プロジェクトマネジメントのための
ソフトウェアの品質向上手法通論

平成２７年度シラバス

２０１５年1月９日

国立情報学研究所

トップエスイープロジェクト

代表者　本位田　真一

１．科目名

プロジェクトマネジメントのためのソフトウェアの品質向上手法通論

２．担当者

古宮 誠一

３．本科目の概要

プロジェクトとは，独自の製品やサービスを創造するために，非定常的な人的組織を組んで行う有期的で非定常的な業務である。この非定常性ゆえにプロジェクトを成功させるには様々な困難が伴う。そのため，プロジェクトを成功へ導くための諸活動が重要であり，そのような諸活動をプロジェクトマネジメントと呼ぶ。

PMBOK（Project Management Of Knowledge）では，プロジェクトを成功へ導くための視点として，①統合マネジメント，②スコープマネジメント，③タイムマネジメント，④コストマネジメント，⑤品質マネジメント，⑥人的資源マネジメント，⑦コミュニケーションマネジメント，⑧リスクマネジメント，⑨調達マネジメント，の９つを挙げている。しかし，**『予定の納期までに，予定以下の開発コストで，予定以上の品質で』という３つの指標を満足させた状態で，顧客と約束した製品やサービスを創造して　顧客に納入できたときに，プロジェクトは成功したと見なされる。プロジェクトマネジメントの中で最もその達成が困難なのは，『ソフトウェアの品質管理』である。しかし，PMBOK はプロジェクトマネジメントの対象を特定していないので，　ソフトウェア開発の品質管理に有効な知識は，ほとんど無いと言ってよい。それ故，この授業科目では，ソフトウェア開発プロジェクトにおいて，ソフトウェアの品質を向上させるために，どのような手段が考えられ，提案されてきたかということを，サイエンスの視点から学習する。**

プロジェクトは，それが持つ非定常性ゆえに，その遂行にはリスクが伴う。ソフトウェアは目に見えないという特性を持つので，プロジェクトの中でも特に，ソフトウェア開発プロジェクトはマネジメントする上で困難が多く，それ故リスクが大きい。 ソフトウェア開発プロジェクトのリスクを低減するためには，中間目標を設定し，設定した中間目標の達成状況をチェックしながら，段階的にプロジェクトを遂行する必要がある。そこで手始めに，１６種類のソフトウェア・ライフサイクルモデルを紹介し，それらはどのような戦略的コンセプトを立ててモデル化しているのか？そのようなコンセプト故に生じるライフサイクルモデルの特徴を明らかにするとともに，プロジェクトの特徴に合わせてそれらを使い分ける方法から明らかにする。以下，**ソフトウェアライフサイクルの工程ごとに，ソフトウェアの品質を向上させるために，どのような手段が考えられ，提案されてきたかということを，サイエンスの視点から学習する**。要求定義工程では，開発するソフトウェアに求められる要求をどのように抽出し，どのように仕様化し，要求仕様の妥当性をどのようにチェックするかを学びます。ソフトウェア設計過程では，どのように効率良く設計し，

設計の正しさをどのように確保するかについて学びます。プログラミングの過程では，ソフトウェアに誤りが混入しないようにするために，プログラミング言語にどのような手段が用意されているか？　それでもどのような危険があり，それに対してどのように対策すべきかについて学習する。ソフトウェアテスト工程やデバッグ工程についても言及する。特に，オブジェクト指向技術を採り上げ，オブジェクト指向技術が持つソフトウェア工学上の問題点を明らかにするとともに，その対策も明らかにする。そして，ソフトウェアの出荷時期を決定する信頼度成長モデルについても言及する。

なお,プロジェクトマネジメントという科目の性質上，他の授業との関係について常に気を配り，それとの関係についてその都度言及する。

４．キーワード

ソフトウェア・ライフサイクルモデル(Software Lifecycle Model)，要求抽出手法(Requirements Elicitation Method)，ラピッド・プロトタイピング(Rapid Prototyping)，議論による要求仕様の妥当性チェック(Requirements Validation Checking by Discussion)，ソフトウェア設計法(Software Design Method)，ソフトウェア設計上の意思決定とその根拠(Software Design Rational)，プログラム理論(Program Theories)，プログラミング言語とデータ型(Programming Language & Data types)，ソフトウェアテスト技術(Software Testing Technique)，オブジェクト指向計算モデル(Object-Oriented Computation Model)，ソフトウェア信頼度成長モデル(Software Reliability Growth Model)

５．前提知識

特になし。

６．講義計画

⑴　第1回

プロジェクトとは何か？プロジェクトの定義を与えるとともに，プロジェクトというものが持つ特徴(非定常的な業務であるということ，など)を明らかにするとともに，プロジェクトが持つ非定常性ゆえに，プロジェクトの遂行にはリスクが伴うのだということを述べる。そして，プロジェクトの中でも，特にソフトウェア開発プロジェクトを対象とするので，ソフトウェア開発固有の特徴ゆえに生じる問題点を明らかにする。しかる後に，プロジェクトが抱えるリスクを低減させるためには，中間目標を設定し，設定した中間目標の達成状況をチェックしながら，段階的にプロジェクトを遂行する必要があることを述べる。このように前置きした上で，１６種類のソフトウェア・ライフサイクルモデルを紹介するとともに，各モデルがどのようなコンセプトの基に設定され，そのコンセプト故に生じる特徴(利点や欠点など)を明らかにする。この中で，ラピッド・プロトタイピングの重要性を明らかにする。しかる後に，どのようにしたら，プロトタイ

ピング支援ツールを作ることができるか，プロトタイピング支援ツールの構築方法の詳細については，別の授業で明らかにすることを述べる。

⑵ 第2回

前回からの続き。１６種類のソフトウェア・ライフサイクルモデルを紹介するとともに，各モデルがどのようなコンセプトの基に設定され，そのコンセプト故に生じる特徴(利点や欠点など)を明らかにする。

⑶ 第3回

ソフトウェア開発プロジェクト固有の問題点である「開発するソフトウェアに求められる要求を抽出することの難しさ」に注目し，要求を抽出する種々の方法を紹介する。

⑷ 第4回

前回からの続き。要求を抽出する種々の方法を紹介する。そして，これらの要求抽出法は５種類に大別できるが，何と言っても有効な方法はインタビュー法(面接法)であることを明らかにする。さらに，抽出された要求を詳細化する方法を明らかにする。しかる後に，経験の乏しい初心者ＳＥでも，要求を漏れなく効率良く抽出できるように，『ＳＥが顧客に対して行うインタビューをシステムが誘導して顧客要求を抽出するとともに，抽出した要求を基に，要求仕様書を自動生成するシステム』が試作されていることを明らかにする。そして，どうすればそのようなシステムを構築できるのか，その実現方法の詳細については，別の授業で行うことを明らかにする。

⑸ 第5回

得られた要求仕様の妥当性やその解釈を巡って，プロジェクトの主なメンバー(経験者を含む)が参加して，様々な視点から議論してチェックする必要があることを強調する。こうすることにより，要求仕様の抽出漏れや誤り・矛盾・曖昧性などの欠陥を事前に検出することができることを指摘する。そして，このときの議論の内容をプロジェクトのメンバー全員(議論に加わらなかった人を含む)で共有することが重要であることを強調する。このときの議論の内容をプロジェクトのメンバー全員で共有するには，得られた要求仕様の妥当性やその解釈を巡る議論をネットワーク上で行い，その時の議論の発言内容を漏らさず記録するために，即時記録するツールが必要であることを強調する。このようなツールは，XML データベースをベースにした Web アプリケーションシステム(プライベート・クラウドシステム)として，容易に開発できる。議論の発言内容を即時記録するための，情報の記録形式のモデルについて議論する。

議論だけでは検出できない要求仕様の欠陥を検出するには，モデル・チェッキングが必要であることを述べる。そして，モデル・チェッキング技術については，モデル検査技術の授業を受講することを勧める。

⑹ 第6回

得られた要求仕様の妥当性やその解釈を巡る議論を即時記録するためのモデルがどのようなものになるかをグループ演習する。

(7) 第7回

信頼性の高いソフトウェアを開発するためには，プログラムの処理アルゴリズムが，それを解読する人にも理解し易い構造になっていなければならない。そのような構造のソフトウェアを実現するために，Dijkstra はプログラムを記述する際に GOTO 文の濫用を制限することを提唱するとともに，プログラムの制御構造を Sequence(逐次処理)，Selection(選択処理)，Iteration(繰り返し処理)の３つに限定する構造化プログラミングの概念を提案した。構造化プログラミングの概念に適合するプログラムを効率良く作成する方法として，種々のソフトウェア設計法が提案された。その１つである Wernier 法の考え方と設計手順を紹介して，その効用を PR する。しかし，ソフトウェア設計法は効率的，かつ，効果的であるが，開発対象となるソフトウェアによって，向き・不向き(相性)がある。このため，複数のソフトウェア設計法に練達し，開発対象となるソフトウェアによって使い分ける必要があることを強調する。ソフトウェア設計法については，別の授業「ソフトウェア設計法通論」を受講することを勧める。

(8) 第8回

ソフトウェア設計作業は，ソフトウェアの品質を作り込む作業である。このため，関係するプロジェクト・メンバーとの間で，設計案とそれを提案した根拠(Design Rational)についてネットワークを介してレビュー(議論)しながら進めると効果的である。そのための議論を即時記録するための支援ツールで用いられる，発言内容を記録するための種々のモデルを紹介する。なお，より詳しくは別の授業を受講することを勧める。

アジャイル開発宣言の見るアジャイル開発が意図と意義，アジャイル開発が成立する条件について考える。また，派生開発の現状とその問題点に触れ，その対策について考える。

(9) 第9回

プログラムの正しさを証明しながらソフトウェアを開発する方法として，形式仕様に基づくソフトウェア開発法がある。形式仕様に基づくソフトウェア開発法の適用において，プログラムの正しさを証明するための理論として Hoare の公理的意味論がある。また，自動プログラミングの本質はプログラム変換だと言われるが，変換前のプログラム・フラグメント(プログラム・ソースコードの断片)と変換後のプログラム・フラグメントとが等価である(= 変換してもよい)ことを証明するための理論として Scott の表示的意味論がある。これらはプログラム理論と呼ばれている。

(10) 第10回

プログラム理論の概要を簡単に紹介する。しかる後に，Hoare の公理的意味論について講義するとともに，これに基づいてプログラムの正しさを証明する方法を紹介する。また，Hoare の公理的意味論に基づき，プログラムの正しさを証明しながら，プログラムを開発する David Gries の手法いついても紹介する。

(11)第11回

ソフトウェアの開発効率を向上させる上で，これまでで最も貢献した技術はプログラム・コンパイラの発明である。プログラミング・パラダイムの視点からプログラミング言語を分類するとともに，各分類に属する主なプログラミング言語を列挙する。しかる後に，これらのプログラミング言語が，プログラムに混入する誤りを検出する技術としてデータ型の利用がある。そもそも何のためにデータ型があるのかの説明に始まり，データ型を利用することによって，どのようなことができるのかについて講義する。

(12) 第12回

プログラムテスト法について講義する。その中でも同値分割，限界値分割，原因－結果グラフ(その用い方を含む)などのブラックボックス・アプローチによる方法について学ぶ。ホワイトボックス・アプローチによるプログラムテスト法について講義する。テストの網羅基準について学ぶ。スタブやドライバを用いた単体テスト，結合テスト，トップダウンテスト，ボトムアップテスト，サンドウィッチテスト，ビッグバンテストなどについても学ぶ。ソフトウェア・テスティングについて，より詳しくは他の授業「ソフトウェアテスト技術」の受講を勧める。

(13) 第13回

前回の続き。HAYST 法について紹介する。また，Back to Back テストについても触れる。しかる後に，プログラムのデバッグ方法についても言及する。オブジェクト指向計算モデルについて講義する。オブジェクトとは何か？オブジェクト指向計算モデルを巡る様々な概念を列挙するとともに，オブジェクト指向計算モデルを巡る様々な立場を紹介する。オブジェクト指向計算モデルを支える重要な仕組みである「継承」の概念と解釈を採り上げる。その中で，単一継承と多重継承が抱えるそれぞれの問題点を議論する。計算モデルがオブジェクト指向言語と類似する言語として，フレーム型知識表現言語がある。両者の間の本質的な相違についても言及する。ソフトウェア工学から見た，オブジェクト指向によるソフトウェア開発の問題点を列挙する。オブジェクト指向プログラミング言語が持つ問題点，オブジェクト指向ソフトウェア開発法が持つ問題点。しかる後に，これらの解決方法について議論する。

(14) 第14回

プログラムの信頼性を評価する技術については，別の講義「メトリククス」の受講を進める。プログラムの出荷時を決定する技術として，ソフトウェア信頼度成長モデルを採り上げ，そのモデルを列挙して，それらの考え方を紹介する。

(15) 第15回

ソフトウェア開発計画(要員割り当てを含む)を自動立案するツールの効用について紹介し，そのようなツールを手にすることの重要性について議論する。どのようにすれば，そのようなツールを開発できるかについては，別の講義を受講することを勧める。

７．評価

出席日数とレポート課題への取り組み状況を総合して評価する。

８．参考文献

[1] Bernard H. Boar, “Application Prototyping: A Requirements Definition Strategy for the 80s,” John Wiley & Sons, Inc. 1984. (邦訳) Bernard H. Boar 著，前川守，伊藤潔監訳, “プロトタイピングの新応用技術と導入法,” 日本技術経済センター, Dec. 1984.

[2] J. Craig Cleveland, “An Introduction to Data Types,” Addison-Wesley Publishing Company, Inc., 1986. (邦訳)J. C. Cleveland 著，小林光夫訳，共立出版, April !990.

[3] J. Conklin, and M. L. Begeman, “gIBIS: A Hypertext Tool for Exploratory Policy Discussion,” CSCW '88 Proceedings, ACM, pp.140-152, 1988.

[4] 淵一博，黒川利明編著, “新世代プログラミング,” 共立出版, Feb. 1986.

[5] 星野匡, “発想法入門,” 日経文庫，日本経済新聞社, March 1989.

[6] 木村泉，米澤明憲, “算法表現論,” 岩波講座－12，岩波書店, May 1982.

[7] Seiichi Komiya, “A Model for Recording Software Design Decisions and Design Rationale, “IEICE transactions on Information and Systems, Vol. E81-D, No. 12, pp. 1350-1363, 1998.

[8] W. Kunz, et al., “Issues as Elements of Information Systems,” No. Working Paper #131, University of California Berkeley, 1970.

[9] 桑名栄二, “ソフトウェア履歴利用の研究動向, “電子情報通信学会, Vol.77, No.5, pp.531-538, 1994.

[10] J. Lee, “SIBYL: A Qualitative Decision Management System,” in Artificial Intelligence at MIT, P. H. Winston, and S. A. Shellard, EDs. pp. 104-133, MIT Press, 1990.

[11] J. Lee, “SIBYL, “A Tool for Managing Group Decision Rationale,” CSCW '90 Proceedings, ACM pp.79-92, 1990.

[12]J. Lee, “Extending the Potts and Bruns Model for Recording Design Rationale, “Proceedings of the 13th International Conference on Software Engineering, IEEE, pp.114-125, 1991.

[13]A. Maclean, R. M. Young, and T. P. Moran, “Design Rationale: The Argument behind the artifact, “In proceedings of CHI'89 Human Factors in Computing System, pp.247-252, 1989.

[14] A. Maclean, R. M. Young, V. M. E. Bellotti, and T. P. Moran, “Question, Options, and Criteria: Elements of design space analysis, Human Computer Interaction, 1991.

[15] 大西淳，郷健太郎，“要求工学,” ソフトウェアテクノロジー・シリーズ９，共立出版，May

2002.

[16] C. Potts and G. Bruns, “Recording the Reasons for Design Decisions, “Procee dings of the 10th International Conference on Software Engineering, IEEE, p p. 418-427, 1988.

[17] C. Potts, “A Generic Model for Representing Design Methods, “Proc. of the 11th International Conference on Software Engineering, IEEE, pp. 217-226, 1989.Charles H. Kepner, Benjamin B. Tregoe, “The New Rational Manager,” Prinston Research Press, Prinston, New Jersey, 1981.（邦訳: C.H.ケプナー, B.B.トリゴー著, 上野一郎監訳, “新・管理者の判断力, 産能大学出版部刊, Feb.28, 1985.

[18] 山田茂, “ソフトウェア信頼性評価技術――ソフトウェア信頼性評価技術――,” HBJ 出版局, May 1989.

[19] 山田茂,大寺浩志, “ソフトウェア信頼性～理論と実践的応用～,“ 株式会社ソフト・リサーチ・センター, May 1990.

[20] 山本修一郎, “ゴール指向によるシステム要求管理技法,” ソフト 2007.・リサーチ・センター, May 2007.

[21]米澤明憲, “オブジェクト指向プログラミングについて,” コンピュータソフトウェア, Vol.1, No.1, pp. 29-41, April 1984.