マルチベンダー環境におけるマネジメント事例 ――課題と対応の骨子

Management of System Development Process in a Multi-vender Environment ――Challenge Facing Management and Its Overcome

中 村 祥 次 郎

要 約 システム開発を取り巻く環境は，情報技術（IT）の進歩により大きく変貌し，従来困難とされていた分野に対しもシステム化が可能となりつつある．今日におけるシステム化ニーズに応えるための開発マネジメントのあり方，及びマネジメントに求められる内容も変わってきている．

例えば，マルチベンダーを前提としたシステム統合及び品質保証の考え方，更には，大規模化に対する作業の可視性確保・統制が従来にも増して重要となってきている．

これらマネジメント課題に対し筆者は，開発計画段階でのシナリオが開発プロジェクトの成否を大きく左右すると考えており，如何なるマネジメント骨子を持って行動したかを述べるとともに，実施段階でどの様な課題に直面しそれを克服してきたかについて，事例を基に本稿で報告する．

Abstract The environment surrounding the system development is changing remarkably with continuous advances in the information technology (IT), which is enabling to computerize the field of application that was conventionally considered as difficult. Circumstances mentioned adove have also changed the development management practice to respond to the today's needs for the computerization,and the contents for which the management is required to make offers.

For example, the view point of the system integration and the quality assurance based on the presumption of the multi-vendor system, and the ensuring and controlling of the visibility of the progress of the very large-scale project, have become important also compared with the former.

This paper discusses that the scenario in a system development planning phase influences the success or failure of a development project to great challenge facing the management,and the spirit of the management underlying the project management performed, and the challenges faced in the implementation phase and how these challenges are overcome,on the basis of an example.

1. はじめに

パソコン（PC）及びコンピュータネットワークの飛躍的な性能向上により，従来システム化が困難とされてきた情報の二次加工・非定型事務等の人間系，さらには高度な情報分析・各種シミュレーション等へとシステム化分野が拡大し，オープンC/SS技術によるシステムは大規模化・高度化してきており，業務要件整理やシステム概念設計に時間とスキルが従来にも増して必要となってきている．

一方，オープンC/SS系の従来のシステム開発においては，比較的小規模なシステムを小集団で開発する事を前提とした開発マネジメント（開発方法論）が実施されてきたが，今日ではマルチベンダーを前提とした技術の評価・統合・調整と言ったマネ

ジメント要素が不可欠となっている．

オープンマルチベンダー環境における従来の開発事例においても，技術（プロダクト）の統合や性能評価の不備が実装段階で表面化したり，業務要件整理に予想以上の手間と時間を要した等の問題を引き起こし，マネジメント上多くの課題を残している．

本稿では，オープンマルチベンダー環境における効率的で高品質な開発を安定的に維持するためのプロジェクトマネジメントについて，事例をもとに考え方・方針を述べる．

2. プロジェクトマネジメントの環境と事前準備

2.1 社会環境（オープン・マルチベンダー）

筆者等コンピュータシステムの開発に携わる者を取り巻く環境は，汎用機が主流であった時代からオープン環境に大きく変化し，確実な足取りでシステム構築の主流と成りつつある．更に，今日においては，特定のベンダーに依存せず，最適なプロダクトの組み合わせによりシステムを構成する方向にあり，マルチベンダー化が定着しつつある．

つまり，単一ベンダーだけでは，求められるシステム機能の実現から運用管理に至る必要プロダクトの調達および意思決定は難しく，ベンダー間の調整など開発に当たっての組織そのものの見直しが必要になってきている．

したがって，開発業務を遂行するプロジェクトマネジャは，多くの他組織とチャネルを持たざるを得ず，多様な角度・手段で折衝を行うことから予期せぬリスクを潜在的に抱えることになるため，プロジェクト全体の透明性と充分なコミュニケーションがプロジェクト運営上欠かせない条件となる．

そこでは開発標準（開発手順・成果物）・設計基準・規約等の開発指針が透明性維持の柱であり，コミュニケーションを行う上で必須のルールとなる．

2.2 コンピュータ技術（Case ツール）の適用

開発生産性の向上，一定の品質保証，保守性の向上を実現するためには，従前の標準・基準だけでは難しく，ある範囲で一連の開発作業を物理的に統制・規制する必要がある．

そのための具体策として DB 設計・入出力設計・プロセス設計等，これらを一体化した Case ツールの適用が生産性・品質保障の裏付けとなる．

2.3 システム化対象業務

従来，人手に依存していた少量多種・非定型・判断といった分野までがシステム化の対象となり，顧客においても過去に経験したことのない業務のシステム化が要求されるに至り，特に，上流工程での業務要件取りまとめ等，コンピュータ技術の専門知識を備えた IT コンサルティング無しではシステム化要件の取り纏めが難しい状況にある．

つまり，最新技術を導入し，顧客の求めるシステム化構想を具現化するためには，現在の技術水準で，どこまで，何ができるか，製造工程に繋がる具体的な業務要件書をどのように定義すればよいか等の技術・ノウハウが不可欠となっている．

2.4　組織と役割定義

コンピュータの適用分野が拡大するにつれ，アプリケーション・システム（AP）開発部隊そのものも従来の切り口には納まらない役割（ノウハウ）の集合体により形成されることになる（図1）.

また，プロジェクトが必要とする組織としての機能は工程の推移とともに変化し，求められるノウハウ，スキルに対し要員面で柔軟に対応しなければならないが，他方で，組織は開発工程を通じて深く・広くなる業務知識を前提としており，開発過程での要員の育成が重要となる.

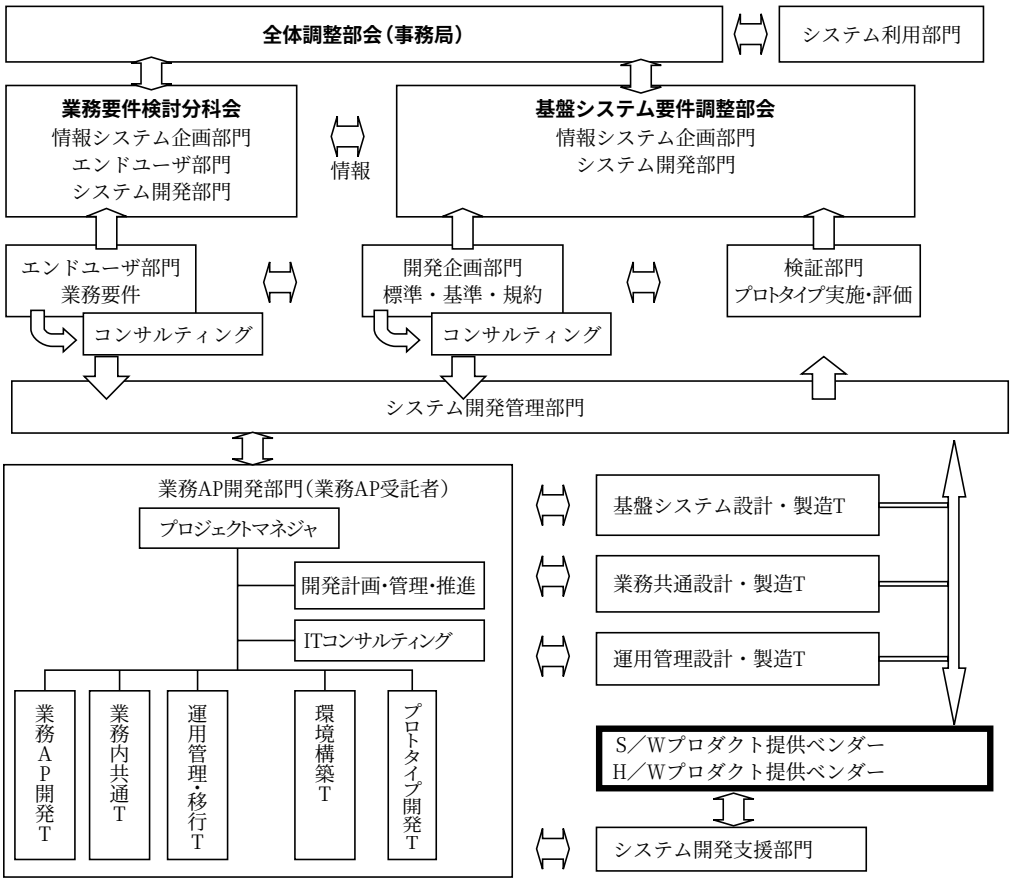

図 1　システム開発体制

2.4.1　会議体の概要（図1）

1）　全体調整部会

システム利用部門と開発部門の合意形成及び各分科会間のシステム要件調整を主たる目的とした会議体である．本会議体は，工程の推移とともに部門間の進捗調整会議へと付議事項が遷移する.

2）　分科会（各業務要件分科会，制御基盤分科会）

業務要件分科会：エンドユーザ部門が取りまとめた業務要件に対し，関連部門

の承認を得る事を主たる目的とした会議体．本会議体は，工程の推移とともに進捗調整および業務課題解決方針の承認会議へと付議事項が遷移する．

制御基盤分科会：基盤システムとして提供される環境・機能に対し業務 AP 開発部門の合意・承認を得るための会議体．本会議体は，工程の推移とともに業務 AP 開発部門に対する設計方針の指導および課題解決に向けた設計方針決定を目的とした会議体へ移行される．

2.4.2 エンドユーザ部門（コンサルティングを含む）

〈役割〉

1） 業務要求定義及び利用部門との調整

利用部門からのニーズを整理し開発部門に対する業務要件としてとりまとめる．また，概算予算枠と開発見積予算の間に乖離が生じた場合は，開発予算削減に向けた業務要件の利用部門との調整を行いシステム化範囲を確定する．

2） システム化方法（実現方式）の承認

システムの利便性及び事務効率（システムの目的達成度）について，利用者部門の立場に立った実現方式の確認を行う．

3） 役割分担

ユーザ　　：利用部門からの要求に基づき業務要求を定義する．

コンサルティング：制御基盤として提供されるであろう技術水準並びに開発標準を前提にシステム化要件を定義する．

注）システム利用部門であるエンドユーザ部門の役割・責任を明らかにすることで，上流工程作業の遅延（仕様確定遅延・仕様変更）が開発コスト増大，マスタースケジュールの見直しになる事を確認し合う事が求められる．

2.4.3 開発企画部門（システム構成企画・標準・基準・規約の制定）

〈役割〉

各部門で行われるマネジメントの指針を具体的に定義することが主たる役割であり，立法府的位置付けである．

1） 役割分担

ユーザ　　：各ベンダーからの提案に基づくシステム構成・開発環境の企画，及び開発標準・基準・規約の定義を行う．

コンサルティング：業務要件実現に向けて必要となるシステム基盤機能の評価，及び導入する Case ツールを前提とした開発標準・基準・規約の提案を行う．

2.4.4 開発管理部門

〈役割〉

開発計画全体の推進を目的とした全部門の統制（部門間調整）及び開発リソース管理が主たる役割となる．全部門の総意としての合意は全体調整会議の場となる．

1） 部門間の開発範囲の調整

マルチベンダー体制による開発では部門間の開発範囲がしばしば問題になり，

担当部門間の行司役として調整に当る.

2) 開発予算の見積（規模・生産性・単価）

エンドユーザ部門と開発部門間での開発予算調整を行い確定する.

3) 他部門進捗の調整

全開発プロジェクトの進捗を管理し，開発・テスト工程の円滑な推進を図る.

4) 品質保証

品質を保証するためには，各工程でどのようなテスト・検証・レビューを行えば当該サブシステムの品質を保証できるかの理論（理屈）を構築することが最も重要となる．実際の理論構築は開発部門で行うことになるので，管理部門としては品質保証の理論的検証・承認を行うことなる.

また，開発の実践に於いて開発標準に則った作業が行われている（行わざるをえない）事の理論的証明になり得る検証ポイントを明確にする.

5) 役割分担

① 検証部門

本格開発に先立ちプロトタイプによる基盤プロダクトの作動確認及び設計方針・開発標準・規約の先行確認と微調整を行う．開発工程期間は主にデザインレビューを実施し，作業規範の確認を行う．テストフェーズ以降は主にシステム性能予想・試験実施・評価等を行う.

② システム開発管理部門

本プロジェクトでは業務要件の詰めと並行してシステム開発を進める手法を取っている事から進捗を単純に捉えることは難しく，仕様の確定度とシステム化範囲の変化に対する予算管理が主な役割となる.

2.4.5 開 発 部 門

〈役割〉

言うまでもなく成果物の開発と一定の品質確保が役割となるが，重要なことは部門内のファンクションが明確化されていることを前提に，各々の立場(発注者と受託者)でのマネジメントが充分に活性化しプロジェクトの弱点を補い合う体制を築くことにある.

1) 業務要件定義

システム化要求定義に基づく開発標準に則った業務要件定義を行うことになる.

主たる作業は入出力設計，概念 DB 設計，プロセス分割とパターン分析，ファイル・プロセス相関の整理である.

2) システム規模の見積

見積理論の構築：一般的な見積技法にとらわれない，受発注者間の合意が可能な客観的な見積理論の構築が必要である.

成果物の定義　：仕様変更を含め開発過程で生産される成果物の種類と深さを明確にする.

3) システム開発

開発過程で発生する以下のマネジメント課題の方針を明確にし,開発を進める.

・工程の進度により発生する仕様変更の取り扱い
・業務課題解消に向けた要員・予算の計画
・一括受託作業とそれ以外の作業の切り分け等

4） 品質保証

設計工程・開発工程・テスト工程の各工程における品質保証の理論を明確にし，管理部門との間で合意する．

品質を保証するために，設計レビューを何回以上実施したとか，莫大なテストケースを消化したからといってシステムの品質を保証するに足りうるかの証明にはならない．つまり，保障に足りうる絶対値としての座標軸が必須であり，その座標軸との同一性を証明する以外にシステムの品質を保証する手段は無く，そのための方法論を定義することになる．

したがって，おのずとサブシステム毎に品質保証の方法論は異なる．

→プログラム生成完了時点で保障される品質の理論的定義と最終工程で到達すべき品質の定義及び保障の証明手段を明確にする．

5） 役割分担

① システム共通開発部門（制御基盤，業務業通，運用管理）

システム共通系の開発，及び業務 AP 開発チームに対するシステムの仕組み・仕掛けに根ざした設計方針の提示を行う．

② システム開発支援部門

開発標準・基準・規約上の問題及び他部門から提供された基盤システム部品の不具合等が有った場合の担当部門及び外部ベンダーとの調整を行う．

③ 業務 AP 開発部門

ⓐ プロジェクトマネジメント

NUL と顧客を単一組織と捉えたプロジェクト運営に関わるマネジメント全般が主たる役割となる．具体的には，企画・管理チームが定義した開発計画に対し，実施に向けた関連部門への協力要請及び調整を行う．

開発過程においては，各種クレーム対応，進捗・品質に対する包括的対応（課題・問題対応）及びリスクの見極めと回避策の計画化等となる．

ⓑ 開発計画・管理・推進チーム

工程毎の開発シナリオの企画と手順化並びにプロジェクト運営に関わる緒手続きの企画と具体的な手順への落とし込みを行う．開発管理面では各種レビューや質疑応答の分析に基づく進捗管理・品質管理・課題/設計ミスなどへの対応指示，及びソフトウェア資産管理．

ⓒ IT コンサルティング

開示された設計方針に対する AP 開発実践に向けた技術的困難性の見極めと設計方針変更要求及びシステム共通系との整合性確保の指示を行う．さらに，提供を受けたシステム共通系成果物及び基準・規約の評価と適用に向けた実装カスタマイズ等，業務開発現場から挙げられる技術的課題の一般化を行う．

ⓓ 業務 AP 開発チーム

業務 AP 開発のコアチームである．本開発は Case ツールの適用が前提になっており，プログラミング作業は基本的に存在しない．全工程を通して設計・テスト計画策定・検証作業等の SE 作業とテストを実施する PE（プログラミング・エンジニアリング）作業から構成される

ⓔ 業務内共通開発チーム

業務内共通関数，コード，用語，データ属性，DB 構造など個々のサブシステムに属さない部品・定義体の開発を行う．

ⓕ 運用管理・移行チーム

データ移行，外部システムとのデータ接続，システム運用に関わる設計と開発を行う．

ⓖ 環境構築チーム

各開発工程で必要となる開発環境の要件整理と構築依頼を実施する（構築作業は他部門にて実施）．さらに，テストで必要となるソフトウェア資産・データの準備と移植作業及びテストツール等の開発を行う．

ⓗ プロトタイプ開発

いくつかの業務 AP を先行開発し実環境での試験稼働を行うことで，業務 AP 設計方針の正当性及び他部門から提供される各種共通部品，ソフトウェアプロダクトの事前検証を行う．

プロトタイプとして開発した AP 資産はそのまま実成果物として継続する．

2.5 契約の考え方

プロジェクトの活動・責任範囲は契約により規定される．当該システムの開発に馴染まない契約内容であったり，契約条件に曖昧さを残していては円滑なプロジェクト運営は望めず，工程毎の契約形態が必要である．

2.5.1 契 約 形 態

契約形態として役務契約，出来高契約，一括受託契約の 3 種類が考えられるが，プロジェクトの性能を最大限に引き出すため，役務契約と一括受託契約を併用する形態とした．

1） 要件定義：コンサルティング契約

どのような成果物を残すかも含めた業務要件定義ノウハウ及びシステム化に向けた知識（成果物）の提供であり，設計・開発とは独立した契約形態となる．

2） 論理設計：役務契約

基準となる生産性に基づく設計成果物の規模に見合った予算枠での役務契約となる．この工程は業務要件の見直し，実装レベルでの設計課題・技術課題解決が前提となるため，一括受託での契約形態ではプロジェクト運営が困難と思われる．

3） 物理設計/プログラムテスト：一括受託

論理設計を受けた規模見積りと生産性から求められる工数で一括受託契約となる．

4） 結合テスト：一括受託

マネジメント負荷軽減を目的に，開発実績規模と生産性から求められる工数で

一括受託契約となる．

なお，結合テスト工程を一括受託する場合，以下の条件が整っていることが前提となる．

① プロトタイプによる制御基盤の稼働実績があること．

② 結合テスト工程での品質保証水準が明確になっていること．

③ 検証ポイントが具体的であること．

④ 前工程でモジュール間インタフェースの確認が完了していること．

5） 統合テスト：役務契約

マルチベンダー環境における統合テストは，他ベンダー提供プロダクトの品質・性能まで含めた稼働保障をする事になるため，顧客サイドに立った専門家による技術支援が必要になり，役務契約と成らざるをえない．

6） 一括受託契約期間の全体統制・調整作業：役務契約

関連部門・関連プロダクト及びサブシステム間との調整・統制に関わる作業については一括受託契約から切り離し，独立した機能として役務契約する．

2.5.2 役割の定義

プロジェクトマネジメントの実施基盤は各々の立場における責任と役割を各マネジメント間で認識しあうことにある．プロジェクト運営上のトラブルは，ほとんどの場合この認識ギャップが原因となる．

一般的に受託範囲・役割については契約書に明記されるが，顧客及び受託者を含む各部門毎・工程毎の役割を公の場で明らかにしておくことが重要となる．

2.5.3 開 発 責 任

1） 発注者責任

一般的には請負業者（製造請負）側の責任がクローズアップされ責任追及の対象となるが，予算超過・スケジュール遅延は，発注者責任が充分果たされない所に多くの原因がある．

例えば，　→開発フェーズにさしかかってから業務要件を変更・追加する．

→開発予算確定後にシステム化範囲・契約外作業を拡大する．

→発注者作業のマスタースケジュールを軽視する．　等

システム化範囲及び実現方法（方針）としての要求仕様と確定スケジュール及び要件変更に伴なう開発予算化責任は発注者にあることを契約行為の中で認識しあう事が重要である．

2） 受託者責任

受託者責任をあえて定義する必要もないが，契約書に明示しづらい品質についての責任が曖昧になりがちであり，明文化できない役割に対する透明性を維持する事が重要となる．

例えば，→進捗管理報告義務の範囲

→開発過程で発生する設計変更に対する許容可能な手戻り作業の範囲

上流工程で作成されたドキュメントのメンテナンス作業

→テスト結果証跡の取得種類

→システム保守・運用に関わるドキュメントの種類と深さ

→外的要因によるスケジュール遅延・手戻り

2.6　規模見積と生産性

1）　開発規模積算根拠

データ中心アプローチを前提に，プロセス毎の機能に着目した現行システムでの実績値としてのステップと入出力項目数から規模を積算する方法論を明示する事により，見積根拠としての理論を確立する．

2）　生産性の客観的根拠

Case ツール提供ベンダーでの開発実績を受け入れることで生産性の根拠とする．その際，Case ツール提供ベンダーも業務受託開発に参加させる事が重要である．

2.7　リスク認識と回避策

リスクの要素としては，規模の見積誤り，手戻り，生産性見込み誤り，制御基盤不良，開発環境不良，性能不良等が一般的な要素として考えられる．

これら個々の対応策については「マネジメント課題」として後述するとして，ここでは開発計画の破綻を招く可能性を秘めたリスク要素である開発規模リスク，業務要件変更による手戻りリスク，マネジメント執行リスク，について回避策を述べる．

2.7.1　開発規模リスク回避シナリオ

許容限度を超える開発規模の見積差異は，発注者・受託者双方の立場において，不利益をもたらすのみならず，開発計画そのものの見直しを強いることになる事から契約形態と契約タイミングを図 2 の通りとする．

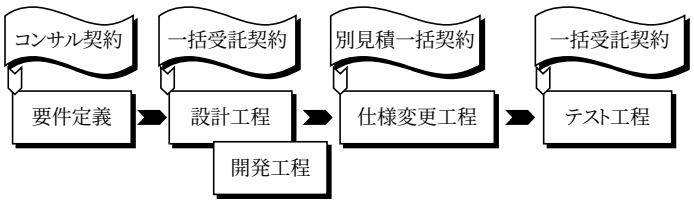

図 2　契約形態と契約タイミング

2.7.2　仕様変更手戻りリスク回避のシナリオ

開発環境でも述べた通り，オープン環境でのシステム化対象業務は，従来システム化が困難とされてきた分野が多く，業務要件の確定に際しては開発工程の進捗に関わらず際限なく続くことになる．

一方，発注者にとっては競合他社が次々と送り出す新商品対応や規制緩和策による変更も開発期間内で吸収する必要があることから，業務要件の変更・追加及び制御基盤等の設計変更リスクについては，マスタースケジュールに仕様変更期間を設け，開発予算については別途見積とする事でリスク発生の規模を極少化する方向で発注者と合意する．

なお，これは仕様変更要求部門へのけん制ともなる．

2.7.3　マネジメント執行リスク回避のシナリオ

1）　間接工数（プロジェクト管理工数）の極少化

一括受発注のメリットを最大限引き出すことの一つとして発注者と受託者のマネジメント力を活用し，プロジェクト全体としての間接工数を極力削減すると共にマネジメント負荷の一極集中を回避する．

→マネージというレンズ（立場）を通さず実体を把握しようとすると，情報の収集・選別を行う必要が有ると共に越権行為に繋がりかねない危険もはらんでおり，ミスジャッジ・混乱・モラルの低下をもたらす恐れがある．また，マネジを通すことで，詳細な役割（責任）及びやる気の実態が把握でき，てこ原理でプロジェクトの実体をコントロール可能となる．

つまり，小さく動いて大きな成果を得る工夫がプロジェクト全体の無駄を削減する事に繋がる．

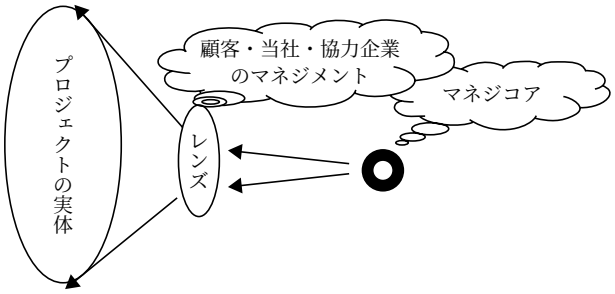

図 3　マネジメント執行

2）　進捗管理の考え方

一括受託契約は成果物に対する契約行為であり，労働時間・要員数などは一切関係しない．

従って，一括受託に於ける進捗・計数管理は成果物を中心とした管理を行うことになる．

以下，管理の座標軸ともいうべき着眼点についての考え方を述べる．

①　方法論

進捗管理を行う場合，一般的には作業者からの報告をもとに成果物の作成状況を計数把握するが，これは開発現場に管理負荷を与えるばかりか，誤った情報であったり，古い情報である可能性をはらんでいる．

これらを排除するために成果物の格納ファイル及びプロジェクト運営手続上発生する各種情報（課題台帳・質疑応答台帳等）をツールでトレースし進捗把握することで管理負荷を軽減すると共に最新の進捗計数を管理する．

②　管理対象分野

設計成果物（ドキュメント及び Case ツール成果物）については，進捗管理ツールによる進捗管理とする．また，課題管理・質疑応答管理については，変更管理の位置付けで進捗計数を管理するが，本質的には業務要件の確定度を把握する尺度として利用する．

ただし，業務仕様変更に繋がるものについては，会議体に付議しマネジメン

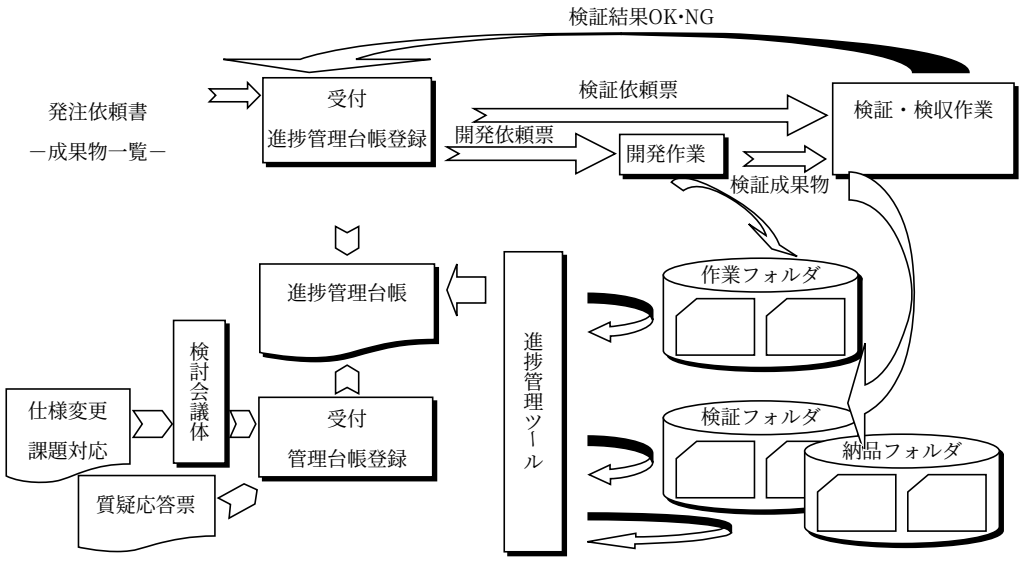

図 4　進捗管理ツールの概念

ト判断対象となる．

3.　マネジメント課題

これまで述べたマネジメント指針に則ったプロジェクト運営を実施したとしても，尚残るマネジメント課題について，事例を基にその対応策を以下に述べる．

3.1　業　　務

1）　開発を止める事なく仕様変更を受け入れられる手順の確立．

仕様変更作業自体は計画された一定期間の集中作業となるが，仕様変更のための設計作業を本来の開発作業と並行で進める必要がある．

また，仕様変更による手戻りを最小限に留めるための作業手順（工程定義）の工夫が必要となる．

2）　下流工程に責任を負わないベンダーが定義した業務要件による業務 AP 開発

上流工程（業務要件定義）は当該業務を得意分野とするコンサルティング会社が受託する事が多く実装に向けた実現性に対する追求は本質的に気薄となり，業務要件の深さ・広がり・精度等の保障はなく下流工程を担当する設計部門が負う事になる．

入出力画面・帳票設計までが業務要件定義となり，本質的な業務要件定義のたたき台的な位置づけで業務要件定義成果物を捉えざるを得ない．

3）　組織合意の時間と手間

利用部門が直接業務要件を整理し，開発部門がそれに基づく開発を行う事が理想ではあるが，利用部門が業務要件を整理する事は非現実的であり，でき上がったシステムを使用し評価する手順となる．

これは，開発部門にとって大きな手戻りリスクを秘めているばかりか，開発ス

ケジュール遅延要因でもあり，最終仕様に対する組織合意の無い中，如何にこのリスクを回避し，手戻りを押さえるかが大きな課題である．

3.2 開発スケジュール

1） 開発手順・設計基準・各種規約策定と並行した業務 AP 開発

マルチベンダーによるオープン系プロダクト群をシステム構成基盤とする業務 AP 開発に当たっては，前提となる開発標準としての開発手順・設計基準・規約に従うことになり，これ無くして効率的で高品質のシステム開発は不可能と言ってよい．

本事例ではマルチベンダーによるオープンプロダクト環境を前提にしたプロジェクトマネジメントを基本理念にしており，設計方針・基準・規約が業務 AP 開発統制の根拠となる．

しかしながら，当該顧客にとってオープン系プロダクト群による初めての本格的なシステム開発であることから，業務 AP 開発を進めながら標準化の整備を行う必要があった．

2） システム基盤構築と並行した業務 AP 開発

業務 AP の設計・開発が本件受託者の使命であり，制御基盤系及び業務共通系の提供については，決定権限の及ばない別プロジェクト（他ベンダ）が担う事になる．

業務 AP 開発部門としては，システム基盤（含む API 開示）の提供時期，動作保障，システム制限等による手戻り・遅延リスクを如何に回避しつつ開発を進めるかが大きな課題となる．

3） 短期開発

顧客にとって競合他社に先駆けてマーケットが求める商品を提供することが最優先される社会背景にあって，必然的にシステム開発の工期短縮が重要テーマとなり，その為の開発手法を企画・立案し実践することが大きなマネジメント課題となる．

3.3 開 発 環 境

1） Case ツール適用環境での業務開発

システム開発から保守全般にわたる品質・生産性の向上を目的に，開発支援ツールとしての CASE ツール導入を前提に開発計画を練り上げているが，CASE ツール利用実績の少ない当該開発環境で計画通りの品質・生産性を維持できるか大きなリスクでもあった．

2） オープン系の大規模開発におけるマネジメント支援

開発を円滑に進めるための企画・管理・技術支援・プロジェクト運営・リスク管理等大規模開発に必要となるノウハウはその経験に帰するところが大きく，一般的には客先支援が必要となる．

これは，一括受託するしないに関わらず必要な機能であり客先のスキル・ノウハウ・要員を補う事を目的に客先を含めたマネジメント業務の支援（コンサルティング）を提供する必要があった．

4. マネジメント課題に対する対応策

ここまで，マネジメント指針・課題について述べてきたが，これら指針・課題と具体的なマネジメント上の対応策は切り口を変えた対応となる．

なぜならば，必ずしもマネジメント課題とその対応については，1：1 の関係ではなく n：n の関係にあるからである．

4.1 契　　約

4.1.1 業 務 要 件

上流工程を担当するコンサルティング会社がエンドユーザ部門との間で定義した業務要件定義は，一般的に業務ニーズを網羅性と実現可能性の両面に渡り満足している保障が無いと見るべきであり，業務要件定義の検証を兼ねた業務要件の詳細化工程を設ける必要がある．

この工程では主に次に示す作業を行う．

① 論理設計へ繋ぐための中間成果物作成

② 業務要件定義書としての説明不足補記

③ システム規模の見積に必要となる客観的基礎数値の調査

以上の作業を通してリスクを回避すると共に見積根拠を明らかにする．

上記工程を踏むことで「物理設計工程以降，結合試験までを一括受託」とする．

システム統合試験（総合テスト，運用テスト）を一括受託契約とすることは，基盤を含めた本番稼働を保障する事になり，マルチベンダー体制を前提とする本システム開発に於いて，一括受託の範囲を越えると判断（SI 契約）でき役務契約とすべきである．

4.1.2 協力会社との契約形態

契約形態の違いからくるリスクを回避するため，対顧客契約と同一タイミング・同一形態の契約とした．ただし，一括受託契約期間中の役務的作業の発生に柔軟性を持たせる必要があり，一括予算とは別枠で役務枠を確保しておくことが重要である．

4.1.3 マネジメント支援

マルチベンダー体制の場合，プロダクト間，AP/プロダクト間，サブシステム間の調整マネジメントが必要になる事から管理業務・技術支援・プロジェクトマネジメントについては，一括受託と別枠でのマネジメント支援できるプロジェクト運営環境を契約面から整える必要がある．

4.2 見　　積

4.2.1 見 積 技 法

FP 法（当プロジェクト向けアレンジ）と WBS の組み合わせにより，一括受託分野の見積を実施する事になるが，この見積の考え方を承認機関の公認事項とする事がマルチベンダー体制における最大のマネジメントテーマであり，後のベンダー間（プロジェクト間）の摩擦及び不信感を避ける意味で円滑な開発の推進に大きく寄与することになる．

1） FP 法

プロセス数・入出力項目数と現行システムの機能別ステップに着目した開発規模の見積を実施する．

上記基礎数値は入出力設計結果と現行システムの調査結果に基づいて積算されており，本システムの機能的特徴（新規性）は，技術支援を通して積算計数に反映させた．

2） WBS

Case ツールベンダーの開発技法に則り，アクティビティ単位の役割分担として活用した．

具体的には，一括受託の役割範囲，一括受託期間中の役務体制の規模，客先の体制に反映させることになる．

これは，一括請負契約の中で定義される生産性の前提条件としての作業範囲を明示する事になると共に開発手順定義の根拠でもある．

4.2.2 見 積 結 果

本開発は Case ツールにより進める前提があり，生成結果としての COBOL ソースと開発作業量は全く連動しないため，受注規模と発注規模は必ずしも一致せず，作業の量と難易度で契約規模を見積もる事になる．

1） 受注規模

認知された見積技法に基づいて積算する．

2） 発注規模

規模に対する開発負荷について発注者と受託者の充分な協議・合意の上，発注規模を確定することが極めて重要となる．

規模に対する開発負荷軽減項目の事例は次の通りである．

→業務内共通の多くはプロトタイプ実施チームで雛形先行開発を行うため，開発負荷が軽減される．

→ボディ（制御ロジック）はプロトタイプ実施チームの雛形先行開発によりスケルトン提供されるため開発が発生しない．

以上 2 点はプロトタイプ実施チームの技術支援により実現させた．

→Case ツールの文法に則った仕様書（設計書）を作成することでソース生成が可能となり，いわゆるコーディング作業が削減できる．

（プログラミング作業は入力イベント処理のみであり，プログラミング工程の主たる作業はプログラムテストのみとなる．）

→定義体（PDP 等）は論理設計成果物からの自動生成となるため，開発が不要となる．

4.3 ノーリスク開発のシナリオ確立

4.3.1 業務要件の追加・変更

システム利用部門であるエンドユーザ部門からの業務要件変更要求を拒否する事はきわめて困難であり，社会背景上発生が予見出来る制度改訂及び競合他社からの新商品対策を開発計画に盛り込む意味を含め，マスタスケジュールに当初から仕様変更工程を設ける．

また，エンドユーザが仕様変更を要求する手続きについても，発注者内の管理機関を通す事務手続きを確立しておくことで，はじめて歯止め無き業務要求の増大リスクが回避可能となる．

これにより，すでに認知された見積技法に則った仕様変更規模見積（仕様変更の予算化）が許され，仕様変更も開発の一つの工程として取り扱う事が開発スケジュール上で可能となる．

4.3.2 制御基盤及び業務共通の変更・遅延

過去の業務 AP 開発においても，基盤変更による手戻りや障害等によるプロダクトの提供遅延で業務 AP 開発が滞る状況を幾度となく経験してきている．

これを回避するためには，基盤プロダクトや業務共通機能と直接インタフェースを持たない仕組みが必要であり，本開発においても制御基盤系ソフトウェアについては全て関数で包む込む仕組みを準備した．

なお，他部門提供機能については，プログラムテスト向けに生成した STUB（Case ツールによる自動生成）を定義し，他部門提供の共通関数仕様変更と提供遅延リスクに備えた．

4.3.3 開発環境の障害

Case ツールを始めとした開発環境の障害による開発作業のストップや成果物の破壊に対する事前の対策を具体的に準備することはきわめて困難である．これについては開発請負契約書として明文化することで歯止めを持たせることが重要となるが，一方で開発手順として段階的開発手法を取り入れることで，環境障害が発生しても全開発作業を止めるまでには至らない手だてを講じている．

結果としてスケジュール調整を余儀なくされた障害も幾つか発生したものの，大小合わせ 20 件程度の障害件数であり，主スケジュールへ大きく影響する事態は発生していない．

なお，本開発においては，Case ツール導入により大幅に軽減されたプログラミング作業の代わりに結合テストの先取りとして，モジュール単体テストは実施せず，exe または dll 単位のプログラムテストの実施を可能とする環境及び作業効率を高める目的で開発者個人にテスト環境を与えるための擬似 Oltp および擬似 Odbc 環境を開発・提供している．

4.4 間接工数（管理工数）の極少化

当プロジェクトにおいては，開発管理業務とマネジメント業務を分け，マネジメント業務については有償化し広く横断的なマネジメントを可能とする一方で，開発管理業務については受託開発の一業務として取り扱っている．

マネジメントの考え方としては，問題となっている実体組織に対し，最も影響力のあるチャネル（人物・情報発信元・人脈・部門）に相談を持ち掛けることでより効果的な問題解決を目指した．

一方，開発管理は作業の進捗と品質が計画通り運んでいるかの検証をすることにあり，進捗・品質の状態指標となる成果物及び手順から発生する文書情報を基にツールで情報収集し管理することを目指し，報告物による管理を排除している．

4.5 品質保証と生産性

4.5.1 開発手順を裏付けとした生産性及び品質の保証

受託先でいかなる作業がなされているかを正確に把握する事が発注者にとっての大きな品質保証となり計画した生産性をも保証することになる．

つまり，開発手順（詳細工程）を定義し，その手順を忠実に実行した場合の生産性を予め定義するとともに,何をいつチェックするかのポイントを定め,第三者が調査・検証することで品質と生産性が保証されるが，約束事と検証だけでは自ずと限界がある．

手順は幹となる部分をツール化することで自由度を持たせつつも手順を忠実に実行せざるを得ない状況を創り出すことが重要となる．

なお，本プロジェクトにおいても，中核となる Case ツールの他に各種設計基準及びツールを開発している．

4.5.2 クリーンルーム開発手法の部分適用

「ノーリスク開発のシナリオの確立」で述べた対応だけでは，あまりにもエンドユーザに高いハードルを要求しているにしか過ぎず，段階的に業務要件（ニーズ）を固められる余地を手順として準備する事も品質・生産性を高める上で重要となる．

当プロジェクトにおいてもインクリメンタル開発プロセスを開発手順として組み入れ，段階的・部分的開発手法を適用した．これにより，エンドユーザ部門がプロセス単位の完全な要件を一気に確定する必要が無く，機能単位に時間を掛けて詰める事ができ，最上流である業務要件からクリーンなシステム開発を目指す事が可能となる．

ただし，プロセス横断的に機能の詰めがなされるため，個別プロセスとしての整合性を損なったり設計方針を逸脱する恐れがあり，クリーンルーム手法でも述べられているデザインレビューの実施が不可欠となる．

5. ま と め

マルチベンダーを前提とした高生産性・高品質実現に向けたプロジェクトマネジメント上の戦略的ポイント及び生産性と規模見積誤差についてをまとめる．

1） 一括受託業務と役務業務の契約分離

マルチベンダーによる開発体制を組織する場合,一括受託契約期間であっても,常に各プロダクトの調査・評価，ベンダー間・部門間の調整・統制が発生するため，これらを実施する体制を一括受託契約と別立てで準備する．

2） 仕様変更対応のマスタスケジュール化

当然発生する積み残し課題・仕様変更については，実行計画書に対応時期・計画を組み入れ別途予算化できる契約条項を組み入れる．

3） 請け負い業者のマネジメント力・情報収集力活用

管理業務，推進・指導等のマネジメント業務については，一括契約の条件として請け負い業者のマネジメント力を活用することにより，開発作業の品質向上及び間接コストの削減を図る．

4） 生産性

Case ツールでの開発において，ステップ規模をもとにした生産性では本質的ではなく，設計・製造成果物に着目した生産性基準値を持つ事が合意形成の絶対条件になる．

なお，本開発では，適用した Case ツールに対応した設計ドキュメントを決め込みながらの開発に成らざるを得なかった背景もあり，ステップ規模を基にした

生産性基準値を設定している．

5) 規模見積

各業務プロセスは，クライアントとサーバに分割された機能が一定の基準でパターン化され構成される．

これらのパターンに対する機能別の計数を設定し，各プロセス毎の画面項目数，帳票項目数，機能ボタン数等を掛け合わせる事で規模を算出している．

見積時点の規模と開発実績は，設計規模は 1670 k 相当であったが，開発規模は 1113 k と評価している．

6. おわりに

本開発は，マネジメント上必要となる基準計数および開発の軸を成す設計基準が無い中でのプロジェクト立ち上げであったと同時に，オープンの名の下，画期的な生産性を求められ，プロジェクトマネジメントも自ずと一定の品質を維持しつつ如何に高生産性を実現するかに集約されることになり，振り返ると品質・生産性・リスク回避との戦いの連続であった．

これら三つのテーマに対し，結果として概ね評価できるが，幾つかの課題も残してきている．

第一に品質については，テストケースを無制限に準備しても本番を保証するに足りうる品質保証には成り得ず実施も困難であり，品質を保証する理論的根拠としての客観的な定義が重要となる．

該システムに於いては，現行システムとの比較により品質の理論的根拠を見出す事が可能となったが，極めて新規性の高いシステムにおける客観的な理論構築が求められる．

第二の生産性向上に向けては，製造技術の進歩により評価できる成果があったと思っているが，オープンマルチベンダー環境において，如何にして，調整・統制・管理のための間接工数・手戻り負荷を軽減するかの解を見出すまでには至っていない．

第三としてリスク回避の基本は，不可避なリスクのマスタスケジュール化とリスク回避策実施に向けた部門間の良好な信頼関係を築く事から始まるが，リスクマネジメントは見積られた開発規模を基に実施され，確度の高い見積り技術の確立が急がれる．

本開発が計画立案段階から開発完了まで良好な状態を保ち得たポイントは，本文で述べてきた Case ツールを前提とした開発プロセスの改善が大きく寄与したと思っている．

なお，開発計画実施に当たり，成功への強い意思，関係者相互の深い信頼関係が基礎をなしていた事は言うまでもなく，組織の信頼関係を損なう行為は，個人々々の創意工夫の努力を失うことになり，結果として更なる問題へと繋がる危険を秘めている事をプロジェクトマネジメントとして充分配慮する必要があると考える．

執筆者紹介　中　村　祥 次 郎（Shojiro Nakamura）
1973 年，滋賀県立彦根工業高等学校機械科卒業．同年，日本ユニシス(株)に入社．主として勘定系 3 次オン開発，系統勘定系 3 次オン共同開発に従事．現在，第二開発センタ開発推進室に所属．