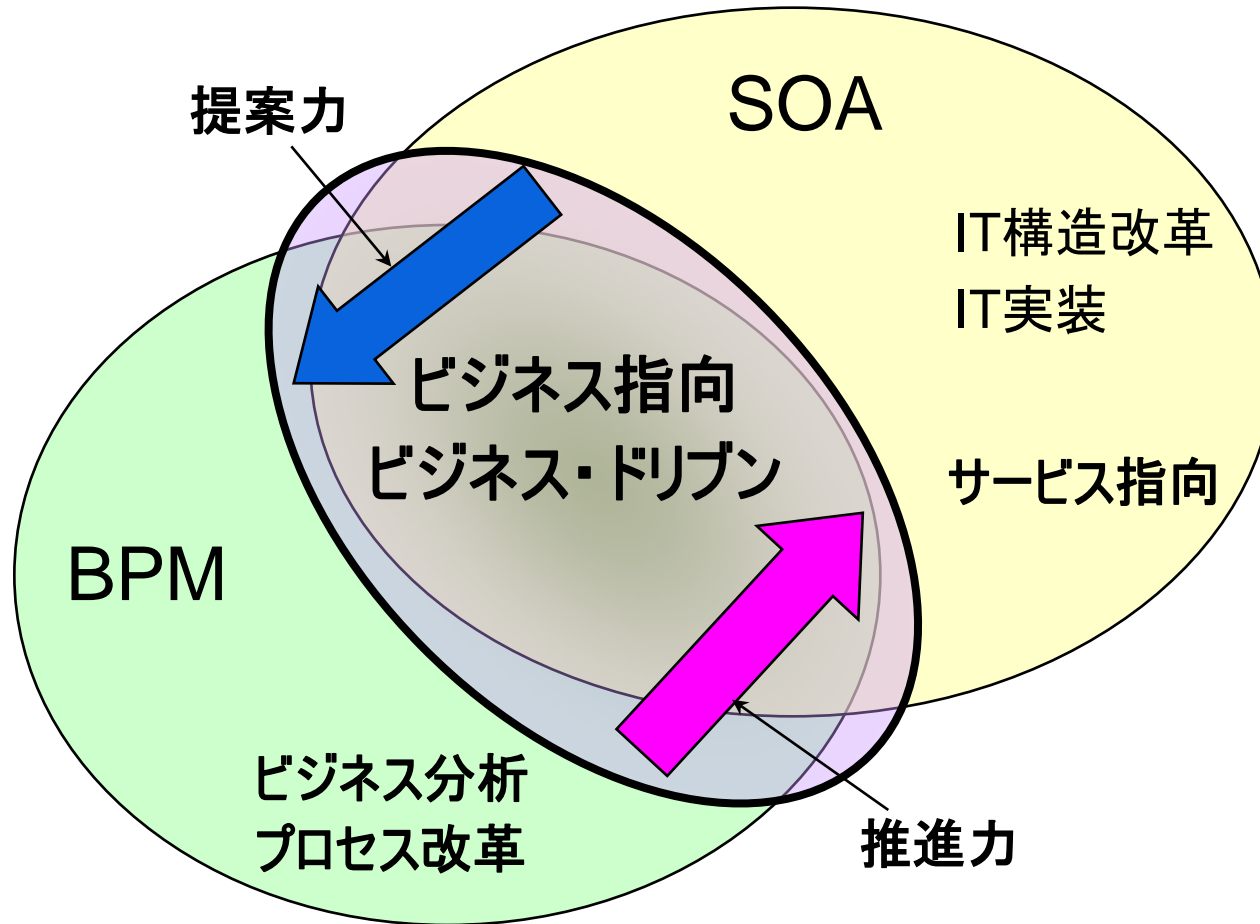


SOAはビジネス・ドリブンであるべき

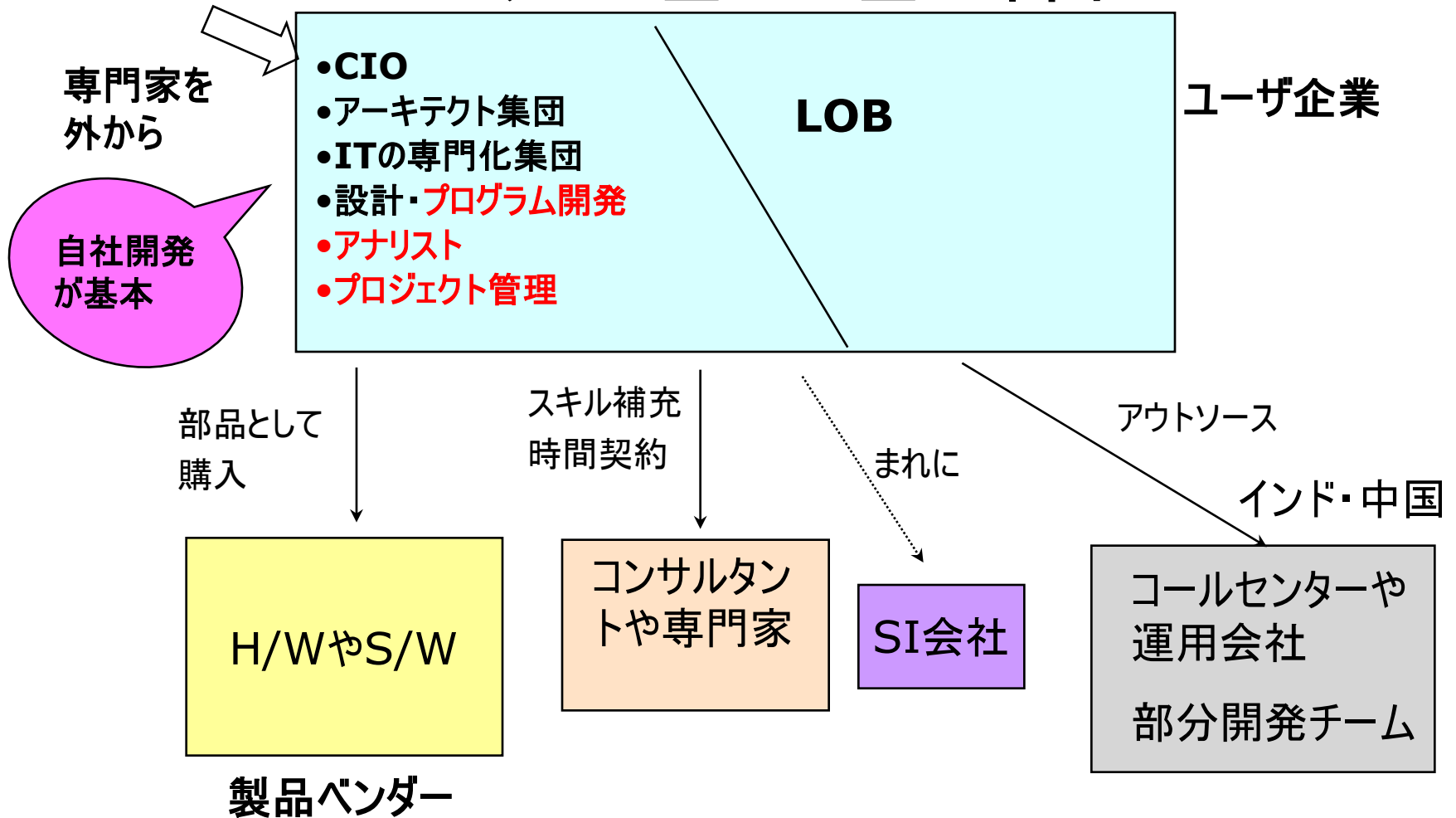


SOAの普及を妨げている要因

- ◆ 技術スキルの成熟度などの問題、**スキル・経験**の領域
 - 使ったことのない技術要素が多い
- ◆ **企業内部の体制的な問題**領域
 - 業務部門とIT部門の関係、予算権限
 - IT部門だけの話題になっている
 - SOAでは、業務部門が開発に参加する度合いが大きく増える
- ◆ 企業内部で完結しない**外部への依存性**に関する領域
 - プロジェクト・スポンサー、発注側の技術・管理能力、依存関係
 - SI会社側の現実、契約形態、多すぎる関係者
- ◆ **技術者**にとっての領域
 - 優秀な技術者ひとりがプログラミングすれば出来るわけではない
- ◆ 過去の成功体験から離れられない**カルチャー**
 - 大規模プロジェクト vs. 小規模に分割したプロジェクト
 - 開発方法論

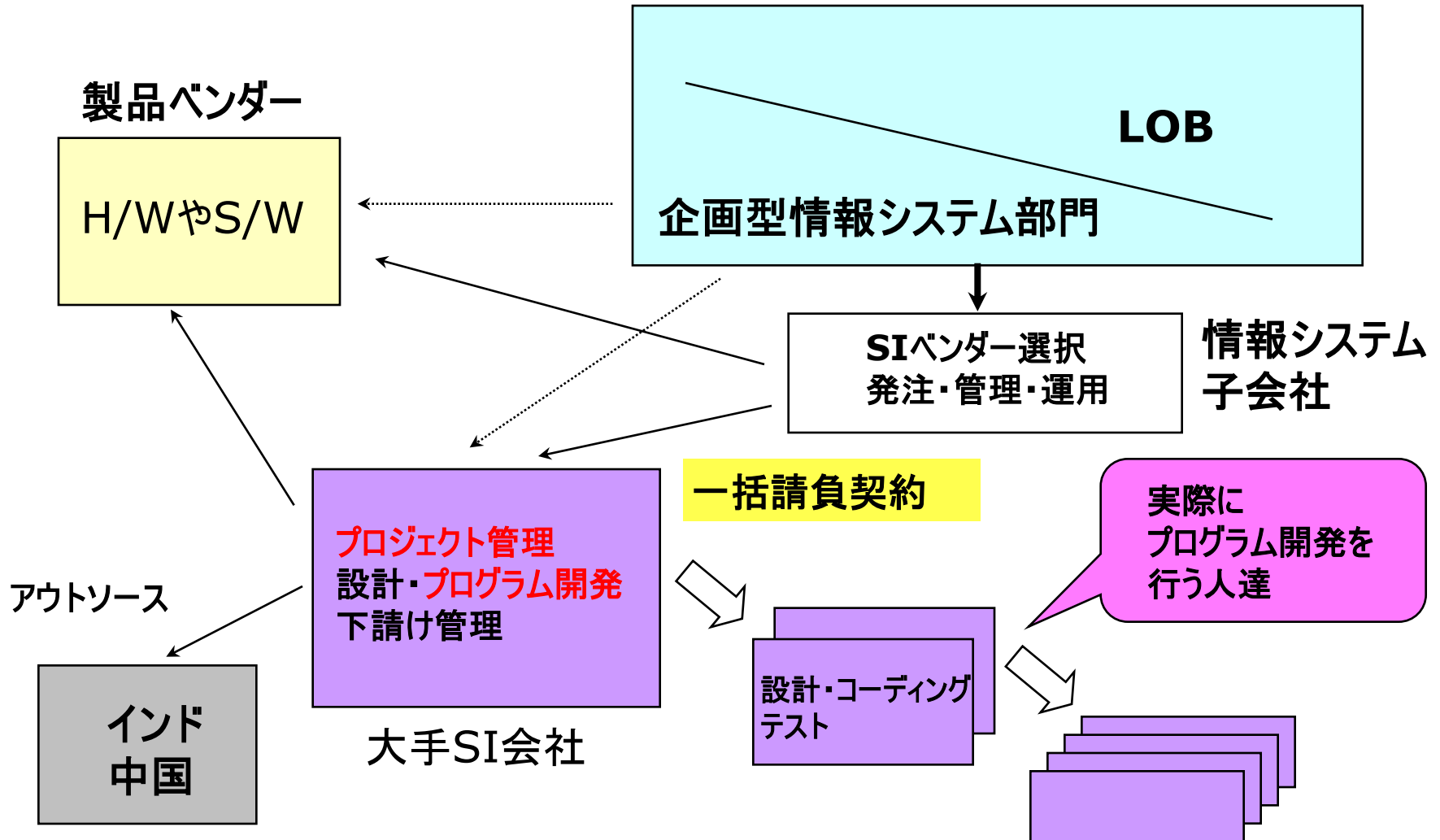
米国の典型的企業のIT組織構造

ガバナンス型・DIY型の米国

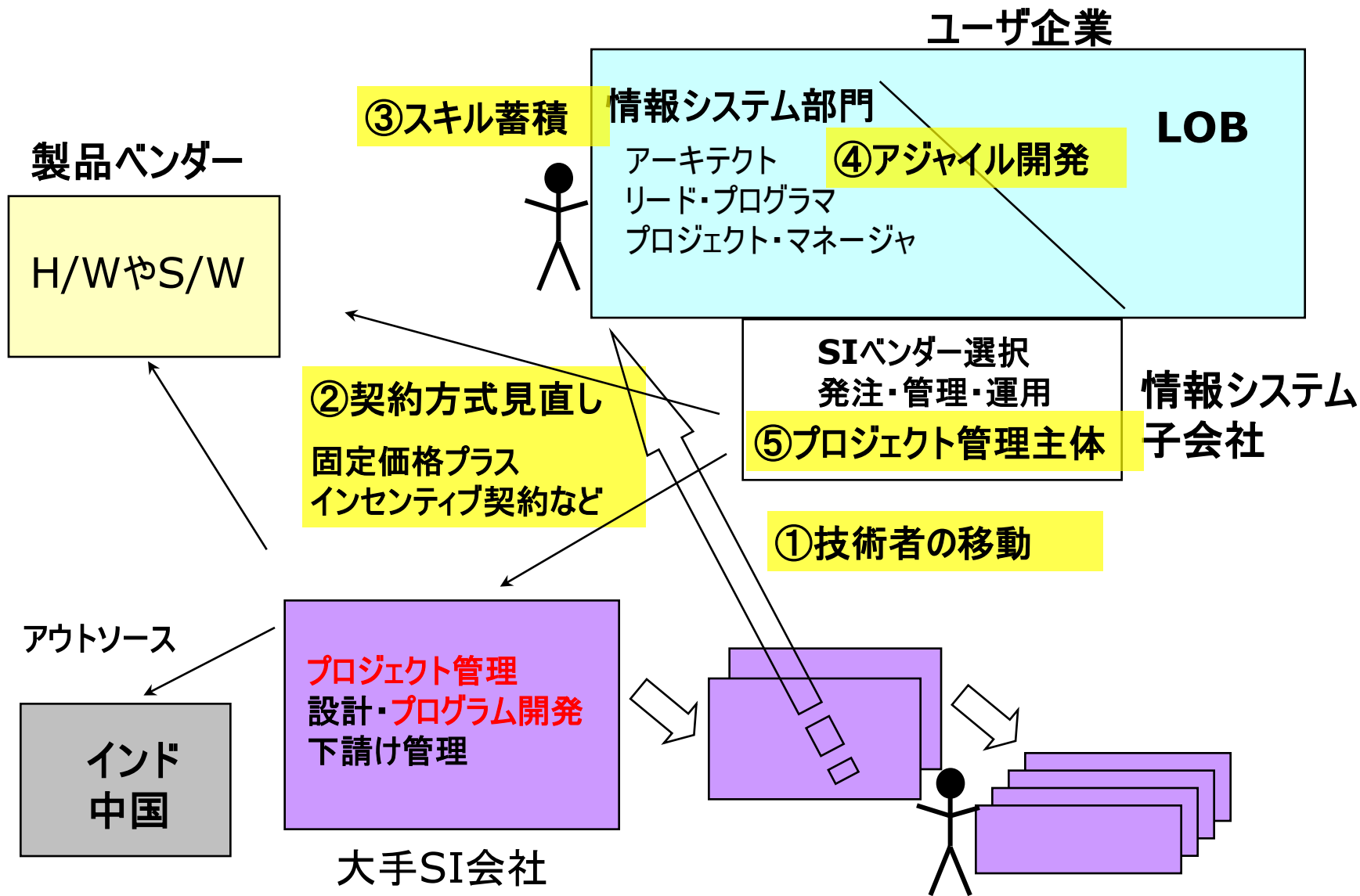


日本の一般的なIT組織構造

ユーザ企業 依存型の日本



ガバナンスを確立してSOAも実現



Backup資料

ウォーターフォール型が最適という神話

- ◆ 一般的にプロジェクトの成功率というのは低いと認識されており、そのため契約時にスコープを決定しておき、責任範囲を明確化するのが、管理できる・成功できるはず、という思い(希望的観測)
- ◆ 今までやってきた方法なので実績として安心できる
- ◆ 大規模開発でも段階的に出来るし、フェーズ毎に別の会社が担当することも可能
- ◆ 大規模プロジェクトはウォーターフォール型でないとできない(と思う)
 - 反復型、スパイラル型などは、小規模のプロジェクト用だと信じられている
 - ✓ 反論: 高層ビルの建築、大型飛行機の開発・製造、高速道路の建設
- ◆ ウォーターフォール型は発注側の負担が少ない(と思う)
- ◆ 要件定義だけ出したら、完成品の納入までは発注側の仕事は少なくて済む(と思う)
 - 完璧な要件定義をプロジェクト開始時に出せると思っている
- ◆ 品質が高いものが出来るとPMが思っている
- ◆ ウォーターフォール型は、一括請負契約が可能で、まとめて売り上げが立つので嬉しい
- ◆ ひとつの案件の発注金額を大きくして発注件数を少なくするほうが、発注側の手間が少なくて済む
- ◆ ウォーターフォール型では、工数の計算がエンジニアのランクに従って機械的に可能なので見積もり易く(請負)価格が決まる
 - 人月の計算は昔の工場労働者のようであり、技術者のモラル低下の一番の原因
- ◆ ウォーターフォール型で成功した人は、最新の製造業の手法などに学ぼうという気持ちはない
 - 最近の自動車製造業の新車開発プロセスは極めて反復的、コンカレント、共同作業である

一括請負型システム開発の問題点

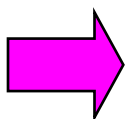
- ◆ 請負型の契約では、発注側の業務部門との共同作業などは考えられずSOA開発に不向き
- ◆ 人月を少なく済ませてコストダウンしようというモチベーションは、SI企業側には出てこない
- ◆ 人月の神話 The Mythical Man-Month : Essays on Software Engineering
- ◆ ユーザーはスキルを買ったのではなく、人月を買ったので決められた時間分働いていることを重視しがちになる
- ◆ 紙のアウトプットの量や、ソースコードの量(ライン数)で評価される
- ◆ 高いスキルを持ち、短時間で少ないコード量で質の高い仕事をするエンジニアよりも、長時間労働で長いだけのコードを作るエンジニアが評価される傾向がある(本当??)
- ◆ 発注者が開発ベンダーを評価する方法が、無いか、あっても科学的ではない
- ◆ 発注側にアーキテクチャーや設計・コードの良し悪しを評価できるスキルが必要
 - 姉齒のような設計をされても問題を発見できない可能性
- ◆ ユーザー企業側がPMを出すことはまれであり、発注した案件の責任も取れないスタイルが多いと思われる
- ◆ ビジネス側の評価を、プロジェクト進行中に適宜得るプロセスがない
- ◆ 要件定義書に100%正しい要件を書けるという大前提から出発している
- ◆ SI企業側からは、要件定義書に書かれていることが金額の攻防ラインになる

発注側の権利と責任

- ◆ SOAではビジネス側に真に役立つサービス設計・開発や組み合わせ・フローによる統合アプリケーションを作るので、開発プロジェクトの初期の段階から今まで以上にビジネス側の参加が必要になる。それなりの参加ワークロードがかかることへの理解が必要
- ◆ SI企業が作成・納品する設計書・コード・運用スクリプトなどは、すべて発注側の資産として再利用が可能なようにするべきである
- ◆ 契約方式の見直し
 - 一括請負契約
 - ✓ 最初に設定した契約金額の範囲で、請負者は利益を確保しながら、指定されたサービスや製品を提供する契約方式。最も一般的な契約方式。受注者が一方的にリスクを負うことになる
 - 単価契約(米国型に近い)
 - ✓ 契約総額を決めずに、サービス単価、製品単価のみを決め、発生した作業料、製品数に単価をかけた金額を受領する契約方式。発注者の知見が必要
 - 固定価格プラスインセンティブ契約
 - ✓ 契約時に契約金額を固定した上で、実コストが予算を下回った場合ボーナスを支払う、上回った場合ペナルティを支払う契約方式。発注者と受注者のリスクバランスを取ることができる

ITエンジニアがスキル向上のモチベーションを持ってない

- ◆ ITエンジニアが誇りを持ってスキル向上も目指し、良い設計・コードを作れるモチベーションは何か？
 - 献身的な心意気や仕事への誇りは、その企業の社員でないと不可能であろう
 - 実際にプログラムを書く人たちがプロジェクトに関わる構造が、米国と大きく異なる
- ◆ ITProの記事より — IT業界のタブー「偽装請負」 鈴木 孝知 = 日経ソリューションビジネス
<http://itpro.nikkeibp.co.jp/article/OPINION/20060113/227252/>
 - 偽装請負、多重派遣は違法であること以外にも問題がある。「いったん多重派遣される側の企業に入った技術者は“プログラミング・マシーン”から抜け出せない」という問題だ。違法な業界構造のなかで、IT技術者としてのキャリアをつぶしてないだろうか。
 - 最初は、「ばりばりのプロジェクトマネジャー」を目指してIT業界に入ったはずの若者が、「人と話すのは1カ月に1度」「開発の最初から最後まで携わったことがない」「指示書のままにプログラミングするだけで、職場でモノを考えることがない」。こうした現場を転々とする生活を10年すると「心が壊れる」。偽装請負についての取材のなかで出てきた実話だ。



下請け構造に関係なく優秀なITエンジニアを、ユーザ企業が直接採用する動きやPostingシステム、流動性向上に期待したい