

アジャイル開発の プラクティス・リファレンスガイド

～アジャイル開発を適切に導入するための工夫と留意点～

Agile Japan 2013
2013年5月24日

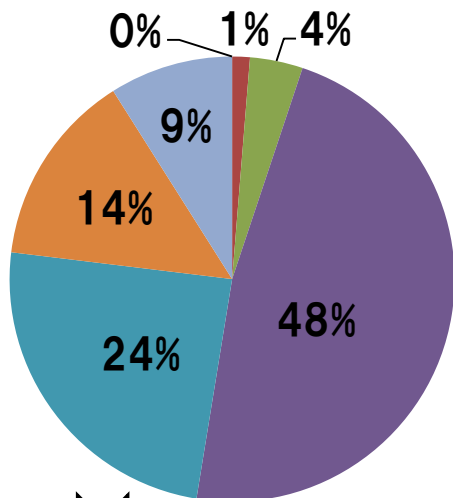
<http://www.agilejapan.org/index.html>

独立行政法人情報処理推進機構 (IPA)
技術本部 ソフトウェア・エンジニアリング・センター (SEC)
山下 博之

プロローグ：アジャイル開発の適用状況（俯瞰）

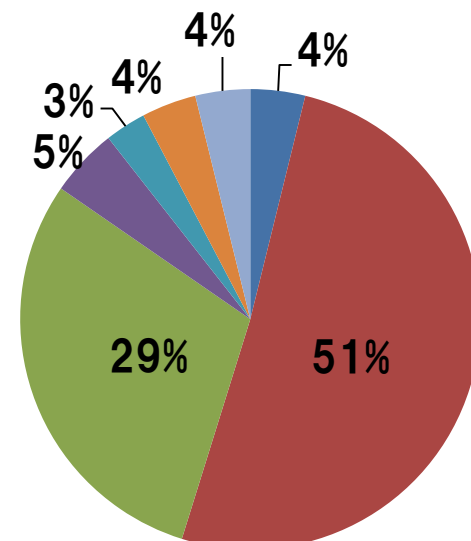
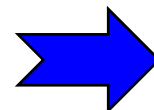
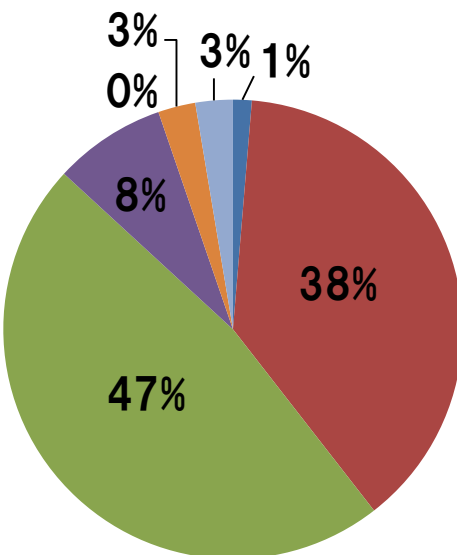
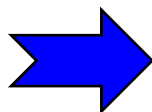
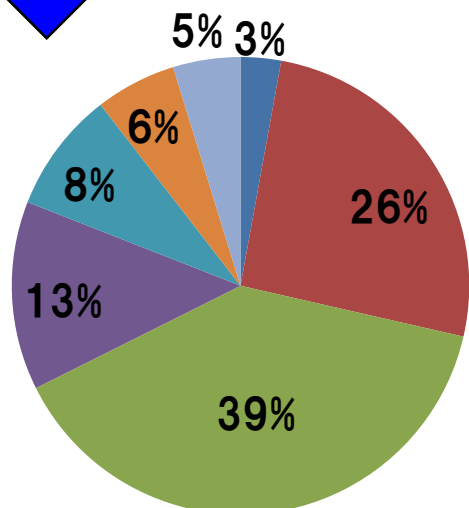
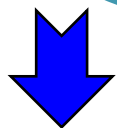
2010年12月2日 横浜 (72名)

～IPA/SECセミナー聴講者アンケート結果から～



- すべてのプロジェクトに適用している
- ほとんどのプロジェクトに適用している
- だいたいのプロジェクトに適用にしている
- ほとんどのプロジェクトに適用していない
- すべてのプロジェクトで適用していない
- 分からない
- 無回答

- 多くのプロジェクトで使っている
- 一部のプロジェクトで使っている
- 使いたいと考えているが実現していない
- 使う予定はない
- よく分からない／関係ない
- その他
- 無回答



2011年11月18日 横浜 (109名)

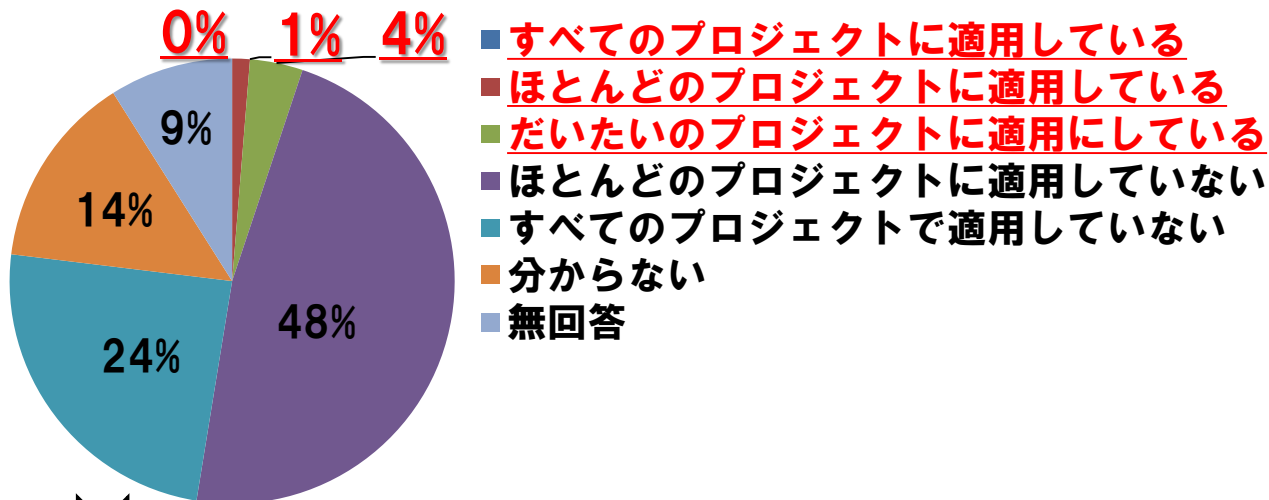
2012年10月24日 東京 (76名)

2013年3月18日 東京 (104名)

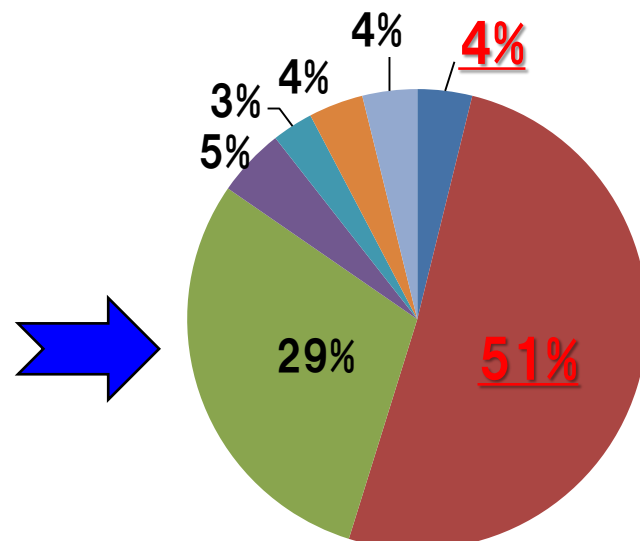
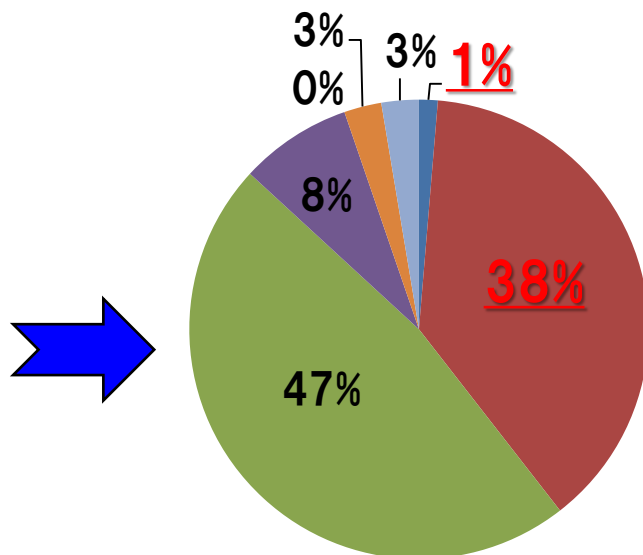
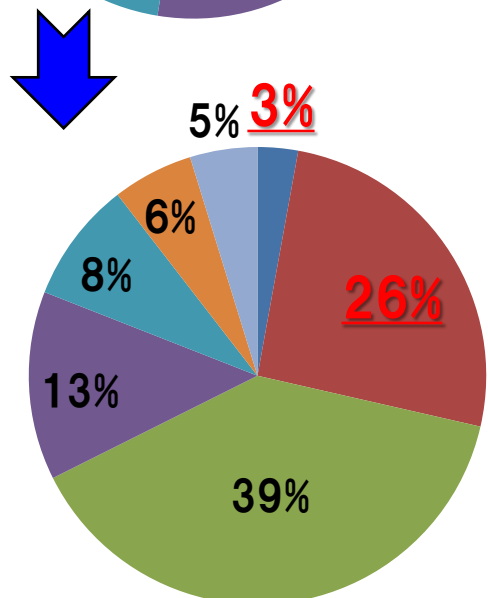
プロローグ：アジャイル開発の適用状況（１）

2010年12月2日 横浜(72名)

～IPA/SECセミナー聴講者アンケート結果から～



- 多くのプロジェクトで使っている
- 一部のプロジェクトで使っている
- 使いたいと考えているが実現していない
- 使う予定はない
- よく分からない／関係ない
- その他
- 無回答



2011年11月18日 横浜(109名)

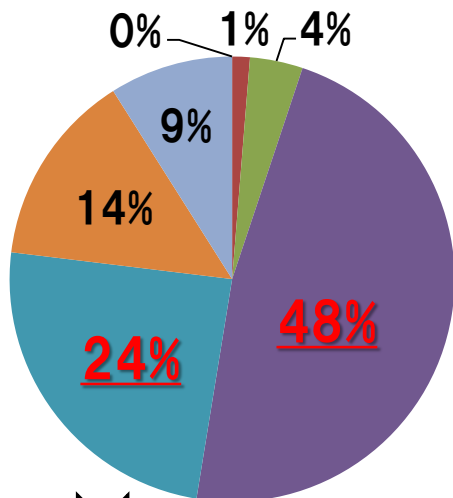
2012年10月24日 東京(76名)

2013年3月18日 東京(104名)

プロローグ：アジャイル開発の適用状況（２）

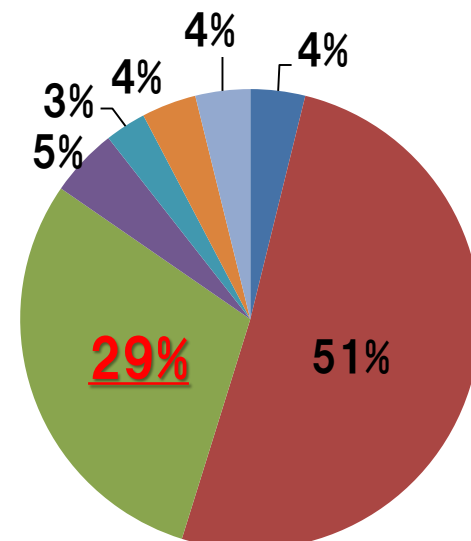
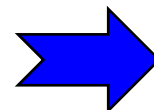
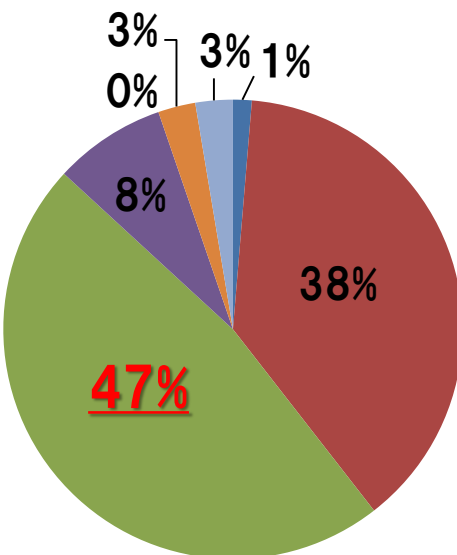
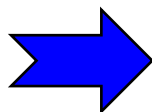
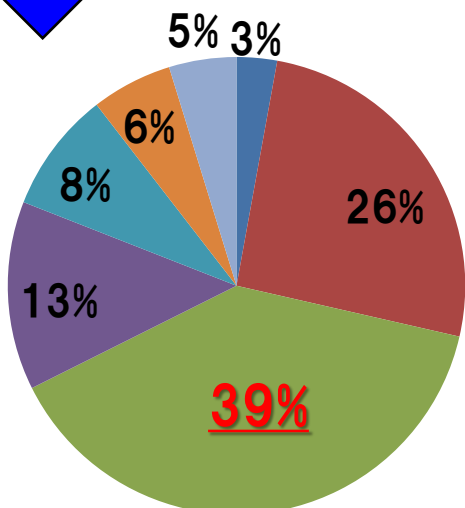
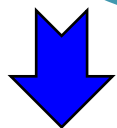
2010年12月2日 横浜(72名)

～IPA/SECセミナー聴講者アンケート結果から～



- すべてのプロジェクトに適用している
- ほとんどのプロジェクトに適用している
- だいたいのプロジェクトに適用にしている
- ほとんどのプロジェクトに適用していない
- すべてのプロジェクトで適用していない
- 分からない
- 無回答

- 多くのプロジェクトで使っている
- 一部のプロジェクトで使っている
- 使いたいと考えているが実現していない
- 使う予定はない
- よく分からない／関係ない
- その他
- 無回答



2011年11月18日 横浜(109名)

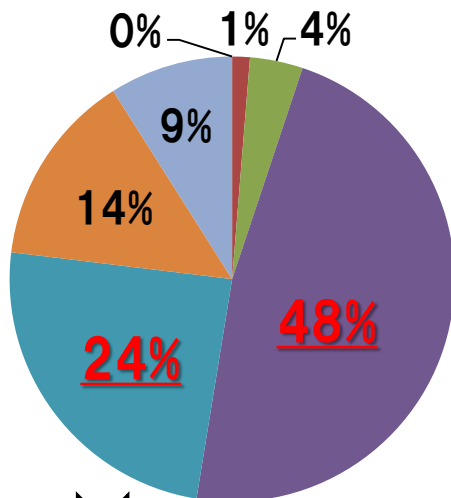
2012年10月24日 東京(76名)

2013年3月18日 東京(104名)

プロローグ：アジャイル開発の適用状況（3）

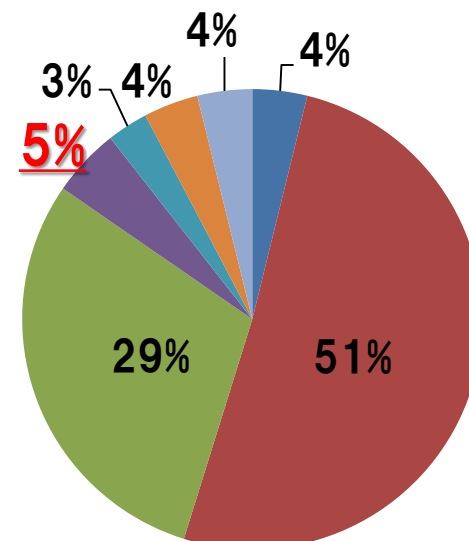
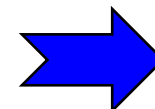
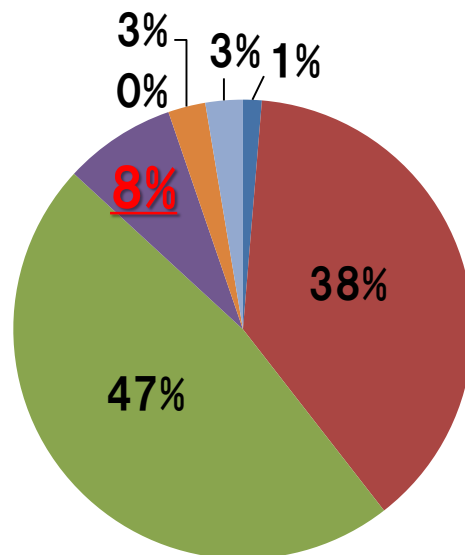
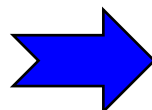
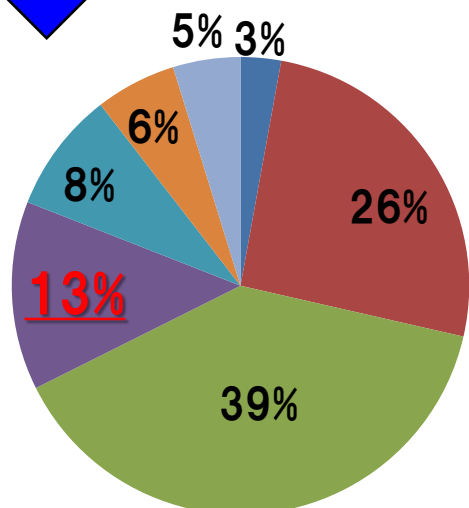
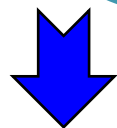
2010年12月2日 横浜(72名)

～IPA/SECセミナー聴講者アンケート結果から～



- すべてのプロジェクトに適用している
- ほとんどのプロジェクトに適用している
- だいたいのプロジェクトに適用にしている
- ほとんどのプロジェクトに適用していない
- すべてのプロジェクトで適用していない
- 分からない
- 無回答

- 多くのプロジェクトで使っている
- 一部のプロジェクトで使っている
- 使いたいと考えているが実現していない
- 使う予定はない
- よく分からない／関係ない
- その他
- 無回答



2011年11月18日 横浜(109名)

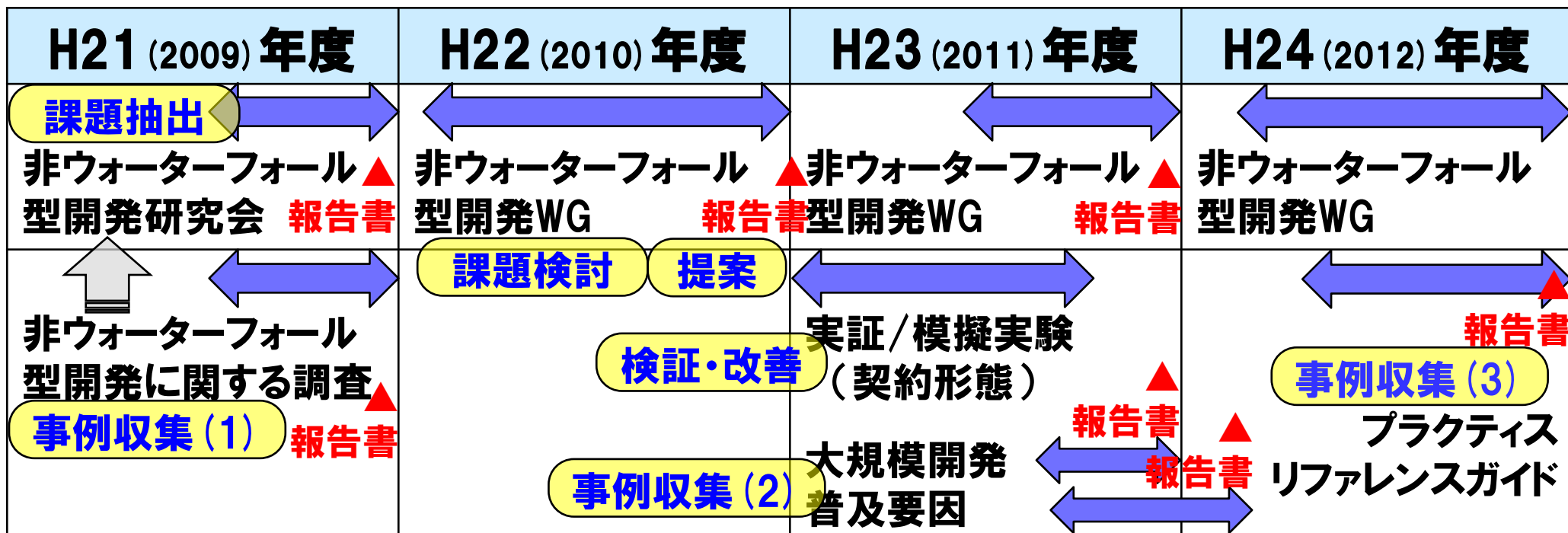
2012年10月24日 東京(76名)

2013年3月18日 東京(104名)

1. IPA/SECの4年間の活動の振り返り
2. アジャイル開発プラクティス・リファレンスガイド
3. アジャイル開発人材の育成

IPA/SECの4年間の活動の振り返り

アジャイル開発に関するIPA/SECの取組み



報告書(公開中)

H21年度版

<http://sec.ipa.go.jp/reports/20100330a.html>

H22年度版

<http://sec.ipa.go.jp/reports/20110407.html>

H23年度版

<http://sec.ipa.go.jp/reports/20120326.html>

(事例)

<http://www.ipa.go.jp/about/press/20120328.html>

(普及要因)

<http://sec.ipa.go.jp/reports/20120611.html>

(プラクティス)

<http://www.ipa.go.jp/about/press/20130319.html>

日本のソフトウェア競争力を高める
生き生きと働ける環境を作る

目指すべき
ゴール

日本におけるソフトウェア開発の在り方

領域
見定め

アジャイル

契約

(日本にふさわしい)契約のあり方

価値評価

ユーザ・ベンダ経営層の意識

環境整備

管理手法や技術面の整備
ツールによる支援

人材

人材の育成と適正配置

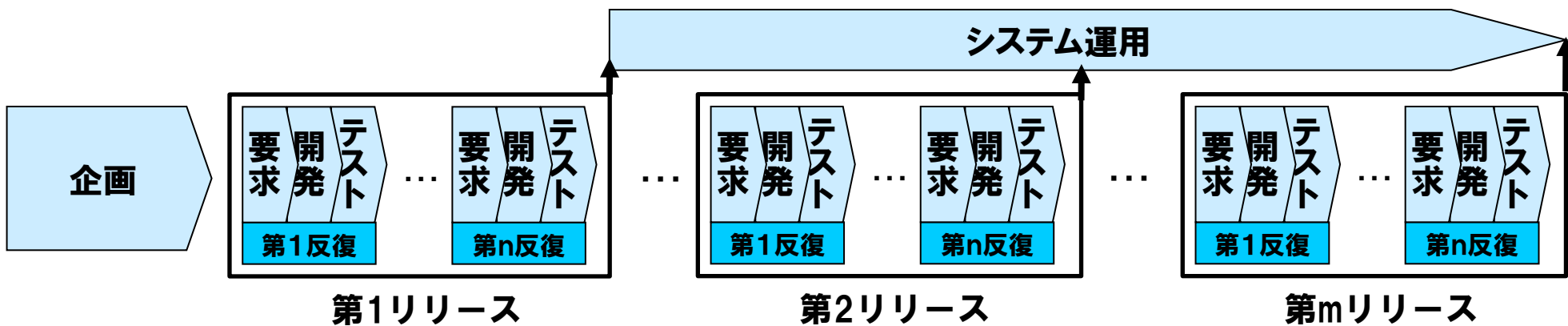
普及

適用領域と適用事例の調査
欧米の競争力(普及要因)の調査

事例収集(1)に
より抽出された
重点課題

- (1) アジャイル開発のプロセスモデル**
- (2) ユーザ・ベンダ経営層が意識すべきこと**
- (3) アジャイル開発に特徴的なスキルと人材育成**
- (4) 海外におけるアジャイル開発の普及要因**
- (5) 日本におけるアジャイル開発に適した契約形態**
- (6) アジャイル開発の適用領域とその拡大**
- (7) アジャイル開発プラクティス・リファレンスガイド**

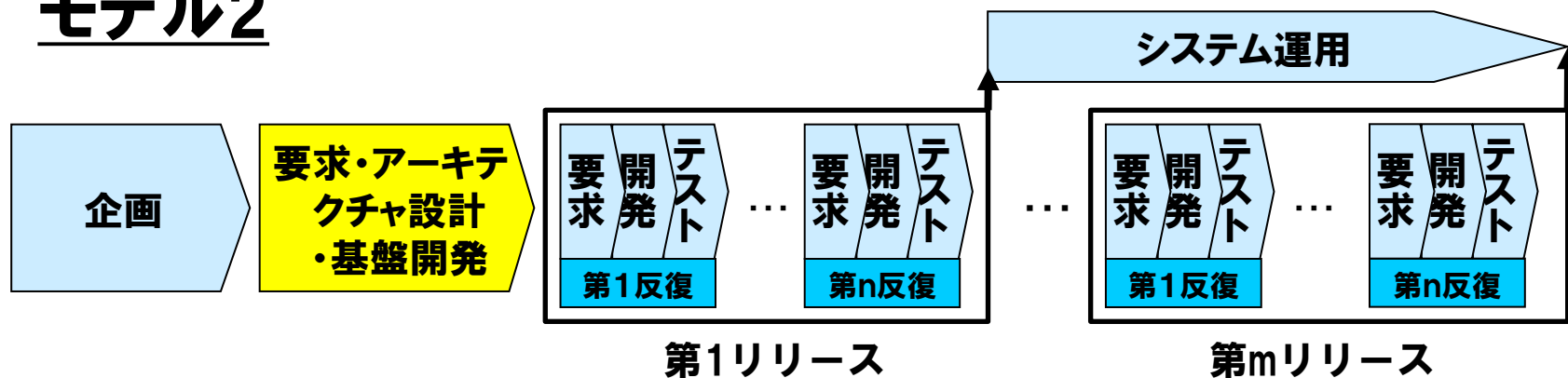
モデル1



- ・ $n=1$ のケースもあり。

考え方
シンプルな基本形

モデル2



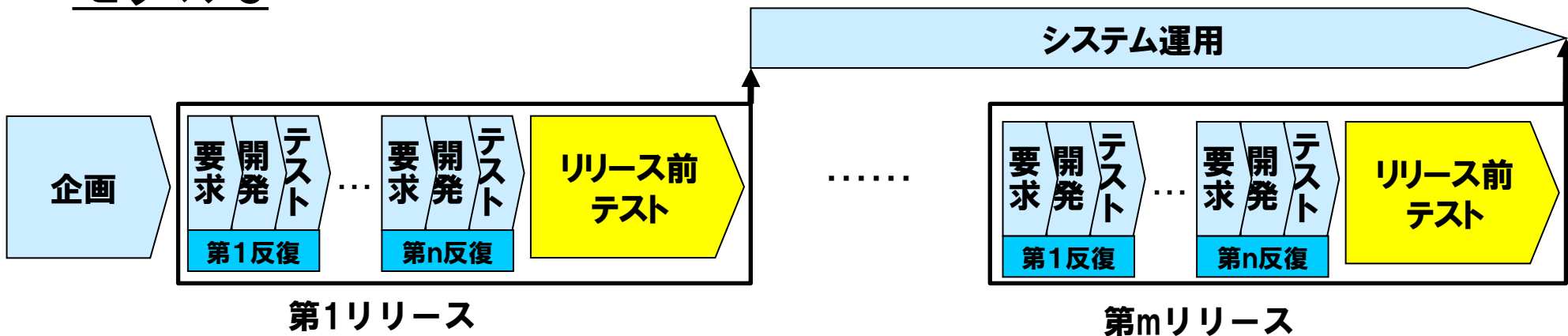
- 比較的大規模システム／新規開発で全体のシステム構造が不明確なケースなど

考え方

拡張形。基盤・共通部といくつかの機能部とから構成されるソフトウェア(右図)において、最初にまず、**基盤・共通部の開発**を終えた後、機能部群について、アジャイル開発を行う。基盤・共通部が確固としていないと、追加・変更時の機能部への影響が大きくなりすぎることを避ける。アジャイル開発では、基盤・共通部の変更は、原則として行わない。



モデル3



- ・ アジャイル開発では反復ごとにリリースできる品質までテストを行うことが原則だが、各リリース工程前に行う重点的なテストを実施することがある。
- ・ リリースは複数回繰り返される

考え方

顧客やビジネスの特徴から、特に高い品質が求められたり、品質がクリティカルであったりする場合に、**リリース前に品質確保**のための特別のアクションを実施する。

顧客・経営層は開発への一層の関与が必要

顧客(ユーザ)経営層

ビジネス環境が激しく変化する現状において、ITシステムに関し、従来のように情報システム部門に任せきりでは適切に対応できない。開発形態(*)にも深く関与する必要がある。

(*) アジャイル開発の採用、クラウドコンピューティングの利用、など

<経営層の責任>

- ・情報システムに関する理解の増進
- ・迅速かつ適切な意思決定
- ・関係部門との経営上の綿密な調整

ベンダ経営層

俊敏な開発の実績を武器に受注を狙う海外勢等に対抗するためには、自ら俊敏な開発を実施できる体制作りに取り組むと共に、その結果を顧客に売り込む必要がある。

ベンダ経営層は開発への一層の関与が必要

顧客(ユーザ)経営層

ビジネス環境が激しく変化する現状において、ITシステムに関し、従来のように情報システム部門に任せきりでは適切に対応できない。開発形態(*)にも深く関与する必要がある。

(*) アジャイル開発の採用、クラウドコンピューティングの利用、など

<経営層の責任>

- ・情報システムに関する理解の増進
- ・迅速かつ適切な意思決定
- ・関係部門との経営上の綿密な調整

ベンダ経営層

俊敏な開発の実績を武器に受注を狙う海外勢等に対抗するためには、自ら俊敏な開発を実施できる体制作りに取り組むと共に、その結果を顧客に売り込む必要がある。

アジャイル開発におけるユーザ側に求められること:

(全ての機能の仕様を洗い出す能力よりも)コアとなる機能を見定め、優先度を図りながら開発プロジェクトの運営を指揮していく能力

明確な仕様を決めなくても良いとはいうものの、定期的なサイクルで実物を見てフィードバックのポイントを増やすことにより、実際のシステムを目で確認しながら、積み上げるように仕様を決定していく

アジャイル開発のベンダ開発者にとって重要なスキル:

- ① プロジェクトのアウトプットに関わる判断ではなく、アジャイル開発の進め方を踏襲させるためのファシリテーションスキル
- ② 反復活動の中で、実際に動くものを作りながら、小規模に、かつトータルにプロジェクトのアウトプットを積み重ねていくスキル
- ③ 設計, コーディング, テストを一貫して実施出来るスキル

Scrum Master等の推移から見るアジャイル開発の普及状況

- アジャイル型開発方法論で最も有名なスクラムに関する資格者が、2005年以降急増
- **米国の取得者が群を抜いて多く、ついで英国が多い。日本は極めて少ない** (試験は英語)

Scrum Master等の人数の経年変化

単位 (人)

| | 2002 | 2003 | 2004 | 2005 | 2006 | 2007 | 2008 | 2009 | 2010 | 2011 | TOTAL |
|-------|------|------|------|-------|-------|--------|--------|--------|--------|--------|---------|
| CSM | 5 | 344 | 907 | 2,647 | 6,841 | 12,857 | 22,514 | 26,886 | 34,601 | 43,028 | 150,630 |
| CSPO | | | | | 83 | 503 | 1,891 | 3,514 | 5,325 | 8,629 | 19,945 |
| CSP | 1 | 2 | 14 | 26 | 38 | 116 | 264 | 366 | 534 | 501 | 1,862 |
| TOTAL | 6 | 346 | 921 | 2,673 | 6,962 | 13,476 | 24,669 | 30,766 | 40,460 | 52,158 | 172,437 |

出典：
Scrum Allianceによる協力

各国の現在のScrum Master等人数 (2012年3月)

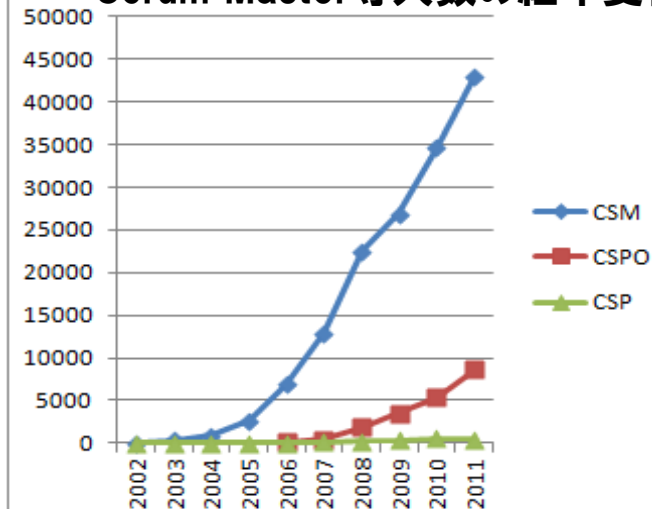
単位 (人)

| | 米国 | 英国 | 中国 | デンマーク | ブラジル | 日本 | TOTAL |
|-------|--------|--------|-------|-------|-------|-----|---------|
| CSM | 67,000 | 11,800 | 3,800 | 3,700 | 4,600 | 350 | 91,250 |
| CSPO | 8,000 | 1,800 | 400 | 750 | 900 | 120 | 11,970 |
| CSP | 1,100 | 0 | 30 | 30 | 60 | 6 | 1,226 |
| TOTAL | 76,100 | 13,600 | 4,230 | 4,480 | 5,560 | 476 | 104,446 |

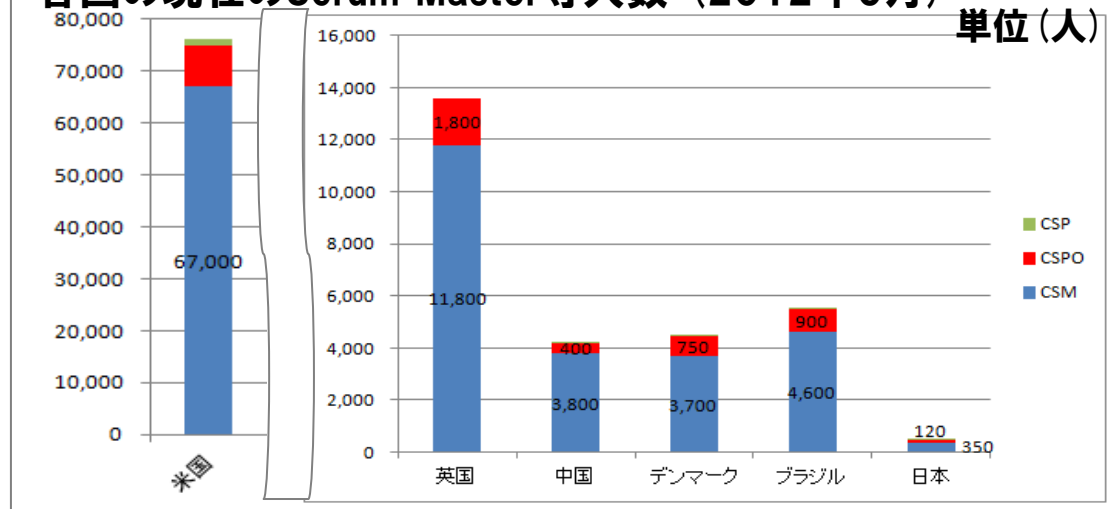
略称説明

- CSM (Certified Scrum Master)
 - チーム全体の支援者
- CSPO (Certified Scrum Product Owner)
 - 製品の責任者
- CSP (Certified Scrum Professional)
 - スクラムの実践者

Scrum Master等人数の経年変化

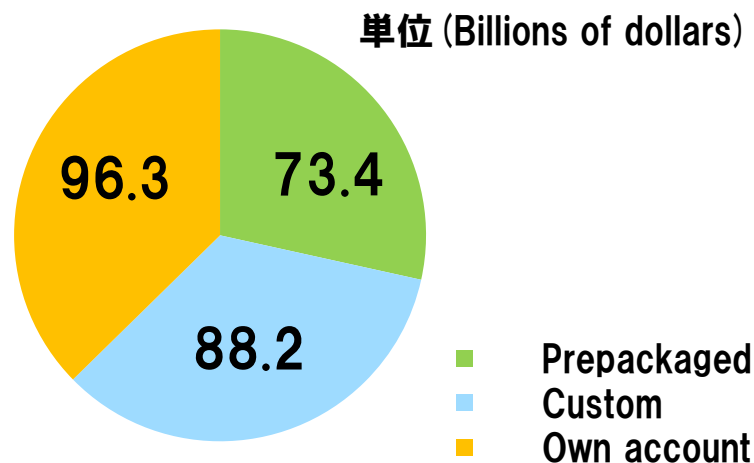


各国の現在のScrum Master等人数 (2012年3月)



ソフトウェア開発プロジェクトの比較【特徴的データ】

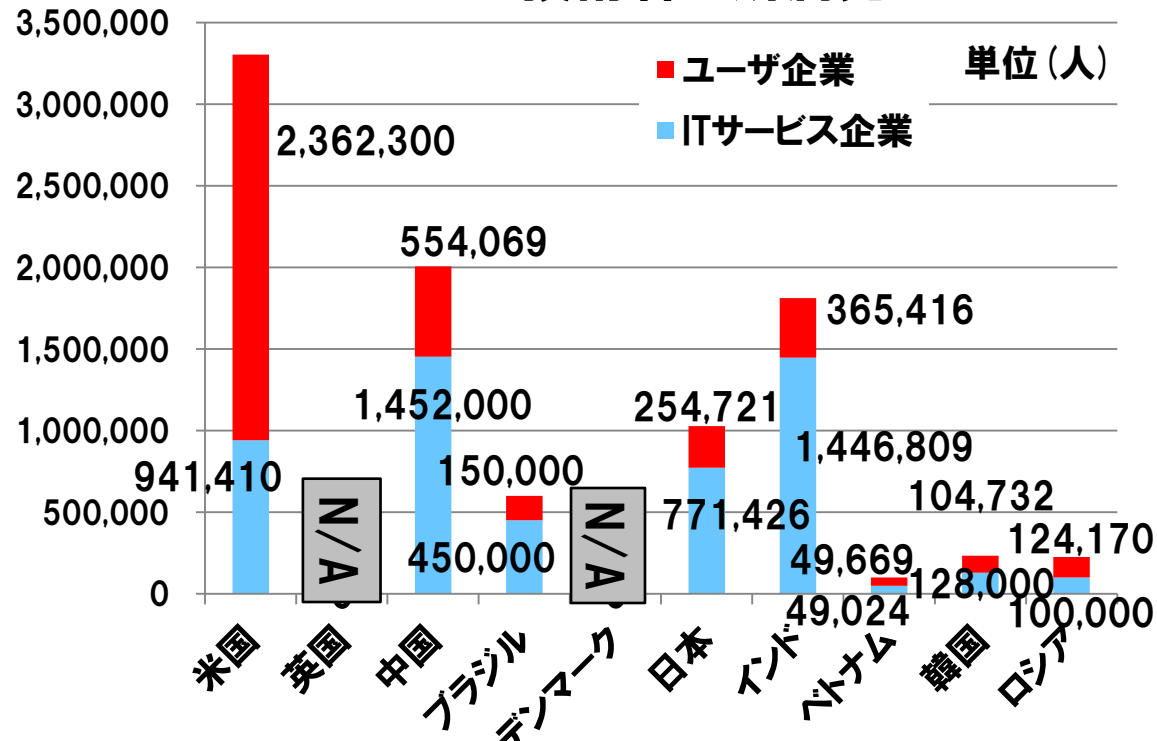
米国では、ソフトウェアに対する投資は「**自社開発(内製)**」「**市販パッケージソフトの利用**」が約2/3を占めている(下図「米国民間部門における ソフトウェア投資」参照)
さらに、他国に比べて**多くのIT技術者がユーザ企業に所属している**(下図「IT技術者の所属先」参照)
✓米国のプロジェクトの形態の特徴は、37%が**内製していること**

米国民間部門における
ソフトウェア投資

Prepackaged : パッケージソフトを購入
Custom : 外部発注作業
Own account : 自社開発ソフト

出典:「Bureau of Economic Analysis
<http://www.bea.gov/national/xls/soft-invest.xls>」

IT技術者の所属先



出典:「グローバル化を支えるIT人材確保・育成施策に関する調査」概要
報告書 2011年 3月 (IPA)

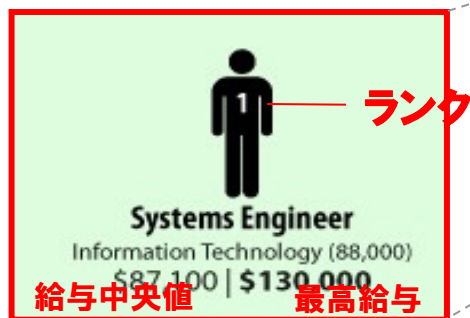
米国では**恒常的にIT関連職の給与が高く、人気も高い**

中国、ブラジルでは、IT関連職の**人材が不足しているため**、IT関連職の給与が高く、**人気が高い**

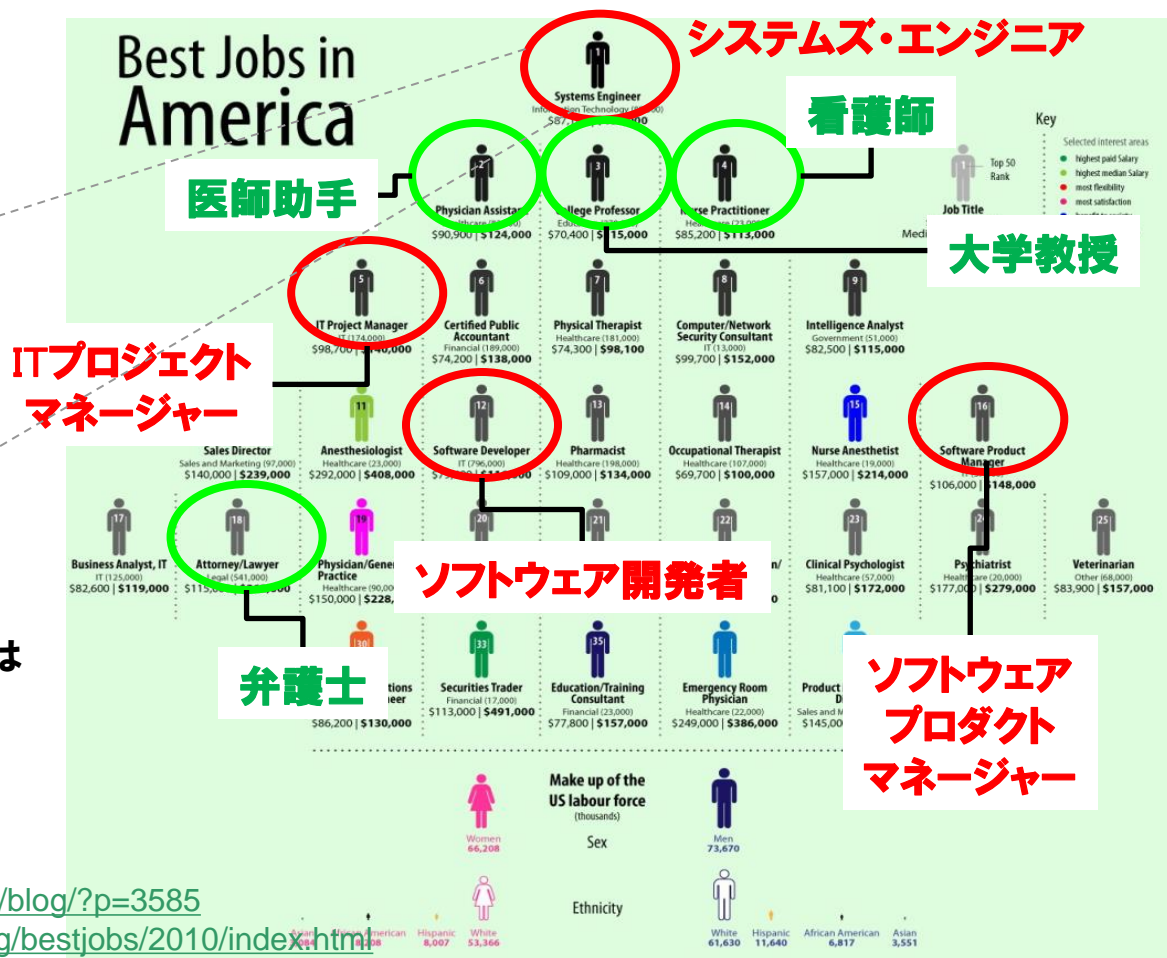
✓ 米国、中国、ブラジルでは、IT関連職が人気であるため、優秀な人材が集まる

米国ではIT関連職が人気の職業

- 2009年:
Systems Engineer が1位
- 2010年:
Software Architectが1位



システムズ・エンジニアとは、日本のSEのニュアンスとは異なる。ソフトウェアを含むシステム全体の構造を検討・定義する役割を持つ職種をさす（出典：CNN Money をもとに編集）



＜出典＞

Focus, Best Jobs in America: <http://kelsocartography.com/blog/?p=3585>

CNN Money: <http://money.cnn.com/magazines/moneymag/bestjobs/2010/index.html>

アジャイル開発には、どんな契約がふさわしいのか?

- 開発内容が決まっていない段階で、開発プロジェクト全体につき、一つの請負契約を結ぶのは適切ではない(何をいくらで完成させるか不明)。
- 他方、開発プロジェクト全体を準委任契約にすることは、ベンダが完成義務を負わない点で、ユーザ側に不安がある(たとえ成果物が完成しなくても、ユーザは対価を支払う必要)。
- また、アジャイル開発の特徴であるユーザとベンダの協働関係を、契約に取り入れる必要がある。

注) ユーザ側での労働者派遣契約に基づく開発チーム構成には、契約上の問題は特にない。
(内製傾向の高まりに伴い増加?)

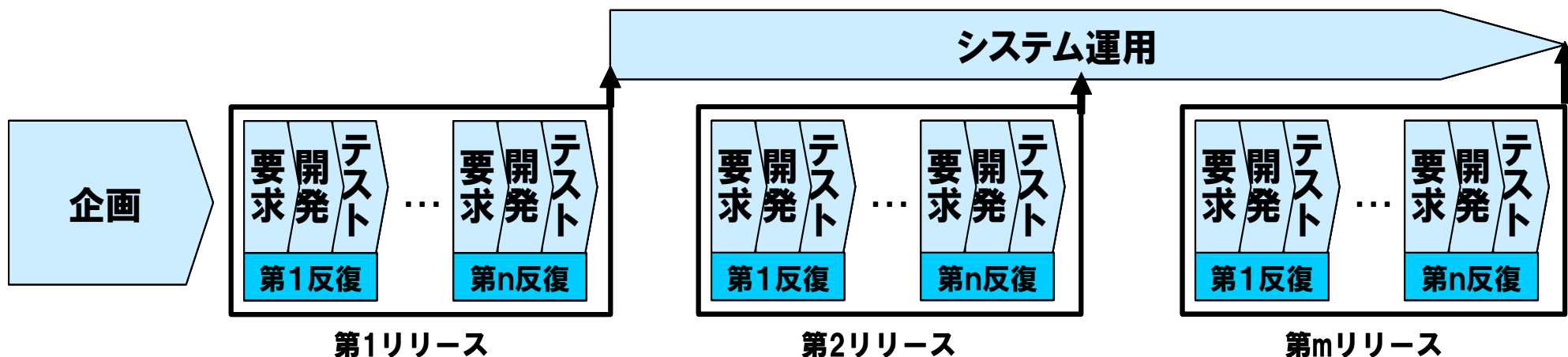
■ 基本／個別契約モデル：

プロジェクト全体に共通する事項につき、基本契約を締結し、小さな機能単位ごとに、開発対象と費用がある程度確定したタイミングで個別契約(請負／準委任)を順次締結する。

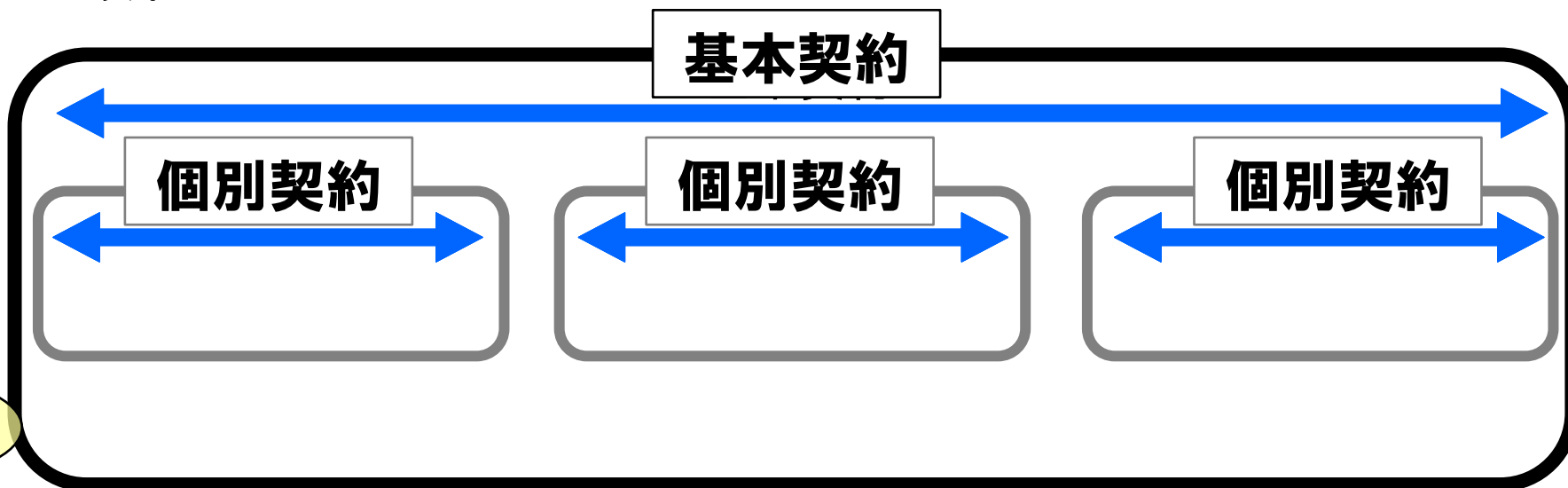
■ 組合モデル：

ユーザとベンダが共同でジョイント・ベンチャーとしての組合を組成し、協力してシステム開発(収益性のあるもの)を企画・実施する(開発された成果から得られた収益は、ベンダとユーザに分配される)。

基本/個別契約モデルの概要



- $n=1$ のケースもあり。



- 例 -

アジャイル開発が得意とし、現在、その適用により効果を挙げている領域：

①ビジネス要求が変化する領域

- ・要求の変化が激しく、あらかじめ要求が固定できない領域。

②リスクの高い領域

- ・不確実な市場を対象としたビジネス領域（市場リスク）
- ・技術的な難易度が高い開発領域（技術リスク）

③市場競争領域

- ・他社に先駆けた製品・サービス市場投入が命題であり、TTM（Time to Market）の短縮が優先となる領域（Webのサービス、パッケージ開発、新製品開発）。

アジャイル開発による経験が十分には蓄積されておらず、現在、チャレンジと創意工夫が求められている領域：

①大規模開発

- ・開発者10人程度を超えると、システム分割、チーム分割が必要。その分割方法、及び、分割されたチーム間のコミュニケーションが課題。

②分散拠点(オフショア含む)開発

- ・開発拠点が分散し、さらに時差によって分断される場合のコミュニケーション手法、また、それをサポートするツールが必要。

③組織(会社)間をまたぐ開発チームによる開発

- ・共通のビジネスゴールを持ったチームを組むことが難しい。

④組込みシステム開発

- ・リリース後のソフトウェア修正が極めて困難であり、採用には工夫要。

| No | 規模 | 部分適用 | 採用手法 | 対象システム種別 | 契約 |
|----|----|------|----------|--------------------|---------------|
| 1 | 大 | | 独自 | B2Cサービス（SNS） | 無（自社内） |
| 2 | 大 | | Scrum | B2Cサービス（ソーシャルゲーム） | 無（自社内） |
| 3 | 大 | ○ | Scrum | ゲームソフト | 受託（未公開） |
| 4 | 大 | ○ | Scrum+独自 | 基幹システム | 受託（準委任） |
| 5 | 中 | | Scrum | B2Cサービス（会員サービス） | 無（自社内） |
| 6 | 中 | | Scrum+XP | B2Cサービス（医療・健康） | 無（自社内）+オフショア* |
| 7 | 中 | | Scrum+XP | B2Cサービス（エンタテインメント） | 無（自社内）+オフショア* |
| 8 | 中 | | XP | B2Cサービス（会員サービス） | 受託（請負） |
| 9 | 中 | ○ | XP | B2Cサービス（ECサイト） | 受託（請負） |
| 10 | 中 | ○ | XP | B2Cサービス（会員サービス） | 受託（準委任） |

中規模：30～100名，大規模：100名以上

*：準委任

独自：特に手法を決めず自ら定義，Scrum+XP：両手法を組み合わせて実践

適用にあたっての主な工夫（一覧）

中・大規模開発特有の工夫

■組織体制

- チーム間ローテーション

■コミュニケーション

- 段階的朝会
- チーム跨ぎのふりかえり

■展開

- 漸進的な人数増加
- 漸進的な展開
- 社内勉強会

■分散拠点開発

- 同一拠点から分散へ
- TV会議

■アーキテクチャ

- 組織の共通基盤アーキテクチャの利用
- アーキテクチャについての教育

■システム分割/インテグレーション

- 同じリズム

■品質

- 第三者テスト

■部分適用

- 必要な部分のみ適用
- 疎結合なチーム
- 工程の見える化

小規模開発とは逆の アプローチをとる工夫

■アーキテクチャ

- 最初のアーキテクチャ構築
- アーキテクチャ専門チーム
- 運用保守チーム

■品質

- テスト・フェーズ

小規模開発と同様だが 特に注意して実施する工夫

■コミュニケーション

- 完全透明性

■展開

- パイロット導入
- 認定研修・コンサルタントの利用

■分散拠点開発

- チケットシステム
- リアルタイムチャット

■アーキテクチャ

- アーキテクチャの改善

■システム分割/インテグレーション

- 疎結合で分割
- 早期からのインテグレーション
- 継続的インテグレーション

■品質

- 重視するビジネス価値
- ビジネス価値の変化
- タイムボックス優先の品質
- 自動単体テスト

アジャイル開発プラクティス・リファレンスガイド

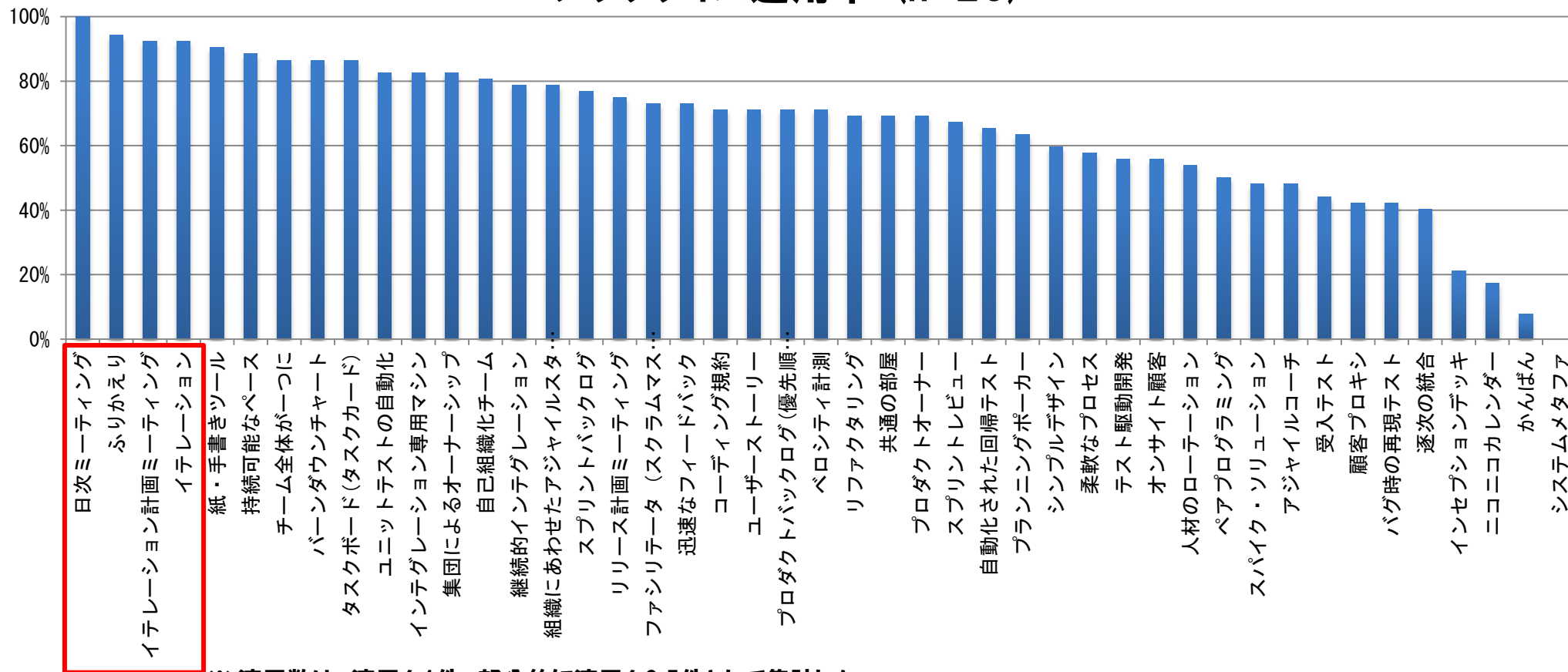
<http://www.ipa.go.jp/about/press/20130319.html>

<http://sec.ipa.go.jp/reports/20130319.html>

- 55個のプラクティス，26個の事例，9つの活用ポイント 計224ページ
- 日本国内の開発現場からのヒアリングにより収集した知見を，
パターン記述形式で取りまとめ
- MS-Wordファイルを公開し，クリエイティブ・コモンズ・ライセンスの下に，
改変自由・営利目的利用可で使用許諾

日次ミーティング、ふりかえり、イテレーション計画ミーティング、イテレーションの順に適用率が高く、これらはアジャイル開発を行う上でのほぼ必須のプラクティスであると言える。これらのプラクティスはScrumとXPに共通するプラクティスである。

プラクティス適用率 (n=26)



※:適用数は、適用を1件、部分的に適用を0.5件として集計した。

※ システムメタファは国内の26事例の中で活用されている事例はなかった。『ガイド編 プラクティス解説』では、海外の事例を調査した。

プラクティスの実態 - 日次ミーティング

状況

チームは、プロジェクトのタスクをこなすためにほとんどの時間を使い、状況や情報の共有のために取れる時間がほとんどない。

問題

情報の共有遅れが問題を大きくする。
情報共有の時間が取れないまま、状況認識と問題対処への判断が遅れると、問題が大きくなるなど、より深刻な状況を招いてしまう。

フォース

関係者が多忙なため、情報共有のための時間が取れない。
情報共有の間隔が空いてしまうと、情報量が増え、共有に必要な時間が余分にかかる。

解決策

必要な情報を短い時間で毎日共有する。
関係者が長時間、時間を取れないようであれば、短い時間（15分を目安に）で済むように、共有を必要な情報に絞る。

利用例

- 事例(9): 遠隔地にいるメンバーも日次ミーティングに参加するため、チャットツールや電話会議システムを利用した。
- 事例(17): 1日3回(朝、昼、夕)10分程度のミーティングを実施。問題を報告/解決するためのリズムが開発メンバー全員に浸透して、短期での問題提起ができています。

留意点

- 必ずしも朝の時間帯にこだわらず、関係者が集まりやすい時間帯に開催する(例えば、終業近い時間帯に開催する夕会)。

プラクティスの実態 - ふりかえり

状況

イテレーション毎に、チームは動くソフトウェアとして成果を出そうとしている。イテレーションを繰り返して、チームはソフトウェアを開発していく。

問題

開発チームは、そこに集まったメンバーにとって最適な開発プロセスを、最初から実践することはできない。

フォース

イテレーションでの開発はうまくいくこともあるが、うまくいかないこともある

解決策

反復内で実施したことを、反復の最後にチームでふりかえり、開発プロセス、コミュニケーション、その他様々な活動をよりよくする改善案をチームで考え実施する機会を設ける。

※1 メンバー全員で、Keep (よかったこと・続けたいこと)、Problem (問題・困っていること)、Try (改善したいこと・チャレンジしたいこと) を出し合い、チームの改善を促す手法。

利用例

- 事例 (25): 当初はKPT^{※1}を用いてふりかえりを行っていたが、ファシリテータの技量にふりかえりの質が依存してしまう、声の大きいメンバーに影響を受けてしまうことに気づいた。そのため、意見を集めるやり方として、555 (Triple Nickels)^{※2}を用いることにした。

留意点

- ふりかえりにチームが慣れていない場合は、進行で各人の意見をうまく引出すようにしないとうまくいかない。
- 問題点を糾弾する場にしてしまうと、改善すべき点を積極的に話し合えなくなってしまう。
- 改善案を出しても、実際に実行可能なレベルの具体的なアクションになっていないと実施されない。

※2 アクションや提案に対するアイデアを出すための手法。5人程度のグループで、各人が5分間ブレインストーミングをしてアイデアを書き出す。5分経過したら紙を隣の人にまわし、新しいアイデアを書き加える。

状況

開発を一定期間のサイクル（イテレーション）で繰り返し行っている。
プロダクトバックログの内容を、チームとプロダクトオーナーの間で合意している。

問題

リリース計画は遠い未来の目標のため、それだけではイテレーションで何をどのように開発すれば良いか分からない。

フォース

ユーザーストーリーのまま、イテレーションの詳細な計画を立て、開発を進めていくのは難しい。

解決策

イテレーションで開発するユーザーストーリーと、その完了までに必要なタスクおよびタスクの見積りを洗い出すミーティングを開く。

利用例

- G社事例 (9): ペーパープロトタイピング^{※1}を用いたUIデザインの共有と受入れ条件の確認をイテレーション計画ミーティングで行っていた。そのため、計画にはかなり時間を要していたが、見積りの前提が具体的になったため、見積り作業時間の削減に繋がった。

留意点

- 見積りに関してチームが水増しする懸念を持つかもしれないが、チームを信じるべきである。プロジェクトの目的を理解したチームは、見積りが大きく外れるようであれば、自らその原因を分析し、次の見積りに活かすはずである。

※1 紙などを使った試作品でユーザビリティテストを行うこと。

事例概要 <<中大規模適用プロジェクトの事例>> 事例(5) C社

プロフィール

既存のサービスのリプレイス開発。単純なサービスのリプレイスではなく、新しい要件も加えながら開発したいとの要望があり、C社から顧客にアジャイル開発を提案して開始した。

リプレイスといいながらも、顧客から要件を聞き出しながら開発を進めていった。要件が固められない部分のみアジャイル開発を行い、要件が明らかな部分についてはウォーターフォール型開発を実施した。

特徴的なプラクティス

- 日次ミーティング: 複数のチームが存在したため、二段階の構成で実施していた。(チーム間→チーム毎)。
- ふりかえり: チーム毎に実施した場合には、他のチームへの不満などばかりになってしまい機能しなかった。そのために、複数チームの混成で実施することにより、問題へ集中するように意識を変えさせた。また、反復毎のふりかえりとは別に、四半期単位でも実施して大きな改善点について話しあった。
- 顧客プロキシ: 当初は顧客に要件管理をしてもらっていたが、機能しなくなったため、C社の社員が顧客の会社へ出向して顧客プロキシとなり全面的に支援した。

| | |
|--------|---|
| システム種別 | B2Cサービス |
| 規模 | 中規模 開発者 32名 インフラ 4名 管理その他 23名 計 59名 |
| 手法 | XP |
| 契約 | 準委任契約(四半期毎に更新) |
| 期間 | 2年 |
| 開発拠点 | 東京、地方を含めた3拠点 |

(1) 短納期、開発期間が短い

開発対象のボリュームに比して、開発期間が短い場合、チームの開発速度を計測し、そのスピード感で、予定している開発量が期限内に完了するのか、常に点検する必要があるため、「**ベロシティ計測**」と、「**バーンダウンチャート**」を活用する。

ベロシティ計測は、関係者であるプロダクトオーナーが理解できる基準で計測する必要がある(H社事例(11))。**バーンダウンチャート**は、関係者と定期的に共有する機会を設けることが活用のポイントである(B社事例(2)、J社事例(17)(18))。

(2) スcopeの変動が激しい

開発中に要求の変更が頻繁に発生することが予想されるプロジェクトでは、チームが扱う要求の全体像と状態、直近のイテレーションで何を開発するかが分かっており、柔軟に優先順位を変えられる必要があるため、「**プロダクトバックログ(優先順位付け)**」、「**スプリントバックログ**」および「**プロダクトオーナー**」を活用する。**プロダクトバックログ(優先順位付け)**は、イテレーション毎に整理を行い、チーム全員で優先順位と内容を合意すると良い(B社事例(2))。

プロダクトオーナーは、業務や全社的に全体最適となる判断を行うこと(G社事例(10))。

(3) 求められる品質が高い

品質要求が高いプロジェクトでは、テストに関するプラクティスである「**自動化された回帰テスト**」、「**ユニットテストの自動化**」を活用する。

自動化された回帰テストや**ユニットテストの自動化**は、プロジェクトの初期段階で、実施有無、実施のための取決め、使用ツールを検討しておくことがポイントである。これを後回しにすると、必ず機能開発が優先され、自動化にたどりつかない(B社事例(2))。

(4) コスト要求が厳しい

必要のないものを作るムダをなくし、必要なものをより素早く提供することがROI(費用対効果)の向上につながり、コスト要求に応えることができる。そのためには、的確に顧客の要求を把握し、認識の相違をなくす必要があるため、「**プロダクトバックログ(優先順位付け)**」を活用する。

また、開発機能がプロダクトオーナーの意図通りになっているか否かの検証のために、「**受入テスト**」を活用する。「**オンサイト顧客**」には、優先順位や仕様の確認がその場で確認することができ、迅速に方針を決められるというメリットがある(K社事例(20))。

(5) チームメンバーのスキルが未成熟

スキルの的に未成熟なメンバーが成長していく機会として、プロジェクトを計画する必要があるため、「**ペアプログラミング**」と「**ふりかえり**」を活用する。

ペアプログラミングは、ベテランとメンバーと一緒に仕事をするすることで、技術的な指導を行うのに適したプラクティスである(C社事例(4))。

ふりかえりは、メンバーの成長の機会として捉えることができる。ふりかえりのやり方自体も見直しながらチームに適したやり方を模索すると良い(E社事例(6))。

(6) チームにとって初めての技術領域や業務知識を扱う

プロダクトの背景にある業界の知識や、要求の理解と実装に必要な業務知識の獲得が必要となるため、「**スパイク・ソリューション**」と「**システムメタファ**」を活用する。

スパイク・ソリューションを適用することは、リスクとなりそうな技術課題について、プロジェクトの初期段階で実験的に小さく試しておくことであり、チームとプロジェクトを後々助けることに繋がる(C社事例(4))。**システムメタファ**は、開発者にとって、なじみの薄い業務知識を理解する手段として、有効と考えられる。

(7) 初めてチームを組むメンバーが多い

初めてチームを組むメンバーが多い場合、チームが向かう方向を明確にすることと、チームビルディングが必要となるため、「**インセプションデッキ**」や「**ニコニコカレンダー**」を活用する。

インセプションデッキは、作成を通じて、プロジェクトの目的や目標が明らかとなる(B社事例(1))。

ニコニコカレンダーは、メンバーの感情や状況を可視化し、チームメンバーのことを知ることがポイントになる(E社事例(6))。

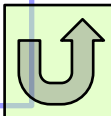
(8) オフショアなど分散開発を行う

プロダクトオーナーと開発チームが別の拠点にいる場合、オンラインでのコミュニケーション手段を検討し、頻繁にコミュニケーションが取れるようにする必要があるため、「**日次ミーティング**」や「**顧客プロキシ**」を活用する。

TV会議システムを使った**日次ミーティング**は、離れた者同士が毎日顔を合わせる機会として、ぜひ活用すべきである(G社事例(9))。**顧客プロキシ**は、分散した環境下でも、迅速なフィードバックが得られる工夫をしなければならない。

(9) 初めてアジャイル開発に取り組む

初めてアジャイル開発に取り組む際には、書籍や文書だけではなく人から人にやり方を伝えることが有効であるため、社内にアジャイル開発に取り組んだ経験のある人がいる場合はその人に、社内にはない場合は、社外から**アジャイルコーチ**を頼んで導入の手伝いをしてもらうのがよい。初めて取り組む場合は、イテレーション期間を短くした上で、**ふりかえり**の中で改善点をチームで考え実行していくことが不可欠となる。



Gemba Pattern Community

<http://gembapatterns.github.com/> (予定)

- リファレンスガイドを単に眺めるだけでなく、現場での活用を促進
- どんな場面で、どのようにガイドを使うかの「使い方」を共有
- 現場のパターンを登録・洗練・共有・利用

<企画者>

懸田剛

合同会社カルチャーワークス 共同代表／アジャイルプロセス協議会 四国支部 Agile459支部長
本橋正成

合同会社カルチャーワークス 共同代表
市谷聡啓

株式会社永和システムマネジメント サービスプロバイディング事業部 アジャイル開発グループ 主任

アジャイル開発人材の育成

アジャイル開発人材育成の取り組み例

- スクラム認定研修... スクラムアライアンス
<http://www.scrumalliance.org/courses/map?country=Japan>
- ワークショップによるアジャイルマインド習得と開発実習... 早稲田大学
http://www.washi.cs.waseda.ac.jp/agile_workshop/
- enPiT 実践的情報教育協働ネットワーク... 産業技術大学院大学, 他
<http://bizapp.enpit.jp/>
- アジャイル検定試験... アジャイルソフトウェア開発技術者検定試験準備委員会
(テクノロジックアート <http://www.tech-arts.co.jp/>)
- PMI Agile Certified Practitioner (PMI-ACP)®
<http://www.pmi.org/en/Certification/New-PMI-Agile-Certification.aspx>
- 北海道ソフトウェア技術開発機構
<http://www.deos.co.jp/hojokoza>
- 島根情報産業協会
<http://agileshimane.doorkeeper.jp/events/2324>
- 民間企業
 - 永和システムマネジメント <http://www.esm.co.jp/agile/consulting/works/tool.html>
 - オージス総研 <http://www.ogis-ri.co.jp/learning/c-01.html>
 - 豆蔵 <http://www.mamezou.com/training/agile.html>
 - NECラーニング <http://www.neclearning.jp/courseoutline/courseId/SD071.0>
 - 翔泳社 <http://event.shoeisha.jp/aa>
 - 富士ゼロックス総合教育研究所 <http://www.fxli.co.jp/seminar/pm/>

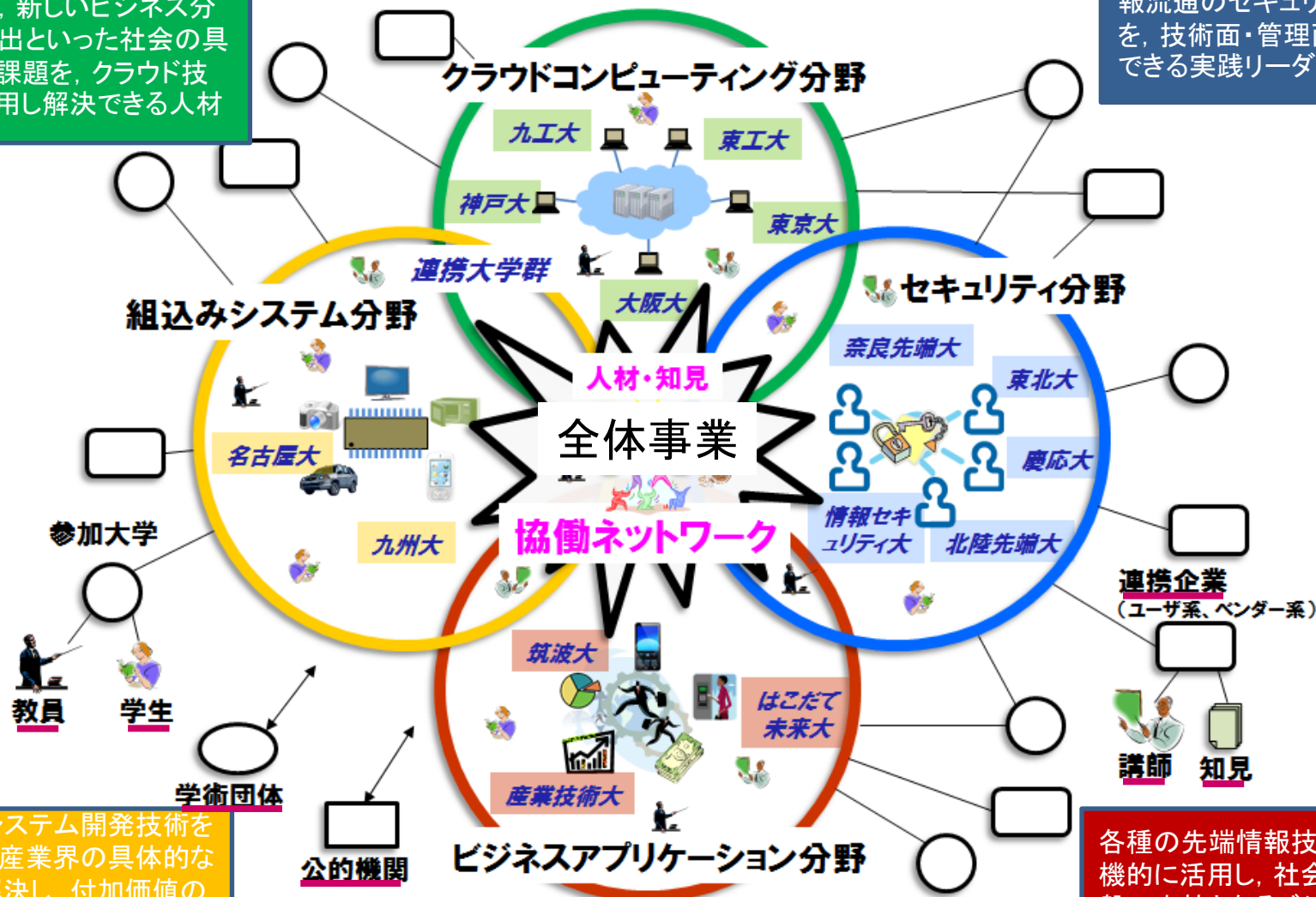
アジャイル開発人材育成の取り組み例

- **スクラム認定研修... スクラムアライアンス**
<http://www.scrumalliance.org/courses/map?country=Japan>
- **ワークショップによるアジャイルマインド習得と開発実習... 早稲田大学**
http://www.washi.cs.waseda.ac.jp/agile_workshop/
- **enPiT 実践的情報教育協働ネットワーク... 産業技術大学院大学, 他**  **本日紹介**
<http://bizapp.enpit.jp/>
- **アジャイル検定試験... アジャイルソフトウェア開発技術者検定試験準備委員会**
(テクノロジックアート <http://www.tech-arts.co.jp/>)
- **PMI Agile Certified Practitioner (PMI-ACP)®**
<http://www.pmi.org/en/Certification/New-PMI-Agile-Certification.aspx>
- **【SEC特別セミナー】アジャイル開発の人材で紹介(2013年3月18日)**
～アジャイル開発の技術者人材像・認定制度と大学・企業での取り組みについて～
※資料公開中!!! <http://sec.ipa.go.jp/seminar/2013/20130318-02.html>
- **民間企業**
 - **永和システムマネジメント** <http://www.esm.co.jp/agile/consulting/works/tool.html>
 - **オージス総研** <http://www.ogis-ri.co.jp/learning/c-01.html>
 - **豆蔵** <http://www.mamezou.com/training/agile.html>
 - **NECラーニング** <http://www.neclearning.jp/courseoutline/courseld/SD071.0>
 - **翔泳社** <http://event.shoeisha.jp/aa>
 - **富士ゼロックス総合教育研究所** <http://www.fxli.co.jp/seminar/pm/>

enPiT全体像

いわゆるビッグデータの分析手法、新しいビジネス分野の創出といった社会の具体的な課題を、クラウド技術を活用し解決できる人材

社会・経済活動の根幹にかかわる情報資産および情報流通のセキュリティ対策を、技術面・管理面で牽引できる実践リーダー



組み込みシステム開発技術を活用して産業界の具体的な課題を解決し、付加価値の高いサイバー・フィジカル・システムズを構築できる人材
Agile Japan 2013 (2013-05-24)

Copyright © 2009-2013 IPA, All Rights Reserved.

各種の先端情報技術を有機的に活用し、社会情報基盤の中核となるビジネスアプリケーション分野の実践的問題解決ができる人材

産業技術大学院大学(AIIT)

情報アーキテクチャ専攻

創造技術専攻

情報システムに
関わる全ての
社会人のための
カリキュラム

Agile
Big Data
Cloud

日本が誇る
ものづくり
プロセスを
体系的に学修

Industrial
Design
3D Printer
MAKERS

品川シーサイド徒
歩3分

東京都が設置
した大学院

社会人の学びを全
力支援

エピローグ:まとめと今後

ウォーターフォール型とアジャイル型との手法の違い

ウォーターフォール型

(開発が)

失敗しないための手法

「プロセス」重視

“計画”駆動型

作るものも使用する技術も明確
例) ビルや橋の建設

最初から綿密な計画を立て
計画に従って着実に進める。

“文化”
が異なる

ケース
バイ
ケース
で
使い分け

アジャイル開発

(ビジネスが)

成功するための手法

「人」重視

(顧客) “価値”駆動型

計画時には、ビジネス上、システム上の課題が未解決、開始後も変更の可能性大

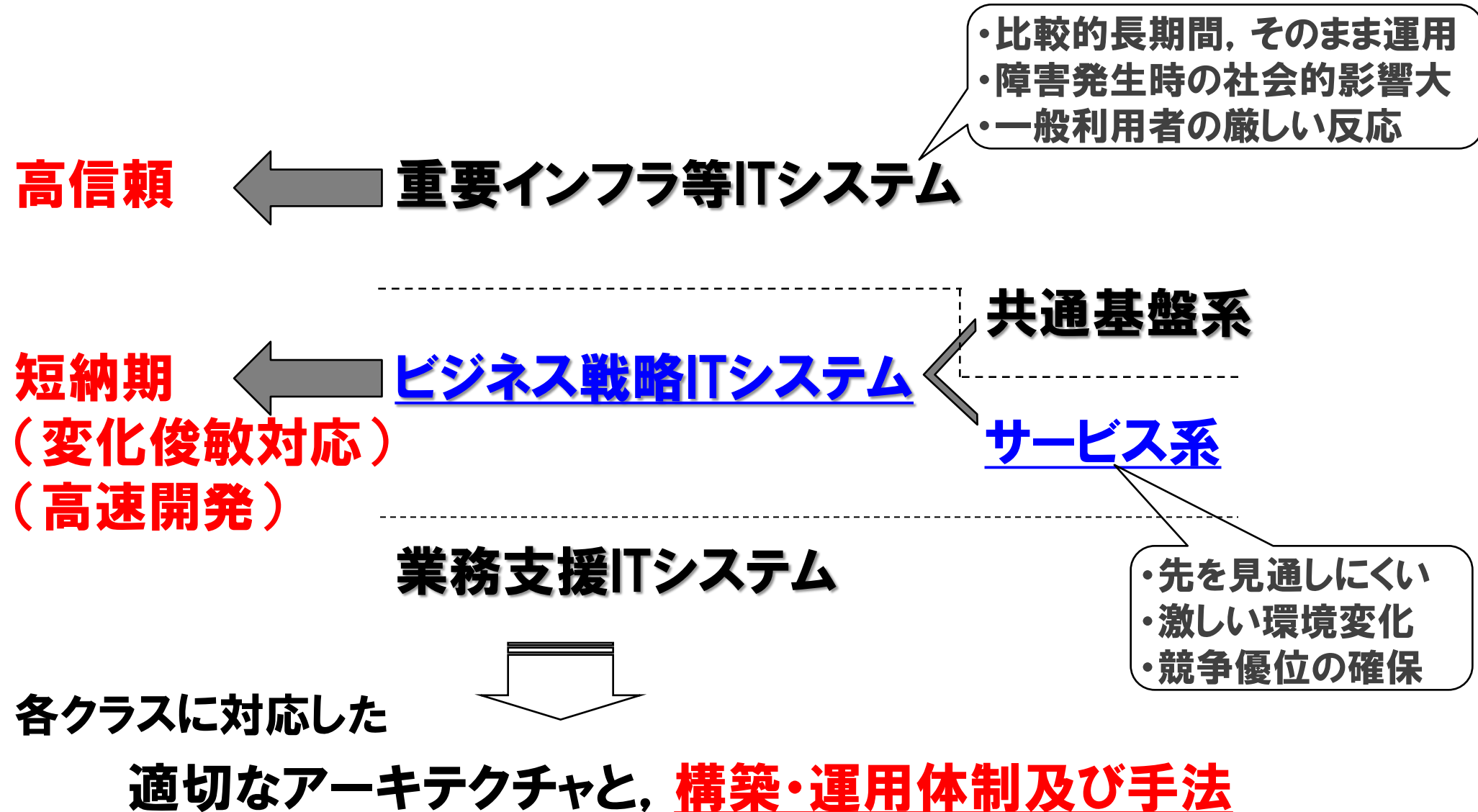
少し試して、その結果に基づいて
次のステップを進める。

多くの組織、チーム、個人にとって、アジャイル開発プロセスへの転換は“**挑戦的**”である。
それは、ある種の**文化的変革**を必要とするからだ。[[Agile transformation](http://www.ibm.com/smarterplanet/us/en/business_analytics/article/agiledevelopment.html), IBM]

http://www.ibm.com/smarterplanet/us/en/business_analytics/article/agiledevelopment.html

アジャイルは、プロセスではなく文化である。

Michael Sahota: “An Agile Adoption and Transformation Survival Guide: Working with Organizational Culture,” 2012.



アジャイル開発は、
環境変化への対応という点で、
もともとスピード感が求められていた
Web (BtoC) サービス系を中心に普及してきた。

環境の変化が激しさを増す中で、今後は、
基幹系 (BtoB等)の一部(小規模システム等)にも、
他の手法と併せて適用が進んでいく。

ただし、成功しその効果を十分高めるためには、
さまざまな工夫と試行錯誤が必要である。
これは、経営層と技術者層の両者に対して言え、
まず、考え方の変革が求められる。

「アーキテクチャ」
の検討も重要
ALMツールの
活用も必要か？

ご清聴, ありがとうございます ございました

<参考>

非ウォーターフォール型(アジャイル)開発の動向と課題,
SEC journal, Vol.8, No.4, Dec. 2012.

IPA/SECホームページ:
<http://sec.ipa.go.jp/index.html>



アジャイルプラクティス・リファレンスガイド 事例一覧 (1)

| 調査先 | No. | 採用手法 ^{※1} | 特徴 | システム種別 | 契約関係 ^{※2} | 開発言語 |
|-----|-----|--------------------|-----|---------------------|--------------------|-----------------------|
| A社 | 0 | Scrum+XP | | B2Cサービス（広告配信） | 自社開発 | Java, PHP, Perl |
| | 1 | Scrum+XP | | B2Cサービス（広告配信） | 自社開発 | Ruby |
| B社 | 2 | Scrum+XP | | B2Cサービス（SNS） | 自社開発 | Java |
| | 3 | Scrum+XP | | B2Cサービス（メール配信） | 自社開発 | Java |
| C社 | 4 | XP+WF | 中規模 | B2Cサービス（メール配信） | 受託開発（準委任） | Java |
| D社 | 5 | XP | | B2Cサービス（SNS） | 自社開発 | Java, PHP, Ruby |
| E社 | 6 | Scrum | 初導入 | 社内システム | 自社開発 | C# |
| | 7 | Scrum+WF | 中規模 | 社内システム | 受託開発（請負） | Java, COBOL |
| F社 | 8 | Scrum+WF | 中規模 | 社内システム | 自社開発 | C# |
| G社 | 9 | Scrum+XP | 初導入 | 社内システム | 実証事業 | Ruby |
| | 10 | Scrum+XP | | 社内システム | 受託開発（請負） | Ruby |
| H社 | 11 | Scrum | | B2Cサービス（音楽配信） | 自社開発 + オフショア（準委任） | Java, C#, Objective-C |
| | 12 | Scrum | | B2Cサービス（エンターテインメント） | 自社開発 + オフショア（準委任） | Java, C#, Objective-C |
| | 13 | Scrum | | 社内システム | 自社開発 + オフショア（準委任） | Java |
| | 14 | Scrum | | B2Cサービス（ヘルスケア） | 自社開発 + オフショア（準委任） | C# |

アジャイルプラクティス・リファレンスガイド 事例一覧 (2)

| 調査先 | No. | 採用手法 ^{※1} | 特徴 | システム種別 | 契約関係 ^{※2} | 開発言語 |
|-----|-----|--------------------|-----------|---------------------|--------------------|-------------------|
| I社 | 15 | Scrum | 中規模(組織展開) | B2Cサービス(広告配信) | 自社開発 | Java, Objective-C |
| J社 | 16 | XP | | B2Cサービス(スマートフォンアプリ) | 受託開発(請負) | Java |
| | 17 | XP | | B2Cサービス(クラウド基盤) | 受託開発(請負) | Java |
| | 18 | XP | | B2Cサービス(クラウド基盤) | 受託開発(請負) | Java |
| | 19 | XP | | B2Cサービス(PaaS) | 受託開発(請負) | Java |
| K社 | 20 | Scrum | | B2Cサービス(ECサイト) | 受託開発(請負) | PHP |
| L社 | 21 | Scrum+UP | | 社内システム | 受託開発(請負) | Java |
| | 22 | Scrum+WF | 大規模 | 社内システム | 受託開発(準委任) | Java |
| | 23 | Scrum+WF | | 技術評価 | 受託開発(請負) | Java |
| | 24 | Scrum | | パッケージ | 自社開発 + オフショア(請負) | C# |
| M社 | 25 | Scrum | 大規模(組織展開) | B2Cサービス(ソーシャルゲーム) | 自社開発 | Perl |

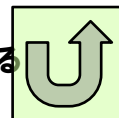
中大規模(30名以上):**6件**

初導入:**2件**

全26事例

※1:XP:エクストリームプログラミング、Scrum:スクラム、
WF:ウォーターフォール、UP:統一プロセス、
もしくは、これらの手法の組み合わせ

※2:自社開発 → 自社組織内に開発部隊あり、一部パートナー(派遣)
受託開発 → 自社組織内に開発部隊なし、外部ベンダに発注している



アジャイルプラクティス・リファレンスガイド プラクティス一覧 (1)

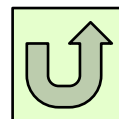
| カテゴリ | サブカテゴリ | プラクティス | 説明 |
|------------|---------|--------------------|--|
| プロセス・プロダクト | プロセス | リリース計画ミーティング | プロダクトリリースのためのリリース計画ミーティング |
| | | イテレーション計画ミーティング | イテレーション(スプリント)ごとのリリース計画やアクティビティなどを計画するミーティング |
| | | イテレーション | ゴールや結果にアプローチするプロセスを繰り返すこと |
| | | プランニングポーカー | スプリント計画時のタスクを見積もるためのプランニングポーカー |
| | | ベロシティ計測 | プロジェクトベロシティの計測 |
| | | 日次ミーティング | 現在の問題を解決するための短いデイリーミーティング |
| | | ふりかえり | 前のスプリント(イテレーション)から学ぶためにふりかえる |
| | | かんばん | ジャストインタイムの継続的なデリバリを強調した管理手法 |
| | | スプリントレビュー | 完了した仕事を表明するスプリントレビューミーティング |
| | | タスクボード(タスクカード) | ボードに貼られたメンバーが継続的に更新するタスク |
| | | バーンダウンチャート | スプリント進捗をモニターするためのバーンダウンチャート |
| | | 柔軟なプロセス | 状況や環境の変化に対応できる柔軟なプロセスにしている、もしくは、プロセスを柔軟に変更している |
| | プロダクト | ユーザーストーリー | 要求についての会話を行うときの開発チームとプロダクトオーナーの間の合意事項 |
| | | スプリントバックログ | プロダクトオーナーとチーム間でのスプリントバックログへの相互コミットメント |
| | | インセプションデッキ | 10の質問によりプロジェクトの属性を明らかにする |
| | | プロダクトバックログ(優先順位付け) | プロダクトオーナーによる優先順位(プロダクトバックログ)の管理 |
| | フィードバック | 迅速なフィードバック | 迅速なフィードバックを得られるような取組みを行っている |

アジャイルプラクティス・リファレンスガイド プラクティス一覧 (2)

| カテゴリ | サブカテゴリ | プラクティス | 説明 |
|--------|--------|--------------|---|
| 技術・ツール | 設計開発 | ペアプログラミング | すべての製品コードはペアプログラミングで開発している |
| | | 自動化された回帰テスト | 自動化された回帰テストを行っている |
| | | テスト駆動開発 | 単体テストを書き、そのテストを通るようなコードを実装する |
| | | ユニットテストの自動化 | ユニットテストの自動化 |
| | | 受入テスト | 受入テストの実施と、その結果を公開している |
| | | システムメタファ | 関係者全員が、そのシステムがどのように動くかについて伝えることができるストーリー |
| | | スパイク・ソリューション | リスクを軽減するために、隠れた問題を探索するための簡単なプログラム(スパイク・ソリューション)の試作 |
| | | リファクタリング | 定常的なリファクタリング |
| | | シンプルデザイン | 設計をシンプルに保つ |
| | | 逐次の統合 | 一度に統合するコードはひとつだけとする |
| | | 継続的インテグレーション | 継続的インテグレーション、または頻繁なインテグレーション |
| | | 集団によるオーナーシップ | 全員がすべてのコードに対して責任を持つ |
| | | コーディング規約 | 同意された標準のためのコーディング規約 |
| | 障害対応 | バグ時の再現テスト | バグが見つかったとき、そのテストがまず最初に作られる |
| | 利用ツール | 紙・手書きツール | ポストイット(付箋紙)やCRC(class-responsibility-collaboration)カードなどの使用 |

アジャイルプラクティス・リファレンスガイド プラクティス一覧 (3)

| カテゴリ | サブカテゴリ | プラクティス | 説明 |
|------------------------|----------------|-------------------|------------------------------------|
| チーム運営・ 組織・チーム 環境 | 人 | 顧客プロキシ | 要件や仕様をまとめるために顧客の業務に精通した顧客プロキシの設置 |
| | | オンサイト顧客 | 顧客といつでも／定期的にやりとりが可能である |
| | | プロダクトオーナー | プロダクトオーナー役の設置 |
| | | ファシリテータ(スクラムマスター) | スクラムマスターによる開発プロセスとプラクティスのファシリテーション |
| | | アジャイルコーチ | アジャイルコーチがプロジェクトに参加している |
| | | 自己組織化チーム | チームメンバーがタスクに志願するなど自律的なチームになっている |
| | | ニコニコカレンダー | ニコニコカレンダーを用いてメンバーの気持ちを見える化している |
| | 進め方 | 持続可能なペース | 継続的なペースで開発している |
| | 組織導入 | 組織にあわせたアジャイルスタイル | 組織にあった適切なアジャイルスタイルを用いるようにしている |
| | ファシリティ・ワークスペース | 共通の部屋 | オープンスペースがチームに与えられている |
| | | チーム全体が一つに | チーム全員がひとつのゴールに向かうような取組みを行っている |
| | | 人材のローテーション | 多能工の育成などのため人材のローテーションを行っている |
| | | インテグレーション専用マシン | 特定のインテグレーション用コンピュータ |



大手IT企業のアジャイル開発に関する取組み

| 企業 | 内 容 |
|--------|--|
| NTTデータ | 若手リーダー対象の研修開始。3年間で計1,000人の人材を育成する計画。 |
| NEC | 大規模案件に適用できる技術者を2012年度に約30人育成する計画。 「品質会計」と組み合わせたアジャイル開発ガイドを作成。 |
| 日立製作所 | 大規模案件に適用するための枠組みや実際の手順等をまとめたガイドラインを策定。 |
| 富士通 | 開発現場で蓄積したノウハウを適用するための開発標準を策定。 |
| 日本IBM | 12年度から顧客に向けアジャイル開発手法に基づくシステム開発の提案を強化。同手法を取り込んだ自社の開発標準を活用。 |

ねらい

- ◆主に大手顧客を対象に、「新規ビジネスを支えるシステムを早期に立ち上げたい」といったニーズに対応
- ◆アジャイル手法をウォーターフォール型開発手法と併用することにより、「仕様変更でプロジェクトが遅延しやすい」といった弱点をカバー

出典:大手IT企業が「アジャイル開発」を強化, 日経コンピュータ, 2012年5月24日号.

SECセミナー情報(例)

日本におけるアジャイル開発に適した契約モデル案と事例

2012年5月23日(水) <http://sec.ipa.go.jp/seminar/2012/20120523.html>

- 顧客との緊密な協働によるアジャイル型開発の成功により高い信頼を得た事例
株式会社プロビスモ 女鹿田 晃和 氏

適用が進み始めたアジャイル開発 ～ベンダー各社における取組みを中心に～

2012年10月24日(水) <http://sec.ipa.go.jp/seminar/2012/20121024.html>

- (1) NTTデータにおけるアジャイル開発の取組み …戸村 元久 氏
- (2) NECにおける非ウォーターフォール開発に対する取組み …岩崎 新一 氏
- (3) 日立製作所におけるアジャイル開発の取組み …向坂 太郎 氏
- (4) 富士通におけるアジャイル開発の取組み …宇野 和義 氏, 和田 憲明 氏
- (5) 住友電工におけるアジャイル(スクラム)の試行と評価 …中村 伸裕 氏

アジャイル開発の先進事例に学ぶ

2012年12月12日(水) <http://sec.ipa.go.jp/seminar/2012/20121212.html>

- (1) 楽天における開発手法改善の試み: 工程改善、アジャイル導入、そしてグローバル化
楽天株式会社 森 正弥 氏
- (2) スクラム拡張事例紹介 ～オフショアを含む複数チームへの適用～
株式会社エムティーアイ 岩崎 奈緒己 氏
- (3) アジャイル開発でよみがえった、プロダクト開発
アクセラテクノロジー株式会社 萩原 純一 氏

<http://sec.ipa.go.jp/seminar/index.html>