GRADUATE SCHOOL OF BUSINESS ADMINISTRATION
KOBE UNIVERSITY
ROKKO KOBE JAPAN

2008-14

ソフトウェア開発における品質的欠陥発生要因と対策

浅田 賢治郎

Current Management Issues

序章

第１節　研究の背景と問題意識

証券システムの障害、銀行 ATM オンラインのダウン、公共料金の超過請求など、情報システムの不具合に起因する報道が後を絶たない。2002 年に発生した銀行の合併に伴う情報システムの品質事故は、口座振替遅延が 250 万件、2 重引き落としが 3 万件発生するなど、非常に大きな被害をもたらした。とはいうものの、最近では 2008 年 5 月 12 日に発生した、三菱東京 UFJ 銀行のシステム統合に伴うシステムトラブルなど、この品質事故が特殊というわけではなく、金融機関は統合の度に情報システムの品質問題が取りざたされている。さらに首都圏 16 鉄道の自動改札機システム障害、NTT 東西における光電話の障害など、ソフトウェアの不具合によるライフラインへ影響を及ぼす品質事故事例についても枚挙にいとまがない。

IT 産業における受注ソフトウェア開発には一般的に「企業、行政機関、教育機関、医療機関等の組織の経営、活動、プロセスを支援するソフトウェア」と定義されるエンタプライズ系ソフトウェア、自動車や家電製品、携帯電話などの製品の一部として「機器に組み込まれて機能を実現しているソフトウェア」と定義される組込みソフトウェアがある。

エンタプライズ系ソフトウェアの品質事故は、自社のみの損害にとどまらず、顧客や取引先、消費者に大きな影響を与えてしまう。それにもかかわらず、日経コンピュータの調査によると、品質・コスト・納期のすべてで予定を満たしたプロジェクトは全体のわずかに 26.7%にすぎないとの報告がある[1]。

一方、エンタプライズ系ソフトウェアだけでなく、国内の組み込みソフトウェアについても品質事故の発生が報告されている。実際に 2004 年にトヨタのハイブリッド車で、組み込みソフトウェアブレーキシステムの不具合により 16 万台のリコールが発生している。さらに公共インフラにおいても、中部電力の駒場ダムで、ダム水位一定制御（計画放流システム）の制御プログラムの欠陥 により、総量約 2 万立方㍍の貯水が異常放流されるという事故や、NEC のソフトウェアの欠陥により国土交通省の航空管制システムの障害が発生したことで、215 便が欠航するという事故も発生している。

保田（1995）が指摘する通り、日本のソフトウェア開発の品質管理は、不良をできるだけ少なくし、ばらつきを押さえるという信頼性、すなわち「当たり前品質」を重視している[2]。その「当たり前品質」を達成するため、多くの国内システム・インテグレータや、ソフトウェア開発企業は ISO9001 等の認証取得、CMM/CMMI[3]といった品質管理に関わる手段を導入している他、様々な品質知識体系の習得にも努めていることは周知の事実である。しかし、そのような様々な取り組みや努力にも関わらず、現在も品質事故は多発している。すなわちプロジェクトに携わる要員の努力や、品質管理や監査の仕組み、知識体系をもっ

---

[1] 日経コンピュータ「特集“プロジェクト成功率は 26.7%”」2003 年 1117 日号 ,50－51 頁

[2] 保田（1995）31 頁

[3] CMMI（Capability Maturity Mode Integration：能力成熟度モデル統合）については SQuBOK 策定部会編（2005）117-118 頁に詳しい。が説明されている。なお CMM（Capability Maturity Model）は CMMI の前身である。

てしても、ソフトウェア開発を含むシステム開発においては「当たり前品質」ですら完全に実現できているとは言い難い状況である。このままでは**図 1** のように市場拡大、受注ソフトウェア売上高が拡大し続ける情報サービス産業においてさらに品質事故が多発するという懸念があり、それに伴い情報サービス産業全体への信頼も失墜し続ける可能性がある。

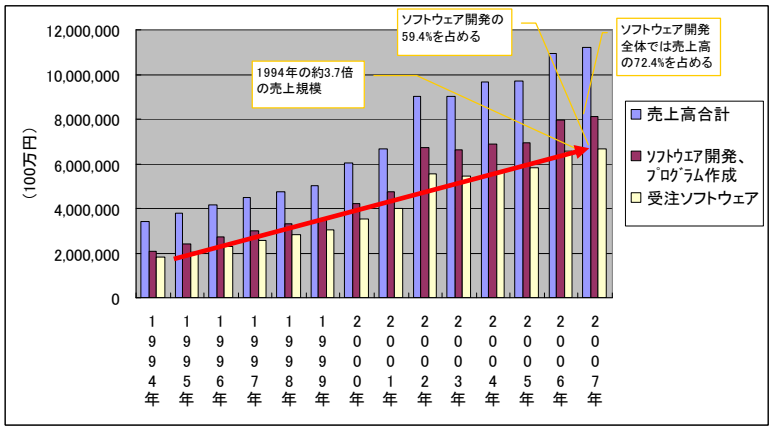

図 1 情報サービス産業の業種種類別売上高
【経済産業省“特定サービス産業動態統計調査 長期データ 情報サービス産業”データより著者作成】

第2節 研究の目的

何故、ソフトウェア開発において品質的欠陥は発生し続けるのか。この点につき、本研究では、IT産業において先に述べたように定義されるエンタプライズ系ソフトウェア、自動車や家電製品、携帯電話などの製品の一部として機能する組込みソフトウェアに焦点を絞り、以下2点につき議論するものとする。

（１）国内ソフトウェア開発企業、組織は、ISO9001 や CMM/CMMI 等の品質管理手法を採用するなどし、品質向上に努めているにも関わらず品質事故の発生を防げないのか。

（２）そもそも、ソフトウェア開発において、高品質は達成可能なのか。可能であるとするならば、どうすれば高品質を達成することができるのか。

第２章　国内ソフトウェア開発の実際

第１節　国内ソフトウェア品質事故事例

第１項　エンタプライズ系（民需）品質事故　みずほ銀行の事例

2002年に発生した旧第一勧業銀行、旧富士銀行、旧日本興業銀行の全面的統合に伴う、システム統合により発生した品質問題は、対外システム系プログラムの欠陥により引き起こされた ATM 障害、及びその後継続的に口座振替のトラブルに大別できる。

前者は、旧第一勧業銀行の対外接続系システムと旧富士銀行の勘定系システムをつなぐため、これらシステムを接続するリレーコンピューターの既存の COBOL プログラムに手を加えたものの、その修正に不良箇所があったことで、旧富士銀行の勘定系システムが孤立。そのため旧富士銀行以外の ATM で旧富士銀行のキャッシュカードを利用した取引ができなくなるというトラブルが発生した。また、後者については、上述のプログラムの不具

合により対外接続系システムに障害が発生したため、情報が滞留し、現金が未払いにも関わらず、残高が減るという事態を生じさせた。この欠陥は旧富士銀行勘定系システムでの残高データを「通信中の電文を保存する領域に一定以上の電文がたまる」、「異なる電文を組み合わせで処理する」といった条件が重なった場合に限り、発生するものであった。また、顧客が持ち込んだデータの中に銀行名は旧、「店番号」が新となっているような不備があったことで、統合と同時に変更された「金融機関コード」や「店番号」に対し、古い番号でも受け付けられるように作られた口座振替プログラムがその誤りを識別できずにエラーが発生した[4]。

2002 年 6 月に発行された金融庁の報道発表において、本トラブルの発生の原因は「システムの機能を確認するシステムテストや運用テストが適切に実施されていなかったなど、最低限必要な準備ができていなかったこと、②テストが不十分であった等の重要な情報が、一部の開発責任行のシステム開発部門内にとどまっていたことなど、グループ内での報告・連絡態勢に重大な問題があり、十分なチェックが働かなかったこと、③CB[5]において大量の事務処理を支える事務インフラが不十分であったこと、等にある」[6]としている。また同報告書や各種の記事ではその他経営者のリスク認識の甘さやなども事故発生の要因として指摘されている。しかし、ソフトウェアの品質的欠陥の発生原因に焦点を絞って考えた場合、納期という制約もあっただろうが、異なるシステムをネットワークで結合したことでシステム間の相互依存関係が複雑化したこと、その相互依存関係に対し、複合的な条件を考慮したテストが工程の中に組み込まれていなかったことにより生じたものといえる。

第２項　エンタプライズ系（公共）品質事故　航空管制システムの事例

2003 年 3 月 1 日に発生した、航空管制システム障害は欠航 215 便、大幅な遅延 1,500 便以上、足止めされた客 27 万人とシステムが引き起こしたトラブルとしては航空史上過去最大のものであった。事のはじまりは、3 月 1 日に防衛庁と東京航空交通管制部が飛行計画をやりとりするために、国土交通省が運用する飛行計画情報処理システム[7]（以下 FDP）に「防衛庁システム対応プログラム」を追加したことで発生した。また後の調査により、この障害は、追加された「防衛庁対応プログラム」の不具合ではなく、前年の 9 月に FDP に追加された「共通データプログラム」に不良があることが原因であることが判明している。ダウンを引き起こした不良は「メモリ空間において、共通データプログラムの読み込み領域を超えるとシステム全体がダウンする」というものであり、FDP 内の様々なプログラムに共通する処理を実行する「共通データプログラム」がデータ領域を読み込んだ際、「防衛庁対応プログラム」の追加により、読み込み領域を超える領域に別のプログラムの領域が再配置されてしまったため、処理の途中で FDP が不正な処理と見なし、システム全体がダウ

---

[4] 日経コンピュータ編(2002)56－69 頁　トラブルの詳細、発生要因については同書に詳しい。

[5] CB とは Corporate Bank の略語である。

[6] 金融庁.“2006 年 6 月 19 日 報道発表資料 みずほフィナンシャル・グループに対する行政処分について”. みずほファイナンシャル・グル・・・:金融庁.

[7] すべての航空機の飛行情報を各航空会社から集約。航空機の管制業務に必要となる行き先、飛行経路等の情報を全国各地の管制部門に対して出力するシステムのこと。

ンした[8]。さらに開発元は当該不良に気づいていたものの、実運用に即したテストが行われなかったことも報告されている。この品質事故の発生を同記事では国交省によるテスト及びシステムのバックアップ体制が不十分であったことが問題との認識を示している。
しかし、実際の品質事故の発生要因は、プログラム追加により新たに構成されたサブシステム間の相互依存関係を考慮しなかったこと、プログラムに品質的欠陥が混入していることに気づきながらも、システム開発工程の中で、不良情報のフィードバックをし、適切な修正を行わなかった品質管理の姿勢と方法に問題があるのではないだろうか。

第３項　組込みソフトウェアの品質事故

組込みソフトウェアの品質事故事例は近年多数発生している。メルセデスベンツでは2004年3月にバッテリーを制御するソフトの不具合で130万台を無料修正、リコールを行っている。またトヨタ自動車も、同年電気モーターとエンジンを併用して走るハイブリッド車「プリウス」で、同じくブレーキ制御のソフトウェア不具合により走行中に突然エンジンが止まるトラブルが発生していることを受け､全世界で約16万台の自主的な無償修理を行っている。また、家電や携帯電話の世界でも多数の不具合が発生、その不具合による多額の回収費用や、信用の失墜が減収要因にもなっているといわれる[9]。

それでは何故このような不具合が発生するのか。この問いに対し、いくつかの記事では規模と開発期間に原因があるとされている。例えば自動車の組込みソフトウェアについて取り上げた記事では「連動には複雑なソフトが必要になる。1台の自動車は様々な部品メーカーが生産した、約3万点の部品からなり、各部品モジュールに使われるソフトがお互いにやり取りできなければならないからだ。反面開発期間は短縮している。この業界でもハードの完成が遅れてソフトの実証時間が短くなるというケースも増えている」[10]としており、携帯、家電でのソフトウェア不具合に言及した記事では「ソフト機能に対する要求が高度になる一方で、開発期間が極端に短くなったことが原因である」､「不具合が多いのはテスト不足のまま出荷せざるをえないから」[11]としている。どちらも規模の拡大に対し、開発期間は短縮し、そのためテストが十分に行えなくなったことが不具合発生の要因であると断じていることが特徴であり、これもまた、品質管理上の問題といえる。また、これ以外にいずれの記事においても、品質向上には「擦り合せ型」の開発工程が重要であるとの認識が示されている。

第２節　高品質ソフトウェア事例

今やソフトウェアはあらゆるものに搭載されている。もちろんそれは宇宙開発の分野や

---

[8] 日経コンピュータ「ニュース＆トレンド 航空管制システム障害は防げた！バグを放置、テストは8時間のみ」2003年 3月 24日号,15－16頁

[9] これらの報告については、日経ビジネス「時流潮流 News&Trend 陥ソフト経済産業省も対策に着手 携帯・家電に不具合相次ぐ」2004年7月5日号,7頁を参照されたい。

[10] 日経ビジネス「特集 ソフト不良が招く品質の危機」2005年4月25日・5月2日号,38頁

[11] 日経ビジネス「時流潮流 News&Trend 陥ソフト経済産業省も対策に着手 携帯・家電に不具合相次ぐ」2004年7月5日号,7頁

軍事システム、航空機や鉄道といった分野においても例外ではない。このような分野で使用されるソフトウェアには、国防、国威発揚、人命といった観点から、非常に高い品質が求められることは容易に想像できる。しかし、このような極めて高い品質が求められるソフトウェア開発でも品質事故は発生している。

例えば航空機においては、2000 年に米国で発生した海兵隊の垂直離着陸機の墜落事故[12]、2002 年に発生したイギリスでの 2 機の旅客機のニアミス[13]、1995 年にコロンビアで発生した墜落事故[14]の報告がある他、国内においても 1994 年に名古屋空港に着陸しようとしていた中華航空 140 便(エアバス A300-600R)で発生した、自動操縦と手動操縦の二つの系統の制御コンフリクトにより、乗員乗客 264 名が死亡、7 名が重傷を負うという大事故が発生している[15]。鉄道においても国内の事例として、2005 年に東海道新幹線にて ATC（自動列車制御装置）のソフトウェア不具合によるブレーキがかからないという品質事故の発生が報告されている[16]。

このように、国内外で本来極めて高品質でなければならないソフトウェアの品質問題に起因した事故の発生が報告される中、高品質ソフトウェア開発の例としてしばしば取り上げられるのが NASA のソフトウェア開発である。

NASA では 1981 年以降、2008 年 3 月末現在で計 122 回のスペース・シャトル打ち上げが行われている。その間、1986 年に発生したチャレンジャー号爆発事故や、2003 年にコロンビア号の空中分解事故が発生しているものの、前者については、ロケット・ブースターのセグメントの繋ぎ目に使用される O リングの不具合[17]によるものであり、後者については、外部燃料タンクから剥がれ落ちた断熱材の破片が衝突したことで、左翼前縁部に亀裂が入り、その状態で大気圏に再突入したことが原因であるとされている[18]。すなわち、いずれもソフトウェアの品質的欠陥により発生した事故ではない。

図 2 は NASA の人間が搭乗する人工衛星関係でのソフトウェアの規模を表したグラフである。人類にとって意義高いアポロ計画ではあるが、スペース・シャトルのソフトウェアと比較すると、スペース・シャトル計画のわずか 5 分の 1 程度の処理の規模である。一方

---

[12] タカミハマダニ・中尾政之.“ソフトのバグによるハイテク航空機の墜落事故”.JST 失敗知識データベース.

[13] マユミビンクス・中尾政之.“管制官の誘導ミスとコンピュータ故障による旅客機のニアミス”.JST 失敗知識データベース.

[14] タカミハマダニ・中尾政之.“データ入力ミスで旅客機が山に激突”.JST 失敗知識データベース.

[15] 張田吉明・中尾政之.“名古屋空港で中華航空 140 便エアバス A300-600R が着陸に失敗炎上”.JST 失敗知識データベース.

[16] 日経ビジネス「特集 ソフト不良が招く品質の危機」2005 年 4 月 25 日・5 月 2 日号,32 頁

[17] Leveson, N.“Report of the Presidential Commission on the Space Shuttle Challenger”.Nancy Leveson’s HomePage at MIT.

[18] 宇宙航空研究開発機構.“2003 年 8 月 26 日 コロンビア号事故調査報告 Volume I（速報版）COLUMBIA ACCIDENT INVESTIGATION BOARD Report Volume I August 2003”. 宇宙ステーション・きぼう広報・情報センター 宇宙航空研究開発機構：JAXA.

スペース・シャトルのプログラムに着目すると、1 秒間に約 3,500 万命令の処理が必要な規模まで拡大している。ソフトウェアは一般に1行に1命令が記述されることを考えれば、後に詳述するエンタプライズ系ソフトウェアや、一部のものを除いて、組込みソフトウェア程度の規模は有するものと推察される。

それでは高品質が求められるソフトウェア開発に対し、NASA ではどのような品質管理を行っているのであろうか。NASA においては ISO9001 の認証取得の他、IBM スペース・シャトル・プロジェクトにおいて CMM/CMMI レベル 5 を達成している。また先にも述べたチャレンジャー事故後に発表された「ロジャー委員会 ソフトウェア報告書」で「第 3 者がチェックすることによって事前に取り除ける不具合」も多数あったとの報告がなされたことが契機となり、NASA ではソフトウェア IV&V[19]と呼ばれる検証プロセスを導入している[20]。NASA では約 100 のプロジェクトリスク評価データを検討し、70 プロジェクトを IV&V の導入候補とし、リスクの高いものから適用するガイドラインが設けられ IV&V の導入が進められた。その効果については、NASA のプロジェクトへの IV&V の適用は肯定的な結果を示し、また高くつくような障害の未然防止もできているとされている[21]。

ここに NASA IV&V Facility MDP [22]にて公開されているデータがある。公開されているデータセットは複数のプロジェクトで開発されたモジュールに関するメトリクスを記したものである。公開データにはプロジェクト毎のモジュールの不具合数不具合率もデータとして公開されているが、その公開されているモジュールのコード行数合計、不具合数合計をそれぞれ集計し、1,000 行毎の不具合率を計算しなおしたものが、**表 1** である。1,000 行毎の不具合率は中央値が 6.95、平均が 7.86 となっており、NASA におけるソフトウェア開発でも極めて多数の不具合が混入しているといえる。つまり、IV&V という検査プロセスを経ることで、不具合の存在が明らかとなり、結果的に欠陥の流出を抑制できている可能性がある。

国内においては、JAXA が 1996 年に NASA からの要請によりこの IV&V を採用しており、1998 年から、ロケットや人工衛星開発に適用が始まっている[23]。また、その定性的な効果として、①技術的欠陥の抽出、②開発作業品質の向上、③マネジメントへの品質の可視化

---

[19] Independent Verification and Validation(独立検証及び有効性確認)の略。IEEE std 610 によれば、開発組織から技術面、管理面、及び財務面で独立した組織が実施する検証と妥当性確認のことであるとされる。

[20] 経済産業省 プロセス改善研究会.“ベストプラクティス調査報告書 ～究極の高品質ソフトウェア開発プロセスを目指して～ 独立行政法人 宇宙航空研究開発機構（JAXA）第 1.0 版 平成 19 年 4 月”. 情報処理推進機構：ソフトウェアエンジニアリング.11 頁

[21] SQuBOK 策定部会編（2005）79 頁

[22] NASA IV&V Facility とは、IV&V プロセスを担う NASA の独立検証機構であり、NASA IV&V Facility MDP とは NASA IV&V Facility が過去プロジェクト成果物の一部をリポジトリに蓄積し、一般に公開している活動である。

[23] 経済産業省 プロセス改善研究会.“ベストプラクティス調査報告書 ～究極の高品質ソフトウェア開発プロセスを目指して～ 独立行政法人 宇宙航空研究開発機構（JAXA）第 1.0 版 平成 19 年 4 月”. 情報処理推進機構：ソフトウェアエンジニアリング.10 頁

が挙げられている[24]。JAXA においては、2003 年の発足以来、打上回数 2008 年 8 月現在で 14 回とその回数は少ないものの、2006 年 H-IIA6 号以外に打上の失敗はなく、この打上失敗についても、固体ロケット・ブースターのノズルに穴が空き燃焼ガスが噴出したため、ロケット・ブースター分離用の導爆線が焼損し、分離できなかったことが要因[25]とされており、ソフトウェアを要因とする事故の発生報告はない。

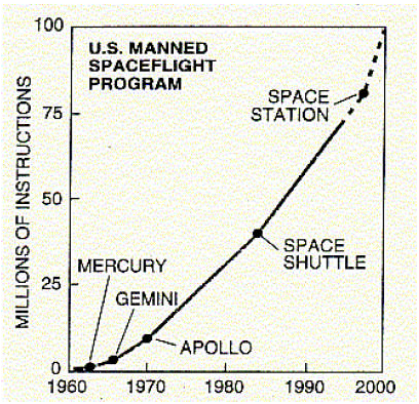

図 2　人間が搭乗するプログラムの規模の推移【Dorfman, Thaiyer (1996) p.5】

	CM1	JM1	KC1	KC3	KC4	MC1	MC2	MW1	PC1	PC2	PC3	PC4	PC5	
①コード行数 （各モジュール行数合計）	16903	457177	42963	7749	25436	66583	6134	8341	25922	26863	36473	30055	161695	
②不具合数 （各モジュールエラー数合計）	70	3179	525	101	253	79	113	37	139	26	259	367	1009	平均
③不具合率 1000×（①/②）	4.14	6.95	12.22	13.03	9.95	1.19	18.42	4.44	5.36	0.97	7.10	12.21	6.24	7.86
		中央値												

表 1　NASA ソフトウェア開発プロジェクト 1,000 行毎の不具合率
【NASA/WVU IV&V Facility, Metrics Data Program を元に著者作成】

第 3 章　先行研究レビュー

第 2 章までは、IT システムを構成するソフトウェア開発における品質事故の発生について、事例分析を行った。これにより、国内ソフトウェア開発の品質的欠陥は、ソフトウェアの相互依存関係性、納期短縮と開発規模増大に伴う開発密度の増大に伴い、製品出荷前の工程におけるテスト（検査）の十分な品質管理工程のための工程が行われていない、あるいは行われてはいるものの、充分ではないことが、ソフトウェアにおける品質的欠陥の発生を引き起こしていることがわかった。その一方、高品質とされる NASA や JAXA においては、特に品質管理における検査工程について強化を図ることにより、品質的欠陥の抑制をはかっている可能性が明らかとなった。

それでは、現状の国内ソフトウェア品質管理の仕組みは、なぜ品質的欠陥の発生を抑制

[24] 経済産業省 プロセス改善研究会.“ベストプラクティス調査報告書 ～究極の高品質ソフトウェア開発プロセスを目指して～ 独立行政法人　宇宙航空研究開発機構（JAXA）第 1.0 版 平成 19 年 4 月”. 情報処理推進機構：ソフトウェアエンジニアリング.15 頁

[25] 宇宙航空研究開発機構.“H-ⅡA ロケット解説資料”. HⅡA LAUNCH SERVICE.

できないのだろうか。

本章ではまず第1にJIS Z8101、ISO8402で、ユーザ要求を満足させる製品やサービスの特性[26]とされる「品質」の定義に従い、その品質を経済的に作り出す手段である品質管理にとって、重要な要素と見なすことのできるソフトウェア特性に関する研究領域、第2に事例研究にて明らかとなったソフトウェアの相互依存関係性や複雑性が品質管理に与える影響に関わる研究領域、第3に日本的なソフトウェア品質管理の特徴に関する研究領域、第4にソフトウェア開発において品質に影響を与える要因に関わる研究領域を概観するものとする。その上で、本章の最後で、本論文の位置づけ及び特徴を明らかにする。

第1節　ソフトウェアの特性に関する研究

ソフトウェア工学に対する古典的な研究の1つであるBrooks（1983）ではソフトウェアには複雑性、順応性、変更可能性、不可視性という4つの特質があるとしている。まず、複雑性という特質は、チームメンバ間の意思疎通を困難にし、その結果製品欠陥やコスト超過、スケジュールの遅延をもたらすとされる。順応性とはインターフェースや仕様の様式に必然性がなく、複雑性の大部分が他のインターフェースにあわせて改変することに起因するため、ソフトウェアだけを再設計しても、複雑性を単純化することはできないという特質である。次に変更可能性は、ソフトウェアはシステムとしての機能を実現したものであることにより、この機能へ変更の圧力がかかること、また融通性に富むため、安易に変更されがちであり、アプリケーション、ユーザ、慣習、文化、機器媒体類からなる文化的構造に左右されることで頻繁な変更が引き起こされるという特質であるとされる。最後に不可視性は､表現関係が存在せず､故に抽象空間に置き換えられないという特質であり、複数の人間で設計する場合のコミュニケーションが妨げられる要因になるとしている[27]。

ソフトウェア製品の特性に関して論じた研究には、保田（1995）がある。保田はソフトウェア自体とソフトウェア開発の持つ、ハードウェアとは異なる特質として以下の4点を挙げている。第1は論理の大規模集合体という特質である。大規模論理集合体を正確に組上げる技術は方向は見えているものの、まだ十分に確立されていないといわれている。そのためどのような技術や管理能力が必要なのかが十分認識されていないこと多いとされ、それが品質管理上の困難さを引き起こしていると考えられる。第2はソフトウェアの開発プロセスは目に見えないという特性である。目にみえないが故、作業進捗やアウトプットを容易に識別できず、問題発生を認識するのが遅れる傾向にあるとされる。第3は特にパッケージソフトの場合、不特定多数のユーザのニーズを把握、仕様化するかが難しいとされる。一方、本論文で取り上げているエンタプライズ系ソフトウェアや組込みソフトウェアのような請負ソフトウェアの場合は、大規模化、及びエンドユーザや経営者の要求が厳しくなることで、要求仕様取り纏めが困難になっているとされている。第4にソフトウェア開発の正否は開発者の個人の能力への依存度が高い。その反面個人の能力を定量的に把

[26] 品質はJIS Z8101で「品物又はサービスが、使用目的を満たしているかどうかを決定するための、評価の対象となる固有の性質・性能の全体」と定義、ISO8402では「ある“もの”の明示された又は暗黙のニーズを満たす能力に関する特性の全体」と定義される。

[27] Demarco,Listar（2006）33-36頁

握することが困難であり、特に外注の場合は、品質の予測が難しく、品質のばらつきが生じるとされる[28]。

第２節　ソフトウェアの相互依存関係性と複雑性に関する研究

事例研究で見たとおり、ソフトウェアには相互依存関係と、相互依存関係から生じる複雑性いう特性が内在する。ソフトウェアの相互依存関係について論じた先行研究に小山・武田（2001）がある。これによればソフトウェア開発では、目的のソフトウェアを開発するために、どこからソフトウェア構成要素を集めてきてどのように組み立てるか、構成間の相互依存関係をいかに削減するかが設計上の中心課題になるとしている[29]。

そして、このような構造間の相互依存関係に着目し、システム複雑性という概念を用いて説明した研究に柴田・玄馬・児玉（2002）があり、この研究ではソフトウェア製品がシステム複雑性を有するものの一つとして取り上げられている。この研究によれば、ソフトウェア製品では同一機能に対して多様な処理方法と選択肢が考えられ､これらの選択肢を、製品出荷前にテストすることは不可能に近いとしており、そのためラピッド・プロトタイピングのような「利用による学習」を製品開発に取り込み、ソフトウェア・システムを改善していくことが、技術開発の本質となっているとしている。しかし、情報処理技術の飛躍的な発展により従来は製品システムに一体化できないと考えられてきた技術情報でも、コンピュータ・ソフトウェアの形の製品として、体化可能になりつつあるため､「利用による学習」によって創出された多くの知識は現在では原理的に製品システムに体化可能であるが、現実にどの程度体化可能であるかはモジュラー、インテグラルという製品アーキテクチャに大きく依存するとしている。さらに、この製品アーキテクチャがインテグラル型アーキテクチャである場合には特定機能の不具合を修正するために相互依存関係を解きほぐす必要があり、かつ特定のサブシステムの修正が他のサブシステムへどのような影響を及ぼすのかが、容易には判断できないとしている[30]。なお、藤本（2004）で銀行のオンラインシステムに代表される、1 品 1 様の大きなエンタプライズ系の「受注ソフト」は基本的に、徹底的に全体の調整を図りながら開発する「擦り合せソフトウェア」であること、また組込みソフトウェアについても、ハードと密接に連動しなければならないということで、カスタム性が高く、コストの制約や圧縮の必要性が大きいので、パッケージソフトに比べれば「擦り合せ」よりのソフトになる こと[31]、すなわち、インテグラル型アーキテクチャであることが述べられている。

前項のソフトウェアの特性においても参照した Brooks（1983）ではソフトウェアの複雑性という特性について、少なくとも命令文単位では類似した部分のない他のどの構造物よりも複雑であり、かつ規模拡大は必ず異なる要素が増加するもので、殆どの場合、それらの要素は非線形に作用し、さらに複雑性を増すとしている。さらにその複雑さというものは偶然の産物ではなく本質的なもので、複雑性を取り除いたシステムは本質をも失うこと

---

[28] 保田（1995）14－17 頁

[29] 藤本・武石・青島（2001）162 頁

[30] 柴田・玄馬・児玉（2002）43－45 頁

[31] 藤本（2004） 154－155 頁

になると論じている。そして、ソフトウェア製品開発における古典的問題の多くは、この本質的な複雑性、及び規模拡大に伴って複雑性が非線形に増加することに起因しているとし、その複雑性のため、プログラムの状態を理解することはもちろん、その状態を記述することも困難となり、結果ソフトウェアは信頼性を欠くこととなるとしている[32]。

延岡（2004）では、日本のシステム開発企業を総括し、ある銀行のシステムを開発しても横展開できない事が多く、システムの中身も複雑なインテグラル形になり、コストが過度にかかり、高い利益には結びつかない事例が少なくないと述べられている[33]。すなわちエンタプライズ系ソフトウェア・システム開発企業の多くが、特定顧客の要求に応えるため、複雑性の原因となるインテグラルなシステム開発を行っていることを示唆している。また、立本（2007）においては、組込みソフトウェアについて、モジュール、インテグラルという概念を用いて「組み合わせ開発・擦り合せ開発」の効果を測定している。結果「擦り合せ量」が品質と開発リードタイムを引き上げないケースが認められたことから、本来必要な擦り合せ量よりも多くの擦り合せが要求され、本来必要のない修正工数が発生することによって必要のない手戻り工数、すなわち「無駄な擦り合せ」が発生しているのではないかと推察している[34]。

これら先行研究によれば、日本のソフトウェア開発企業は擦り合せ（インテグラル）型な開発を行っているため、システム複雑性が増大、その結果、品質的欠陥の検出ができないといった事態や、品質向上に寄与しない無駄な擦り合せが発生していることになる。しかし､その一方で品質に影響を与える複雑性そのものは 20 年以上前よりソフトウェア製品の本質として捉えられており、現在でもその複雑性が引き起こす品質的欠陥に対する根本的な解決方法は存在しない。

第３節　日本的ソフトウェア品質管理に関する研究

保田（1995）によれば日本のソフトウェアの品質管理は、歴史的にハードウェアで培ってきた日本的品質管理を導入した歴史的経緯もあって、ハードウェアと同様に工業製品としてとらえている点にその特徴があるとしている[35]。具体的には、ハードウェアと同等の工場制度の導入、合否判定と製品出荷の可否についての権限を有する検査部門の役割、コーディング後のテスト工程での不良管理を行う中間工程品質管理、QC サークル等全員参加の品質管理をその特徴としている[36]。

しかし、多くの先行研究ではソフトウェアの品質管理はハードウェアのそれとは異なるものと主張される。例えば、櫻井（2001）は、ソフトウェア産業における品質管理の意義について､ユーザの仕様通りに機能（適合品質への対応）をさせるだけでは十分ではなく、バグのなさ、使い易さ、性能、拡張性といった諸要因、一言でいえば、品質を重視すると

---

[32] Demarco,Listar（2006）33-34 頁

[33] 延岡（2004）259-261 頁

[34] 藤本（2007）399－402 頁

[35] 保田（1995）31 頁

[36] 保田（1995）31-32 頁

いう見方が定着しつつあるとしている[37]。また、伊藤（2001）は、ソフトウェアはサービスと同様の無形財であるものの、他方でサービスとは異なり、ソフトウェアはディスクなどに保存することができ、しかも情報としての価値を摩耗させることなしに、容易に複製できるという点に特徴を有することから、サービスとハードウェアの中間に位置する経験財であるとし、ソフトウェアの品質管理は、ハードウェアのそれとは切り離して議論する必要がある[38]と述べている。さらに、Cusmano（2004）は、最も基本的なこととして、ソフトウェア開発は製造活動ではなく、むしろ製品設計である[39]と主張しているし、「ソフトウェア品質知識体系ガイド」においてもソフトウェアは物理的実態を持たないために生産工程が存在しないこと、その製品特性により異なる点は多いが、どちらかといえばハードウェアの設計工程に近いものと述べられている[40]。

しかし、その一方でソフトウェアをハードウェアと同等の「ものづくり」と位置づける先行研究も存在する。藤本（2007）は、ソフトウェアを無形だが耐久的な媒体に設計情報が乗った商品であり、サービス業と製造業の中間であるとする[41]。そもそも藤本（2007）の主張は、「ものづくり」の核心とは「もの」というより、「設計」であり、顧客を喜ばせる新しい設計情報を創造し、媒体に転写し、顧客に向かう「設計情報のよい流れ」を作ることであり、それが物財（製造業）であろうと、サービスであろうと、等しく「ものづくり」[42]とするものである。

藤本（2003）では、このような「ものづくり」の概念を前提とした上で、製品の品質については「総合品質」、それを形成する「設計品質」、「製造品質」とに分類している。さらに、設計品質とは製品設計情報の内容（性能やスタイルなど）が消費者のニーズを的確に反映しているかどうかを示す指標であり、「製品品質」とは実際の製品がどの程度製品設計どおりに作られているかを示す指標としている[43]。

この「設計品質」と「製造品質」という品質概念を用いて、品質管理について論じた研究に藤本（2001）がある。これによれば品質管理とは本来「総合品質」と「設計品質」、「製造品質」に対応するはずであるが、実際には品質管理といった場合にはその殆どが「製造品質」のことである[44]と述べられている。品質管理活動のマネジメント対象である「製造品質」について、藤本（2003）では工程から素材への設計情報の転写の精度であるとする。

藤本（2001）では品質管理は「転写精度の累積的な劣化（誤差の累積）」という問題を抱えており、これをいかに防ぐかで製造品質が決まってくると述べられている[45]。また同概念を用いた場合の製造品質の管理・改善の手段（品質管理）は、①発信源である機械・作

---

37 櫻井（2001）69 頁
38 伊藤（2001）102 頁
39 Cusmano（2004）202 頁
40 SQuBOK 策定部会編（2005）63 頁
41 藤本(2007) 289 頁
42 藤本（2007）285-286 頁
43 藤本（2003）117-118 頁
44 藤本（2001）265 頁
45 藤本（2001）249 頁

業者の情報ストックの質を保つこと、②通信チャネル（加工作業）におけるノイズ除去、③製品側で受信された情報と発信された情報の事後照会と情報不適合（不良）のスクリーニング、④そもそもノイズに強い設計情報にすること、⑤設計情報を受信、吸収しやすいような製品側のメディアをあらかじめ選択すること[46]であるとしている。藤本（2003)、藤本（2007）ではこれらの手段に加え、いったん不具合が発生した場合、それが迅速に発見・是正されるように、品質不良情報のフィードバックループをなるべく小さくする（自主検査・自動化・仕掛品在庫削減・1個流し）のが基本であるとしている[47]。さらに、日本企業の多くやジャスト・イン・タイム企業の典型的な品質管理、すなわちTQCに代表される日本的品質管理のもとでは、そもそも不良を出さないことに力を入れる「品質作り込み」という生産思想が用いられているとも述べられている[48]。

藤本（2003）及び藤本（2007）では、このような品質管理の仕組を、日本的な「ものづくり統合生産システム」の製造品質面の組織能力とし、発信者側による情報転写精度（品質作り込み）を優先的に考える、次に受信者側の転写精度のチェック（検査）を考えるという「発信者優先のシステム設計」[49]、工程内の不良（誤転写）に関する情報を、発信源に迅速かつ確実にフィードバックできるループを確保する「不良情報のフィードバック」という2点に要約できるとしている[50]。

先に取り上げた「ソフトウェア品質知識体系ガイド」においても、ハードウェアの品質管理の基本原則、すなわち、「全員参加」の原則、「品質第一」の原則、「改善」の原則、「次工程はお客様」の原則という「5 ゲン主義」を適切にソフトウェアに読み替えて適用する必要があるとしており、具体的に役立つ“悪さ”の知識の抽出、体系化、蓄積を行うこと、また「品質の作り込み」により、より上流で“悪さ”の知識を子細に活用し障害を排除することで、継続的な品質改善が実現できるとしている[51]。

第4節　ソフトウェアの品質への影響要因に関する先行研究

ソフトウェア品質についての影響要因については特に米国で多数の研究がなされている。例えばHarter,Slaughter（2003）では10年間にわたるIT企業のシステム・インテグレーション部門の調査において、CMMレベルが1から3の範囲の中において、CMMプロセス成熟度の1%の増加が製品品質における1.25%の改善に繋がっており、継続的なCMMプロセス成熟度の向上が製造品質向上に関係していることを発見している[52]。

またKrishan,Kriebel,Kekre,Mukhopadhyay（2000）ではソフトウェア開発研究所の1つにおける56のプロジェクト観察によって、コード行数から測定されたソフトウェア製品の

---

[46] 藤本（2001）266頁

[47] 藤本（2003）131頁,藤本（2007）163頁

[48] 藤本（2001）266－273頁

[49] 藤本（2003）131頁では「品質保証（通信精度向上）の責任を受信者側（検査重視）ではなく、むしろできるだけ発信者側に負わせる」ことを「品質作り込み」としている。

[50] 藤本（2003）131－132頁,藤本（2007）161－164頁

[51] SQuBOK策定部会編（2005）64－65頁

[52] Harter,Slaughter（2003）pp.784-800

サイズ、プロジェクトメンバーの人員能力、プロジェクト初期段階での資源投資、CMMプロセス成熟度の遵守がそれぞれソフトウェア品質に統計的有意であることを発見しており、特に人員能力、CMMプロセス成熟度の遵守が製品品質に著しく影響しているとしている[53]。従ってこれらの先行研究からはプロセス成熟度が高くない組織においては、開発規模、要員能力とプロジェクト初期段階での資源投資といったものの他、ソフトウェア品質管理手法の一つであるCMMプロセス成熟度の遵守と向上がソフトウェア品質に関係することが明らかになっている。そしてCMM/CMMIの効果についてはその品質改善効果は非常に高く、中央値の改善度合いで48%もの効果がある[54]とされる。また国内事例においても、レベル5プロセス実践の効果として「要件定義・設計・コーディングの上流フェーズでの誤り検出率が60%から84%へ増えた。その結果、上流品質が向上したことで出荷後の不具合件数を40%以上も削減し，計画と実績の乖離半減の目標を達成できた」[55]との報告もある。

しかしその一方で、高いプロセス成熟度の組織においても品質問題は発生する。この点に関連する研究に、Argwal,Chari（2007）がある。この研究では 4 つの組織の 37 の CMM レベル 5 プロジェクトから収集したデータについて、製品サイズ（SIZE)、製品複雑さ（COMPLX)、スケジュール圧力（SP)、チームサイズ（TEAM)、人員能力（PERCAP)、必要条件仕様書品質（REQUAL)、管理者の経験（MGROL)、ソフトウェア産業の経験（INDEXP)、モジュール間の依存性に見る複雑性（MOD）をそれぞれの独立変数とし、従属変数に品質を据えての線形回帰分析を行っている。その結果、製造品質に統計的有意な要因はコード行数と機能ユニット数から測定された製品サイズのみであり、製品サイズが1%増加することで欠陥が 0.3%増加することが明らかになっている[56]。この結果はプロセス成熟度が高い組織における欠陥の発生は、製品規模のみに比例しているということであり、不良の密度は一定であることを示している。すなわち、この研究は、プロセス成熟度を高めても、不良率を改善することが難しいことを示唆している。

また、その他の研究として、Austin（2001）がある。この研究においてはゲーム理論モデルを使用することで、スケジュール圧力下の元では、開発者が品質について妥協するであろうことが示唆されている[57]。すなわち、スケジュールが短い場合、どのような品質管理手法が採用されていたとしても､品質は妥協され､品質的欠陥発生に繋がることとなる。

第5節　本論文と先行研究の違い

これまで、ソフトウェア品質影響要因と、品質管理に関わる主要な研究をレビューしてきた。これらの既存研究から明らかなことは、まず第1に日本のソフトウェア開発はハードウェアとは異なる特性を有するが、国内のソフトウェア開発はハードウェアと同様の品

---

[53] Krishan,Kriebel,Kekre,Mukhopadhyay（2000）pp745-759

[54] Giveson,D.L&Goldenson,D.R&Kost,k.“performance Result of CMMI -Based Process Improvement”.Software Engineering Instistute Carnegie Mellon.pp.11-12

[55] 藤原良一・本間敏夫・細谷和伸・中前雅之・遠藤和彦.“プロセス改善による高品質ITソリューションの提供に向けたCMMIレベル5達成への軌跡”.三菱電機 技報.5頁

[56] Argrawal.Chari(2007) pp.145-156

[57] Austin(2001) pp.195-207

質管理が採用されているということである。そして第2にソフトウェアとは、複雑な相互依存関係からなる非常に複雑な論理構造物で、その複雑性とはソフトウェアが有する本質的なものであるが故に、そもそも品質管理方法を含めて、品質的欠陥を抑制する根本的な解決策は存在しないこと、さらに国内ソフトウェア開発においては擦り合せによる開発が主流であるため、その複雑性をさらに増大させている可能性があるということである。第3に、ソフトウェアの品質向上に影響する要因は先行研究でおおよそ明らかになっているものの、それらは品質に影響するという関係を説明するにとどまり、現在の品質管理方法ではソフトウェアの開発規模が拡大すれば品質的欠陥は必ず発生するということである。

既に見たとおり、国内ソフトウェア開発企業は、ソフトウェア開発をハードウェアと同等の「ものづくり」と見なし、日本的な品質管理を実践してきた。ところが、先行研究には「ものづくり」における品質管理の視点から、ソフトウェアの品質的欠陥の発生に言及したものはなく、十分な説明がなされていないといえる。また、先行研究からすれば、ソフトウェア開発における品質的欠陥は、どのような品質管理をおこなったとしても、その発生をなくすことは不可能に等しいといえる。

そこで、先行研究を概観することで明らかとなった、このような課題に対し、本論文では以下のようなアプローチを取ることとする。まず、第４章で、国内ソフトウェア開発の現状においての品質的欠陥の発生における要因を検証する。第５章では、エンタプライズ系ソフトウェアと組込みソフトウェアの開発力強化、品質向上に関わる研究に取り組み、その成果を実践・検証するため、先進ソフトウェア開発プロジェクトを産学官の枠組みを越えた展開を実施、さらに藤本の「ものづくり論」の観点からの共同研究を行っているIPA/SECの所長である鶴保 征城氏に対して、現状の品質管理上の課題や問題点、及び藤本によって提唱される「ものづくり」の品質管理の概念を用い、ハードウェアと同等の品質管理が機能不全を引き起こす要因について、インタビューを通して、明確化する。

また、第６章においては、第４章で明らかにした、国内ソフトウェア開発企業における「ものづくり」と同等の日本的品質管理をソフトウェアに採用したことによる課題や問題点、及び生産機能を阻害する要因について、国内でNASAと同様に宇宙開発を担い、高品質ソフトウェア開発のベストプラクティスとも紹介されるJAXAに対し､どのように品質的欠陥の発生を抑制しているか、さらに一般的なエンタプライズ系、組込みソフトウェアとの違いはどこにあるのかについて、インタビューを行い、分析を行う。その上で、第７章において第４章で抽出した要因と、第５章、第６章の分析結果を基とし、国内ソフトウェア開発における適用可能性の検討及び、提言を試みることで、国内ソフトウェア開発における品質に関する先行研究を発展させるとともに､品質向上のため一助とすることを試みる。

第４章　国内ソフトウェア開発の現状

第１節　ソフトウェア開発規模と開発期間

冨永（2005）はソフトウェアの規模増大がちょっとした間違い（グリッジ）を招き、それが大事故に繋がるリスクを高めていると主張する。その中でも特に組込み系の膨張が急速であるとするが、大規模化・複雑化は他でも同様であり、エンタプライズ系ソフトウェア・システムでも膨らみがちであるとしている。そしてそのような複雑化は欠陥増加につ

ながり危険を増す[58]としている。そこでソフトウェア開発規模について確認を行うと、組込みソフトウェアにおいては**図 3** の通り、プログラムのサイズ、コード量について、「組込みソフトウェアの総行数（新規開発と既存の合計）」は、既に 1,000 万行以上の開発規模の製品が存在していることがわかる。また、その平均行数はすでに 100 万行近くになっており、この点でも相当量の規模となっているといえる。さらに、組込みソフトウェアについては先の事例で引用した記事にて、開発規模が拡大する反面、開発期間が短くなっていることが指摘されていた。例えば**図 4** は，携帯電話における開発期間と開発規模の推移を示しているが、これによれば、携帯電話におけるソフトウェアの規模は 2 倍にまで膨張する一方で、開発期間が半分になっている。すなわち、ソフトウェア開発企業は、約 4 倍の密度で開発を完了しなければならないということなり、このような構図が品質事故の発生に繋がっている可能性は十分にある。なお、組込みソフトウェアの平均開発期間については、企画・調査要求獲得からハードウェアとソフトウェアを含めたシステム総合テストまでの期間が約 50 週、組込みソフトウェアの製品開発のみに焦点を絞ると、ソフトウェアアーキテクチャ設計から、ソフトウェア総合テストの完了までが、約 30 週、約 7.5 ヶ月となっている[59]。

一方エンタプライズ系ソフトウェアにおいては、**図 5** のように 10 万行以下のプロジェクトが 6 割を占め、その平均も約 30 万行となっている。また 100 万行以上のプロジェクトも存在するモノの、ごく少数にとどまっている。さらに、システム稼働後 6 ヶ月間の 1,000 行毎の累計発生不具合数を示した**図 6** を見ても、小規模でもばらつきが大きいことから、コード行数で見た開発規模の大きさが直ちに発生不具合数に結びついているとはいえない[60]。また、工期計画値が、中央値が 8 ヶ月、平均が 9.3 ヶ月なのに対し、実績値は中央値が 6.5 ヶ月、平均が 8.3 ヶ月となっており[61]、当初計画よりも早く開発を完了しているということができる。この点を踏まえれば、組込みソフトウェアとエンタプライズ系ソフトウェアの開発規模において、規模の拡大や開発期間が焦点となるのは組込みソフトウェアであり、エンタプライズ系ソフトウェアにおける品質的欠陥の発生については、これらの影響を受けているためと断じることはできない。

---

[58] 冨永章.“ソフトウェア・グリッチの予防”.JISSJ Toc Member.3 頁

[59] 経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2007 年版組込みソフトウェア産業実態調査報告書－プロジェクト責任者向け調査－第 1 版改訂 平成 19 年 10 月”. 情報処理推進機構：ソフトウェアエンジニアリング.35 頁　なおここではソフトウェア要求定義の期間は設計期間と捉え除外した。

[60] IPA・SEC（2007）213 頁　なお、1,000 行毎の不具合密度につき、中央値を確認すると「新規開発」が 0.026、「再開発」が 0.032、「改修・開発」が 0.004、「拡張」が 0.000 となっており、「再開発」が最も高い値となっている。ただし、「改良・開発」の規模の対象範囲には変更・追加分のみが計測範囲となっており、システム全体の規模を示していない。

[61] IPA・SEC（2007）43 頁

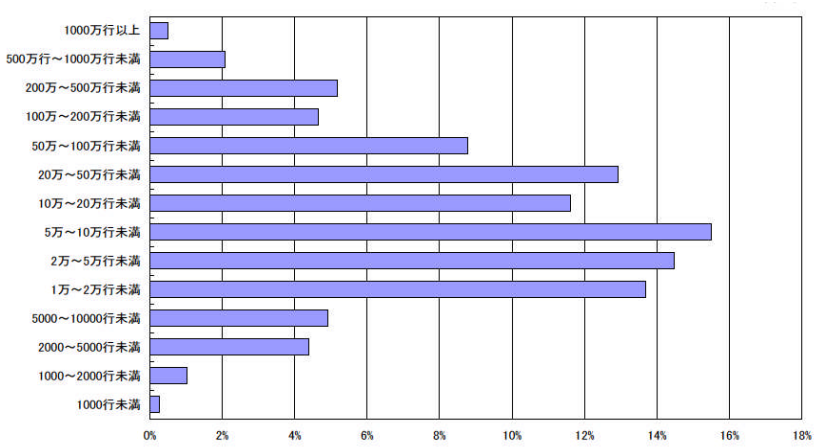

図 3　全行数（新規開発と既存の合計）

【経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2006 年版 組込みソフトウェア産業実態報告書－プロジェクト責任者向け調査－第 2 版 平成 18 年 8 月”33 頁】

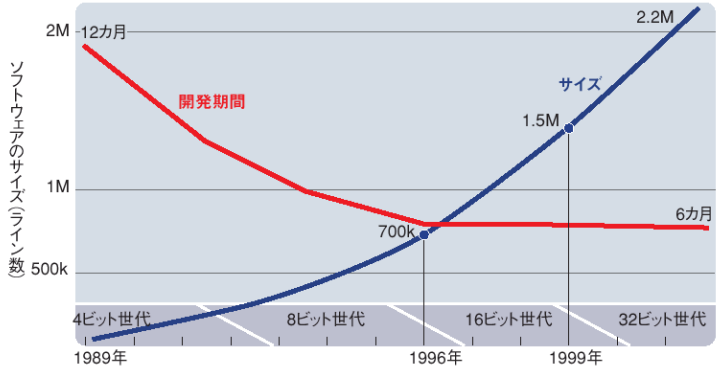

図 4　携帯電話のプログラムサイズの推移

【室修治“組み込み最前線(1) 難問山積のソフト開発 現場の頑張りはもう限界”
出典：平山雅之 ET2002 TB-6 「組込みシステム開発における品質向上の施策」】

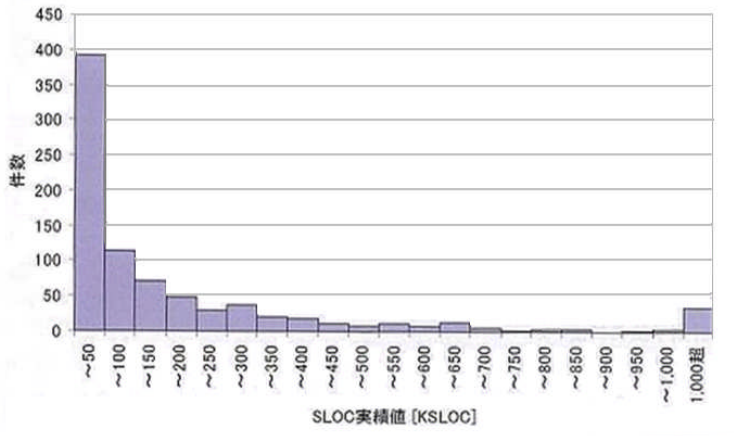

図 5　エンタプライズ系ソフトウェアの開発規模

【IPA・SEC（2007）「ソフトウェア開発データ白書 2007」42 頁】

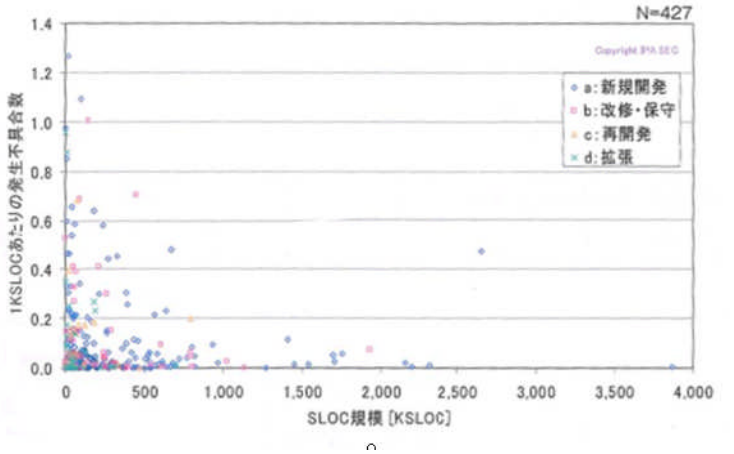

図 6 コード行数（SLOC）規模と不具合密度（主要言語混在）
【IPA・SEC(2007)「ソフトウェア開発データ白書 2007」213 頁】

第２節　ソフトウェア開発プロセス

Cusmano（2004）は、日本のソフトウェア開発プロセスの特徴を、「日本企業の組織的プロジェクトは殆ど「ウォーターフォール型」の手順に従うもので、概要設計から詳細設計、機能設計、プログラム作成、テストといった一連のステップを順番に進めるもの」[62]と述べている。ウォーターフォールモデルの問題点はいくつか存在する[63]が、最大の問題は不具合や要求事項との相違が開発プロジェクトの最後に行われるテストの段階まで抽出できないことである。最終段階まで不具合を発見できないため、上流で発生した不良を修正することがスケジュール的にも、コスト的にも困難となる。また変更要求が発生した場合には、その変更がソフトウェア全体に影響を及ぼし、膨大な手戻り作業を引き起こし、結果多大なコストの発生や納期に間に合わないこととなり、重大な品質事故の発生にも繋がっている可能性がある。

この国内ソフトウェア開発プロジェクトにおける、このウォーターフォール型プロセスの採用状況を確認すると、エンタプライズ系ソフトウェアでは**図 7** の通り 96.6%が[64]、また、組込みソフトウェア開発では**図 8** の通り 36.2%と最も高い採用率となっている[65]。また、組込みソフトウェアにおいてはスパイラル方式[66]、インクリメンタル方式[67]、アジャ

---

62 Cusmano（2004）204 頁

63 ウォーターフォールモデルの問題点は小泉・辻・吉田・中島（2003）17 頁に詳しい。

64 IPA・SEC（2007）34 頁

65 経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2005 年版 組込みソフトウェア産業実態報告書－開発プロジェクト責任者向け調査－改訂版 平成 17 年 6 月”. 情報処理推進機構：ソフトウェアエンジニアリング.80 頁

66 スパイラル方式とはスパイラル開発のこと。小泉・辻・吉田・中島（2003）19 頁に詳しい。

67 ここでいうインクリメンタル方式とは、インクリメンタル開発のこと。小泉・辻・吉田・中島（2003）132 頁、阪田・高田（2006）178 頁に詳しい。

イル方式[68]といった反復型プロセスの採用率が合計で 36.28%と、エンタプライズ系ソフトウェアに比べ高い採用率となっており、ウォーターフォール方式とほぼ同率での採用であることがわかる。しかし、プロセスと製品出荷後の不具合率の各指標について、これら反復型の方式とウォーターフォール方式の不具合発生率を比較した場合、**図 9** に示される通り、スパイラル方式がいずれの指標においてもやや低い水準にある以外、ウォーターフォール方式よりも不具合の発生率が高くなっている。「品質の未達成率」に焦点を絞り確認しても、インクリメンタル方式、アジャイル方式はいずれもウォーターフォール方式と比較し、その不具合発生率は高い水準となっている。さらに言えば、アジャイル方式については「開発費用の超過」が他のプロセスと比較して際だって高くなっている他、アジャイル方式、インクリメンタル方式ともに「開発期間の超過件数」において、標準的な開発プロセスがない場合とほぼ同等の水準となっている。以上より、ソフトウェア開発において注目を集めるアジャイル方式、インクリメンタル方式は、残念ながら現時点のところでは効果的な適用をおこなうことは困難であると言えそうである。

特徴的なのが、プロトタイピング方式が機能・性能や品質の確保に有効であるという事実である。プロトタイピングとは、「いくつかのシナリオについて設計者がプロトタイプを試作し、ユーザの検証・評価の結果をフィードバックして、プロトタイプを繰り返し修正し、要求仕様の獲得と確認をねらう」[69]プロセスを指す。保田（1995）では、このプロセスを、プログラム仕様、特に端末等のユーザインターフェースのビジュアル化による仕様の早期確定に有力な技術[70]と説明している。しかし、その一方で、阿部・吉沢（1998）においては、ユーザ仕様と乖離したプロトタイプ機能を開発してしまうこと、ユーザからの過度の仕様変更・ユーザ要求とプロトタイプ機能のミスマッチによって作業が長期化すること、また、過去のプロトタイピング情報が履歴として残っておらす、再利用やプロセス改善に利用できないという問題点も指摘されている[71]。すなわち、このプロセスを採用するプロジェクトでは、開発すべき機能や達成すべき性能といった明確な場合については、品質を確保しやすいものの、そうでない場合においては品質的欠陥の発生に繋がる可能性が高いものといえる。すなわち、開発プロセスのみに着目した場合、いずれのプロセスを採用しても、現状では品質的欠陥の発生を抜本的に解決することはできないといえる。

---

[68] ここでいうアジャイル方式とはアジャイルモデルのことであり、開発対象を多数の小さな機能に分割し、1 つの反復で 1 機能を開発、この反復のサイクルを継続し、1 つずつ機能を追加開発してゆく。Larman（2004）p.25 ではアジャイル開発について、「適応型の計画を行い、タイムボックスを適用しながら反復型、進化型の開発を行い、段階的に出荷し、機敏性（Agility）を高めるための価値やプラクティスを奨励する。アジャイル型にモットーがあるとしたら、それは変化を受け入れることである。アジャイル手法に戦略上重要な点があるとしたら、それは機敏性である。」と定義している。

[69] 小泉・辻・吉田・中島（2003）31 頁

[70] 保田（1995）35 頁

[71] 阿部・吉沢（1998）75－76 頁

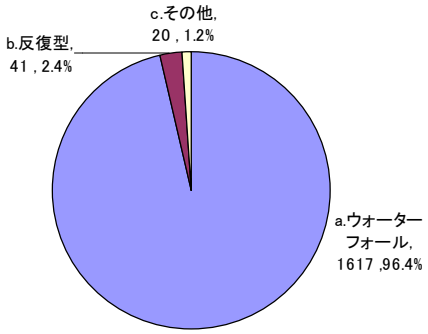

図 7 エンタプライズ系ソフトウェア開発ライフサイクルモデル
【IPA・SEC（2007）「ソフトウェア開発データ白書 2007」34 頁】

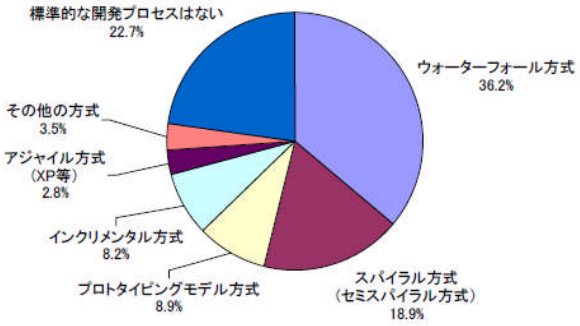

図 8 プロジェクトで採用した開発プロセス
【経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修“2005 年版 組込みソフトウェア産業実態報告書－開発プロジェクト責任者向け調査－改訂版 平成 17 年 6 月”93 頁】

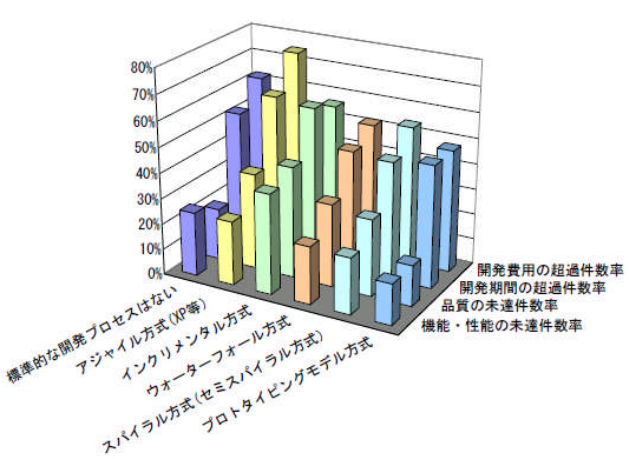

図 9 採用した開発プロセス別のプロジェクトの計画未達成率
【経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修“2005 年版 組込みソフトウェア産業実態報告書－開発プロジェクト責任者向け調査－改訂版 平成 17 年 6 月”116 頁】

第3節　受注ソフトウェア産業の構造

エンタプライズ系、組込みソフトウェアに関わらず、受注ソフトウェア業界には元請け・下請けのピラミッド構造が存在する。事実、IPA の調査によれば、**表 2** の通り、労働生産性は元請けの方が下請けより高くなっており、同報告書ではその理由として、元請け企業は労働集約的な業務を下請けに外注化する傾向があることが考えられること、および下請けの中でも「元請会社は系列会社（あるいは親会社）である」と回答した企業の労働生産性は際立って低い結果となっていることが報告されている[72]。

この調査報告では、主として自社内でのソフトウェア開発を行っている元請け企業の割合は、約 90%と高い数値となっているものの、一方で、例えば櫻井（2001）では、カスタム・ソフトウェア開発におけるソフトウェア産業の原価構造は、「ソフトウェア産業では伝統的に外注や下請けが多く、上場しているような大手の業者であれば実質的な人件費である 60～70%のうち、30%強は外注費である」[73]としている。すなわちエンタプライズ系では外注、下請けが多いこと、そしてそのコスト比率が高いことを示唆している。実際に、エンタプライズ系の外注に関わる調査では、一部を国内向けに外注しているケースが 40%強、国内派遣も 20%強の回答者が関係する業務で利用され、外注していない業務よりも多いこと、さらに、オフショアリングは 10%程度、関係する業務で利用されていることも報告されている[74]。さらに組込みソフトウェアに関しても 78.5%が開発を外部委託していること[75]、そしてそのうちの 25.4%が海外への外注を行っており、海外外注先としては中国が 39.6%で最も高く、次いでインドの 16．5%、米国の 12．9%となっていること[76]も報告されている。以上より、外注は国内受注型ソフトウェア産業においては当たり前に行われており、その一部はオフショアリングされていることは間違いない。

西村・峰滝（2004）では、このような受注ソフトウェアを含む、国内情報サービス産業における「外注化」を①「製品アーキテクチャのモジュール化」②「生産のモジュール化」③「企業間システムのモジュール化」のうち、「生産のモジュール化」として捉え、「生産

---

[72] 独立行政法人 情報処理推進機構.“第 29 回情報処理産業経営実態調査報告書（概要）”．情報処理推進機構：IT ベンチャー支援.3 頁　なお、本調査対象はエンタプライズ系、組込みソフトウェア開発企業 4,000 社となっている。

[73] 櫻井（2001）22 頁

[74] 独立行政法人 情報処理推進機構.“2007 年度産学連携ソフトウェア工学実践拠点事業エンタプライズ系ソフトウェア技術者個人の実態調査報告書 2008 年 5 月”,情報処理推進機構：ソフトウェアエンジニアリング.22 頁

[75] 経済産業省 商務情報政策局，情報政策ユニット情報処理振興課，組込みソフトウェア開発力強化推進委員会監修.“2007 年版 組込みソフトウェア産業実態報告書－経営者・事業責任者向け調査－第 1 版改訂 平成 19 年 10 月”．情報処理推進機構：ソフトウェアエンジニアリング.
128 頁

[76] 経済産業省 商務情報政策局，情報政策ユニット情報処理振興課，組込みソフトウェア開発力強化推進委員会監修.“2007 年版 組込みソフトウェア産業実態報告書－経営者・事業責任者向け調査－第 1 版改訂 平成 19 年 10 月”．情報処理推進機構：ソフトウェアエンジニアリング.134 頁

のモジュール化」が生産性向上に寄与しているかどうかについての研究を行っている。同（2004）では「外注比率（売上高に占める外注費の割合）」を「モジュール化の」代理変数とし、生産性に対する定量的な効果の測定を行った結果「外注比率」は情報サービス産業の生産性に正の効果を与えていないばかりか、むしろ負の効果を与えているという事実を発見している[77]。

「外注化」が生産性に負の効果を与えているとするのであれば、それはソフトウェアの品質にどのような影響を及ぼすのであろうか。櫻井（2001）は、品質の向上は、設計とレビュー、テストの徹底、開発工数全般にわたる強化策により達成しうるが、品質向上を徹底させようとすると、どうしても工数を増加させ、従って労務費の増加を招来し、少なくとも短期的には生産性が低下するという現象をみることになるとしている[78]。すなわち、情報サービス産業全体において観察された、生産性の低下、あるいは生産性への負の効果を及ぼす「外注化」は、さらに「生産性」を低下させることとなる「品質向上」のための工程は､犠牲にならざるを得ないという状況が発生していることが想像に難くない｡また、エンタプライズ系、組込みソフトウェアに関わらず、労務費全体の低減と、生産性向上を実現するため、オフショアリングによる開発を行っていることも間違いないであろう。

	企業数	労働生産性（円）
元請け	223	6,415
主として自社内でのソフトウェア開発を行っている	189	6,451
主として外注業者によってソフトウェア開発を行っている	23	6,533
下請け	236	3,719
元請け会社は系列会社（あるいは親会社）である	29	3,480

※元請け・下請けと回答した企業のうち労働生産性が計測可能な企業（459社）を対象とした。

表 2　元請け・下請けと労働生産性

【独立行政法人 情報処理推進機構“第 29 回情報処理産業経営実態調査報告書（概要）”3 頁 】

第 4 節　ソフトウェアの品質特性

ソフトウェアの品質モデルはソフトウェアの品質を階層構造のモデルで表現したものである。歴史的にみれば 1976 年 TRW 社の Boehm が階層構造を持つ品質モデルを提案した。次いで 1977 年 Walter と McCall が、品質特性を「利用者視点」､「開発者視点」､「測定可能なメトリクス」の 3 層構造を持つ品質モデルとして提案。その後ベンダが続いた。しかし、品質を比較評価するためには、複数の品質モデルの併存は、ソフトウェア品質を比較評価するのに不都合が生じる。そこで 1980 年代の後半から ISO で標準化作業が始まり、その後 1991 年に ISO/IEC 9126「ソフトウェアの品質特性」が規定された[79]。日本では 1994 年にこれが翻訳され「JIS X1029 ソフトウェア品質の評価－品質特性及びその利用要領」として発行されている。

「JIS X1029 ソフトウェア品質の評価－品質特性及びその利用要領」ではソフトウェア製造品質に対して「内部品質及び外部品質」の品質モデルを規定、ソフトウェアの内部品質及び外部品質モデルを機能性、信頼性、使用性、効率性、保守性、移植性の 6 つの品質

77　西村・峰滝（2004）168－205 頁

78　櫻井（2001）69 頁

79　SQuBOK 策定部会編（2005）26 頁

特性及び、それぞれの特性で細分化した品質副特性で表現していた。しかし、その後 2001 年には ISO/IEC9126 を発展強化した新しい ISO/IEC 9126 シリーズが発行され、新たに有効性、生産性、安全性、満足性からなる「利用時の品質」が品質特性として追加されている[80]。

以上のようなソフトウェア品質特性の変遷を**表 3** に示す。保田（1995）が指摘するとおり、品質特性は普遍的なものではなく変化するものであり、ソフトウェアの適用分野が広がるにつれて、要求される品質特性の範囲は拡大傾向にあること[81]は間違いない。

また、これらの品質特性、品質副特性を定量的に計測する方法は品質メトリクスと称されるが、ISO、JIS、IEEE 規格、研究者によりその定義は様々であるとされており[82]、表の通り、ベンダ独自のものも存在する。従って業界の標準といえるものは存在しないものといえる。

	Boehm	MaCall他(RADIC)	SQMAT(NEC)	SQUALAS(IBM)	ISO/IEC 9126(2001)
外部及び内部品質		Correcteness	正確性	機能性	機能性
	Reliability	Reliability	信頼性	信頼性 アベイラビリティ サービサビリティ	信頼性
	Human Engineerling		使い易さ	操作性 習得容易性	使用性
	Effeciency	Effeciency	効率	パフォーマンス	効率性
	Maintainability ├ modifiability ├understandability └testability	Maintainability Testability	保守性	保守性 拡張性	保守性
	Portability	Reuseability Portability Flexibility Interoperability	 接続性		移植性
			セキュリティ	セキュリティ インテグリティ	
利用時品質					有効性
					生産性
					安全性
					満足性

表 3　品質特性の比較表【保田（1995）6 頁に一部著者追記】

第 5 節　日本のソフトウェア品質管理の現状

第 4 節にて論じたような、拡大する品質特性に対処するために、エンタプライズ系ソフトウェア、組込み系のソフトウェア開発企業は ISO9000 ファミリーで規定される品質マネジメントシステムと、その品質マネジメントシステムを組織的かつ継続的に改善するための、例えば CMM/CMMI のような、プロセスアプローチの導入を推進している。事実、エンタプライズ系ソフトウェアについても**表 4** の通り、例えば日本 IBM や NTT データ等多くのエンタプライズ系ソフトウェアのソフトウェア開発を手がける組織が、ISO9000 シリーズにおける品質マネジメントシステムの要求事項を規定する ISO9001 の審査登録、及び CMM/CMMI での成熟度レベル認定に積極的に取り組んでいる。また、経済産業省がまとめた「2005 年版組込みソフトウェア産業実態調査報告書」によれば、「品質管理として採用している方法」という問いに対し、**図 10** の通り「ISO9000 シリーズの準拠」については回答事業部門の約 7 割が、「CMM に準拠した標準的な管理プロセス」については約 3

[80] 「利用時の品質」の品質特性に対する品質副特性は ISO/IEC 25000 「Software product Quality Requirements and Evaluation (SQuaRE)」シリーズで検討中である。

[81] 保田（1995）5 頁

[82] SQuBOK 策定部会編（2005）200-201 頁

割が採用していると答えている[83]。

品質マネジメントの管理対象は主に欧米を中心として発達した結果系に焦点をあてる検査重視型のマネジメントと、要因系、すなわち日本において発達してきた、製品とサービスを作り出すプロセス重視マネジメントがあり、CMM/CMMI やシックスシグマはこのようなプロセス系重視のマネジメント[84]とされる。ISO9000 シリーズが一般的に検査重視型のマネジメントであるとされることを考え合わせるのであれば、ソフトウェア開発企業の多くはこれまでの結果系に要因系の品質管理手法を採用し、結果系、要因系両輪での品質管理により品質向上を達成しようとしていることが伺える。

企業名	グループ会社など	ISO9001（特定部門・組織のみも含む）	CMMIレベル（2008年8月現在）	CMMI認定組織
CSK	独立系	認定済み	5	中部事業本部：レベル4 関西事業本部：レベル5
CSKシステムズ	CSK	認定済み	5	西日本事業本部
JFEシステムズ	JFE	認定済み	–	
NECソフト	NEC	認定済み	–	
NTTコムウェア	NTT	認定済み	5	システム開発部門の4つのパイロットプロジェクト
NTTデータ	NTT	認定済み	2～4	国内外グループ企業
TIS	三菱東京UFJ銀行	認定済み	4	全事業部
TKC	独立系	認定済み	–	
SRA	独立系	認定済み	3	
アイネス	日立製作所	認定済み	–	
インテック	独立系	認定済み	2	
インフォメーション・ディベロップメント	独立系	認定済み	–	
オービック	独立系	認定済み	–	
さくらケーシーエス	三井住友ファイナンシャルグループ	認定済み	–	
ジャステック	独立系	認定済み	5	
住商情報システム	住友商事	認定済み	–	
トランスコスモス	独立系	認定済み	5	大宇宙信息創造（中国）有限公司）全社
日本IBM	米IBM	認定済み	5	サービス・デリバリー部門（SI・AP運用保守）
日本ビジネスコンピュータ	日本IBM	認定済み	–	
日本ユニシスソフトウェア	日本UNISYS	認定済み	5	金融部門
日本システムソフトウェア	独立系	認定済み	–	
野村総合研究所	野村證券	認定済み	5	公共保険向け部門
菱友システム	三菱重工業	認定済み	3	BS事業部（本社）
日立ソフトウェアエンジニアリング	日立製作所	認定済み	5	
富士通ビジネスシステム	富士通	認定済み	4	
東芝インフォメーションシステムズ	東芝	認定済み	5	開発センター
三菱総研DCS	三菱総合研究所	認定済み	–	
三菱電機インフォメーションシステムズ	三菱電機	認定済み	5	金融および流通・サービスの2事業分野

表 4　国内の主なエンタプライズ系ソフトウェア開発企業の品質管理手段【著者作成】

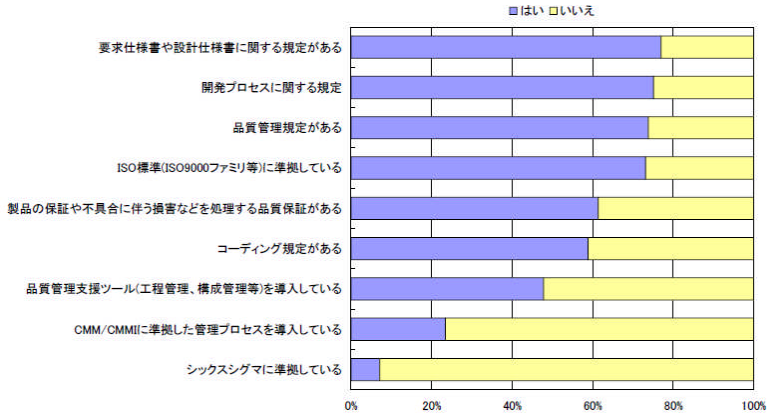

図 10　組込みソフトウェアの品質管理として採用している方法

【経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2006 年版 組込みソフトウェア産業実態報告書－経営者・事業責任者向け調査－第 2 版 平成 18 年 8 月 115 頁】

[83] 経済産業省 商務情報政策局，情報政策ユニット情報処理振興課，組込みソフトウェア開発力強化推進委員会監修.“2005 年版 組込みソフトウェア産業実態報告書 －経営者・事業責任者向け調査－改訂版 平成 17 年 6 月”. 情報処理推進機構：ソフトウェアエンジニアリング.80 頁

[84] SQuBOK 策定部会編（2005）57 頁

第６節　品質への意識と人に関わる諸問題

先行研究からすれば、ソフトウェアは個人への依存度が高い。これはすなわち、個人の能力やモラールに依存する部分が大きいことを示している。エンタプライズ系、組込みソフトウェアそれぞれにおいて、品質に関わる意識、及び人員の能力につき確認を行ったところ、エンタプライズ系においては、「品質計画」について特に問題ないとする回答が 38.7%、「品質管理」について特に問題ないとの回答が 36.2%を占めており、現状の品質に関わる意識については決して高いものではないという実態を示していた。また、問題と認識されている割合の高い「品質管理の方法」や「品質基準の達成方法が決まっていない」、「品質管理の方法が明確になっていない」など、品質に関する問題意識は**表 5** の通り、下請けに行くほど強いという結果も出ている。しかし、2 次下請以下では改善が進んでいないこと、および外注管理相手の品質コントロールは難しいとの結果も報告されている[85]。

また、同表を見ると、人員能力についても、「必要なスキルを備えた人材が不足している」、「プロジェクトメンバーのスキルが足りない」、「チーム力が発揮されない」等、人に係る問題意識がどの階層でも高い傾向にあることが明らかとなっている。その他、解決すべきとする課題として認識されている割合についての報告では「スキル」30.9%、「要員調達」26.4%となっているほか、現在 3 人に 1 人は転職を考えており、これが業界における人材流動化の実態となっていることも指摘されている[86]。すなわち、人員の慢性的な不足、ただでさえ不足している人員の流動、そしてスキルの問題と、エンタプライズ系ソフトウェア開発では人に関わる問題が深刻であることを見て取れる。

一方組込みソフトウェアにおいては、「組込みソフトウェア企業の問題意識」として品質を高めることが非常に重要との回答が**図 11** の通り 70%と最も多く、品質に対する意識は高いものの、**図 12** の通り、スキルや経験を有する人材が不足していること、また品質に関わる部分で言えば、特に QA スペシャリストや開発プロセス改善スペシャリストの不足が深刻であるといえよう。

---

[85] 独立行政法人 情報処理推進機構. “2007 年度産学連携ソフトウェア工学実践拠点事業エンタプライズ系ソフトウェア技術者個人の実態調査報告書 2008 年 5 月”, 情報処理推進機構: ソフトウェアエンジニアリング. 183 頁

[86] 独立行政法人 情報処理推進機構. “2007 年度産学連携ソフトウェア工学実践拠点事業エンタプライズ系ソフトウェア技術者個人の実態調査報告書 2008 年 5 月”, 情報処理推進機構: ソフトウェアエンジニアリング. 183 頁

順位	ユーザ	(%)	元請	(%)	1次下請	(%)	2次下請	(%)	
1	必要なスキルを備えた人材が不足している	46.65%	必要なスキルを備えた人材が不足している	56.91%	必要なスキルを備えた人材が不足している	60.81%	必要なスキルを備えた人材が不足している	53.59%	スコープ
2	プロジェクトメンバーのスキルが足りない	36.19%	プロジェクトメンバーのスキルが足りない	49.41%	プロジェクトメンバーのスキルが足りない	53.85%	チーム力が発揮されない	41.99%	タイムスケジュール
3	チーム力が発揮されない	35.39%	チーム力が発揮されない	45.43%	チーム力が発揮されない	42.12%	プロジェクトメンバーのスキルが足りない	39.23%	コスト管理
4	「情報共有」の方法が明確でない、決められていない	28.95%	コストの見積もりが正しくできていない	41.22%	コストの見積もりが正しくできていない	38.46%	品質管理の方法が明確になっていない	38.12%	品質管理
5	コストの見積もりが正しくできていない	27.88%	要求定義・仕様が不明確	41.22%	要求定義・仕様が不明確	37.00%	要求定義・仕様が不明確	34.25%	人的リソース
6	責任の所在が明確になっていない	26.54%	「情報共有」の方法が明確でない、決められていない	34.66%	品質基準の達成方法がきまっていない	31.87%	リスクが明確になっていない	33.15%	コミュニケーション
7	品質管理の方法が明確になっていない	25.74%	責任の所在が明確になっていない	33.02%	「情報共有」の方法が明確でない、決められていない	28.57%	品質管理の方法が周知されていない	31.49%	リスク管理
8	要求定義・仕様が不明確	24.40%	進捗管理の方法が明確になっていない	31.85%	品質管理の方法が明確になっていない	28.21%	品質基準の達成方法がきまっていない	29.83%	
9	役割分担が明確になっていない	23.86%	実績報告を適切なタイミングで行われていない、行っていない	30.21%	そもそも無理なスケジュールを組んでいる	28.21%	そもそも無理なスケジュールを組んでいる	29.83%	
10	進捗管理の方法が明確になっていない	21.72%	品質基準の達成方法がきまっていない	29.98%	責任の所在が明確になっていない	27.47%	コストの見積もりが正しくできていない	29.28%	

表 5　産業階層別の問題意識の相違

【独立行政法人 情報処理推進機構 “2007 年度産学連携ソフトウェア工学実践拠点事業 エンタプライズ系ソフトウェア技術者個人の実態調査報告書 2008 年 5 月”182 頁】

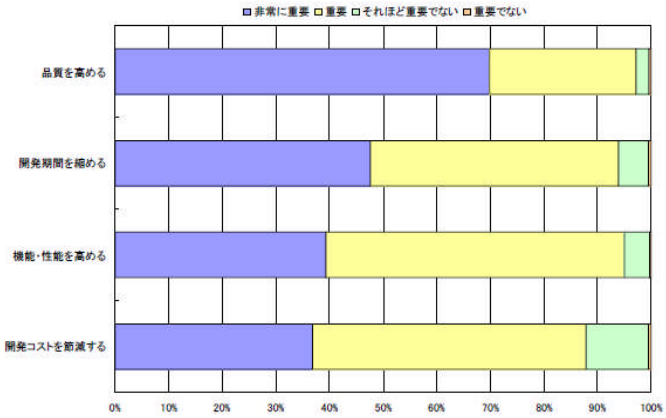

図 11　組込みソフトウェアの改善すべき領域

【経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修“2004 年版組込みソフトウェア産業実態調査報告書 平成 16 年 6 月” 33 頁】

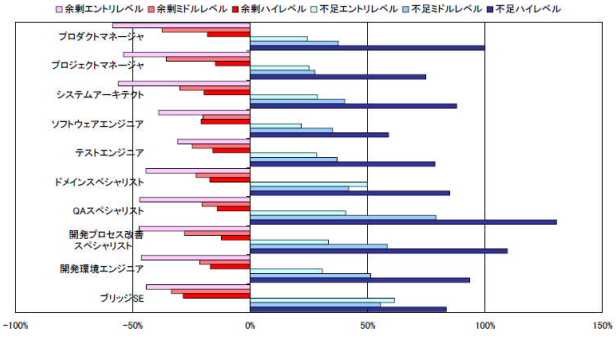

図 12　技術者のキャリアレベル別不足率：現状と最適の比較・社内と社外の合計

【経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2007 年版組込みソフトウェア産業実態調査報告書－プロジェクト責任者向け調査－第 1 版改訂 平成 19 年 10 月” 71 頁】

第７節　日本のソフトウェア品質の実際

Cusmano,MacCormack,Kemerer,Cranbell（2003）では日・米・インド・欧州における出荷後12ヶ月でのトータルソースコード行数の1,000行毎不具合率の中央値について比較が行われている。これによれば**表 6**の通り、日本のソフトウェア開発の不具合率は0.020となっており、米・インド・欧州と比較し、およそ1/13～1/20という低い値となっている[87]ことが指摘されている。すなわち他国と比較した場合には極めて高い品質水準にあるということができる。しかし、エンタプライズ系ソフトウェア、公共系を含むソフトウェア開発プロジェクト、組込みソフトウェアでも、前述の通りソフトウェアの品質的欠陥混入による品質事故が発生しており、また現在も発生し続けている。

実際に国内の組込みソフトウェア産業の調査結果からは、製品出荷後の品質問題発生原因の46.3%がソフトウェアの不具合に起因していることが明らかになっている[88]。またエンタプライズ系ソフトウェアにおいては納品後の品質に関して、全く問題がなかった、あるいはそれ程大きな問題がなかったとの回答が8割以上を占めるものの、問題が頻発する企業も散見されるとされる[89]他、別の調査では保守・追加開発の内容として「バグ修正」が20%を占めること、また保守作業後1年以内にバグが発生する確率が16%であることも報告されている[90]。

	India	Japan	US	Europe & other	Total
Number of projects	24	27	31	22	104
Median output[1]	209	469	270	436	374
Median defect rate[2]	.263	.020	.400	.225	.150

1. No. of new lines of code / (avg. no. of staff × no. of programmer-months).
2. No. of defects reported by customers in 12 months after implementation / total source LOC. We adjusted this ratio for projects with less than 12 months of data.

表 6　各国の欠陥密度の中央値
【Cusmano,MacCormack,Kemerer,Cranbell（2003）P.32】

第８節　品質的欠陥発生要因とその影響の考察

ここまで検証してきた結果から、国内ソフトウェア開発における品質的欠陥の発生要因を考察する。ソフトウェアは今やものづくりや、社会や企業において欠かせないものとなった。そのため、特に組込みソフトウェア分野においては規模と規模拡大、擦り合

---

87 Cusmano,MacCormack,Kemerer,Cranbell（2003）P.32-33

88 経済産業省 商務情報政策局，情報政策ユニット情報処理振興課，組込みソフトウェア開発力強化推進委員会監修.“2007年版 組込みソフトウェア産業実態報告書－経営者・事業責任者向け調査－第1版改訂 平成19年10月”．情報処理推進機構：ソフトウェアエンジニアリング.170頁

89 ソフトウェア・エンジニアリング・センター.“ソフトウェアエンジニアリアリングの実践強化に関する調査研究「エンタプライズ系ソフトウェアソフトウェアにおけるSE度の実態調査」”．情報処理推進機構：ソフトウェアエンジニアリング.14頁

90 日経コンピュータ「ニュース＆トレンド 保守費は5年で初期開発費の8割強に JUASが82プロジェクトの保守・追加費を調査」2006年5月1日号,16頁

せ開発により複雑性が増大しているにも関わらず、納期は短縮化する傾向にある。一方エンタプライズ系については、規模の拡大や、納期までの期間が短いという事実は明らかではないものの、新規開発プロジェクトで不具合発生数が多い、あるいは、開発、導入後に多くのバグ修正が発生していることは間違いない。このことは、特にエンタプライズ系においては、開発されたソフトウェアそのものが、初期開発時には製品として本来出荷できるレベルに達していない可能性が高いことを示唆する。さらに、ソフトウェアに求められる品質特性は拡大し続けているため、「不具合」「欠陥」と見なされる範囲そのものも、拡大を続けているともいえるであろう。

また、プロセスとして、様々な問題が指摘されるウォーターフォールモデルを採用し続けること、あるいは、注目を浴びる反復型開発が、品質のみならず、コストや開発期間においても問題が多く、さらに、ソフトウェア開発においては、下請け構造、すなわち「外注化」が品質的欠陥を発生させる一因となっていること、そして、第 7 節で明らかにした海外と比較して非常に低いレベルに抑えている欠陥率を、オフショアリングすることによって悪化させている可能性があることも注目に値する。すなわち、現在の国内ソフトウェア開発のプロセス、産業構造には、根本的に品質的欠陥を抑制する方策がないばかりでなく、むしろ、この問題を増大させている可能性すら存在するのである。また、エンタプライズ系においては要員の品質への意識の低さ、要員・スキルの不足、組込みソフトウェアにおいてはキャリアを積んだ人材の不足が問題であり、意識改善やスキル向上、人材の確保といった取り組みも必須であるといえよう。また、品質的欠陥の発生に関連するこれら諸要因は、労働集約産業という特徴を持つソフトウェア開発において、最終的に開発要員への圧力や負担という形へと変容し、新 3K とまで揶揄されるような労働環境を生みだし、業界の魅力の減退に繋がり、優秀な人材を遠ざけ、結果、人材が不足し、そのことが新たな品質事故の発生に繋がっているという可能性すら示唆するものである。

このような現状に対し、国内ソフトウェア開発企業は特に品質管理に力を入れることで、問題の解決をはかろうとする姿勢が見て取れる。つまり ISO9001 や CMM/CMMI 等、欧米で発展を遂げた品質管理手法を、人、コスト、時間という多くの資源を消費し、採用することで、品質的欠陥の抑制に努めているといえる。もちろん、その根底には「そもそも不良を入りこませない」ことを前提とした、日本的品質管理が行われているとされていることも忘れてはならない。しかし、品質的欠陥が発生し続けている現状を踏まえれば、その取り組みは十分に機能していない、あるいは機能不全が生じているのではないのであろうか。

第５章　国内ソフトウェア開発企業における品質的欠陥発生要因

第１節　調査概要

本調査の目的は、エンタプライズ系、組込みソフトウェアという、国内受注型ソフトウェア開発における、「ものづくり」と同等の品質管理をソフトウェアに適用した場合の、品質的欠陥の発生に関わる要因を明らかにすることにある。調査方法には、定量的調査手法であるアンケートを基にした調査と、定性的なインタビュー調査が存在する。第３章でも

触れたように、先行研究には日本的な「ものづくり」における品質管理の視点から、ソフトウェアの品質的欠陥の発生に言及したものはなく、また定量的調査はもちろんのこと、定性的調査による説明変数の抽出すらなされていない。従って、本論文では品質的欠陥発生要因の仮説探求を行うことが主要な目的となるため、インタビューによる調査を実施することとした。

調査対象は、エンタプライズ系、組込みソフトウェアに関わらず、国内ソフトウェア開発において「ものづくり」という視点から幅広い領域での調査・研究、あるいは実践している企業や機関が望ましい。そのため、本調査においては、本論文でも度々引用している、エンタプライズ系ソフトウェア開発統計調査を行った「ソフトウェア開発データ白書」の編纂を行い、また、本論文の基礎となる「ものづくり」の品質管理の概念を提唱した藤本が主催する東京大学経済学部ものづくり経営研究センター（MMRC）との共同調査研究プロジェクトで行う調査研究を行うなど、国内ソフトウェア産業における開発効率化・品質向上の手法と環境の整備を行っている IPA/SEC の所長である、鶴保 征城氏に対し半構造的インタビューを実施した。

第２節　インタビュー結果と分析

品質管理が機能しない要因、すなわち情報転写精度を劣化させる誤差を累積させ、結果的に品質的欠陥を発生させることとなる要因について、第２章、第３章、第４章において行った分析からすれば、以下の通り考えられる。すなわち、多くのエンタプライズ系ソフトウェアや、組み込みソフトウェア企業に関わらず、ハードウェアと同等の工場制の下で採用されている、ウォーターフォールプロセスでは、上流から下流に常に工程が流れるため、誤転写情報のフィードバックループは、必然的に大きくならざるを得ない。さらに、工場制の下では、合否判定と製品出荷の可否については製造部門とは異なる検査部門が権限を有し、コーディング後のテスト工程での不良管理を行う中間工程品質管理を行うとされていることからも、検査重視であり、決して品質作り込みを重視したものとなっていないことが想定される。にもかかわらず、事例研究に見たとおり、その検査そのものも、充分に行えていない。つまり、そもそも不良を入り込ませないという思想に基づいた、ハードウェアと同等の品質管理を採用しているといいながら、実際にはハードウェアのような品質管理は行うことができない、あるいは機能していないため、不良が入り込むものと考えられる。

本節では、IPA/SEC の所長である、鶴保 征城氏へのインタビュー結果について、これらの分析結果を基に、最初にソフトウェア開発において「ものづくり」と同等の品質管理を行っているにも関わらず不良が入り込む要因を、次に「ものづくり」と同様の品質管理で転写精度の累積的劣化を引き起こす要因について整理・分析を行った。

第１項　「ものづくり」と同等の品質管理で不良が入り込む要因

まずはじめに、現在の国内ソフトウェアにおける「ものづくり」と同等の品質管理を行うソフトウェア開発で、なぜ不良が入りこんでしまうのかという根源的な問題がある。そこで、この点に関する質問を行った結果、以下のような回答を得た。

ソフトの場合はハードにおける製造というものが、殆どゼロなんですね。ハードウェアにおけるような製造工程がないと言ってもよい。つまり設計がずっとつづいているという感じなんですね。その違いが非常に大きい。　　（SEC 所長 鶴保 征城氏）

すなわち、ソフトウェア開発とは設計であり、もともと製品を製造するための具体的な工程の存在する「ものづくり」とは異なり、製造品質の転写精度を高めるための仕組みである品質管理が行えない可能性が高いといえる。しかし、このインタビュー結果は、先行研究における「ソフトウェアとは設計である」との主張がソフトウェア開発における認識として存在していることが、確認できたにとどまり、「ものづくり」と同等の品質管理が機能不全を引き起こす要因とはいえない。また、例えば事例研究に見たとおり、組込みソフトウェアにおいては、現在はソフトウェア無くして製品機能は実現できず、それ故、ハードの中核部品の一つとなっているともいえる。ソフトウェア開発は確かに製造であっても実際は設計であるともいえるが、そもそも設計が 100%のものになれば、モノは動くのか。そこに「ものづくり」で確立されたとされる品質管理はどのように活かされているのか。

第２項　転写精度の累積的劣化を引き起こす要因

藤本（2001）において、製造品質の管理・改善の手段である日本的な「ものづくり」の品質管理は、発信者側による情報転写精度（品質作り込み）を優先的に考え、次に受信者側の転写精度のチェック（検査）を考える、第 2 は工程内の不良（誤転写）に関する情報を、発信源に迅速かつ確実にフィードバックできるループを確保するという 2 点に要約できるとされている。すなわち、「ものづくり」と同等の品質管理で、情報転写精度の累積的劣化を防止できない要因を明らかにするには、この 2 点が実際のソフトウェアの品質管理でどのような状況にあるかを確認することが重要といえる。

（１）　検査重視

（ソフトウェア開発を）擦り合わせ型でやっているということは、組み合わせ型に比べたら、品質保証が難しい面があるんですね。組み合わせ型っていうのは、それぞれの部分が正しく動くことを保証すれば、全体を組み合わせたものも、ある程度正しく動くと。それは組み合わせ型の特徴であり、アーキテクチャという議論ですね。

日本人はどちらかというとそういう考え方が苦手なんですね。つまり、モジュールに分けるよりも、いわゆる垂直統合で作るのが得意という点がありますね。

だから、テストや、品質保証の方から見ると、非常にやりにくい方式なんだけれども、日本人のきめ細やかさで、特にハードは良い品質のものをつくってきたわけですね。しかし、このやり方は規模が大きくなると不利なんですね。　　（SEC 所長 鶴保 征城氏）

上記は、「ソフトウェアにおける日本的品質管理上の負の側面」という質問に対しての回答である。先行研究で見たとおり、エンタプライズ系、組込みソフトウェアといった受注開発ソフトウェアの製品アーキテクチャは擦り合せ型であること、さらに擦り合せ型アーキテクチャは、もともと検査による品質管理を行うことが困難であるばかりか、規模が拡大すればさらに品質保証（品質管理）を難しくする要因となっているという。

しかし、日本的な「ものづくり」の品質管理は本来、検査ではなく、発信者側による情

報転写精度（品質作り込み）を優先的に考えるものであったはずである。

例えば藤本（2003）によれば、自動車においては、30 万点の構成部品を利用し、新モデルではその構成部品のうち、特殊設計部品を 60%～80%利用している。そして、このことをもって自動車をインテグラルな製品として分類しており[91]。トヨタではそのインテグラル型製品における製造品質を確保するため、検査重視ではなく、品質作り込みを優先する生産システムを確立しているのである。すなわち、ソフトウェア開発における品質管理は「ものづくり」と同等の品質管理を標榜しているにも関わらず、現実にはその理念や思想とは全く異なる、検査優先の品質管理を行っていることが伺える。

テスト項目をどうやって選ぶかについても重要なシステムでも明確な基準がないんじゃないかと思うんですね。単体試験やなんかはテスト項目の選び方など検討されるが、総合試験になってくると人にまかされている傾向が強い。　　（SEC 所長 鶴保 征城氏）

しかも、鶴保氏によればソフトウェアにおいてはテストの基準すら明確でないという。ソフトウェアにおける総合試験とは、藤本（2001）でいう、ハードウェアの出荷検査（最終検査）に近いものである。同（2001）によれば、このような出荷検査（最終検査）は、検査基準の厳しさ、正確性、検査が不正確である場合には、不良品が出荷されるリスクがある[92]とする。この点は、第 2 章の事例研究の結論である、テスト工程不足の問題にも関係するものといえる。ただし、保田（1995）によれば、検査とは、「ソフトウェア製品を何らかの方法で試験（テスト）した結果を、品質判定基準と比較して、合格・不合格の判定を下すこと」としており、また、英語で検査に該当する「Inspection」とは、ハードウェア分野では日本語の「検査」とほぼ同等であるものの、ソフトウェアにおいてはレビュー技法という意味で使われるのが普通であるとしている[93]。なお、その相違は**表 7** にまとめた通りである。

すなわち、テストにおける基準がないとは、レビューと呼ばれる公式の評価手法によりそのテストの内容や結果について合否を判定するための基準もないということになり、少なくともソフトウェア製品の最終的な出荷成否を決定するための最終検査においては、検査重視であるともいえない。

その意味では、予め決められた明確な合否判定基準に従って結果の記録と妥当性の評価をする ISO9001 のソフトウェア産業における実行の実態について疑問を持たざるを得ない。

最近はインドなどで、テストだけを請け負う会社があるんですね。ところが、テストを請け負うときに、仕様書を要求するわけですね。日本の場合はその仕様書がわりときっちり書いてないから。インドの業者から見たらテストもできないという問題が発生しているんですね。　　（SEC 所長 鶴保 征城氏）

---

[91] 藤本（2003）100-104 頁

[92] 藤本（2001）270 頁

[93] 保田（1995）37-38 頁

また、オフショア開発の進展により、テスト工程のみを外部委託することも増加しつつある昨今、本来そのテストを行うために必要な情報が仕様書の問題で正確に伝わらず、そもそもテストすらまともにできないことが指摘されており、この点についても結果的に品質的欠陥が発見できないという事態を生じさせている可能性がある。

検査の定義(JIS)	検査の定義(ISO8402)	Inspectionの意味	Inspectionの定義(IEEE)
品質をなんらかの方法で試験した結果を、品質判定基準と比較して、個々の品物の良品・不良品の判定を下し、又はロット判定基準と比較して、ロット合格・不合格の判定を下すこと	ある“もの”の各特性の適合性を確定するために、一つ又はそれ以上の特性を測定、審査、試験又はゲージ合せをして、その結果を要求事項と比較する活動	検査{けんさ} 綿密{めんみつ}な調査{ちょうさ} 点検閲覧{てんけん えつらん} 検閲{けんえつ} 検品{けんぴん} 査察{ささつ} 実況見分{じっきょう けんぶん} 視診{ししん} 検討{けんとう}	1) ソフトウェア要求仕様、設計またはコードを、作成者以外の個人またはグループが詳細に調査し、欠陥や開発規格の違反及びその他の諸問題を発見するための公式の評価手法 2) 品質管理の1過程であって、そこでは試験、観測又は測定などの手法によって、材料、支給品、構成要素、部品、付属品、システム、処理又は構造などが、あらかじめ定めた品質要求に合致するかどうかを判定する

表 7 検査と Inspection の定義【保田（1995）25-26 頁をもとに著者作成】

（２）フィードバックループの確実性と迅速性

第 2 節冒頭に、多くのソフトウェア産業が採用するウォーターフォールプロセスが誤転写情報のフィードバックループを大きくしている可能性について述べた。「品質作り込み」に関する質問をおこなったところ、このウォーターフォールプロセスに関する見解として、インタビュイーより以下のような回答を得た。

設計工程においても品質のつくりこみは重要です。この設計に対して、ウォーターフォール型の開発モデルが、適用が難しいという問題があり、スパイラル型だとか、繰り返し型の議論がされています。このことが一番大きい問題ですね。　（SEC 所長 鶴保 征城氏）

すなわち、ものづくりにおける品質管理上の要諦ともいえる、不良情報のフィードバックループを増大させる要因として想定していた、ウォーターフォールプロセスがその要因ではなく、そのフィードバックループを縮小するための手段でもある、繰り返しプロセスが、「品質作り込み」上の問題であるという。
鶴保氏の回答を要約すれば、製造途上での顧客の変更要求がなければ、ウォーターフォールモデルは品質保証をしやすい。しかし、国内ソフトウェア開発においては、製造途上での変更要求が発生するため、第３章で掲示したようなアンケート上ではウォーターフォールの切れ目で契約を行うため、ウォーターフォールとの回答が多くなるが、その開発形態はアジャイル開発に代表される反復型となっており、ウォーターフォールがうまく適用できないことが「品質作り込み」を困難にしているというのである。

最近のマーケット情勢から判断して、こういう機能が優先的だとか、他との会社の関係でこの機能が欲しいとか、それだったら、納期に間に合うとか、間に合わないとか議論しながら、完成度を高めていくわけです。こういう開発スタイルというのは、ウォーターフォールの機能変更、追加要求でもあるし、アジャイルとも言える。(SEC 所長 鶴保 征城氏)

アジャイルは、ウォータフォールの考え方で仕様書をつくって、後で修正する、優先順位が変わるということと、同じですね。はじめに決めた仕様で、納期まで変わらないということはありえないんですよね。

後は作り方の問題ですね。どういうことかと言うと、はじめに仕様を決めたとしても、受注側としては発注側の様子を見ながら、この機能は一応こう書いてあるんだけど、変わる可能性が強いというところは作らないですね、後で発注側とまた相談しようと思って。そうやっているわけだね。現実は。（SEC 所長 鶴保 征城氏）

相手側との議論の中で、この機能は仕様書にはこういう風に書いてあるんだけれども、危ないとか、この機能はもう殆んど間違いないから、最優先で作っていこう、という風にプロジェクトマネージャは考えているわけですね。アジャイルというのはこのような現実の開発スタイルを形式化したものと考えるのが自然だと思う。

だから、組込みにしろ、エンタプライズにしろ、多くのプロジェクトはアジャイル的な開発スタイルになっていると思う。（SEC 所長 鶴保 征城氏）

すなわち、実際にアジャイル開発プロセスにおいて行われるとされる段階的な出荷を行うという開発を行っているということではなく、ウォーターフォールであるとされていても、追加・変更が発生することを開発当初より想定しているため、実質的な開発においては、プロジェクトマネージャーやソフトウェア開発企業が個別に反復型プロセスのような開発を行っていると述べている。

藤本（2007）によれば、製造業らしい製造業の場合、生産過程で媒体に転写される設計情報は「構造情報」であり、それは消費空間でユーザに使用されて「機能情報」に変換されるとする。これに対し、サービス業らしいサービス業では、設計情報を運ぶ媒体は無形のエネルギーであり、設計情報は有形のものを媒介せず、直に顧客に転写される[94]とする。

ソフトウェアは製造業とサービス業の中間とされるが、サービス業のように設計情報の保存がきかないわけではない。むしろ製造業と同じく、生産過程で媒体に転写される設計情報としての「構造情報」は電子媒体に保存され、それが消費空間、すなわち実際にユーザに使用されて始めて、「機能情報」として転写されている。しかし、媒体に設計情報を転写したとしても、1 顧客のみに向けた受注生産であるが故、使用が行われない製造途中には様々な機能変更要求がユーザサイドから生じ、その都度「構造情報」は変化するといえる。そのため、プロジェクトマネージャーや開発組織は反復型のような工程設計を実質行っていると考えられる。

しかし、これだけでは「品質作り込み」ができない理由にはならない。なぜなら、変更が生じるであろう受注生産制はなにもソフトウェア開発に限ったことではなく、また、この内容はあくまで転写される設計情報が常に変化するというだけで、フィードバックループを小さくできないという説明にはならないからである。

それでは「品質作り込み」、製造品質の組織能力の 1 つである、特にフードバックループが小さくならない、あるいはそれができない理由に焦点を当てた場合、どこに問題があ

94 藤本（2007）290－291 頁

るのであろうか。インタビュー結果を整理すれば以下のようになる。

発注側では、この機能はこういう風に変えないといけないということが、わかってるんだけれども、モジュール化された後工程、例えば、受注側の中国の方ではそういう話は全然聞いていないということが起こるとこれは最悪なんですね。

最後の総合試験の時に、発注側は設計変更は言ってあるじゃないかと言い、元請けはうちは聞いてますと言うが、下位の下請けは聞いてませんということになる。それが一番まずいわけですね。発注側なり、元請けが意図したことが、末端まで伝わらないといけない。誤りなく伝われば、中国で作っていても、なんとかなるかもしれないのだけれども、そのためには、例えば受発注システムで代理店と製造会社が議論して、こういう風に仕様が変わったんだということと、各モジュールにどういう風な影響があるんだということが把握できる必要があります。（SEC 所長 鶴保 征城氏）

すなわち、変更そのものがフィードバックループそのものに影響を及ぼすわけではなく、下請け構造が事実として存在する国内ソフトウェア開発の構造において、そもそも設計変更の内容が、例えば下請けソフトウェア開発企業や、昨今拡大する海外オフショア等に対し、モジュールまで含め、明確に伝わらないことこそ問題であり、結果的にその変更に起因するものが不良として顕在化した場合、当然ながら設計変更を伝えるルートとは逆のルートとなるフィードバックも遅くなることを示唆するものである。すなわち、本来の情報転写においては、情報の発信元である元受が仕様書という形で体化した情報 M が、本来であれば、第 2 工程にあたる下請け企業で M+A、第 3 工程にあたる孫受けで M+A+B という形で製品設計情報（付加価値）が転写されることにならなければならない[95]。しかし、変更が生じた場合、その情報が、第 2 工程、第 3 工程に伝わらないことにより、結果として不良となる可能性がある。このことは同時に、各工程における検査工程における不良情報も原因工程に伝達されるまで時間がかかることとなり、そのため作業改善の勧告まで長い時間が必要となったり、不良と認識できないまま、出荷される可能性があるといえる。

第 3 項　その他の品質影響要因

ソフトウェア開発とは典型的な労働集約産業である[96]。それ故、先行研究でも明らかな通り、個人への依存がいまだ大きく、品質に関しても、設計者や開発者といったプロジェクト要員の能力に依存する部分が存在している。

この点に関し、鶴保氏は製造工程で多数の人員が入った場合に、これらの人員は仕様書にある「決まったもの」を作ればよいとだけ考え、例えば変更が合った場合にも、その影響について熟考しない等、モラールやモチベーションの問題を指摘する。

また、特にエンタプライズ系のソフトウェア開発においては、自分でソフトウェア開発を行えるだけの、設計能力、製造能力が欠如していることも大きな問題との認識を示す。例えばトヨタの場合は、トヨタテクニカルディベロップメントというトヨタ自動車 100%出

[95] 詳しくは藤本（2001） 266-271 頁を参照されたい。

[96] 櫻井（2001） 22 頁

資のトヨタの子会社があり、デンソーやアイシンといったパートナー企業向けのソフトウェア開発を委託するため、十分に開発を行うに足る設計能力、開発能力に裏打ちされた、コーディングまでも記載された詳細な「仕様書」作成を行っている。ところが、特にエンタプライズ系ソフトウェア開発企業の場合、受注元であり下請け企業への発注元となる企業や組織のプロジェクト要員は、設計能力、実装能力を有さないままで「仕様書」を作り、下請け企業や子会社に任せることで、品質、価格面にも影響を生じさせているとする。

このような要員に関わる能力については、IT 業界で問題視されており、例えば山下（2007）では、IT ベンダ企業では、新卒者全体で IT 教育研修を受けても業務に従事できないレベルが 2 割、組込みソフトウェアでも 1 割存在することも報告されている。また、新卒時のプログラム能力はインドや中国にかなわないともしている。しかしその上で、企業内教育と現場経験によって培われた職人的技術者の能力と、製造業から引き継いだ生産管理等の手法を組織的に適用したことで、日本のソフトウェア開発能力は国際的に引けをとるものではないとする[97]。とは言うものの、特にエンタプライズ系元請け組織の要員、また下請け開発を行う組込みソフトウェアの元請け組織の要員においても、下請け構造により、ソフトウェア開発そのものを外部委託してしまえば、現場経験で設計や開発、実装に関わる能力を醸成する機会も少なくなる。その結果として鶴保氏が指摘するように設計能力、開発、実装能力が失われることになり、その結果、品質へ影響を及ぼしているとは考えられないだろうか。

第 4 項　ビジネス構造の品質への影響

「ものづくり」の品質管理の仕組みを阻害する要因の一つとして、不良情報のフィードバックの問題や、検査重視という問題がオフショアなども含む下請け構造の問題により発生している可能性が高いことこれまでの調査により明らかとなった。さらに品質にも影響を及ぼす設計能力、開発能力が要員や組織から失われた要因についても、下請け構造により引き起こされている可能性がある。そこで、本節では、鶴保氏のインタビューから、特に下請け構造が生じた原因について分析する。

生産性を上げるためには部品化しなきゃいけないというところにあります。モジュール化ですね。これは経済の常識ですね。自社でつくらない部品を外部から調達するということですね。しかし、ソフトの場合、部品でものをつくろうという発想がなかなかうまくいかなかった。その代わりに、モジュール化の代理変数として、工程のモジュール化っていうのを考えたわけですね。つまり、工程を分割して一部を外部に出せれば、ハードのモジュール化に匹敵するような経済効果があるんじゃないかと、発想したのです。例えば、設計は自社でやるけども、コーディング、テストは外部に出すと。その結果、業界の多重下請構造ができあがっているわけですね。しかし、一方で、設計、コーディング、テストというものが、なかなかうまく分割できないということがわかってきたけれども、業界構造は残ってしまった、というのが、今の業界の大きな問題であり、不幸なところがあるわけですね。（SEC 所長 鶴保 征城氏）

[97] 山下（2007）44－58 頁

つまり、生産性を高めることを目的とし、工程分割、モジュール化したものの、機能させられず、しかし、業界には多重下請け構造がそのまま残ってしまったことが原因であり、この多重構造こそが、ソフトウェア開発での「ものづくり」と同等の品質管理が機能不全を引き起こす、あるいは組織や要員から設計能力、開発能力を奪う大きな要因となっているものといえる。

第３節　議論と分析

本論文では、エンタプライズ系、組込みソフトウェアというハードウェアと同等の「工場制」を敷く、受注型ソフトウェア開発における品質的欠陥発生要因に対する解釈の一つの視点として、藤本のいう「ものづくり」における品質管理の概念を用いて分析を行ってきた。本節では、改めてこの概念の枠組みに照らし合わせてインタビュー結果を整理する。

ハードウェアと同等の品質管理を行うソフトウェア開発には、ハードウェアのような工程が殆ど存在せず、設計が続いている。藤本（2003）によれば、いったん設計図面に固定された情報は「製品設計→工程設計→工程→製品」という「品質連鎖」を通じ次々に転写されていくが、その過程で伝言ゲームのように、転写精度の劣化（誤差の累積）が生じることとなり、これをいかに防ぐかで製造品質が決まる[98]とする。ソフトウェア開発においては生産そのものの工程が殆どなく、設計が続いているとするのであれば、「製品設計→製品」となり、事実上、製品設計情報そのものがダイレクトに顧客に転写されることとなる。すなわち、製品設計情報の成否こそ品質そのものであるといえる。しかし、実際のところ、設計だけではソフトウェアは動かない。一般的に「仕様書」や「フローチャート」として整理された設計情報を解釈し、その「仕様書」や「フローチャート」など抽象的な設計文書の内容を、プログラミング言語を使って具体的なコードに変換するコーディングを行い、単体テスト、組み合わせテスト、総合テストを行い、媒体に転写し、事後的に品質判定基準に照らし合わせての検査による合否判定を行い、最終的に出荷される。つまり、そこに工程は存在している。また、事実、工程が存在しているからこそ、「工程のモジュール化」が行えるといえるのではないだろうか。

しかし、実態として「ものづくり」においての品質管理の精神は活かされているとは言い難い。そもそも「ものづくり」における品質管理、すなわち統合生産システムの製造品質面における組織能力とは本来、発信者側による情報転写精度（品質作り込み）を優先的に考え、次に受信者側の転写精度のチェック（検査）を考える、工程内の不良（誤転写）に関する情報を、発信源に迅速かつ確実にフィードバックできるループを確保するという仕組みでなければならない。しかし、ソフトウェア開発においては、「品質作り込み」よりもテスト・検査重視であるばかりか、そのテストや検査の基準も明確でない。また、工程のモジュール化によって生じた多重下請け構造は、設計変更に関わる情報の伝達不良情報の伝達速度を劣化させ、フィードバックや作業改善を行えない原因となっていることも示唆している。

以上より、国内ソフトウェア開発においては、「ハードウェア」と同等の品質管理を行

---

[98] 藤本（2003）118 頁

っているとされてはいるものの、自動車に見られるような「ものづくり」における品質管理が機能していないことが確認されたといえる。

第４節　対策

本論文では品質的欠陥の発生要因のみでなく、その対策についても明らかにすることを目的としている。以下に本聞き取り調査から明らかとなったソフトウェア品質向上のための対策について整理を行う。

第１項　製品アーキテクチャ

まず、本調査から明らかとなったのは、ソフトウェアの製品アーキテクチャに関する対策である。先行研究でも、組込みソフトウェアの調査において、本来必要な擦り合せ量よりも多くの擦り合せが要求され、本来必要のない修正工数が発生することによって必要のない手戻り工数、すなわち「無駄な擦り合せ」が発生していることは既に述べた通りであり、この研究の帰結は「現状の組込みソフトウェア開発では、どこを組み合わせで行い、どこを擦り合せるかという議論があまりにも少ない。そのため擦り合せ部分が多すぎる」[99]というものであった。また、先行研究からも、聞き取り調査からも、国内受注型ソフトウェア開発においては、擦り合せ開発を行っていることは明らかである。

鶴保氏によれば、特に規模の拡大が進む国内ソフトウェア開発においては、オープンなインターフェースのもとに､製品アーキテクチャやプラットフォーム という考え方をはっきり打ち出すこと、特にモジュール化、すなわち藤本・武石・青木（2001）の言うモジュラー化[100]を進めることが、品質保証という観点においては重要であるとしている。モジュラー化そのものは藤本・武石・青島（2001）で「モジュール内の相互関係とモジュール間の集約された相互関係に分けられる。構成要素間の調整、統合化は簡略化されて、必要とされる複雑性処理能力の総量は減少されることになる」[101]と指摘していることからも、ソフトウェアが具有する相互依存関係や複雑性について縮減できうる可能性がある。しかし、同時に藤本・武石・青島（2001）ではモジュラー・アーキテクチャについて､「各部品で見ると、それぞれ自己完結的な機能があり、一つ一つの部品に非常に独立性の高い機能が与えられている」[102]としており、これは第３章で見た、組込みソフトウェアで採用されているプロセスモデルの１つであるインクリメンタルモデル、すなわちソフトウェアを独立性の高いサブモジュールに分割する方式に近いものと考えられる。しかし、当該モデルによる開発では、現状では品質に問題を残しているとの結果を考え合わせるのであれば、そのモジュール自身の「欠陥」をゼロとすることなしには、品質的欠陥の発生は撲滅できないであろう。

---

[99] 立本（2007） 402 頁

[100] 藤本・武石・青木（2001）32 頁にて「モジュラー化」と「モジュール化」は同一の概念を表す言葉として扱われている。

[101] 藤本・武石・青木（2001） 5 頁

[102] 藤本・武石・青木（2001） 54 頁

第２項　設計段階

鶴保氏によれば、ソフトウェア開発の場合何を実現するかという部分が最終的な品質に強い影響を及ぼすため、記述方法含め、上流工程である設計段階で、設計の方法そのものを変えていく必要があり、そのために特に重要なのは「モデリング」であるとしている。

「モデリング」とは例えば自動車等の製造業においては、一般的に CAD/CAM[103]を利用した設計開発、特に CAD による設計を「モデリング」と称する。これに対し、ソフトウェア開発における「モデリング」とは「目的意識を持って対象を抽象化し、それを読み方、書き方の定められた記述方法に基づいて表現すること」[104]を指すが、いずれにせよ情報から関係や構造などを形式的に表現する手段であるといえるであろう。

藤本（2001）においては、「モデリング」を行うためのソフトウェアである CAD/CAM が、いくつかのボディ設計のステップの省略、情報伝達の自動化によるヒューマンエラーの排除の手段であり、品質管理における転写精度累積的な劣化（誤差の累積）を防ぐための対策とされている[105]。また、延岡（2004）では、3 次元 CAD がもたらす効果として、商品開発の初期段階における製品設計者と生産技術者の間のコミュニケーションを促進、これを媒介することで、後工程と前工程の間で協同による問題解決が促進されること、製品設計者自らが生産条件を盛り込んだ設計図面を作成しやすくなること、設計者自身が比較的高度な解析を実施できるようになることでフロント・ローディング[106]が可能となることを指摘する[107]。また、その一方で組込みソフトウェア開発においては、フロント・ローディングをして、前段階のシステム設計とソフトウェア設計の段階で問題を発見し解決することが求められるものの、実際には、後工程の実装段階とテスト段階で、ほとんどの問題が顕在化することが多いこと、フロント・ローディングのベストプラクティスと呼ぶことができる仕組みがないことが述べられている[108]。

鶴保氏は、エンタプライズ系であれば、BPMN（Business Process Modeling Notation）[109]と呼ばれる表記法、組込みソフトウェアであれば UML（Unified Modeling Language）[110]

---

[103] CAD（Computer Aided Design)とは一般的に作業者がコンピュータの画面上で設計、3 次元形状化をするためのソフトウェアであり、CAM（Computer Aided Manufacturing）とは製品の製造を行うために、CAD で作成された形状情報を入力データとして、加工用の NC プログラム作成などの生産準備全般をコンピュータ上で行う為のソフトウェアのことである。

[104] 阪田・高田（2004）147 頁

[105] 藤本（2001）249 頁

[106] 延岡（2004）209 頁 にて「フロント・ローディング」とは関連する（相互依存性が存在する）部品間や機能間での擦り合せを、プロジェクトの少しでも早い段階で、徹底的に実施することであるとしている。

[107] 延岡（2004）223－224 頁

[108] 延岡（2004）231－232 頁

[109] BPMN（Business Process Modeling Notation ）は、White,S.A“Introduction to BPMN”.BPMN Information Home.p.1 において、BPMI（Business Process Management Initiative）によって開発された標準のビジネスプロセスモデリング表記法であるとされる。

表記、状態遷移表[111]を使う必要があると述べている。また、これらの表記法を用いた設計をおこない、現在できつつある「モデリング」された内容を自動的にプログラム化する技術を用い、試作品や一部分のプログラムを生成する、ただし、パフォーマンス等には課題を残すため、最終版の主要部分については手作りコーディングを行うというといった仕組みを導入することが品質を確保するためには重要であるとする。

前述の藤本、延岡の先行研究からすれば、品質管理における転写精度累積的な劣化（誤差の累積）を防ぐための対策としても、ソフトウェア開発におけるフロント・ローディングを実現するための一つの方法となることも期待できる。

第３項　欠陥への対処

先行研究で明らかな通り、ソフトウェアの品質に有意である CMM/CMMI のレベル５を取得していたとしても、規模が大きくなれば欠陥は混入する。本調査においても鶴保氏は現状では限りなく欠陥をゼロに近づけることはできるものの、完全にゼロにすることはできないとの認識を示した。

そのため、止まった時の影響を考え、人手で対処するための仕組みを組み込むことが重要であるとする。また、レアケースやイレギュラーなケースについては、それをソフトウェアに作り込むことで、相当にややこしいものとなり、それが新たな欠陥を発生させることも考えられるため、ソフトウェアの作り込みの対象から除外すること、またオーバーコントロールとなった場合の対処についての専門家を交え、充分に議論し、手動や物理的な対処の仕組みを考えることが必要であるとする。

欠陥は発生するものとして、重要な機能に対する欠陥発生時の対処策を十分に想定し、その回避策をあらかじめ製品全体で考え、実装しておくことは、生命や財産を守るという点において、非常に重要である。ただし、ソフトウェア産業界全体では、欠陥をそもそも発生させないための方法や仕組みについて、さらに検討を進めていく必要があるであろう。

第６章　高品質事例の調査と分析

第１節　調査概要

本章では、第５章までの結論を基にし、ソフトウェアの品質的欠陥の発生による報告が認められておらず、外部から観察した場合に、高品質を達成するための一つの対策となっているとも考えられるソフトウェア IV&V を導入する、JAXA の情報・計算工学センター 主幹開発員　片平 真史氏、情報・計算工学センター 開発員 宮本 祐子氏に対し、半構造的インタビューを実施した。

---

[110] 阪田・高田（2004）151 頁において、UML とはオブジェクト指向のソフトウェア開発における、標準化されたモデリング記法と説明されている。

[111] 阪田・高田（2004）151 頁において、状態遷移表とは、内部状態と入力イベントに対するリアクティブ（即時に反応する）な振る舞いを捉える状態遷移図を表による表現にしたものと説明されている。

第２節　分析結果

国内ソフトウェア開発において、宇宙開発であっても、ソフトウェアの品質的欠陥発生の要因として大きな違いはないはずである。それにも関わらず、これまで見てきたようにエンタプライズ系、組込みソフトウェアにおいては品質的欠陥の発生が報告されているにも関わらず、JAXAにおけるソフトウェア開発においては高品質を達成できているのか。もちろん、エンタプライズ系、組込みソフトウェアで報告される品質的欠陥の報告数と、JAXAにおけるソフトウェア開発には、もともと製品として出荷される根本的な母数の違いが純然たる事実として存在するものの、エンタプライズ系や組込みソフトウェア開発とは一線を画す環境にあることや、品質管理への取り組み方法や思想が異なることも考えられる。

従って本節では第４章にて検証したソフトウェア開発上品質的欠陥の発生要因、品質確保のための取り組みについてどのような差異があるかにつきインタビュー結果を分析した。

第１項　品質的欠陥発生要因に見る差異

表 8 は本インタビュー結果において、これまでの議論における品質的欠陥要因に関してまとめたものである。結論からすれば、これまで見てきたソフトウェアの品質的欠陥の要因とも考えられる事項について、ソフトウェアの開発期間、品質特性への対処、品質への意識といった点に受注型ソフトウェア産業に見られる特徴との違いが見られる。

区分	規模(SLOC)	開発期間	プロセス	外注(下請け)	品質特性への対処	品質管理手段	品質への意識
JAXAのソフトウェア開発	・数万行～数十万行	・衛星の開発期間：4～5年 ・そのうちの一定期間 ※詳細は公表できず	・基本はウォーターフォール ・特異な例として反復型あり	あり	・品質メトリクスによる詳価を要求 しかし、特に基準は定めていない ・検査としてソフトウェアIV&Vを実施	ISO9001 （事業部で異なる）	品質を最重視
組込みソフトウェア開発	・(平均)100万行 ・1,000万行以上のプログラムも存在	・企画・調査要求獲得 ～システム総合テスト： (平均)約1.25年 ・ソフトウェア・アーキテクチャ設計～ ソフトウェア総合テスト： (平均)約7.5ヶ月	・ウォーターフォール （36.2%） ・反復型合計 （36.28%）	あり （うち海外 約10%）	各社各様	ISO9001 CMM/CMMI	高いとはいえない
エンタプライズ系 ソフトウェア開発	・平均30万行 ・100万行以上のプログラムは少数	・実績値：(平均)8.3ヶ月	・ウォーターフォール （96.6%）	あり （うち海外 約20%）		ISO9001 CMM/CMMI	エンタプライズ系に比較して高い

表 8　JAXA のソフトウェア開発と国内受注型ソフトウェア産業との違い【著者作成】

（１）ソフトウェア開発規模と開発期間

まず、本調査から明らかとなったのは、開発規模と開発期間における違いである。JAXAにおける特にロケット等に搭載されるソフトウェア開発規模についてはそれ程大きいものではなく、詳細については明らかとしていただけなかったものの、数万行、数十万行程度の規模であるという。この点については、平均行数が 100 万行を超え、1,000 万行を超える規模のものも存在する組込みソフトウェアと比較すると、それ程大きな規模ではないといえる。また開発期間については、例えば衛生全体を開発するのに、4、5 年を要するため、その中で比較的長い時間をかけて開発するようである。なお、品質を重視するため、基本的にクイックリリースを求めないということでもあった。従って、組込みソフトウェアに見られるような、開発密度が飛躍的に拡大するといった事態の発生もないものといえる。

（２）開発プロセス

次に開発プロセスに関しては、JAXA におけるソフトウェア開発においても基本的にはウォーターフォールモデルを採用しているということであった。インタビューによれば、JAXAではソフトウェア単体での開発が少なく、ハードウェアやシステム製品そのものが上流から下流へと流れるウォーターフォールでの開発工程となっているため、特異な例として反復型を採用する場合もあるものの、基本的にはウォーターフォールプロセスであるとのこ

とであっ、た。すなわち、ウォーターフォールでの開発が多いというアンケート結果が得られているエンタプライズ系、組込みソフトウェアとの違いはない。

（３）産業構造

JAXA では実質的なソフトウェア開発を開発メーカーに委託している。ソフトウェア開発における位置づけからすれば、JAXA そのものは発注者たる顧客であり、この開発の委託を受けるメーカーが 1 次請け企業となる。従って、受注型ソフトウェア産業の特徴である多重下請け構造は JAXA のソフトウェア開発においても存在していることとなる。

（４）品質特性

JAXA において品質を評価するための基準となる品質特性を定量把握するための品質メトリックスについては、その基準を設定し、取得、分析し、開発に活かすようにと要求をおこなっている。ただし、それぞれの企業、それぞれの開発領域によって、どういった設定をするかというのは異なるということであった。

（５）ソフトウェア品質管理

ソフトウェア開発企業及び NASA においては ISO9001、CMM/CMMI を取得し、ソフトウェアの品質管理を行っているが、JAXA においては、例えば、ロケットだとか、人工衛星だとか、宇宙ステーション等それぞれの事業本部で、ISO9001 を取得しているが、取得している部門と取得していない部門があるので、いちがいに全部とはいえないとのことであった。また、CMM/CMMI については、特に要求はしておらず、委託先である開発メーカーによるとのことであった。ただし、この CMM/CMMI を ISO が国際規格化した ISO15504[112]で成熟度評価を行うための取り組みを始めているとのことであった。

（６）品質への意識

JAXA におけるソフトウェア開発における品質への意識は高く、この意識は組織の文化にまで昇化されているという。そして、同時にソフトウェア開発に関わる全員、品質が最も重要だとの認識を有しているとする。また同様のことは品質のチェックを行うための、試験工程にもあてはまるということであった。

例えば上流の仕様書をしっかり、正確に書こうと、いう気持ちが開発の現場にあるかないかっていうのが大きいと思うんですよね。ただ、そのステップ バイ ステップで、一つ一つの製品、各段階での製品をですね、例えば仕様書とか、そういったものを作っていくというのが、私たちはこう、半分、強制的かもしれないですけど、ある意味では自主的に、そういう文化ができあがっているので、それが一番下支えになっているんじゃないかなと。同じ事が試験工程にも言えて、しっかりとした試験仕様を作って、試験をしようという考え方がありますので、それがまさしく品質そのものだと思うんですけどね。で、おそらく他の業界さんと比較して、当たり前のようにそういったことがなされる文化にはなっているかなと思いますけど。（主幹開発員　片平 真史氏）

ただし、このような品質に対する意識は、何かの取り組みにより形成された訳ではなく、自分たちが作ったものを、製品として宇宙に上げるという目的があることで、自然発生的

[112] ISO15504 については SQuBOK 策定部会編（2005）116 頁に詳しい。

に形成されているものであるということであった。

第２項　品質管理方法における差異

（１）品質作り込み重視

第５章の分析では、国内受注型ソフトウェア開発が、「ものづくり」における品質管理に見られるように、品質を作り込むことが優先とはなっておらず、検査重視であることを確認した。JAXA においては、NASA と同様に IV&V という検査プロセスを導入しており、また、JAXA からは定量的データは公表されていないものの、NASA の品質メトリックスのデータにおいては、その IV&V プロセスで多くの不良を発見していることからも、検査プロセスをさらに強化することで、不良の流出を防いでいる可能性がある。この点につき確認した。

私たちが考えている品質というのは、何かそういう点検活動をすることによって確立できるものではなくて、そもそものモノづくりをしっかりすることの方が、品質つくりこみがまず重要なんで、そういった活動に重点を置いています。（主幹開発員　片平 真史氏）

すなわち、検査ではなく、そもそも不良を出さないように、品質を作り込む活動を重視する、「ものづくり」の品質管理の活動にのっとったものであるという。そこで、IV&V プロセスそのものの意義について確認を行ったところ、以下のような回答を得られた。

IV&V で品質が確保されているわけではなくて、ものづくりは現場の人が作り込む、特に品質は作り込むものなので、IV&V をやったからといって、品質を確保できているっていう位置づけでは、私たちは IV&V はやっていないですね。　（主幹開発員　片平 真史氏）

やらなくても、当然宇宙に上げられるだけの品質を確保できるというように考えています。その上で、IV&V をやることによって、NASA もホームページ見てもらうと同じ事が書かれていますが、例えば宇宙機でいくと、宇宙機のミッションだとか、人命に関わるような大きな問題が起きないということを、より確度の高い確認をするために、IV&V を実践しているという位置づけに。ですから、高信頼性、安全性の確保のために、さらなる違った観点、品質でチェックをかける、という位置づけになります。　（主幹開発員　片平 真史氏）

NASA プロジェクトの、IV&V やっていないプロジェクトもいっぱいありますよね。じゃあ、それを上げられないかというとそんなことはなくて、なんでやるのかっていう時に、品質を確保するんじゃなくて、先ほど申し上げた通り、ある特別の目的で、再度チェックをすると、いうことですかね。より、確度を高くするために。（主幹開発員　片平 真史氏）

IV&V とかで少しでも他の人の視点を入れるということは、開発側にも刺激になりますので、違った視点でそういったダブルチェックをかけるというのは、たとえ小さなコストであっても、大きな効果が出る可能性があると思います。　（主幹開発員　片平 真史氏）

経済産業省プロセス改善研究部会が 2007 年に発行する「ベストプラクティス調査報告書

～究極の高品質ソフトウェア開発プロセスを目指して～　独立行政法人 宇宙航空研究開発機構（JAXA)」においては、JAXA は、この IV&V と SQA[113]を組み合わせることで、信頼性の高いソフトウェア構築を実施する組織として紹介されている[114]。しかし、本調査で明らかとなったのは、IV&V プロセスそのものは、安全性や信頼性を高めるためのチェック機能を提供するものであり、検査プロセスを強化することで高品質が達成されているわけではないこと、また品質を最初に作り込むという点に重点を置いていた活動こそ、高品質を実現するためには最も重要であるとの認識を示した。さらに、その品質作り込みがなぜ行えるのかという点については工程毎の品質をきちんと順番に作りこんでいくこと、及び実際の開発をするメーカー側の開発者が厳選されており、その開発者は十分な能力を有していることが、品質作り込みへと繋がっているという認識が示された。

また、ソフトウェア開発における「検査」に該当するレビューについては、厳密な基準をもって行われているのではないかと想定していたが、実際は開発メーカーによってその殆どが開発されることもあり、開発時点のレビュー手法については開発メーカーによって異なるということだった。しかし、要求レベル、設計レベル、コードレビュー等についての要求事項はルール化され、IV&V 等複数の検査をおこなっているとのことであった。

（２）フィードバックループの確実性と迅速性

それでは、もう一つの「ものづくり」における品質管理の特徴である、フィードバックループの確実性と迅速性という点で、JAXA ではどのような仕組みを採っているのか。ウォーターフォールモデルが誤転写情報のフィードバックループを大きくしている可能性があるとするなら、その開発の殆どについてウォーターフォールモデルを採用し、また不良情報の伝達を阻害する要因ともなる委託開発を行う JAXA でも対処が必要なはずである。

まず、全てのソフトウェアの開発活動については、JAXA の方で、特に品質保証の確保のための要求は決めていて、それは親請けであろうと、子供であろうと、孫請けであろうと、基本的には全て網羅する形、それを適用していただく形で展開をしています。その中に例えば不良情報の発生時にどのような処理をするかといった要求も決まっていて、各ヒエラルキーの中で、どのレベルの不具合というか、不良情報はどのレベルで処置をして、上に上げなければいけないものは何でということは一応ルール化はされている。まあ、本当はですね、その中に細かく何日以内にとかいうことも本当は書かれているんですけれども、

---

[113] 経済産業省 プロセス改善研究会.“ベストプラクティス調査報告書 ～究極の高品質ソフトウェア開発プロセスを目指して～ 独立行政法人　宇宙航空研究開発機構（JAXA）第 1.0 版 平成 19 年 4 月”.宇宙ステーション・きぼう広報・情報センター 宇宙航空研究開発機構：JAXA.4 頁によれば SQA（Software Quality Assurance）とは開発部門から独立した組織が、開発プロセス規定に則って成果物が作成されたことを保証する活動と説明されている。

[114] 経済産業省 プロセス改善研究会.“ベストプラクティス調査報告書 ～究極の高品質ソフトウェア開発プロセスを目指して～ 独立行政法人　宇宙航空研究開発機構（JAXA）第 1.0 版 平成 19 年 4 月”.宇宙ステーション・きぼう広報・情報センター 宇宙航空研究開発機構：JAXA.4 頁

そこらへんは運用でうまく処理をしていると。　　（主幹開発員　片平 真史氏）

JAXA におけるソフトウェア開発も開発メーカーへの委託開発であり、不良情報のフィードバックを確保するために、品質保証要求を明確化することで、この問題を解決しているということであった。すなわち、各工程における品質保証要求で、不良情報を規定、そのルールを明確化することが、この課題の解決策となっているとするものである。

第２節　要約と結論

これまで、高品質ソフトウェア開発を行っているとされる JAXA のソフトウェア開発について、面談調査の結果を整理、分析してきたが、JAXA のソフトウェア開発が、一般的なエンタプライズ系、組込みソフトウェアの開発と比較して、異なる点として注目できる内容は４つある。すなわち、第１に規模がそれ程大きくないソフトウェア開発において、クイックリリースを求めず、各工程で品質を作り込むということに重点を置くということ、第２に、品質作り込みを重視しているといいつつも、やはり IV&V といった検査プロセスの導入等、検査プロセスの強化、あるいは品質メトリックス等による評価、開発へ反映を求めていること、第３に発注者となる JAXA 自体が品質を最も重視するという意識を有しているということであろう。

また、そもそもの情報の発信者という位置づけで見た場合に、副次的に品質向上に寄与する要因として、実際の開発者たるメーカーへの品質保証要求としての、各工程におけるレビューのルール化、不良情報に関するフィードバックルール化、また、能力を有する人材の不足が指摘されるエンタプライズ系、組込みソフトウェアとは異なり、開発メーカー側において厳選された人材による開発が行われていることも、品質確保のための重要な要素といえるであろう。

すなわち、JAXA において高品質達成要因としては、時間とコストをかけ、徹底的に品質を優先してのソフトウェア開発を行うこと、またそのようにして作られたソフトウェアについても、特にリスクの高いソフトウェアについては、IV&V プロセス等の検査プロセス等により、欠陥の除去を行うという２点に要約できるものといえる。

第７章　まとめ

第１節　各章における発見事実

本論文では第２章における事例研究を通じ、国内受注型ソフトウェア開発における品質的欠陥の発生要因は、ソフトウェアの相互依存関係性、納期短縮と開発規模増大に伴う開発密度の増大、製品出荷前の工程における十分なテストがなされていないことに起因するものであることを確認した。しかしその一方で、高品質とされる NASA や JAXA においては、特に品質管理における検査工程について強化を図ることにより、品質的欠陥の抑制をはかっているとの可能性を示した。

次いで第３章では、既存研究を概観し、ソフトウェアはハードウェアとは異なる特性を有しているものの、ハードウェアと同等の品質管理を行っていること、そして、現在の品質管理ではソフトウェアの開発規模が拡大した場合、品質的欠陥発生を抑制できないこと、

しかし、先行研究では「ものづくり」における品質管理の視点から、ソフトウェアの品質的欠陥発生に関し、理論的にも十分な説明がなされていないことを明らかにした。

第４章では、国内ソフトウェア開発において、品質的欠陥を招来する要因に対し、エンタプライズ系、組込みソフトウェアそれぞれにおいて、開発規模と開発期間、開発プロセス、産業構造、品質特性、品質管理の現状、品質への意識について、公開情報を元に確認を行うことにより、それぞれにつき、品質的欠陥の発生を引き起こす可能性があること、そして、国内受注型ソフトウェア産業においては「ものづくり」と同等の品質管理によって、その発生を抑制しようとしていることを確認した。しかし、実際にはそのような品質管理がソフトウェア開発においては、有効に機能していない、あるいは機能不全が生じており、そのことが、品質的欠陥の発生に繋がっている可能性を示した。

第５章においては、ソフトウェアの品質的欠陥の発生要因を「ものづくり」における品質管理という視点から明らかにすることを目的に行ったインタビュー調査結果について、藤本の品質管理の概念を元にインタビュー調査の分析を行った。この調査・分析により、国内受注型ソフトウェア産業においては、ハードウェアと同等の品質管理を行っているとされながらも、実際にはその仕組みや精神は活かされていないこと、具体的には、検査重視の開発であること、また、不良情報の迅速性と確実性に問題があることを確認した。そして、その原因が工程のモジュール化という思想のもとに行われた、下請け構造にあることを確認するとともに、モジュラー化、設計段階での「モデリング」、エラー時の手動対応という対策について明示した。

そして、第６章では、高品質とされる JAXA へのインタビューを通じ、国内受注型ソフトウェア産業における第３章で議論した品質的欠陥の要因それぞれと、藤本の品質管理の概念について、その間に存在する差異について分析を行った。これにより JAXA におけるソフトウェア開発においては、品質作り込みに重点を置き、クイックリリースを開発メーカーに対して求めないという事実、検査強化の姿勢、品質への意識の違いという差異を確認するに至っている。

第２節　本論文におけるインプリケーション

本論文の重要な課題の一つはそもそも不良を入り込ませないという思想に基づいた「ものづくり」と同等の日本的品質管理を行う国内受注型ソフトウェアで、なぜ品質的欠陥が発生するのか、そして、その品質的欠陥を発生させる要因は何であるのかについて検討を行うことであった。ソフトウェアの品質的欠陥の発生要因については、ソフトウェアの特性、相互依存関係性、複雑性など、ソフトウェアそのものが有する特性から生じる品質確保のための困難さや、ソフトウェアの使用者たるユーザやハードウェア開発元からもたらされる変更や追加を正確に反映、制御する仕組みやそもそもの管理能力の未熟さ、あるいは規模の増加に伴う開発密度の拡大や、それに伴うテスト量の級数的増加によるテスト不足から生じているとする見解、さらにはプロセス、人員能力、CMM/CMMI などの品質管理手法等、品質そのものに影響するであろう様々な要因に着目した研究が行われており、その都度、ソフトウェアで品質的欠陥の発生を抑制することが難しい、あるいは撲滅することができないことが説明されてきた。しかし、それら通説は、そもそも、「ものづくり」と同等のそもそも不良を入り込ませないことを目的とした、日本的品質管理を行う国内受注型

ソフトウェア産業において、なぜ品質的欠陥が発生し、そしてその発生を抑制するためにどのような対策が考えられるのかについては明らかにしてこなかった。

しかし、本論文における一連の分析は、「ものづくり」に見られるようなそもそも日本的品質管理の仕組みが、企業や組織によって程度は異なるであろうが、国内受注型ソフトウェア開発産業では機能しているとは言い難く、すなわち、「ものづくり」に見られるような日本的品質管理がソフトウェアの品質を高めることに繋がっているとは、必ずしも証明できるものではないということを確認するに至っている。そして、そのことは同時に、様々な品質影響要因に対し、現在国内受注型ソフトウェア開発で行われている開発手法やテスト手法、品質管理に関わるテクニカルな手法といったものが、少なくともソフトウェアの品質を品質管理により達成しようとする現状においては、品質問題に対して1次的な対応となってしまっており、根本的な問題解決となっていない可能性が高い。

それでは国内受注型ソフトウェア開発において、高品質を達成するために、ソフトウェア開発企業は何をしなければならないのであろうか。まずは、第5章において論じたいくつかの対策が、ソフトウェア開発における品質向上のための手段となるが、第6章の JAXA からの面談調査からすれば、品質を最重要視するという組織的、個人的な意識の下で、品質を確保するためには納期やコスト以上に、徹頭徹尾品質を作り込むという姿勢が必要といえよう。しかし、限られた納期の中、限られたコストで、多数の変更要求にさらされながらも、顧客が望む機能を有するソフトウェアを提供するという責務をソフトウェア開発企業は有している。従って、品質至上主義による製品開発を行うことは、実質的に不可能であると言わざるをえない。

加登（2005）ではこの点について言及し、経済性のみを優先する品質管理活動、及び品質至上主義、すなわち品質重視で経済性や他の要因を軽視する品質活動、いずれの品質活動の問題を克服する有力な視点として、ゼロディフェクト（欠陥ゼロ）の精神を活かしながら、高いレベルの品質を、適正レベルの資本を含む経営資源の投入に達成するという ROQ（Return of Quality）[115]の考え方の導入が必要であるとしている。またその他に日本的品質管理の再生のための処方箋として、①TQM エキスパートの知恵を組織知とするために再度雇用すること、システム、データベース、マニュアル等を再整備する、あるいは品質管理に関する新たな知識を習得すること、②品質管理を源流時点からの全社活動とすること、③品質管理教育の徹底すること、④これまでに蓄積されてきた原価企画、環境型設計開発と称される DfE（Design for Environment）やプロジェクトマネジメントなどの実践経験と研究結果を統合した製品開発マネジメントシステムを構築すること、⑤改善活動をオープンにすること、⑥品質では差別化ができないが故、品質管理に関する知識を社会全体で共有することを挙げている[116]。

これをソフトウェア開発における品質管理活動に読み替えるのであれば、例えば IV&V、SQA を併用した、品質への影響が高いと考えられる検査品質プロセスへの投資、先行研究で見たような品質に正の影響を及ぼす各種品質影響要因への投資といった経済活動内における合理的な経営資源の投入について、適正レベルの資本の投入で実施することが必要に

[115] ROQ については梶原（2005）に詳しい

[116] 加登(2005) 8-17 頁

なるといえよう。また、その他の処方箋については、①個人の能力への依存度が高く、また人員の流動が激しいことが指摘される国内ソフトウェア開発産業においては、組織的取り組みとして、個々人に蓄積された職人的技能や知識を、システム、マニュアル、データベースとして蓄積しておくこと、②源流である要件定義や設計段階からの品質管理活動を組み入れること、③国内受注型ソフトウェア産業が広く取得する ISO9000 シリーズの適切なファイリングやドキュメンテーションを研修プログラムへ組込むこと、④品質管理部門に所属する人々だけでなく、プロジェクトマネージャーや開発者、自社だけでなく、品質管理に対して有効な手立てを有さないとされる下請け企業に所属する品質管理部門の人々や開発者に対する研修教育を行うこと、④改善活動の結果や効果を例えば単に ISO9000 を取得している、あるいは CMM/CMMI のレベルを公表するだけでなく、その効果やサクセスストーリーについて業界全体に対しオープンにする活動を行うこと、⑤主に製造業の分野では研究の進んでいる原価企画の分野を設計段階に適応すること、あるいは IT 業界で様々な研究や実践がなされているプロジェクトマネジメントや知識体系を統合した製品開発マネジメントシステムの再構築を行うこと、⑥ハードウェアの場合とは異なり、当たり前品質が達成できていない本業界では、品質が差別化の源泉ともなっていることが想定されるため、困難であることが想定されるが、ソフトウェアの社会的な重要性がさらに増している現在の状況を鑑みれば、品質問題に関わる情報を共有し、問題解決にあたり、その結果を広く公開するといったことが、有効な対策になるのではないか。さらに付け加えれば、日本的品質管理がソフトウェア開発で有効であり、また、開発コストと生産性の両立のため、オフショア開発を行うことが不可避というのであれば、日本的品質管理の仕組みや精神を海外ソフトウェア開発企業に適応、浸透させる手段を講じるべきではないだろうか。

第３節　本論文の限界と今後の課題

本論文では、国内受注型ソフトウェア産業において、品質的欠陥の発生要因について解明するため、事例、アーカイバルデータ、サーベイデータという異なるデータの調査、分析を通じての研究を行った。そして、その結果から明らかとなったのは、ハードウェアとは異なる様々な特徴や特性を有するソフトウェアそのもの及びソフトウェア開発を、ハードウェアと同等と見なし、「ものづくり」と同等の品質管理を行っていること、そしてそもそも不良を最初から入り込ませないという思想から形成されているはずの日本的品質管理が実際には充分に機能していないということこそが、品質的欠陥の発生要因となっているのではないかとの考えに基づき、調査と分析を試みた。その結果、藤本のいう統合的生産システムにおける製造品質面の組織能力という面からすれば、そのような組織能力を下支えする品質管理の仕組みが産業全体で機能していない、あるいは実質的に実行されていないという事実を確認することができた。しかしながら、本論文では、あくまで日本的品質管理が産業の傾向として機能していないことを確認できたにとどまり、実際に個別の企業、組織やプロジェクトにおいて、「ものづくり」における品質管理の仕組みが、ソフトウェア品質に影響を及ぼす真の要因であるかどうかについて、統計的に明らかとなっていない。また、第６章では高品質事例の調査として JAXA に対する面談調査を実施したが、「品質作り込み」を重視しているとの見解は明らかとなってはいるものの、具体的な生産システムの中身や、その定性的、定量的効果についてまで言及できていないことも事実であり、「も

のづくり」と同等の品質管理をどのようにすれば機能させることができるのか、あるいは「品質作り込み」という個別の生産システムどのような仕組みとして確立していけよいかという具体的な対策についても明らかとなっていない。

この点に関しては、多数のソフトウェア開発プロジェクトの高・低品質プロジェクト事例の観察、プロジェクトデータの収集を行い、統計的手法を用いた分析により、日本的品質管理をソフトウェアに読み替えた場合の仕組みが、ソフトウェア開発における品質向上に本当に有効な手段となりうるか、また、ソフトウェア開発においてどのような仕組みが品質向上のための生産システムとして有効かの効果測定を行うことができれば、違った切り口での示唆や定量的な洞察、提言が可能となるであろう。さらに第３章で議論した品質的欠陥の要因となり得る要素をそれぞれ変数として捉え、各変数と品質の因果関係が説明できれば、より定量的で客観性の高い研究になるものと推察される。

謝辞

本論文のインタビュー調査においては IPA/SEC 所長 鶴保征城氏、及び JAXA 片平 真史氏、宮本 祐子氏にお会いしてお話をお伺いし、メールや電話でのご質問をさせていただくことなしには実現しえなかった。また、インタビュー内容を記載することについても、ご理解を頂くことで、初めて本論文への掲載が可能となった。本論文はここに挙げさせていただいた皆様のご協力の上に成り立っており、ここに深く謝意を表させていただきたい。また、執筆に際しては加登豊先生（神戸大学）から示唆に富むご質問、ご助言をいただいた。記して深く感謝申し上げる。

参考文献

Abran,A＆Moore,J.W“Guide to the Software Engineering Body Of Knowledge”.Guide to the SWEBOK:PDF Format of SWEBOK. （オンライン）, 入手先<http://www.swebok.org/pdfformat.html>,（参照 2008-8-20).

Agrawal,M&Chari,K (2007) “Software Effort,Quality,and Cycle Time:A Study of CMM Level 5 Projects”*IEEE TRANCEACTION ON SOFTWARE ENGINEERING,* Vol.33,No.3, pp.145-156.

Austin,R.D (2003)“The Effect Of Time Pressure on Quality in Software Development:An Agency Model” *Information System Research,* Vol.12,pp.195-207.

Cusmano,M＆MacCormac,A＆Kemerer,K.M＆Cranbal.B“Software Development Worldwide: The State of the Practice”*IEEE SOFTWARE*,Nol.20,No.6,pp.28－34.

Cusumano,M (2004) *The Bussiness Of Software,*Free Press
（サイコム・インターナショナル監訳『ソフトウェア企業の競争戦略』ダイヤモンド社,2004).

Demarco,T.Listar,T (1990) *Software-State-Of-The-Art : Selection Paper*, Dorset House.
（児玉公信監訳『ソフトウェアエンジニアリング論文集 80‘s』翔泳社,2006）.

Dorfman,M&Thaiyer,R.H(1996) *Software Engineering*,Ieee Computer Society.

Giveson,D.L&Goldenson,D.R&Kost,k.“performance Result of CMMI -Based Process

Improvement”.*Software Engineering Instistute Carnegie Mellon*,（オンライン）.入手先<http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06tr004.pdf>（参照 2008-6-17).

Krishnan,M.S&Kreibel,C.H&Kekre,S&Mukhopadhyay,T（2000）“An Emprical Analysis of Productivity and Quality in Software Products” *Managemant Science*, Vol.46,No.6, pp.745-759.

Larman,C（2003）“Agile and Iterative Development：A Manager’s Guide”*Addison-Wesley*.

Leveson, N.“Report of the Presidential Commission on the Space Shuttle Challenger”.Nancy Leveson’s HomePage at MIT.（オンライン）,入手先<http://sunnyday.mit.edu/accidents/challenger-table-of-contents.html>,（参照 2008-6-17).

NASA/WVU IV&V Facility, Metrics Data Program, available from http://mdp.ivv.nasa.gov; Internet; accessed [05 07 2008].

SQuBOK策定部会編(2005)『ソフトウェア品質知識体系ガイドーSQuBOK Guide-』オーム社。

White,S.A“Introduction to BPMN”.BPMN Information Home.（オンライン）, 入手先<http://www.bpmn.org/Documents/Introduction%20to%20BPMN.pdf>,(参照 2008-8-20).

阿部昭博・吉沢正（1998)「品質機能展開を用いたプロトタイピング手法」『社団法人品質管理学会』Vol.28,No.2,73-85頁。

伊藤嘉博（2001)『環境を重視する品質コストマネジメント』中央経済社。

梶原武久（2005)「ROQアプローチによる日本的品質管理の再構築」『Business Insight』第13巻第1号,30-45頁。

宇宙航空研究開発機構.“2003年8月26日 コロンビア号事故調査報告 Volume I（速報版）COLUMBIA ACCIDENT INVESTIGATION BOARD Report Volume I August 2003”. 宇宙ステーション・きぼう広報・情報センター 宇宙航空研究開発機構：JAXA.（オンライン）,入手先<http://iss.jaxa.jp/shuttle/sts_accident/nasda_reports/prompt_report0826.html>,（参照 2008-6-17).

宇宙航空研究開発機構.“H-ⅡAロケット解説資料”.HⅡA LAUNCH SERVICE.（オンライン）,入手先<http://h2a.mhi.co.jp/en/RSC/mtsat1/documents/h2a_j.pdf>,（参照 2008-7-12).

加登豊（2005)「日本的品質管理の再生」『Business Insight』第13巻第1号,8-17頁。

金融庁.“2006年6月19日 報道発表資料 みずほフィナンシャル・グループに対する行政処分について”.みずほファイナンシャル・グル・・・:金融庁.（オンライン）, 入手先<http://www.fsa.go.jp/news/newsj/13/ginkou/f-20020619-1.html>,（参照 2008-6-10).

経済産業省.“特定サービス産業動態統計調査 長期データ 情報サービス産業”.特定サービス産業動態調査 経済産業省.（オンライン）,入手先< http://www.meti.go.jp/statistics/tyo/tokusabido/result/result_1/xls/hv14303j.xls>,（参照 2008-6-30）.

経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2004年版組込みソフトウェア産業実態調査報告書 平成16年6月”.情報処理推進機構：ソフトウェアエンジニアリング.（オンライン）,入手<http://sec.ipa.go.jp/download/dl.php?filename=/report/200406/es04r002.pdf>,（参照 2008-8-8）.

経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力

強化推進委員会監修.“2005 年版 組込みソフトウェア産業実態報告書 －経営者・事業責任者向け調査－改訂版 平成 17 年 6 月”.情報処理推進機構:ソフトウェアエンジニアリング.（オンライン）,入手先<http://sec.ipa.go.jp/download/dl.php?filename=/report/200506/es05r002.pdf>,（参照 2008-6-26）.
経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2005 年版 組込みソフトウェア産業実態報告書－開発プロジェクト責任者向け調査－改訂版 平成 17 年 6 月”.情報処理推進機構：ソフトウェアエンジニアリング.（オンライン）,入手先<http://sec.ipa.go.jp/download/dl.php?filename=/report/200506/es05r003.pdf>,（参照 2008-6-26）.
経済産業省 商務情報政策局，情報政策ユニット情報処理振興課，組込みソフトウェア開発力強化推進委員会監修.“2006 年版 組込みソフトウェア産業実態報告書－経営者・事業責任者向け調査－第 2 版 平成 18 年 8 月”. 情報処理推進機構:ソフトウェアエンジニアリング.（オンライン）,入手先<http://sec.ipa.go.jp/download/dl.php?filename=/report/200606/es06r001.pdf>,（参照 2008-6-26）.
経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2006 年版 組込みソフトウェア産業実態報告書－プロジェクト責任者向け調査－第 2 版 平成 18 年 8 月”.情報処理推進機構：ソフトウェアエンジニアリング.（オンライン）,入手先<http://sec.ipa.go.jp/download/dl.php?filename=/report/200606/es06r002.pdf>,（参照 2008-6-26）.
経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2007 年版 組込みソフトウェア産業実態報告書－経営者・事業責任者向け調査－第 1 版改訂 平成 19 年 10 月”.情報処理推進機構：ソフトウェアエンジニアリング.（オンライン）,入手先<http://sec.ipa.go.jp/download/dl.php?filename=/report/200706/es07r001_r2.pdf>,（参照 2008-6-26）.
経済産業省 商務情報政策局・情報政策ユニット情報処理振興課・組込みソフトウェア開発力強化推進委員会監修.“2007 年版組込みソフトウェア産業実態調査報告書－プロジェクト責任者向け調査－第 1 版改訂 平成 19 年 10 月”.情報処理推進機構：ソフトウェアエンジニアリング.（オンライン）,入手先<http://sec.ipa.go.jp/download/dl.php?filename=/report/200706/es07r002_r2.pdf>,（参照 2008-8-8).
経済産業省 プロセス改善研究会.“ベストプラクティス調査報告書 ～究極の高品質ソフトウェア開発プロセスを目指して～ 独立行政法人 宇宙航空研究開発機構（JAXA）第 1.0 版 平成 19 年 4 月”.情報処理推進機構:ソフトウェアエンジニアリング.(オンライン),入手先<http://sec.ipa.go.jp/download/dl.php?filename=report/200705/JAXA.pdf>,（参照 2008-6-4).
小泉寿男・辻秀一・吉田幸二・中島毅（2003)『ソフトウェア開発』オーム社。
阪田史郎・高田広章（2006)『組込みシステム』オーム社。
櫻井通晴（2001)『ソフトウェアの管理会計』白桃書房。
柴田友厚・玄馬公規・児玉文夫（2002）『製品アーキテクチャの進化論　システム複雑性と分断による学習』白桃書房。
ソフトウェア・エンジニアリング・センター.“ソフトウェアエンジニアリアリングの実践強

化に関する調査研究「エンタプライズ系ソフトウェアソフトウェアにおける SE 度の実態調査」” .情報処理推進機構：ソフトウェアエンジニアリング.（オンライン）,入手先 <http://sec.ipa.go.jp/download/dl.php?filename=report/200710/SE_level_research_2006_2.pdf>,（参照 2008-6-26）.
タカミハマダ二・中尾政之.“データ入力ミスで旅客機が山に激突”.JST 失敗知識データベース. 入手先<http://shippai.jst.go.jp/fkd/Detail?fn=0&id=CA0000293&>,（参照 2008-7-5）.
タカミハマダニ・中尾政之.“ソフトのバグによるハイテク航空機の墜落事故”.JST 失敗知識データベース. 入手先<http://shippai.jst.go.jp/fkd/Detail?fn=0&id=CA0000486>,（参照 2008-7-5）.
張田吉明･中尾政之.“名古屋空港で中華航空 140 便エアバス A300-600R が着陸に失敗炎上”.JST 失敗知識データベース. 入手先< http://shippai.jst.go.jp/fkd/Detail?fn0&id=CA0000621&>,（参照 2008-7-5）.
独立行政法人 情報処理推進機構.“第 29 回 情報処理産業経営実態調査報告書（概要)”.情報処理推進機構：IT ベンチャー支援.（オンライン）,入手先 <http://www.ipa.go.jp/software/hosyo/pdf/summary29.pdf>,（参照 2008-7-1).
独立行政法人情報処理推進機構.“2007 年度産学連携ソフトウェア工学実践拠点事業エンタプライズ系ソフトウェア技術者個人の実態調査報告書 2008 年 5 月”.情報処理推進機構：ソフトウェアエンジニアリング.（オンライン）,入手先<http://sec.ipa.go.jp/ reports/20080522/ent-engineer_20080522.pdf>,（参照 2008-8-8）.
独立行政法人 情報処理推進機構・ソフトウェア・エンジニアリング・センター・経済産業省.“組込みスキル標準 2005 年版キャリア基準ー組込みプロフェッショナルの戦略的育成に向けてー”.情報処理推進機構：ソフトウェアエンジニアリング,入手先 <http://sec.ipa.go.jp/download/dl.php?filename=/report/200505/ETSS_career_Draft_20050520.pdf>,（参照 2008-8-8).
独立行政法人 情報処理推進機構（IPA）,ソフトウェア・エンジニアリング・センター（SEC）（2007）『ソフトウェア開発データ白書 2007』日経 BP 社。
冨永章.“ソフトウェア・グリッチの予防”.JISSJ Toc Member,入手先 <www.yy.ics.keio.ac.jp/issj/annualconf2_2006/pdfs/HIS(5)/HIS-01-tominaga.pdf>（参照 2008-6-5).
西村清彦・峰滝和典（2004)『情報技術革新と日本経済「ニューエコノミー」の幻を超えて』有斐閣。
日経コンピュータ編（2002)『システム障害はなぜおきたか みずほの教訓』日経 BP 社。
日経コンピュータ「ニュース＆トレンド 航空管制システム障害は防げた！バグを放置、テストは 8 時間のみ」2003 年 3 月 24 日号, 15－16 頁。
日経コンピュータ「特集“プロジェクト成功率は 26.7%”」2003 年 11 月 17 日号, 50-71 頁。
日経コンピュータ「ニュース＆トレンド 保守費は 5 年で初期開発費の 8 割強に JUAS が 82 プロジェクトの保守･追加費を調査」2006 年 5 月 1 日号,16 頁。
日経ビジネス「特集 ソフト不良が招く品質の危機 自動車編 加速する品質競争」2005 年 4

月 25 日・5 月 2 日号,37-39 頁。
日経ビジネス「時流潮流 News＆Trend 欠陥ソフト経済産業省も対策に着手 携帯・家電に不具合相次ぐ」2004 年 7 月 5 日号,6-7 頁。
延岡健太郎（2006)『MOT［技術経営］入門』日本経済新聞社。
藤本隆宏・武石彰・青島矢一（2001)『ビジネス・アーキテクチャ』有斐閣。
藤本隆宏（2007)『ものづくり経営学―製造業を超える生産思想』光文社。
藤本隆宏（2004)『日本ものづくり哲学』日本経済新聞社。
藤本隆宏（2003)『能力構築競争』中央公論新社。
藤本隆宏（2001)『生産マネジメント入門Ⅰ【生産システム編】』日本経済出版社。
藤原良一・本間敏夫・細谷和伸・中前雅之・遠藤和彦.“プロセス改善による高品質 IT ソリューションの提供に向けた CMMI レベル５達成への軌跡”.三菱電機 技報.（オンライン）,入手先< http://www.mitsubishielectric.co.jp/giho/0609/0609201.pdf>,（参照 2008-7-15).
マユミビンクス・中尾政之.“管制官の誘導ミスとコンピュータ故障による旅客機のニアミス”.JST 失敗知識データベース.（オンライン）, 入手先 <http://shippai.jst.go.jp/fkd/Detail?fn=0&id=CA0000425&>,（参照 2008-7-5）.
室修治.“組み込み最前線(1) 難問山積のソフト開発 現場の頑張りはもう限界”.組込み最前線（１）: IT Pro.（オンライン）,入手先< http://itpro.nikkeibp.co.jp/prembk/NBY/techsquare/20050328/158076/>,（参照 2008-8-5).
保田勝通（1995)『ソフトウェア品質保証の考え方と実際：オープン時代に向けての体系的アプローチ』日科技連出版社。
山下徹（2007)『高度 IT 人材育成への提言　国際競争力の復権に向けて』NHK 出版。

ワーキングペーパー出版目録

番号	著者	論文名	出版年
2007・1	小杉　裕	シーズ型社内ベンチャー事業へのＶＰＣの適用 ～株式会社エルネットの事例～	4/2007
2007・2	岡本　存喜	マネジメントシステム審査登録機関Ｙ社 のＶＣＰ（Ｖａｌｕｅ　Ｃｒｅａｔｉｏｎ　Ｐａｔｈ）の考察	4/2007
2007・3	阿部　賢一	F 損害保険会社における ＶＣＰ（Ｖａｌｕｅ　Ｃｒｅａｔｉｏｎ　Ｐａｔｈ）の考察	3/2007
2007・4	岩井　清一	S 社における VCP（Value　Creation　Path）の考察	4/2007
2007・5	佐藤　実	岩谷産業の VCP 分析	4/2007
2007・6	牛尾　滋昭	（株）森精機製作所における VCP(Value Creation Path)の考察	4/2007
2007・7	細野　宏樹	ＶＣＰ（Ｖａｌｕｅ　Ｃｒｅａｔｉｏｎ　Ｐａｔｈ）によるケーススタディー ケース：株式会社 電通	4/2007
2007・8	外村　衡平	VCP フレーム分析による T 社の知的資本経営に関する考察	4/2007
2007・9	橋本　敏行	企業における現金保有の決定要因	10/2007
2007・10	森本　浩嗣	百貨店 A 社グループのシェアードサービス化と その SS 子会社によるグループ貢献の VCP 分析	4/2007
2007・11	山矢　和輝	みすず監査法人の知的資本の分析	4/2007
2007・12	山本　博紀	S 社の物流（航空輸出）に関する VCP(Value Creation Path)の考察	4/2007
2007・13	中　智玄	A 社における VCP(Value Creation Path)の考察	5/2007
2007・14	村上　宜洋	ＮＴＴ西日本の組織課題の分析 ～Value Creation Path 分析を用いた経営課題の抽出と提言～	5/2007

2007・15	宮尾　学	健康食品業界における製品開発 －研究開発による「ものがたりづくり」－	5/2007
2007・16	田中　克実	医薬品ライフサイクルマネジメントのマップによる解析評価 －Product-Generation Patent-Portfolio Map の提案－	9/2007
2007・17	米田　龍	サプライヤーからみた企業間関係のあり方 ～自動車部品メーカーの顧客関係についての研究～	10/2007
2007・18	山田　哲也	経営幹部と中間管理職のキャリア・パスの相違についての一考察　－日本エレクトロニクスメーカーの事例を基に－	10/2007
2007・19	藤原　佳紀	供給サイドにボトルネックが存在する場合の企業間連携の評価 －原子力ビジネスにおいて－	10/2007
2007・20	加曽利　一樹	通信販売ビジネスにおける顧客接点複合化の検討 ～ 株式会社ゼイヴェルの事例をてがかりに ～	11/2007
2007・21	久保　貴裕	高付加価値家電のデザイン性のマネジメント	12/2007
2007・22	川野　達也	「自分らしい消費」を促進するアパレル通販 －インターネット・メディアとの連動－	11/2007
2007・23	東口　晃子	1994 年～2007 年のシャンプー・リンス市場における マーケティング競争の構造	12/2007
2007・24	茂木　稔	デバイスマーケットのデファクト・スタンダード展開 ～後発参入でオープン戦略をとったＳＤメモリーカード～	12/2007
2007・25	芦田　渉	地域の吸引力～企業誘致の成功要因～	12/2007
2007・26	滝沢　治	製薬企業の新興市場戦略『中国医薬品市場における「シームレス・バリュー・チェーン」の導入』	12/2007
2007・28	南部　亮志	e コマースにおけるパーソナライゼーション ～個々の顧客への最適提案を導く仕組みと顧客情報～	12/2007
2007・29	坪井　淳	ホワイトカラー中途採用者の効果的なコア人材化の要件に関する一考察	12/2007
2007・30	石川　眞司	アップルとサプライヤーとの企業間関係に関する考察	1/2008
2008・1	石津　朋和 白松　昌之 鈴木　周 原田　泰男	技術系ベンチャー企業の企業価値評価の実践－ダイナミック DCF 法とリアル・オプション法の適用－	5/2008
2008・2	荒木　陽子 井上　敬子	医薬品業界と電機業界における M&A の短期の株価効果と長期の利益率	5/2008

杉　一也
染谷　誓一
劉　海晴

2008・3	堀上　明	IT プロジェクトにおける意思決定プロセスの研究 －クリティカルな場面におけるリーダーの意思決定行動－	9/2008
2008・4	鈴木　周	M&A における経営者の意思決定プロセスと PMI の研究 －リアル・オプションコンパウンドモデルによる分析－	10/2008
2008・5	田中　彰	プロスポーツビジネスにおける競争的使用価値の考察　プロ野球・パシフィックリーグのマーケティング戦略を対象に	10/2008
2008・6	進矢 義之	システムの複雑化が企業間取引に与える影響の研究	10/2008
2008・7	戸田 信聡	場の形成による人材育成	10/2008
2008・8	中瀬　健一	BtoB サービスデリバリーの統合～SI 業界のサービスデリバリーに関する研究～	10/2008
2008・9	藤岡　昌則	生産財マーケティングアプローチによる企業収益性の規定因に関する実証研究	11/2008
2008・10	下垣　有弘	コーポレート・コミュニケーションによるレピュテーションの構築とその限界：松下電器産業の事例から	11/2008
2008・11	小林　正克	製薬企業における自社品および導入品の学習効果に関する実証研究	11/2008
2008・12	司尾　龍彦	マネジャーのキャリア発達に関する実証研究　管理職昇格前のイベントを中心として	11/2008
2008・13	石村　良治	解釈主義的アプローチによるデジタル家電コモディティ化回避	11/2008
2008・14	浅田 賢治郎	ソフトウェア開発における品質的欠陥発生要因と対策	11/2008