を含む、広範囲の分析技術に対応できる。
  - 我々は、原則的には、モデルが安全性のような特性を具体化していることの証明や、プログラムが仕様に関して正確であることを証明することができる



形式的仕様記述言語

- 形式的仕様記述言語は計算機システムのモデルを表現する為に使用される形式的言語。
- そのような言語は、抽象化と厳格さを支援する。

General Purpose

VDM-SL

Z

RSL

Act One

Clear

Special Purpose

CCS

CSP

Real-Time Logic

Deontic Logics

形式的仕様記述言語: VDM-SL

- ウィーン開発手法(Vienna Development Method)(VDM)
- 仕様記述言語はVDM-SL
- ISOで標準化されており、完全に形式的
- 支援ツールがそろっている
- 産業界での使用の実績がある
- データと機能の抽象化を支援

講義の概要

- イントロダクション
- 形式的モデルのガイド付きツアー
- 論理
- 基礎的な抽象化記法
- 主要な抽象化記法: 集合、シーケンス、マッピング
- 状態に基づいた記述、および関数的記述スタイル
- 確認（*Validation* ）

講義内容の原則

- フォーマルメソッドは理論的な計算機科学ではなく実際的なシステム工学の一部である!
- 我々の例はすべて、実用の使用環境で開発されていた実際のフォーマルモデルに基づく
- 使用されたモデル化言語の強い特色十分伝えるために例と練習課題を幅広く使用する