# ソフトウェア業界航海術

## ～業界の基礎知識、現状分析、そして慶の戦略～

＜試読版＞

※試読版に含まれる内容は１章までとなります。

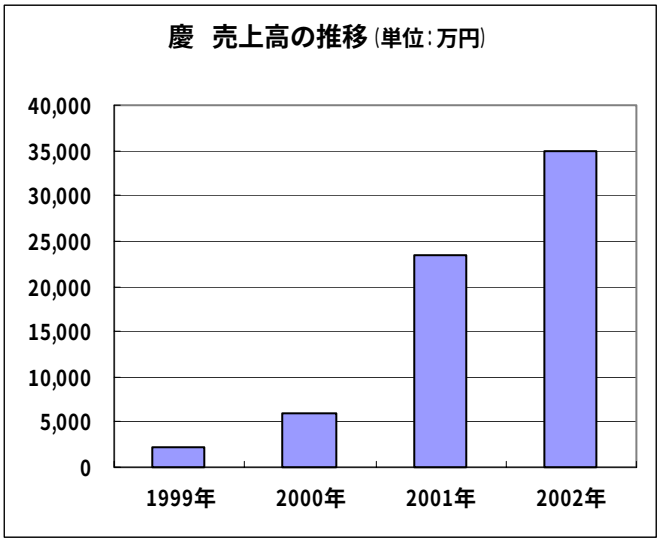
株式会社 慶 代表取締役

**蒲生 嘉達**

# 目　次

# まえがき

下図「慶　売上高の推移」が示すとおり、株式会社 慶はこれまで順調に成長してきました。




しかし、ソフトウェア業界は 2001 年以降、急速に不況色を強めています。その厳しい状況下で、慶が 2003 年以降どのようにしたら成長を続けられ、利益を確保できるのかが本書のテーマです。

第１章ではソフトウェア請負開発について考察します。
ソフトウェア請負開発には元々危険性が内在していましたが、近年その危険性が特に顕在化しています。請負開発の危険性に直面した時、どのような戦略を取るかによってその会社の性格と将来は決定付けらます。
驚異的に生産性の高いチームを作ることによって請負開発会社として成功できることを解説します。

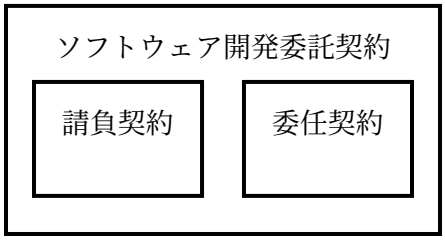
第２章では技術者派遣について考察します。
ソフトウェア業界における技術者派遣市場は非常に規模が大きく、且つ未開拓分野も大きな市場です。この巨大な市場に対しどのような戦略で挑むのかということも、請負に対する戦略同様、その会社の性格と将来を決定付けます。技術者派遣事業について分析し、慶の戦略を記します。

第３章では、慶が請負開発と技術者派遣の成果を生かして、労働集約型の会社から知識集約型の会社に成長することを示します。

2003 年 3 月
株式会社 慶 代表取締役
蒲生 嘉達

## １.ソフトウェア請負開発の基礎知識・分析・戦略

ソフトウェア会社がソフトウェア開発の仕事を受ける場合の契約形態には請負契約と委任契約の２つがあります。この契約形態の違いを正確に理解することは非常に重要です。契約形態の違いによって事業としての性格が大きく異なってくるからです。



## 1.1 請負契約と委任契約

### 1.1.1. 主な違い

請負契約と委任契約の主な違いを下記に示します。

請負契約と委任契約の主な違い

| | 請負契約 | 委任契約 |
|---|---|---|
| 契約の目的 | 「仕事や物（プログラムなど）の完成」を約すこと。 | 委任が契約は「プログラムを開発するために知識や労力といったサービスの提供」をすること。 |
| 完成と対価 | 対価を得るためには、プログラムを完成させなければなりません。 | プログラムを開発するために知識や労力などのサービスを提供しなければなりません。しかし、委任においてはプログラムの完成ということは約されていません。すなわち、プログラムが完成しなくても、今までになした履行（サービスの提供）の割合に応じて報酬を受けることができます。 |
| 経費 | 定額の請負（後述しますが変額の請負もあり得ます）においては、原則として経費が増加しても契約金額を変更することはできません。 | サービスの提供のために必要な費用は償還請求ができます。つまり超過時間の精算ができます。 |

### １.1.2.　用語の整理

#### (A)委託

一般に「委託」と言う言葉もよく使われます。例えば、「ソフトウェア関連業務委託基本契約書」とか「甲が乙に委託するコンピュータシステムの請負業務に関して、次のとおり基本契約を締結する」といった使われ方をします。

この「委託」という言葉は取引用語であり、法律用語ではありません。委託という言葉が使われている場合は、契約全体をみて請負契約であるか、委任契約であるかを判断する必要があります。
その判断のポイントは「仕事の完成を約束しているかどうか」です。
仕事が完成しなければ代金が支払われない、つまり代金の支払いの条件が仕事の完成となっているのであれば、たとえ「委託契約」という名称になっていても法律的には請負契約です。

#### (B)委任と準委任

細かいことをいうと、法律行為を他人に委託することを「委任」といい、法律行為以外の事務の委託をすることを「準委任」といいます。したがって、ソフトウェア開発での委任は、法律行為以外の委任なので「準委任」です。
しかしながら「準委任」という用語は一般にはなじみが薄いので、本書では「委任」とします。

#### (C)SE 支援・SE サポート・業務請負・作業請負

取引上「SE 支援」「SE サポート（略して SES）」「業務請負」「作業請負」という言葉を使う場合がありますが、これは「委任契約」と同義です。

#### (D)技術者派遣

本書では「技術者派遣」という用語も使っていますが、これは「委任契約に基づいて技術者を派遣する（客先でサービスを提供する）」ことを意味します。

#### (E)一括請負

取引上「一括請負」という言葉もよく使われます。これは「請負契約」と同義です。

### １.1.3.　請負と技術者派遣の事業面での比較

請負と技術者派遣の事業としての一般的な特徴を下記にまとめます。

| | 請負 | 技術者派遣 |
|---|---|---|
| 技術の蓄積 | 開発過程で得られる技術を蓄積して他の仕事に活用することが可能です。 | 独自技術の蓄積面では不利です。 |
| 人材育成 | 各開発工程を通して社員のスキルアップをおこなえるために人材育成面で有利です。 | 計画的な人材育成という面では不利です。 |
| 利益率 | 料金設定は「１プロジェクト単位」となるため、少ない人数でプロジェクトを完遂できれば利益率は高くなります。一方、プロジェクトが失敗し、大赤字になる可能性もあります。ハイリスク・ハイリターン型です。 | 売上は「工数×単価」が基本となるために労働集約的な色合いが強まり、利益率は低くなります。但し、極端な赤字が発生することもありません。ローリスク・ローリターン型です。 |
| 売上・利益・技術者数 | 売上・粗利・技術者数は必ずしも比例しません。 | 売上・粗利・技術者数は比例します。したがって、派遣契約数が減少すれば社員の給料支払い負担が急激に資金繰りを悪化させます。 |
| 設備投資資金 | 開発環境をすべて社内で整備する必要があるために大きな設備投資資金が必要となります。 | クライアント側の設備を利用できるため、自社設備に大きな資金を必要としません。 |
| 資金繰り | プロジェクトが完成するまでは全く売上になりません。売掛金、立替金負担が非常に大きくなります。大きな仕事になるほど開発期間は長引き資金繰りは厳しくなります。 | 月単位の支払いがある場合が多く、資金繰りは比較的安定しています。 |
| 入金遅れ | 品質が悪ければ検収が遅れ、入金が遅れる場合もあります。 | 検収が遅れるということはありません。 |

## 1.2 ハイリスク・ローリターンとなった請負

「1.1.2.請負と技術者派遣の事業面での比較」で記したとおり、請負は本来、技術を蓄積し、人材を育成し、高い利益をもたらす可能性をもっています。しかし、最近の請負型ソフトウェア会社の実状はかなり違います。

- 利益率が高いどころか、プログラムを納期までに完成させることができずに、大赤字を出している会社が増えています。
- 人材を育成する余裕のない会社が増えています。
- 毎回新しい技術で開発するので社内に技術が蓄積されていない会社がほとんどです。

請負は本来ハイリスク・ハイリターンという性格をもっているのですが、最近はハイリスク・ローリターンになっています。どうしてこのようになったのか本節で考察します。

### １.2.1.　請負契約が元来持っている危険性

システム開発請負契約の基本的な問題点として、「契約締結時点においては、具体的にどのようなものをつくるか（最終的な詳細仕様）が決められない」ということがあげられます。

契約締結時点においては、発注者より発注仕様が提供されるどころか、どのようなものを作るかということを決定すること自体が業務内容になることもよくあります。システム設計という仕事は、どのようなものを開発したらよいのかを具体的に明らかにする仕事だからです。

このような仕事を請負契約にするということは、「何が完成なのかが明らかになっていないにもかかわらず、仕事の完成を約する」ということになります。

その結果として、システム開発請負契約においては、完成とは何かということについてユーザの主観にゆだねられ、ユーザの意図した結果が出るまで手直しが要求されるという危険性が内在しているのです。

したがって、システム設計時は委任契約にし、プログラムの作成だけを請負型にするという手法もしばしば採用されます。それによって危険性はかなり軽減されますが、それで問題が全て解決されるわけではありません。その理由は下記のとおりです。

- ユーザ要件は必ず変化します。
- 詳細仕様まで全て決めてからプログラミングに入るということは事実上不可能です。それをやろうとすると、非効率的な文書中心主義を生み出してしまいます。

## １.2.2.　請負の最近の傾向

技術革新と不況が請負の様相を変えています。

### １.2.2.1. 技術革新

ソフトウェア業界は技術の変化が激しい業界ですが、常に一定のスピードで変化しているわけではありません。飛躍的に変化した時代と大して変化しなかった時代とがあるのです。例えば、1980 年代は、汎用機、UNIX、PC ともに比較的技術変化が緩やかな時代でした。それに対し、1990 年代後半から現在までは、技術の変化が激しい時代です。

ロバート・Ｘ・クリンジリーは「コンピュータ帝国の興亡」（株式会社アスキー発行）の中で下記にように記しています。同書は 1991 年夏に出版されたものであることを念頭において読んで下さい。

> パーソナルコンピュータ業界では、私たちは少なくとも 10 年間にわたって多かれ少なかれ同じ問題を解き続けている。・・・（中略）・・・この 10 年間でソフトウェアにおいて本当に技術的に進歩したと言えるのは、リサのマルチタスク・オペレーティングシステムとグラフィカル・インタフェース、アドビのポストスクリプト印刷技術、そしてローカルエリア・ネットワークによるユーザのリンクしかない。
>
> ・・・（中略）・・・私たちはいまこの無風状態から脱出しようとしている。業界に変化を強いる大きな転換が、いままさに始まろうとしているのだ。標準を前提としたコンピューティング、ＲＩＳＣプロセッサ、先進的半導体、そしてメインフレームの死。

クリンジリーが同書を執筆した後で、Web の急速な普及があり、1990 年代後半以降は彼の予想以上のスピードで変化しています。これには、政治・経済の世界での変化の激しさも背景にあるのでしょう。

汎用機での開発が激減し、クライアント/サーバ型の開発すら急速に減り、Web、Java、XML、.Net へと急速に移行しています。

このことは、毎回未経験の技術での開発が求められるということを意味しています。未経験の技術は些細な過ちやエラーの温床であり、十分な調査期間が与えられないと品質低下を招きます。ところが、後述のとおり不況により予算が削減され、開発期間は短縮され、十分な調査期間は与えられなくなって来ています。そのため、品質が低下し、プロジェクトが予定通りには収束しないという危険性が高まっているのです。

### １.2.2.2. 不況と請負開発

#### (A)1990 年代前半までの請負

「2.2.1.請負契約が元来持っている危険性」で解説したとおり、請負契約というものは元来危険なものです。

しかし、1990 年代前半までのソフトウェア業界の元請けと下請けとの関係は系列的な関係でした。請負といえども「大儲けはさせないが、かかった分は出してやる」という雰囲気がありました。また、元請けに下請けを育てるという余裕がありました。

元請けと下請けは長期的な関係であり、あるプロジェクトで赤字になっても、次のプロジェクトで工数を上乗せしてもらえるなど、長期的には補填してもらえました。開発期間が長かったこともあり、ある工程の区切りで再見積もりすることも許されました。

仕事は豊富にあり、人手が不足していたので、下請けを長期的に囲い込んでおくことに意味があったからです。

### (B)不況と請負

この不況によって、エンドユーザもコンピュータメーカも金と力が無くなりました。プロジェクトの数が減り、開発費が削られ、開発期間は短縮され、単価が切り下げられました。現在の請負の状況を要約すると下記のようになります。

(a) 毎回未経験の技術による開発が求められるのに、調査に必要な予算と期間が与えられず、そのことによる技術的なリスクを受託側が一方的に抱えることが要求されています。

(b) 請負契約が本来持っている契約上のリスク「何が完成なのかが明らかになっていないのに完成を約する」が受託側に過酷な形で顕在化してきています。下請けを系列的に囲い込むメリットが無くなったため、工数オーバーしてもユーザや元請けが補填する処置を取らなくなったからです。

(c) 予算的にも期間的にもプロジェクトに初心者を参加させ教育する余裕が無くなっています。

(d) 未払いリスクも背負うようになりました。

請負は元々売掛金が多くなる傾向があり、資金繰りを圧迫します。それでも納期後に約束通りの期日で顧客が支払ってくれれば問題は少ないのですが、最近は支払期日延長や分割払いを要請されることが弊社でもたまにあります。

さらに顧客が倒産して売掛金が焦げ付く場合すらあります。もちろん、これは委任契約でも発生し得ることですが、売掛金が大きくなる請負で未払いが発生した場合の打撃は非常に大きく、弊社の周囲でもそれが原因で経営が傾いた例も珍しくありません。

### １.2.3.　多くのソフトウェア会社が採用している戦略

請負に対しどのような戦略を選択するかが、そのソフトウェア会社の性格と将来を規定します。

多くのソフトウェア会社が採用している主な戦略を概観します。

#### １.2.3.1. 請負をやらない会社

多くのソフトハウスは請負を諦め、技術者派遣のみ行っています。

これでは技術の蓄積と人材の育成という請負のメリットは享受できないのみならず、技術者派遣を戦略的に捉える姿勢が欠けているため技術者派遣のメリットも享受することができません。

### １.2.3.2. 常駐型請負

最近は社内持ち帰り型の典型的な請負が減り、常駐型請負が増えています。請負をやっているはずのソフトハウスを訪問しても大概技術者は全員出払っていて、机だけが並んでいるという状況になっています。

常駐型請負はけっして悪いことではありません。むしろ、下記の点では発注側・受託側双方にとってメリットがあります。

- 要求仕様の確定はユーザの近くでやる方が効率的です。
- 一つのプロジェクトをいくつかのチームが共同して行う場合、プロジェクトの初期の段階で同じ場所で作業し、相互に理解し合い信頼し合うことは有益です。

しかし、最初から最後まで常駐型請負でやると、どうしてもある程度発注側から管理されるようになります。これは一面では、受託側のリスクとプロジェクト全体のリスクを軽減する面もありますが、次のようなデメリットも出てきます。

- システムの完成に対する契約であるにもかかわらず、それに何人投入しているかを発注側に監視されることになります。その結果、請負が本来持っている「受託会社が生産性を上げて利益を上げることができる」というメリットが薄れてきます。それにもかかわらず、完成に対する責任だけは負わせられます。
- 発注側の標準プロセスに従って作業することとなるので、主体性を持って技術やノウハウを蓄積することができなくなります。

ちなみに 1990 年代前半まではコンピュータメーカが圧倒的な技術力を持ち、優秀なソフトウェア技術者がコンピュータメーカに集まっていたため、そこに常駐すると技術力を高めることができました。しかし、現在ではコンピュータメーカの技術力は必ずしも高いわけではなく、メーカ常駐の技術的・教育的メリットは少なくなっています。

### １.2.3.3. 管理主義の強化

#### (A)小さなソフトウェア会社

小さなソフトウェア会社では請負プロジェクトについては経営者が細部までチェックすることによって対応している会社もよく見かけます。技術畑出身の経営者がプロジェクトの細部にまで目を光らせるのです。

技術に明るく収支についての感覚が鋭く顧客との交渉も巧みな経営者なら、１プロジェクトとしての請負を成功させることができるでしょうが、その会社の規模は、その経営者が目の届く範囲に限定されてしまいます。また、ラインとスタッフの分離が起きず、近代的な企業にはなり得ません。

#### (B)システムインテグレータ

優秀なシステムインテグレータは、プロジェクト管理方法を研究しマニュアル化し、管理専門家を養成することにより、請負を成功させようとしています。

この方法により、I 社はある程度成功しているように見えます。

しかし、I 社のやり方は従来型のソフトウェア工学の延長線上にあるやり方であり、I 社が受託しているような（不況のため渋くなったとは言え）潤沢な予算のある大規模プロジェクト向きであり、小規模開発の生産性を飛躍的に高める方法論ではありません。また、I 社の受託案件でも、表面に出てこないだけで、依然としてかなりの確率で失敗プロジェクトは発生していると思います。

#### １.2.3.4. アウトソーシング

自社では営業とコンサルティング・要件定義等の上流工程のみやり、製造工程はアウトソーシングするという手法も多くのソフトハウスが憧れています。
しかし、中小規模案件の場合は、コンサルティングの予算は少なく、要件定義からテストまで同じチームが担当する方が効率がよいので、結局これも大規模開発向けの解決策と言えます。
また、開発自体の生産性と品質の向上については解決にはなっていないので、リスクと薄利を下請けに押し付けただけとも言えます。

## 1.3 **慶の戦略**

### 1.3.1. 開発現場の戦略：驚異的に生産性の高い小規模チーム

私は前章で解説したような請負の現状を見て、請負で利益を上げることについて一時期悲観的になりました。

しかし、「ソフトウェア職人気質」(ピート・マクブリーン著　ピアソン・エデュケーション発行)を読んで、請負で利益を上げることも可能であると考えるようになりました。

「ソフトウェア職人気質」の思想を受け継いで、請負を成功させる方法を次章にまとめました。

驚異的に生産性の高い小規模チームを作り出すことにより、高品質で堅牢なアプリケーションを安くかつ短期間でユーザに提供し、顧客とソフトウェア会社両方に利益をもたらすための方法です。

驚異的に生産性の高い小規模チームづくりに成功した近未来の慶の原則という形でまとめてみました。

#### １.3.1.1. 組織の原則

慶の組織は階層的な組織ではなく、効率の良い小規模チームが複数できる組織です。そのチームは、ある単発プロジェクトのために人が集められ、プロジェクトが終われば解散させられるグループではありません。次々にプロジェクトをこなしていく熟練者の下で一緒に働く長期的なチームです。

慶では熟練者といえども一人では作業をしません。もしも半年後を納期とする 3 つの開発案件があり、3 人の開発者がいる場合、それぞれの開発者に 1 案件を割り当て半年で開発させるのではなく、 3 人 1 チームで順次 3 つの開発をそれぞれ 2 ヶ月で完成させます。
その理由は下記のとおりです。

- アプリケーションの詳細知識は最低でも 2 つの頭に中にしまっておくべきです。たった一人のキーマンの頭の中にしか詳細知識は存在しないという状況は、引き継ぎに時間がかかる上、不慮の事故があった場合、開発も保守もできなくなってしまうからです。
- 優れた開発者同士のチームの生産性は彼らそれぞれの生産性を合計したものよりもずっと高くなります。適切な分業が行われ、また相互に知的に刺激し合うからです。
- チームによる開発は初心者にソフトウェア開発プロジェクトに参加する機会を提供します。（チームでは１人ないし２人の初心者を指導します。小さなプロジェクトなら、例えば「熟練者１名＋中級者１名＋初心者１名」という構成です。大きなプロジェクトなら、例えば「熟練者３名＋中級者６名＋初心者６名」という構成です。）
- チームによる開発は「開発・リリース・フィードバック」サイクルを短期化させ、開発者に何がうまくいき、何がうまくいかないかを学ぶ機会を増やします。

#### 1.3.1.2. **技術者の原則**

(a) 慶の技術者は「アプリケーション全体を一人の技術者が理解できる」と断言します。仕事の一部だけしかできない幅の狭い専門家ではなく、仕事を最初から最後まで全てやり遂げることができる真の技術者だからです。

(b) 慶の技術者はアプリケーション全体に対して責任を持ちます。

(c) 慶の技術者は高品質で堅牢なアプリケーションを構築する能力に強い関心を寄せ、ソフトウェア開発の熟達度をさらに向上させようと絶えず努力します。

(d) 慶の技術者はソフトウェア開発を楽しみます。そして、ソフトウェア開発に対する自分の意気込みと熱意を、同僚に波及させます。

#### １.3.1.3. **管理者の原則**

作業をどのように行うかという知識は管理職の頭に中にではなく、開発者の頭の中にあります。したがって、慶の管理者は技術者に対して詳しい指示をするのではなく、世話役と指導担当者として振る舞います。主な仕事は下記のとおりです。

・チーム間の調整
・必要となるリソースのスケジュール調整
・開発者、顧客間の関係を円滑にすること

管理者は技術者との対話をたやさず、命令とコントロールではなく、交流と対話によって管理します。

#### １.3.1.4. 開発の原則

(a) 開発標準

慶では、技術者はプロジェクト内で統一された開発標準に従って開発します。しかし、これは全てのプロジェクトが同一の開発標準に従うという意味ではありません。プロジェクトが置かれている特定の状況に基づいて、開発標準は変化します。

(b) コンポーネントの再利用

慶では、コンポーネントの再利用を無条件には奨励しません。
特定プロジェクト内で使用するために開発されたコンポーネントの再利用は容易に行えるものではありませんし、コンポーネントの長期的な再利用は基本的に危険です。慶では自社製フレームワークによって、注意深くしかし効果的にコンポーネントを再利用します。

(c) アプリケーションの分割

慶では、巨大な一枚岩のアプリケーションではなく、互いに協調して動作する小規模アプリケーションに分割して構築します。

(d) モジュールの分割

慶では、アプリケーションを凝縮度の高いモジュールへ適切に分解します。これが保守を容易にします。

(e) インクリメンタルな開発

慶では、設計やコーディングフェーズからではなく、要件定義フェーズからインクリメンタルな進化的開発を行います。特に新規性のある項目をプロジェクトの初期に開発します。

(f) 簡潔な設計

慶では、汎用性のある複雑な設計ではなく、可能な限り変更を容易にした簡潔な設計をします。

(g) 未経験の技術

慶では、未経験の技術を用いる場合には、その技術を調査できるだけの十分な時間をチームに与えます。また、開発チーム内の品質保証やテストといった段階に対して、十分な要員を配置します。

(h) 見積もり

慶は、見積もりを削ってまでプロジェクトを受注することはしません。精度の高い見積もりの作成するために時間と工数を確保します。

[コーヒーブレイク]　慶という社名

株式会社 慶 の社名は運慶に代表される慶派から取りました。
慶派は鎌倉時代の仏師の集団で、分業体制で短期間に巨大なプロジェクトを完成させたことで有名です。彼らの作品は写実的で力強い躍動感に満ち溢れています。

東大寺・南大門の金剛力士像（阿形・吽形の２体）は 1203 年に作られた日本最大の木彫の仁王像です。本に載っている写真ではあの像の迫力、素晴らしさは分かりません。下から見上げる効果が計算しつくされた像なのです。
この巨大な彫像を運慶らはわずか 69 日で仕上げました。高さ 8.4m に及ぶ巨像を 3,000 もの部品を組み合わせることで極めて短期間に作り上げたのです。

適切な部品化、周到な計画性、チーム内で標準化された作業手順、磨きぬかれた技、堅牢な設計、グループとしての強烈な個性、プロジェクトを完成するための強靭な意志・・・800 年前の慶派は請負開発を成功させるために何が必要であるかを示唆しています。

#### １.3.1.5. **教育の原則**

(a) 学習の主導権

学習は、教える側ではなく、教えられる側の方にはるかに大きな主導権があります。

慶では、初心者は熟練者から教えられるのではなく、仲間の助けを借りながら、どう作業すればよいか自分で考えるよう求められます。 熟練者が初心者の作業を監督する上で観察する必要があるのと同様に、初心者も熟練者の作業のやり方を観察し、学ぶのです。

熟練者は初心者に対して教育を行うのではなく、プロジェクトに参加する機会を提供するのです。互いに指導し合う実践の場を数多く与えます。

入門者は、彼らよりも数ランク上の中級者から教えられます。

(b) 採用

慶は、最小限度の監督しか必要としない、指導しがいのある自立した初心者のみ採用します。

(c) 教育期間

慶では、手っ取り早い一夜漬けの教育モデルは採用しません。

ソフトウェア開発での熟達は、高品質で堅牢なアプリケーション構築を積み重ねることによって達成されるのであり、長い期間が必要なのです。

それなりの規模のプロジェクトを最小限度の監督で任すことができるようになるまで少なくとも 5 年はかかります。

(d) レビュー

中級者、初心者はチームによって作り出された全ての成果物を全員が読みます。

同僚に対して自分の作業内容を見せ、自己の研鑽のために同僚の作業内容に目をやり、できる限り多くの漏れ、欠陥、過ちを指摘することが求められるのです。

#### １.3.1.6. 報酬の原則

慶は報酬での平等主義を排します。

飛び抜けて生産性の高い開発者というのは、プロジェクトの成否を決める存在です。彼は１人で平均的な開発者５人分以上の価値をプロジェクトにもたらします。プロ・スポーツの分野で花形選手よりもコーチの方に高い報酬が支払われることがないように、慶では優れた開発者は管理者よりも高い報酬を得ます。一方、平均的な開発者は必ずしも高い報酬を得ることはありません。

### １.3.2.　法務戦略

受注時に下記の点に注意しなければなりません。

#### １.3.2.1. **仕事の完成**

請負契約とは「仕事の完成」を約する契約なので、契約時に「完成とは何か」を明確にしなければなりません。

発注仕様書等で「仕事の完成」が不明確な場合には、作業内容とその作業の結果作成される成果物を一覧にし、その成果物ごとにサンプルを契約書に添付しておくことが望ましいです。つまり、作業ではなくて、作業の結果としての成果物に注目して業務範囲を特定するのです。

#### １.3.2.2. **納期**

納期が外部的要因によって左右される可能性がある等、納期を確定しにくい場合は、下記のような契約書の書き方も考えられます。

「乙は、本件業務を別紙の作業予定に従って行います。ただし、検収のための成果物の確定的な引渡日は、乙より甲に対して１か月前に事前通知することによって定めるものとします。」

#### １.3.2.3. **請負金額の決め方**

請負契約では普通は契約締結時に請負代金の総額を確定します。（これを定額請負と呼びます。）しかし、契約確定時にどうしても不確定な部分があれば、その部分のみ限定的に単価請負または概算請負にするという方法もあります。

(a) 単価請負

単価だけを取り決めておき、契約締結時に予定の投入量で請負代金を見積りますが、それは固定されたものではなく、作業の終了時において実績に基づき請負代金を確定する方法です。「工数契約」ともいわれます。

(b) 概算請負

概算金額で請負代金を決めておき、後で実際に要した費用に基づいて精算する方法です。

単価請負、概算請負ともに最低金額、最高金額を決めておいて双方の不安を少なくすることも考えられます。

## １.3.3.　営業戦略

### １.3.3.1. ジョン・スポールストラのように

「1.3.1. 開発現場の戦略：驚異的に生産性の高い小規模チーム」で述べたことは、才能のあるスポーツ選手を集めて訓練し最強のチームを作ることに似ています。
しかし、最強のチームが必ずしも興行的に最も成功しているわけではありません。弱くても興行的に成功しているチームもあります。例えば、「エスキモーに氷を売る」の著者ジョン・スポールストラは NBA で最弱(27 位)のチームを使ってそこそこの観客動員数(12 位)を達成しました。
一朝一夕で最強チームを作れない以上、営業がジョン・スポールストラのように知恵を絞って、限られた資源を最大限に生かして良い興行成績をあげていかなければなりません。

### １.3.3.2. 請負営業の使命

請負営業の使命は、一定量の良い仕事を確保することです。
安定した顧客が存在している場合、ソフトウェア会社の営業はあまりやることがありません。顧客が安定した仕事量を確保してくれるからです。
しかし、この不況により、安定して仕事を発注できる顧客が減りました。そのため、多く請負型ソフトウェア会社は営業的に不安定な状況にあります。また開発期間が短くなっていることも仕事量を不安定にしている原因の一つです。次に示すようなことを多くのソフトウェア会社が経験しています。

- 短納期でボリュームのある仕事を受注し、それをこなすために技術者を増やした。しかし、その次の仕事が見つからず、技術者が空いてしまった。
- プロジェクトが発生しそうだというので要員を確保していたら、顧客の予算の関係でプロジェクトが無くなり、投入を予定していた要員が空いてしまった。

安定した仕事量を確保する具体策としては下記が考えられます。

- 後述の技術者派遣営業と連携しながら相乗効果を狙っていく。
- 効果的な宣伝
- 人的リソース未来予測の確度を高める。

### 1.3.4. 課題

「1.3.1. 開発現場の戦略：驚異的に生産性の高い小規模チーム」で述べたことは世界的に見ても実現されている例は少ないので、慶がそれを実現できれば、経済的にも文化的にも大きな意義があります。ソフトウェア産業は、日本の人件費の高さを生産性の高さで補える数少ない基幹産業になり得ます。
また、技術的な強みを持つということは「第 3 章　知識集約型企業へ」で解説する知識集約的な新しいビジネスモデルを生み出すために不可欠です。

しかし、慶の（そして多くの中小ソフトハウスの）現実に立ち返って見るなら、依然として多くの課題を抱えていることに気付かされます。

#### １.3.4.1. **時間的な問題：一朝一夕にできない**

(a) チームを率いることができる熟練者の絶対数が不足しています。
社内だけでなく、社外にも少なく、すぐに集まるものではありません。

(b) 初心者が熟練者になれるか否かは、その人の素質と情熱に依存します。
全員が熟練者になれるわけではありません。また、ある人がなれたとしても、それまでに非常に長い時間が必要です。

#### １.3.4.2. **組織上の問題**

ライン部門とスタッフ部門がバランスのとれた形で分離しません。教育、収支管理、人事管理、営業すらもライン部門で行われることになり、一旦ライン部門が暴走しだすと止められなくなります。

#### １.3.4.3. **財務上の問題**

(a) 売掛金負担が大きく、資金繰りが苦しいという請負本来が持つ構造は変わりません。

(b) 固定費率が高まります。
安定したチームを作ろうとすれば、長期の雇用が前提となり、固定費率は高まります。契約社員として雇用しても長期の雇用が前提となり、実質的には固定費となります。したがって、仕事が減った場合、あるいは予定していたプロジェクトが無くなった場合のダメージは大きくなります。

(c) 労働集約的な構造は変わりません。

#### １.3.4.4. **人事面の問題**

後述の技術者派遣の場合、技術者は常に自分自身の市場価値を意識します。個人売上が明確なので自分自身の市場価値が自明なのです。しかし、チームで請負をやる場合、個人売上が明確になりません。そのため、甘えの温床になる可能性があります。甘えの構造を生み出さないためには、技術者の市場価値というものを常に意識させる仕組みが必要です。

#### １.3.4.5. **その他の問題**

受け入れ可能な初心者の数は限られています。したがって、とりわけ現在の日本において社会的に極めて重要な雇用拡大にはすぐには繋がりません。

・ ソフトウェア開発・利用契約と契約文例事例集
（http://www.juas.or.jp/usc/manual/text-1/Default.htm）
法律知識については、このサイトを参考にしました。

・ ソフトウェア職人気質
ピート・マクブリーン著（ピアソン・エデュケーション　発行）

・ コンピュータ帝国の興亡
ロバート・X・クリンジリー（アスキー　発行）

・ エスキモーに氷を売る
ジョン・スポールストラ著（きこ書房　発行）

著者紹介
**蒲生 嘉達**（がもう　よしさと）
昭和 33 年生まれ。
中央大学法学部卒業後、特許情報関連の企業に就職。
その後、ソフトハウスに転職し、13 年間在籍。SE・プロジェクトリーダーとして数多くのシステムを手がける。平成 10 年、株式会社慶を設立。

---

**ソフトウェア業界航海術**
**～業界の基礎知識、分析、そして慶の戦略～**

---

平成 15 年 3 月 14 日　初版第 1 刷発行
■著者　蒲生 嘉達
■発行所　株式会社　慶
〒170-0013　東京都豊島区東池袋 1-15-10 池袋ビル 2F
電話(03)5951-8490
FAX(03)5391-3676
URL　http://www.kei-ha.co.jp/
e-mail　kei@kei-ha.co.jp

表紙及び裏面の写真：Photo by (c)Tomoyuki.U
URL(http://www.yun.co.jp/~tomo/photo.html)

# ソフトウェア業界航海術

～業界の基礎知識、現状分析、そして慶の戦略～