ソフトウェア工学 102-27
（1995．1.27）

教示的生産管理システムの提案

～人，上司，作業マニュアルを含めた生産システム～

君島 浩
㈱富士通ラーニングメディア

ソフトウェア開発は人間が中心になる活動である．ソフトウェア技術者への指導（instruction)による生産管理が大切である．人，上司，作業マニュアルを含めた生産システムの枠組みを提案する．この枠組みは過去の品質問題や生産性問題を解決し，ソフトウェアパッケージやクライアントサーバシステムなどの新しい商品の開発を容易にする．

A Proposal of Instructional Production Control System

－ Production System with human, manager, and work manuals －

Hiroshi Kimijima
FUJITSU Learning Media Limited

Software development is human-centered activity. The production control with instruction to software engineers is important. This paper proposes the concept of new production system which contains human, manager, and work manuals. This concept solves the past problems on quality and productivity and it eases new software packages and client/server systems development.

1．はじめに

従来のソフトウェア開発の問題点は，教育的な観点で見ると①ＯｆｆＪＴとしては作業マニュアルを基準にした作業や反省がされていない，②ＯＪＴとしては上司による日程管理や指導がなされていない，③自己啓発としては学会活動や文献調査が不足している，の三点に要約できる．従来のソフトウェア生産システムのとらえ方は，工程段階をどのように中間生産物が流れていくかという図式であった．ソフトウェア技術者を指導する作業マニュアルや上司の存在が抜けていた．

本稿はソフトウェア技術者に対する作業マニュアルや上司の指導を教示（instruction,指示ともいう）として考え，教示の存在を明確に含めた生産管理システムを提案する．これはスーパーマーケットなどのほかの業種や米国のホワイトカラーの生産管理を参考にしたものである．この枠組みは過去の問題点を解決するとともに，ソフトウェアパッケージやクライアントサーバシステムなど，今後強化すべき取組にも貢献する．

2．教示的生産管理システムの概要

教示的生産管理システム（instructional production control system ）とは，作業マニュアルや上司によるソフトウェア技術者の指導の存在を明確に位置づけた生産管理システムである．この方式はスーパーマーケットや米国では定着しており，それをソフトウェア開発に適用してみた（図１）．

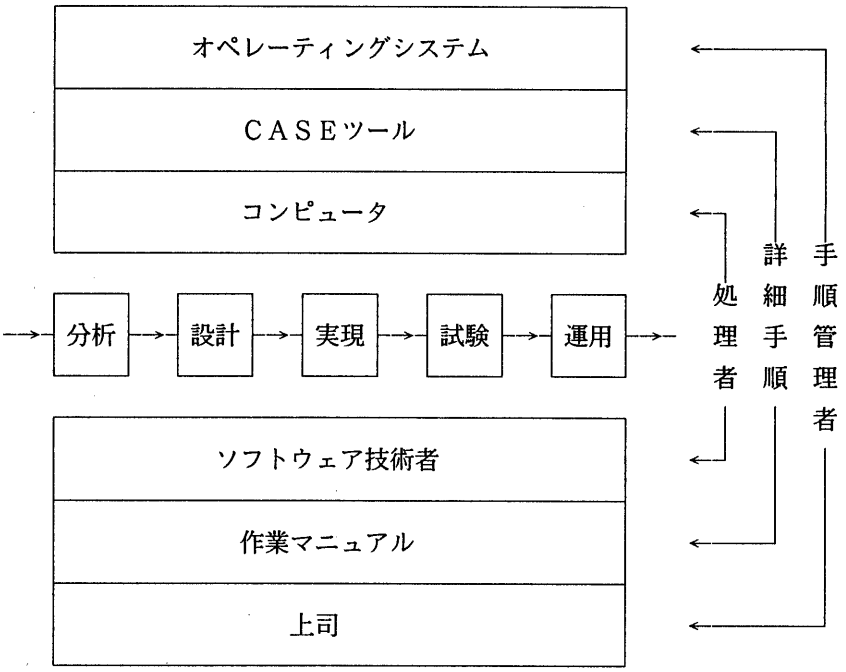

図１　教示的生産管理システム

ソフトウェア開発という使命（ミッション）の生産活動は，ソフトウェアライフサイクルプロセスSLCPを参考にすれば，工程段階（フェーズ），任務（タスク），ルーチン，ステップというように細分される．これをメニューにして，作業手順を設計し遂行する．その管理は階層の深さによって変えるべきである．

段階レベルの作業手順は，ウォータフォールやプロトタイピングなどの雛形工程を参考にして，プ

ロジェクトごとに設計することが大切である．ウォータフォールなどの雛形をワンパターンで適用してはいけない．任務レベルの作業手順は，計画の時には決めずに，数日単位の日程管理（スケジューリング）で場当たり的に決めるべきである．これは米国では常識であり，オペレーティングシステムもこの場当たり的な管理（タスク管理）を踏襲している．しかし，ルーチン・ステップのレベルを上司が指導するのは手間がかかるので，あらかじめ作業マニュアルに書いておいてそれを参考にする．このようにすれば，作業手順決定という納期短縮の重要な事項を，ソフトウェア技術者の判断にまかせることなく，経験のある冷静な上司が制御することができる．

3．規則による命令と作業マニュアルの教示の区別

やるべきこと，やってはいけないことを規則といい，規則に従って部下を制御することを命令という．例えば，プログラムを変更するときに設計書を更新して保守に役立つようにすることは規則であり，命令で強制すべきことである．白黒がはっきりしているからだ．

作業マニュアルとは過去の作業方法を定義した参考文献であり，規則ではない．作業マニュアルの内容を教えたり，それから逸脱する新しい方法を指示したりすることを教示という．作業マニュアルを教室で教える講師の活動は教示であり，職場で上司が指導するＯＪＴも教示である．教示という言葉は，一応最初に提示する方が先達であること，相互の会話によって変更が可能なこと，臨時の通達であることの三つのニュアンスを持っている．例えば，作業マニュアルの年度改訂までの間，変更事項を書いた通達を指示書（instruction)という．

作業マニュアルから逸脱したことが成功すると，それは次の改訂に反映する．作業マニュアルに従ったために失敗したことは，再発防止策を次の改訂に反映する．これによって作業マニュアルは信頼できる参考文献として進歩していく．

規則と作業マニュアルの違いは日本では知られていない．例えば，スカイラークは作業マニュアルによる硬直化を反省して，最低限守るべきことを残して作業マニュアルを簡素化した．これは作業マニュアルを規則として誤解していることであり，参考文献としての作業マニュアルを捨てたことになる．これではきめこまかな教示や品質向上はできなくなる．

4．個別品質管理による障害の絶滅

ソフトウェアの障害のデバッグに統計を使わないのが当然のように，根本対策にも統計を使わないのが当然である．このことがソフトウェアの分野では知られていない．ハードウェアの欠陥設計に対しては統計的品質管理は使わない．欠陥設計が発生すると設計作業マニュアルを改訂する．

ソフトウェアの障害が１件でも発生すると，レビューやテストがもれていたのは明らかである．既存の設計マニュアル，レビューマニュアル，テストマニュアルを守ったために障害を見逃したのである．百件ほど障害が出るのを待って統計を取ったのでは，根本原因は不明になる．そこで例えば，その障害を開発時に発見するための記事をテストマニュアルに書き加える．テストマニュアルとは主にテスト項目選択基準を箇条書きなどで定義した文書であり，それに箇条書きの新しい条項を書き加えるのである．ソフトウェアのテストは抜き取りテストなので，作業マニュアルがないと個人差が発生する．作業マニュアルによって個人差をなくし，作業マニュアルの改訂によってノウハウを蓄積して，後世に伝えるのである．こうしない限り品質は向上しない．

従来のソフトウェアの統計的品質管理は，前に述べたような生産管理の定石や設計品質管理の定石からはずれたものであった．筆者のような教育担当者がまとはずれな教育をしてきたからだ．

5．日程管理とＯＪＴと人事管理

従来のソフトウェア工学では，進捗管理とは「進捗度を測定して遅れていたら対策を取ること」と教えていた．これは生産管理の常識からいうと間違っている．生産管理における日程管理とは「納期を短縮できるように運用中に作業手順を指示すること」である．作業手順を教示することが中心なのである．日程管理（スケジューリング）は次のように進める．

①１～３日単位ごとに日程管理会議兼レビュー会議を開く（製造工場なら現場で行う）．
②部下は生産物を説明する．
③上司はまず作業速度を評価し，まずければどこがまずかったかを指導する（事後ＯＪＴ）．
④上司は次に生産物のできばえを評価し，まずければ改善するよう指導する（事後ＯＪＴ）．
⑤上司はチーム全体の進捗度を計算する（部下に進捗報告させるのではない）．
⑥上司はSLCPや商品体系をメニューにして，次に行うべき作業項目を洗い出す（部下と相談）．
⑦上司は洗い出した作業項目の中から，すぐに行う優先項目を選抜する（部下と相談）．
⑧上司は部下の教育受講，学会参加，会議出席，出張，病気，休暇などの都合を確認する．
⑨上司はそれぞれの作業項目に対して担当者を割り振る（担当者に作業を割り振るのではない）．
⑩上司は担当者に対して必要なノウハウを指導する（事前ＯＪＴ）．必要なら作業マニュアルから逸脱する作業手順を指導する．
⑪作業マニュアルから逸脱して成功したことを作業マニュアル改訂提案書として出させる．
⑫作業マニュアルに従って失敗したことを作業マニュアル改訂提案書として出させる．

このようにして，任務（タスク）は進捗中に順序を決める．タスクがどのぐらいの日数で終わるかは，計画段階には予測できないからである．計画段階に「人間に対して作業群を割り振る」と人間がさぼったり，病欠者が出たときに困ったりする．

このようにしてレビューすれば，ＯＪＴをしたことになる．また上司は部下の業績（速さとできばえ）を自分の目で見ているので，公正な人事査定ができる．ＯＪＴや実力給制度の成功の鍵は，年度単位の計画制度や面談制度ではなく，日々の日程管理なのである．この日程管理が，作業マニュアルを基礎にした指導や逸脱の指導の中心になる．ソフトウェア技術者の素人仕事をなくす要点である．

クライアントサーバシステムの短期開発のときには特に，上司による臨機応変なスケジューリングが必要である．計画時にウォータフォール型からプロトタイピング型に変更するだけでは済まない．

6．教育の共通基礎としての生産管理論

生産管理は社会人の共通素養である．しかもコンピュータには密接な関係がある．

①新入社員へのオリエンテーションは，会社とは何か，仕事とは何かを厳密に定義する生産システムの教育を中心にするべきである．
②ソフトウェア開発は生産活動の一つであり，その教育には生産管理が前提知識になる．
③プログラムとは作業マニュアルをモデルにしたものであり，生産管理が前提知識になる．
④オペレーティングシステムとは上司の生産管理活動をモデルにしたものであり，生産管理が前提知識になる．
⑤システム分析とは顧客の生産活動を分析することであり，生産管理が前提知識になる．

このような考えを元にして教育体系を変更すれば，教育負担が減って教育効果が向上する．これらの教育において重要なのは，機能（作業の役割）や性能（作業の速度）を話題の中心にすることである．コンピュータに慣れた教育者は，コンピュータが速いということを忘れがちであるが，初心者に

はコンピュータを使う理由として性能の話から始めるべきである.

7．人間と機械の協業システムの設計・運用

ソフトウェア開発は人間と機械とが協業する典型的なクライアントサーバシステムである．この場合には，人間の性能（作業速度）が先行し，機械化が迅速に追随することが大切である．一方だけを改善してもシステムとしての性能は上がらない．そこで人間・機械を区別せずに生産システムを設計することが出発点である.

ソフトウェア開発という作業群を，段階・任務・ルーチンに分解する．それから適材適所で人間と機械に作業を割り振って，人間のためには雛形工程・雛形日程・作業マニュアルを教育エンジニアが開発する．機械のためにはシェルスクリプト・タスク構造・応用プログラム（ＣＡＳＥ）を情報システムエンジニアが開発する．人間と機械の会話の部分は次のような階層と見て，状態遷移図などで相互関係と視覚的構造（画面デザイン）を設計する.

作業マニュアル	作業	応用プログラム（ＣＡＳＥ）
章	段階	画面（応用プログラム）
節	任務	ウィンドウ（タスク）
段落	ルーチン	パラグラフ（図・写真・箇条書きなどの単位）

ソフトウェア技術者を教育するには，ＣＡＳＥの操作方法を教えることよりも，人間側の作業マニュアルの内容を主体に教えることが大切である．事務計算のような機械中心のシステムに対して会話型の人間・機械システムでは，人間の方の作業の遅さがネックになりやすい．ワードプロセッサを導入しても，文章技術やタッチタイピングができない人には文章を速く書けないのがその例である．人間と機械の速度向上をバランスを見ながら推進するにはこの方法が有効である.

この考え方は，顧客のクライアントサーバシステムを開発するときにも適用できる．画面設計は人間の作業分析結果を含めて設計すること，コンピュータシステムだけでなく人間側の作業マニュアルも開発して教育することが大切だ.

作業マニュアルの電子化や教材のＣＡＩ化が進むと，それを職場のワークステーションで使えるようになる．実務ツールと教育ツールが一体化したワークステーションシステムを，実務学習支援システム（PLSS: performance/learning support system)という．今後は実務ツールと学習ツールの一体化が進むので，以上のように人間と機械を一緒に分析してシステム化する作業方法は重要性が増すであろう.

8．商品管理

ソフトウェア工学はソフトウェア生産の一般論である．ソフトウェアの発明・開発は，個別の定石を蓄積した情報工学や情報システム学に依存する．しかし，情報工学をソフトウェア技術者が習得し実践しているかどうかは管理しにくい．そこで時間軸に沿った横軸の生産管理に対して，縦軸の商品管理という管理方法を交差させる.

商品管理とは，社会，会社，使命，職務，段階，任務，ルーチンという階層としてソフトウェア商品群を眺めて，具体的な商品名と上下の利用関係だけを図やデータベースで把握することである．例えば，Ａ運輸，Ｂ運輸，Ｃ運輸が共通の流通パッケージを使っていることから出発して，応用プログラム，ミドルウエア，アルゴリズム，共通部品，システム関数の利用状況を眺める．これらを参考にして，共通でなかったことへの共通商品を発明したり，アルゴリズムや部品を再利用したりするので

ある．この商品管理という制御をすることによって，情報工学・情報システム学の習得・実践を確認するのである．これと論文や特許の管理を連携させると更によいであろう．

こういったことを励行しないと，ソフトウェアパッケージやミドルウェアや部品ライブラリでは米国に勝てないことになる．米国では3社から共通の応用システムを受託するといった複数顧客プロジェクトの経験を積み重ねて，共通点を見抜く能力を増強してきた．現在，米国では流通システム・金融システム・生産管理システムなどを連結するメガプログラミングや，企業間の納品の流れを電子化する電子調達システムCALSという壮大な事業が進んでいる．システム関数の把握，共通モジュールの見つけ出しという小さなレベルから教えて，会社を越えるパッケージを発明できるよう指導したい．

１０．まとめ

作業マニュアルや上司による教示を含めた教示的生産管理システムという枠組みを提案した．この方法は米国やほかの業種で実用されている．ソフトウェア開発の分野に導入すれば，品質や生産性などの過去の問題の解決と，今後の発展への貢献を次のように進めるだろう．

①作業マニュアルの改訂を軸にした個別品質管理による品質問題解決
②日程管理による生産性問題解決とプロトタイピング工程時代への対応
③生産管理の共通教育による教育の合理化
④人間・機械協業システムの設計による画面デザイン・教育推進
⑤商品管理による情報工学・情報システム学の職場管理

＜参考文献＞

1) 森五郎，「人事・労務管理の知識」，日経文庫，1968.
2) 山田雄一，「社内教育入門　13版」，日経文庫，1976.
3) 篠山勲，「ソフトウェア開発のプロジェクト管理」，日刊工業新聞社，1989.
4) 君島浩，「障害１件だけで根本原因を究明する方法」，ソフトウェア信頼性シンポジウム，1991.
5) G.Gery，「Electronic Performance Support Systems」，Weingarten，1991.
6) P.Wegner，「Toward Mega programming 」，CACM，Vol.35，No.11，1991.
7) 君島浩（共著），「日本的デザインレビューの実際」，日科技連，1993.
8) 君島浩，「社内教育システム開発方法論」，日科技連，1993.
9) 販売革新別冊，「スーパーマーケットオペレーション教本」，商業界，1993.
10) 人見勝人，「入門編　生産システム工学（新訂版）」，共立出版，1993.
11) 山本恭逸，「ソフトウェア産業人事制度」，コンピュータエージ社，1993.
12) R.ワーマン，松岡正剛訳，「理解の秘密：マジカル・インストラクション」，ＮＴＴ出版，1993.
13) 大野豊監修，「システム開発取引の共通フレーム SLCP-JCF94 」，通産資料調査会，1994.
14) 君島浩，「ＵＮＩＸ実務の学び方」，ソフトリサーチセンター，1994.
15) 澤田善次郎，「ＣＩＭと経営管理」，日刊工業新聞社，1994.
16) 島田達巳，「情報システムマネジメント」，日科技連，1994.
17) 中央情報教育研究所，「教育エンジニア育成カリキュラム」，中央情報教育研究所，1994.
18) 吉岡誠編著，「ＳＧＭＬのススメ」，オーム社，1994.
19) 「変革期のソフトウェア工学シンポジウム論文集」，情報処理学会，1994.