IPA Better Life with IT

Information-technology Promotion Agency, Japan

Software Reliability Enhancement Center

アジャイル開発の現状と課題

～ アジャイル開発への関心の高まりに応えるために ～

アジャイルジャパン2014
北陸サテライトイベント@金沢
2014年7月5日

独立行政法人情報処理推進機構 (IPA)
技術本部 ソフトウェア高信頼化センター (SEC)
山　下　博　之

http://www.ipa.go.jp/sec/index.html

プロローグ：ビジネスを取り巻く状況とIT

ビジネスの3要素

SEC
Software Reliability Enhancement Center

5

ビジネスの3要素　ヒト

モノ

カネ

4要素

5要素

参考

システム開発におけるQCDの優先順位

SEC Software Reliability Enhancement Center

システム企画工程におけるQCDの優先順位

- 品質：29 (28,29) %
- コスト: 23 (23,24) %
- **納期：48 (49,47) %**

調査で収集した1021 (918,801) プロジェクトのうち,「QCDのうちのどれかを優先した」という回答 (446 (377,313) プロジェクト) の内訳 [() 内は前年度,前々年度の結果]

<出典> ソフトウェアメトリックス調査2014 (2013,2012), 一般社団法人 日本情報システム・ユーザー協会 (JUAS).

そうした事業環境の中，いわゆる「QCD」のうち，特に納期を重視してものづくりを進めている．品質の確保は当たり前．開発・製造期間を短縮して製品の投入スピードをいかに速くできるかが，世界を相手に競争優位を築くカギになる．

<出典> CIOの哲学：三菱重工業 児玉敏雄氏，日経コンピュータ，2012.10.25.

開発(構築)手法の選択

SEC
Software Reliability Enhancement Center

環境の変化に対する俊敏な開発(構築)が求められる場合

俊敏な開発(構築)手法

a. 非ウォーターフォール型開発(アジャイル開発) **少しずつ作って，確かめながら**

b. クラウドコンピューティング **作らないで，使う**

「超高速開発」

c. 自動コード生成／ビジネスルールマネジメントシステム(BRMS) **パラメータを変更するだけ**

1つのシステム全体を単一の手法で開発(構築)することが適切ではない(かもしれない)

異なる手法で開発した部品の組合せ？

参考

各種開発手法の比較

SEC Software Reliability Enhancement Center

図表9-2a 各種開発法

<出典> ソフトウェアメトリックス調査2014，一般社団法人 日本情報システム・ユーザー協会（JUAS）.

ITシステムのクラスと特徴

SEC
Software Reliability Enhancement Center

・比較的長期間，そのまま運用
・障害発生時の社会的影響大
・一般利用者の厳しい反応

高信頼 ← 重要インフラ等ITシステム

本日の主対象

共通基盤系

短納期
（変化俊敏対応）
（高速開発）

ビジネス戦略ITシステム

サービス系

・先を見通しにくい
・激しい環境変化
・競争優位の確保

業務支援ITシステム

各クラスに対応した

適切なアーキテクチャと，構築・運用体制及び手法

参考

システム構築時の重視事項

SEC Software Reliability Enhancement Center

図表8-3-1 システム構築時の重視事項（1位、2位の合計%）

	基幹系	業務支援情報系	Web・フロント系	管理業務系
データ数	989	966	963	974
品質	76.8	59.2	59.3	76.9
コスト	41.2	54.8	53.1	50.2
開発スピード	14.3	35.9	43.5	12.3
変更容易性	27.7	33.1	32.4	23.6
継承性	34.9	14.5	7.3	33.0

・品質重視、継承性の基幹系、管理業務系
・開発スピード重視の業務支援、Webフロント系
・全システムともコストは一様に重視
　などの特徴があれている。
・期待に応えるアクションが必要

IT動向調査2014 by JUAS

品質，継承性重視

開発スピード，変更容易性重視

重要インフラ等ITシステム

ビジネス戦略ITシステム

環境変化への対応体制 – ITシステム構築・運用

SEC
Software Reliability Enhancement Center

ユーザ企業

情報システム部門任せ → （コスト・時間をかけて）信頼性重視で構築

市場の監視 → 業務部門主導 → IT投資判断 IT予算執行

経営の「柔軟性」マインドの転換

ユーザ企業には，
ビジネス・イノベーションを
実現し，競争優位を維持し
つつ持続するために，
変化が求められている．

顧客・経営層は開発への一層の関与が必要

SEC
Software Reliability Enhancement Center

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

開発手法の話に留まらない
企画も一選択肢に

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．

俊敏な開発（構築）手法：アジャイル開発への注目

SEC Software Reliability Enhancement Center

俊敏な開発（構築）手法

a. 非ウォーターフォール型開発（アジャイル開発）

b. クラウドコンピューティング

c. 自動コード生成／ビジネスルールマネジメントシステム（BRMS）

最近の傾向：アジャイル開発への注目度がより高まってきている背景

1. ビジネスの俊敏さへの対応要求の増大
2. グローバル化の拡大
3. ウォーターフォール型開発に適合しにくいケースの増大

「しゃべってコンシェル」

会社に足りないとされるのが経営のスピード感。「まずは七分でよし。利用者のお叱りを受けながら100%に磨き上げていく」（加藤薫・NTTドコモ社長）

2012.7.21 朝日新聞朝刊

従来：100%にしてから世に出す
現在：エンドユーザとの共同作業により，よいものにしていく（β版文化）

2012.7.23 総務省・谷脇康彦氏講演

ベンダ経営層は開発への一層の関与が必要

SEC
Software Reliability Enhancement Center

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

アジャイルは世界の主流

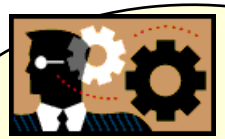
ベンダ経営層

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．

システム開発ベンダにも求められる「柔軟性」

SEC
Software Reliability Enhancement Center

人材のクラウド




企画 | 要求 | 設計 | 製造 | 試験 | 運用

親会社

多様性

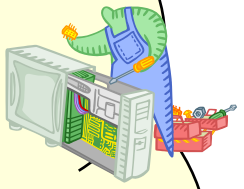
子会社　子会社

孫会社　孫会社　孫会社

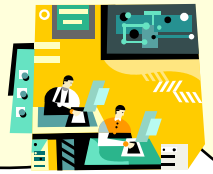
協力会社

協力会社（子会社）

連携会社





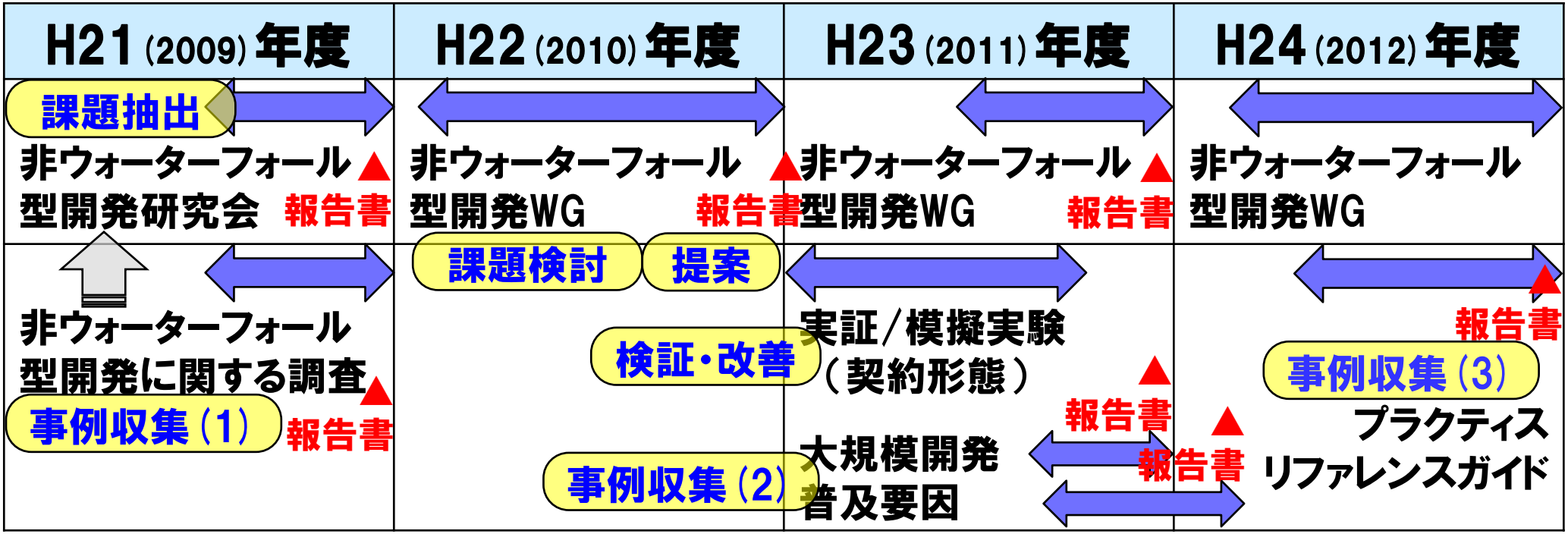

ウォーターフォール型

アジャイル型（非ウォーターフォール型）

個人に求められるスキル：
“マルチタレント”
（IT人材白書，他）

アジャイル開発に関するIPA/SECの取組み

SEC Software Reliability Enhancement Center



報告書（公開中）

H21年度版	***http://www.ipa.go.jp/sec/softwareengineering/reports/20100330a.html***
H22年度版	***http://www.ipa.go.jp/sec/softwareengineering/reports/20110407.html***
H23年度版	***http://www.ipa.go.jp/sec/softwareengineering/reports/20120326.html***
（大規模開発）	***http://www.ipa.go.jp/about/press/20120328.html***
（普及要因）	***http://www.ipa.go.jp/sec/softwareengineering/reports/20120611.html***
（プラクティス）	***http://www.ipa.go.jp/about/press/20130319.html***

本日の講演内容

SEC
Software Reliability
Enhancement Center

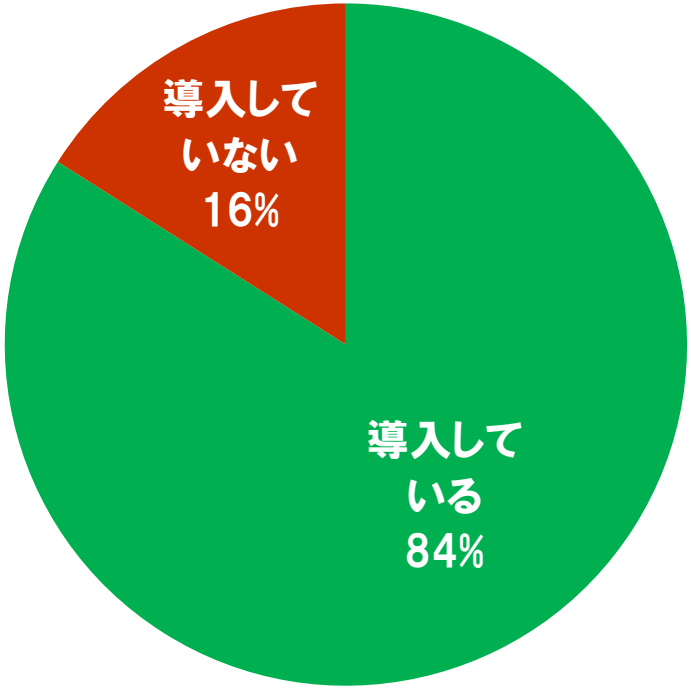
1. アジャイル開発の導入状況
2. アジャイル開発の現状
3. アジャイル開発プラクティス活用リファレンスガイド
4. アジャイル開発人材
5. アジャイル開発の課題

アジャイル開発の導入状況

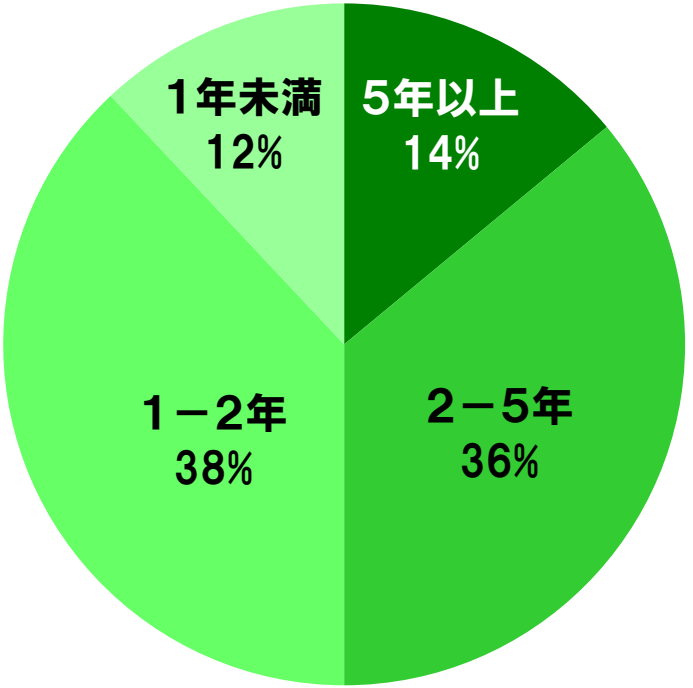
アジャイル開発の導入状況-海外-（2013年）

SEC
Software Reliability Enhancement Center

海外企業におけるアジャイル開発の導入状況



（アジャイル開発を導入している企業のうち）アジャイル開発を行っている期間

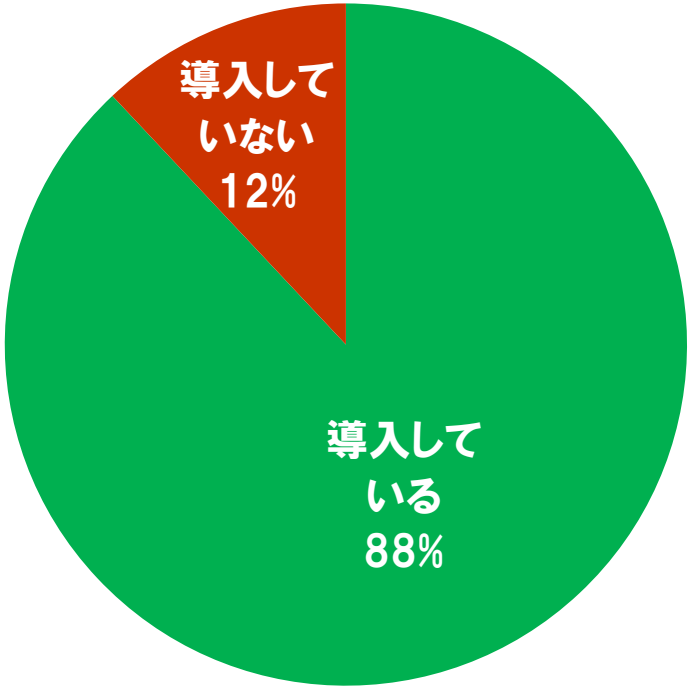


VERSIONONE®: 7th ANNUAL STATE of AGILE DEVELOPMENT SURVEY, 2013
http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf

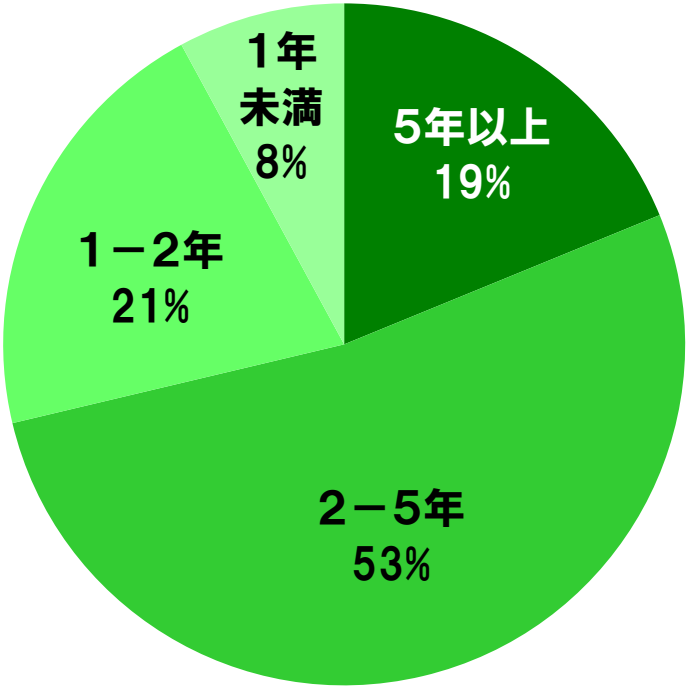
アジャイル開発の導入状況-海外-（2014年）

SEC
Software Reliability Enhancement Center

海外企業におけるアジャイル開発の導入状況



（アジャイル開発を導入している企業のうち）アジャイル開発を行っている期間



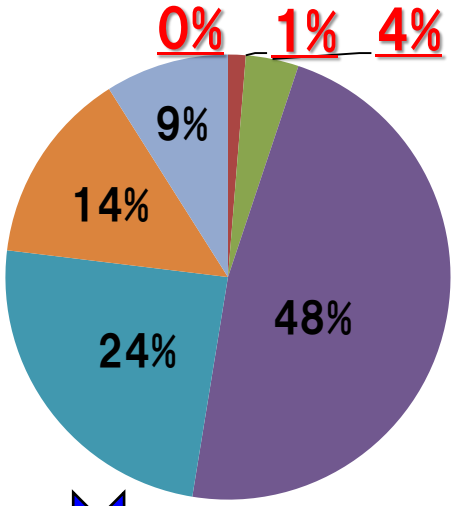
海外でのアジャイル開発の導入は拡大傾向

VERSIONONE®: 8th ANNUAL STATE of AGILE DEVELOPMENT SURVEY, 2014
http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf

アジャイル開発の適用状況-国内-(１)

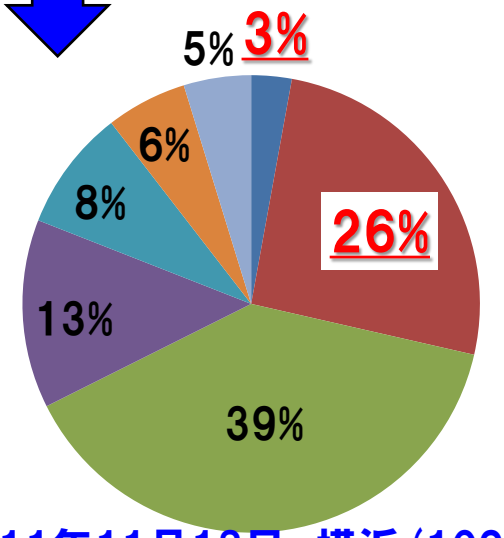
SEC Software Reliability Enhancement Center

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～

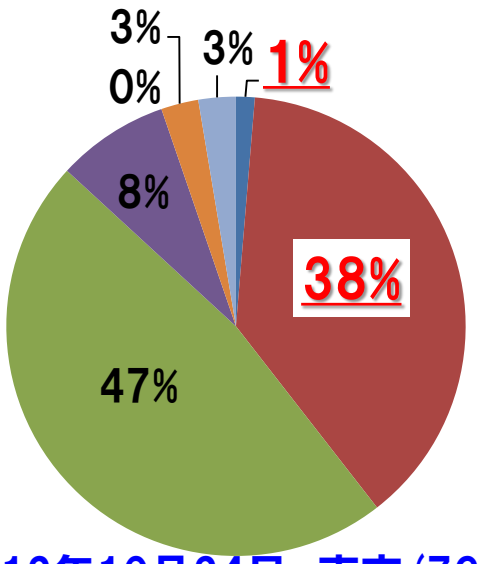


- すべてのプロジェクトに適用している
- ほとんどのプロジェクトに適用している
- だいたいのプロジェクトに適用にしている
- ほとんどのプロジェクトに適用していない
- すべてのプロジェクトで適用していない
- 分からない
- 無回答

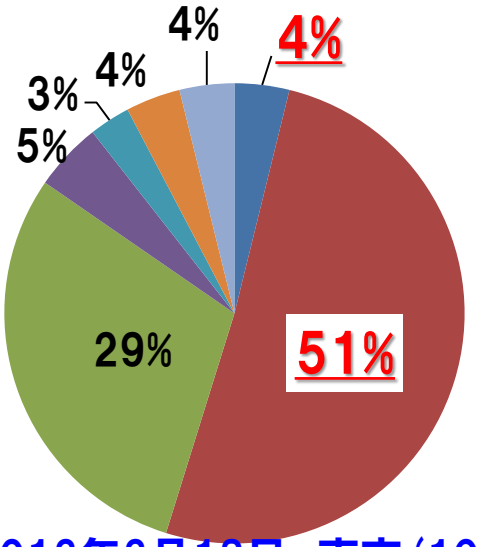
- 多くのプロジェクトで使っている
- 一部のプロジェクトで使っている
- 使いたいと考えているが実現していない
- 使う予定はない
- よく分からない／関係ない
- その他
- 無回答



2011年11月18日　横浜(109名)



2012年10月24日　東京(76名)

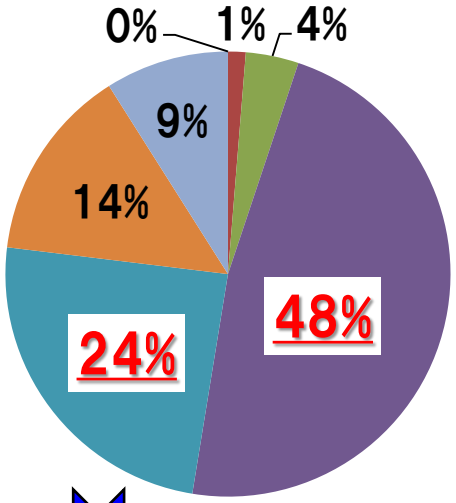


2013年3月18日　東京(104名)

アジャイル開発の適用状況-国内-(２)

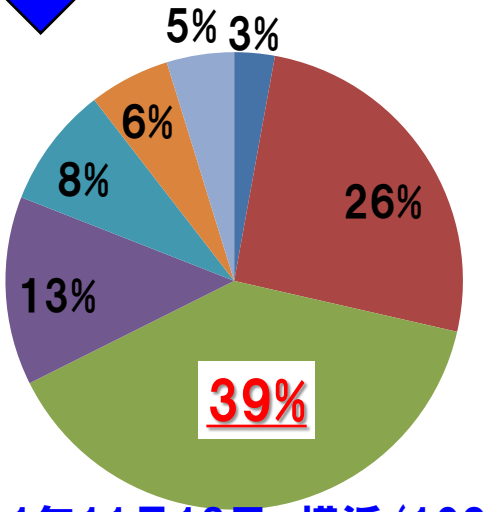
SEC Software Reliability Enhancement Center

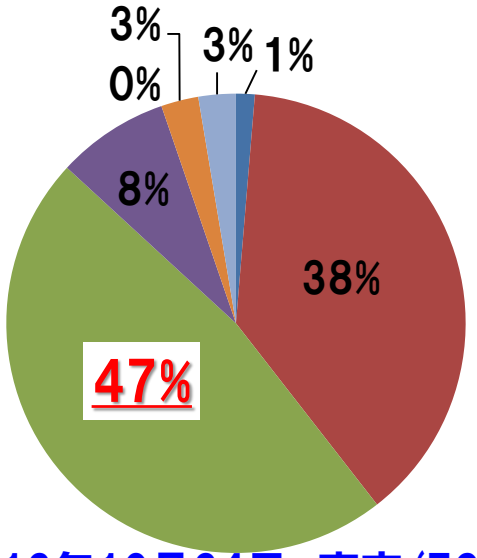
2010年12月2日　横浜(72名)　～IPA/SECセミナー聴講者アンケート結果から～

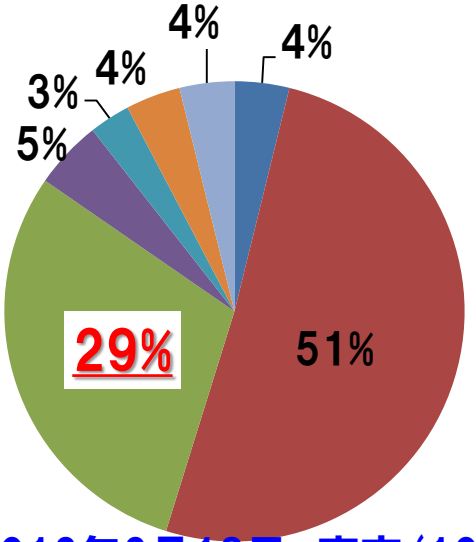


- すべてのプロジェクトに適用している
- ほとんどのプロジェクトに適用している
- だいたいのプロジェクトに適用にしている
- ほとんどのプロジェクトに適用していない
- すべてのプロジェクトで適用していない
- 分からない
- 無回答

- 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





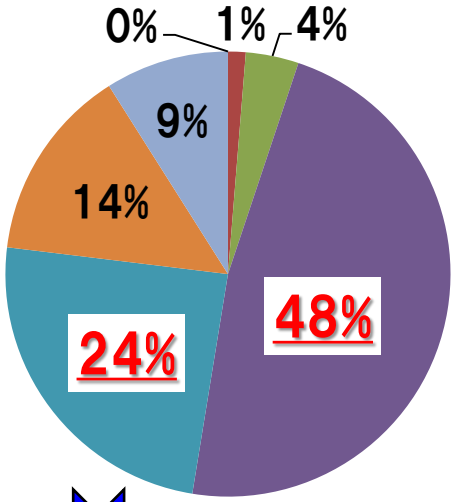


2011年11月18日　横浜(109名)　2012年10月24日　東京(76名)　2013年3月18日　東京(104名)

アジャイル開発の適用状況-国内-(３)

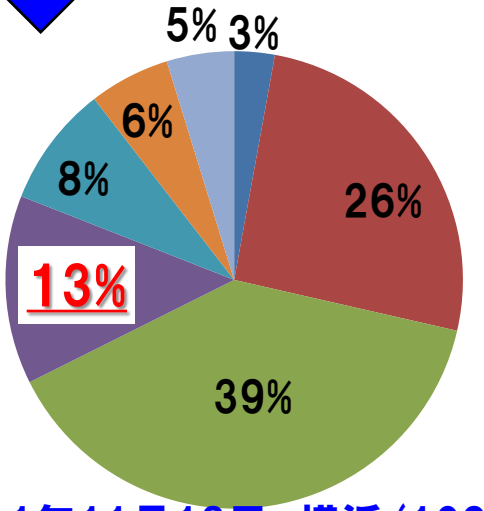
SEC Software Reliability Enhancement Center

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～

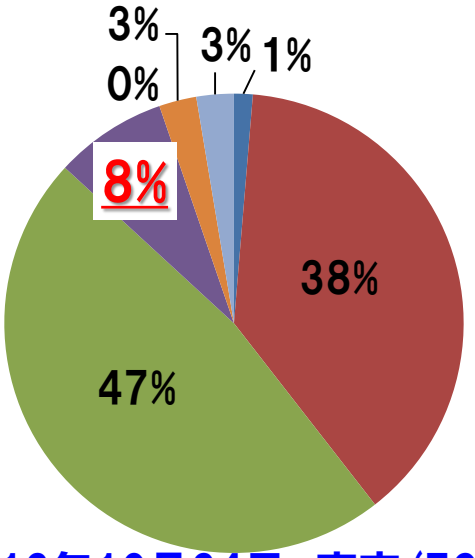


- すべてのプロジェクトに適用している
- ほとんどのプロジェクトに適用している
- だいたいのプロジェクトに適用にしている
- ほとんどのプロジェクトに適用していない
- すべてのプロジェクトで適用していない
- 分からない
- 無回答

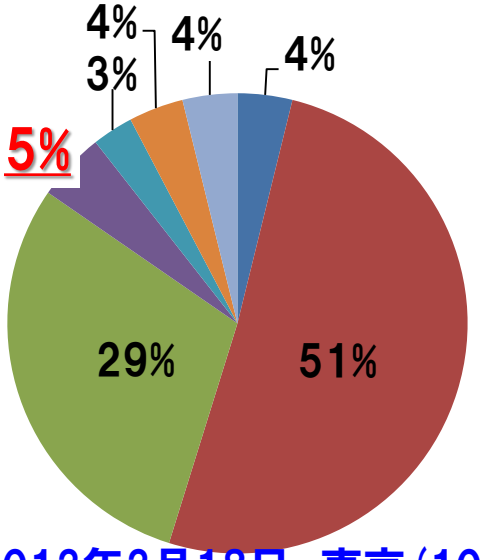
- 多くのプロジェクトで使っている
- 一部のプロジェクトで使っている
- 使いたいと考えているが実現していない
- 使う予定はない
- よく分からない／関係ない
- その他
- 無回答



2011年11月18日　横浜(109名)



2012年10月24日　東京(76名)

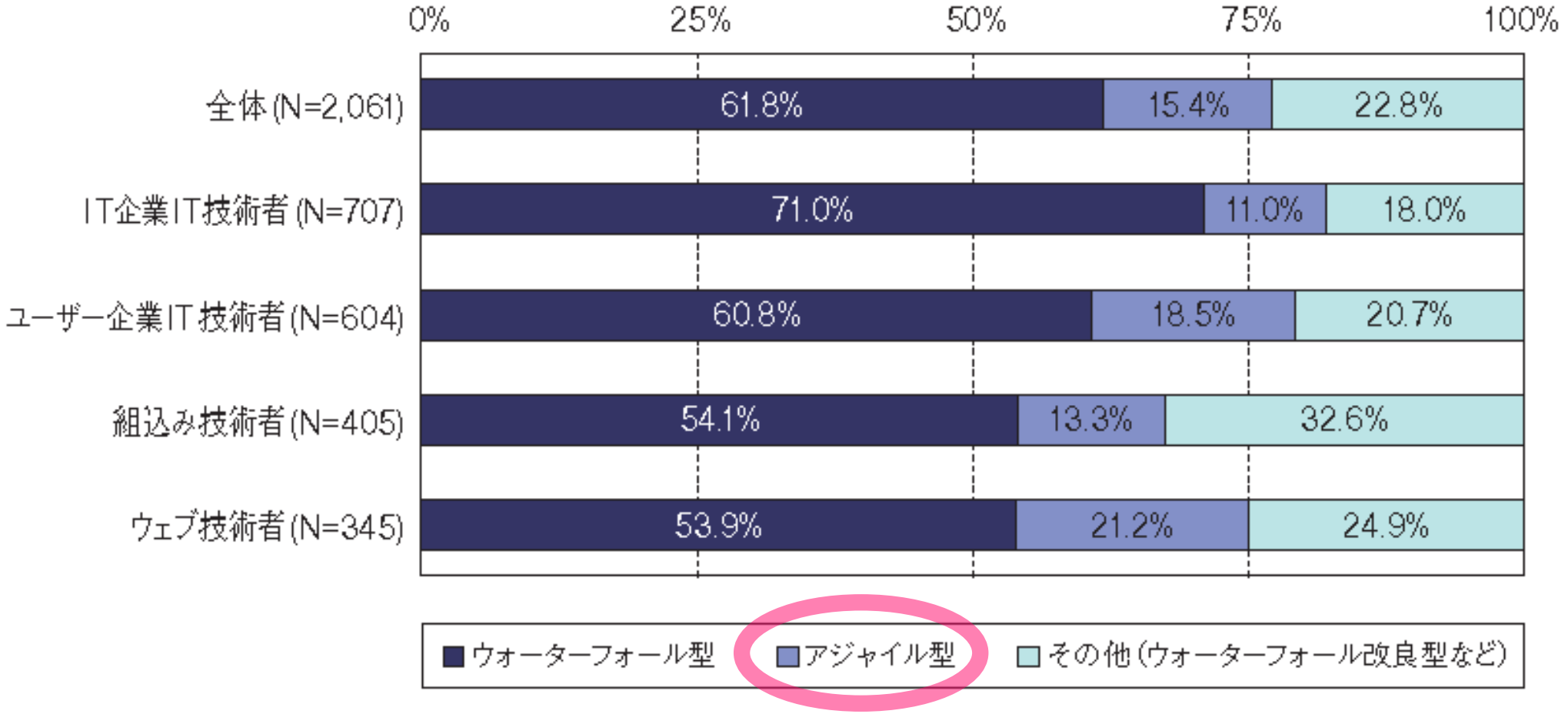


2013年3月18日　東京(104名)

アジャイル開発を用いる技術者

SEC
Software Reliability Enhancement Center

業務において最もよく用いられる開発プロセス（技術者別）



出典：「IT人材白書2014」, IPA, 2014年4月25日.
http://www.ipa.go.jp/jinzai/jigyou/about.html

アジャイル開発の現状

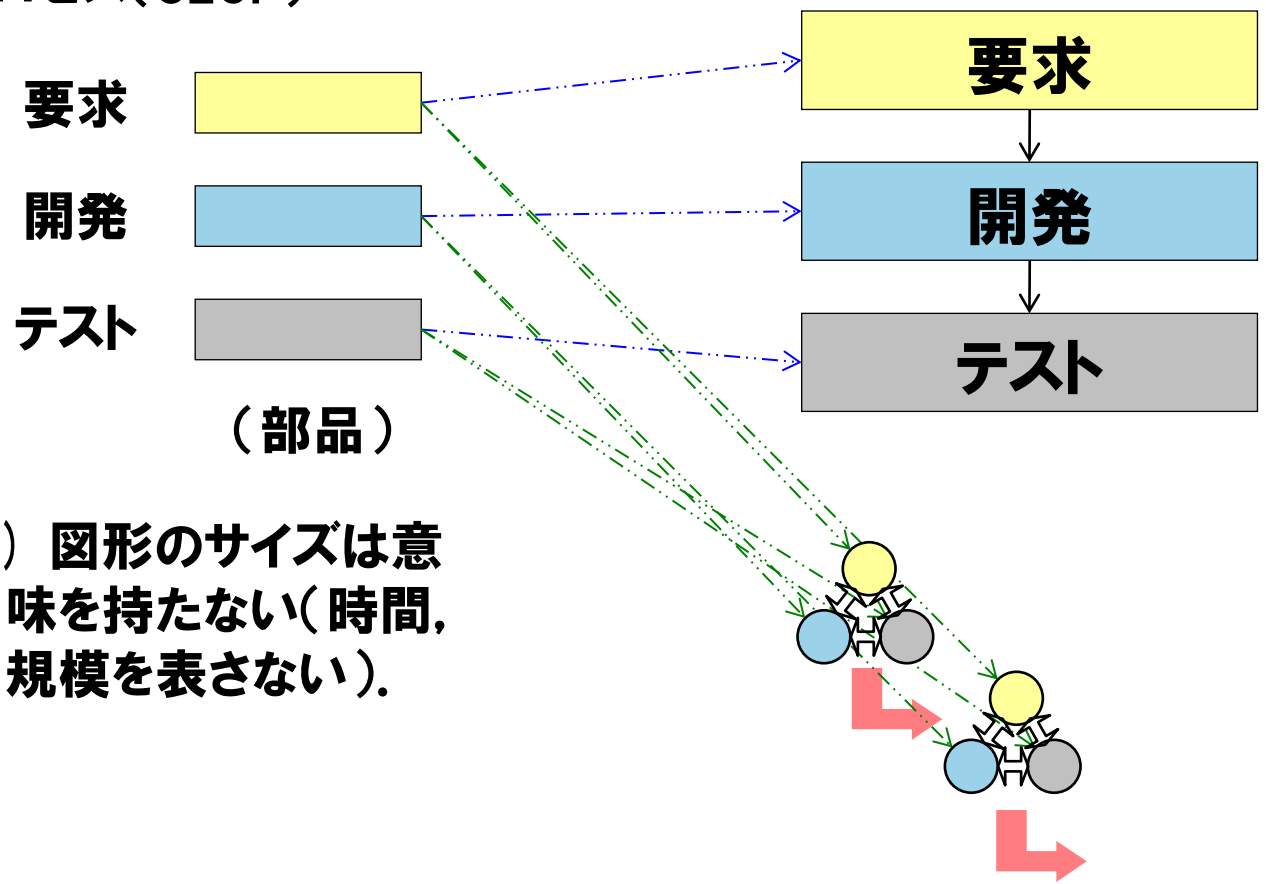
＜参考＞
非ウォーターフォール型(アジャイル)開発の動向と課題,
SEC journal, Vol.8, No.4, Dec. 2012.

（1）アジャイル開発の特徴

アジャイル開発のモデル – プロセスの対応 –

<標準>
ソフトウェアライフサイクル
プロセス(SLCP)

<実際>



ウォーターフォール型

大きなプロセスを
順に実施し,
それを1回で終了

注) 図形のサイズは意味を持たない(時間,規模を表さない).

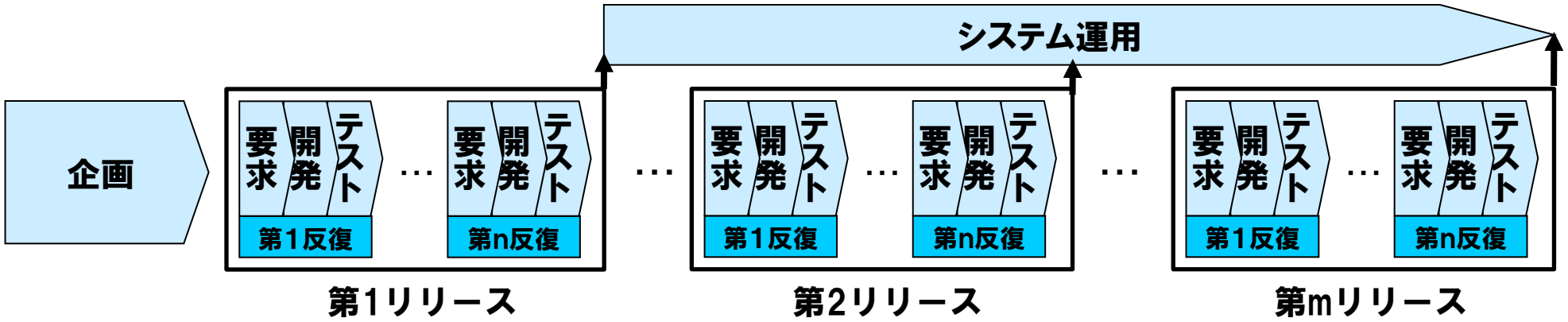
アジャイル型

小さなプロセスを
行き来しつつ実施し,
それを何回も反復

注) 図形のサイズは意味を持つ.

調査事例から導かれた開発プロセス・モデル（1）

モデル1



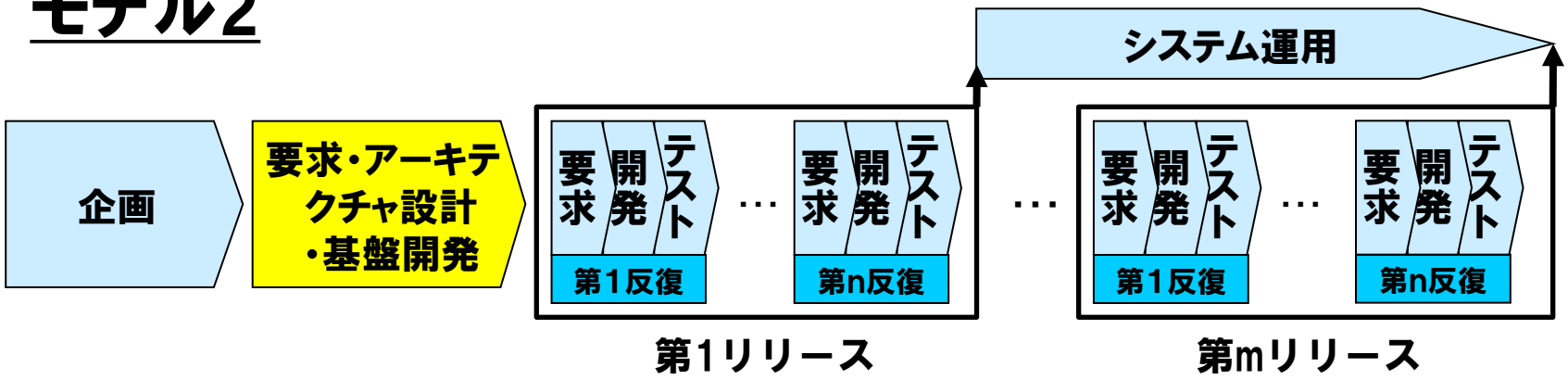
- n=1のケースもあり。

考え方
シンプルな基本形

調査事例から導かれた開発プロセス・モデル (2)

SEC
Software Reliability Enhancement Center

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

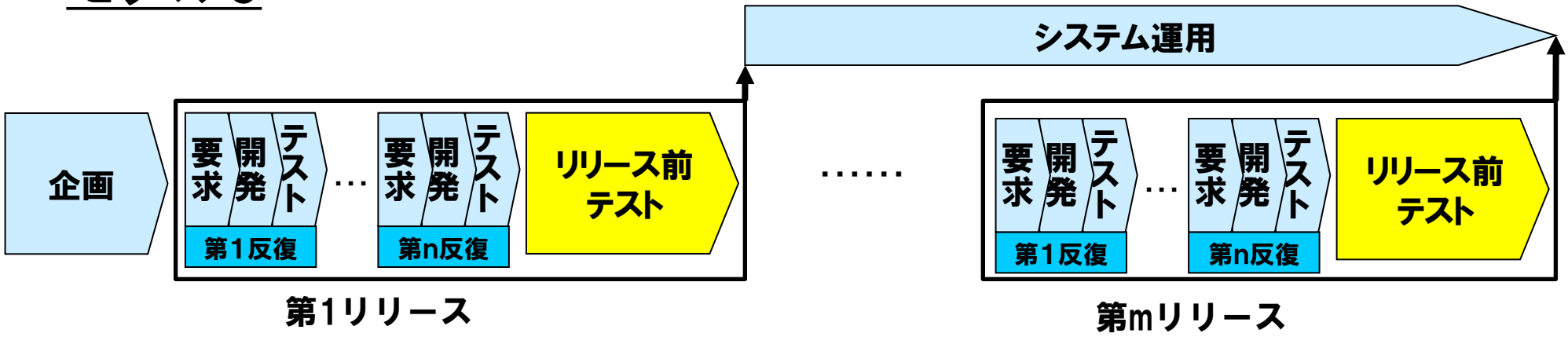
考え方

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．

調査事例から導かれた開発プロセス・モデル（3）

SEC
Software Reliability Enhancement Center

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．

ウォーターフォール型とアジャイル型との手法の違い

SEC Software Reliability Enhancement Center

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

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

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.

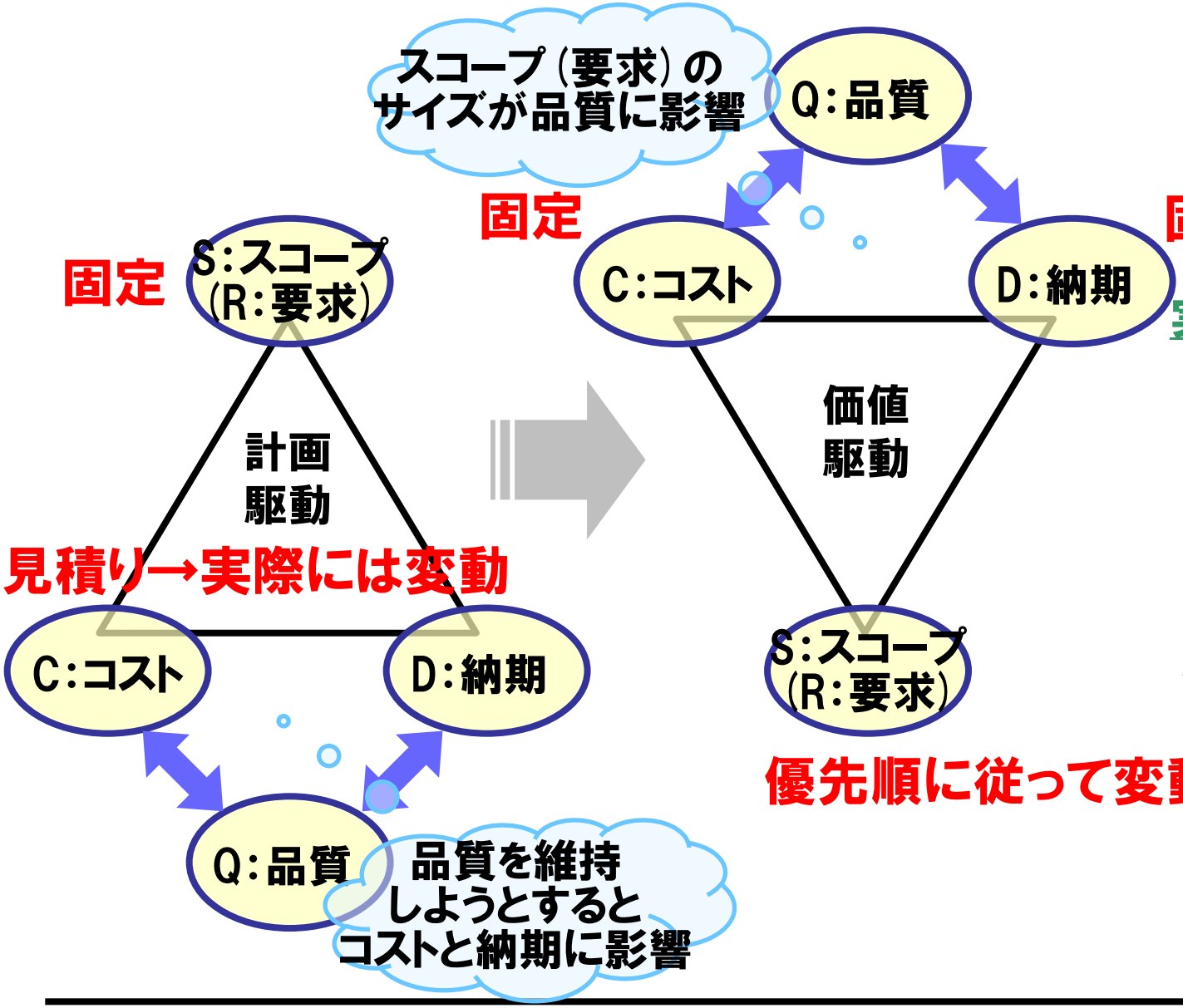
計画時には，ビジネス上，システム上の課題が未解決，開始後も変更の可能性大

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, 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.

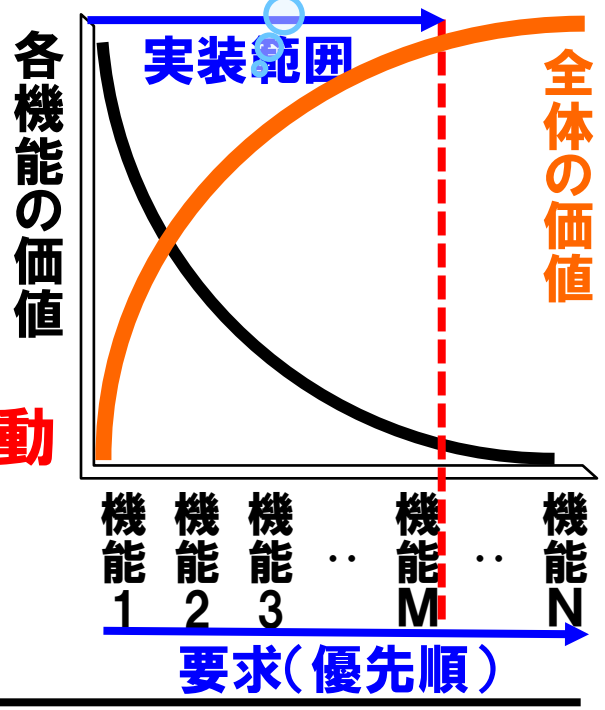
開発プロジェクトのパラメータ間の関係



参考
QCDの優先順位

固定

優先度の低い機能は実装しても結局は使われない
→無駄な実装はしない

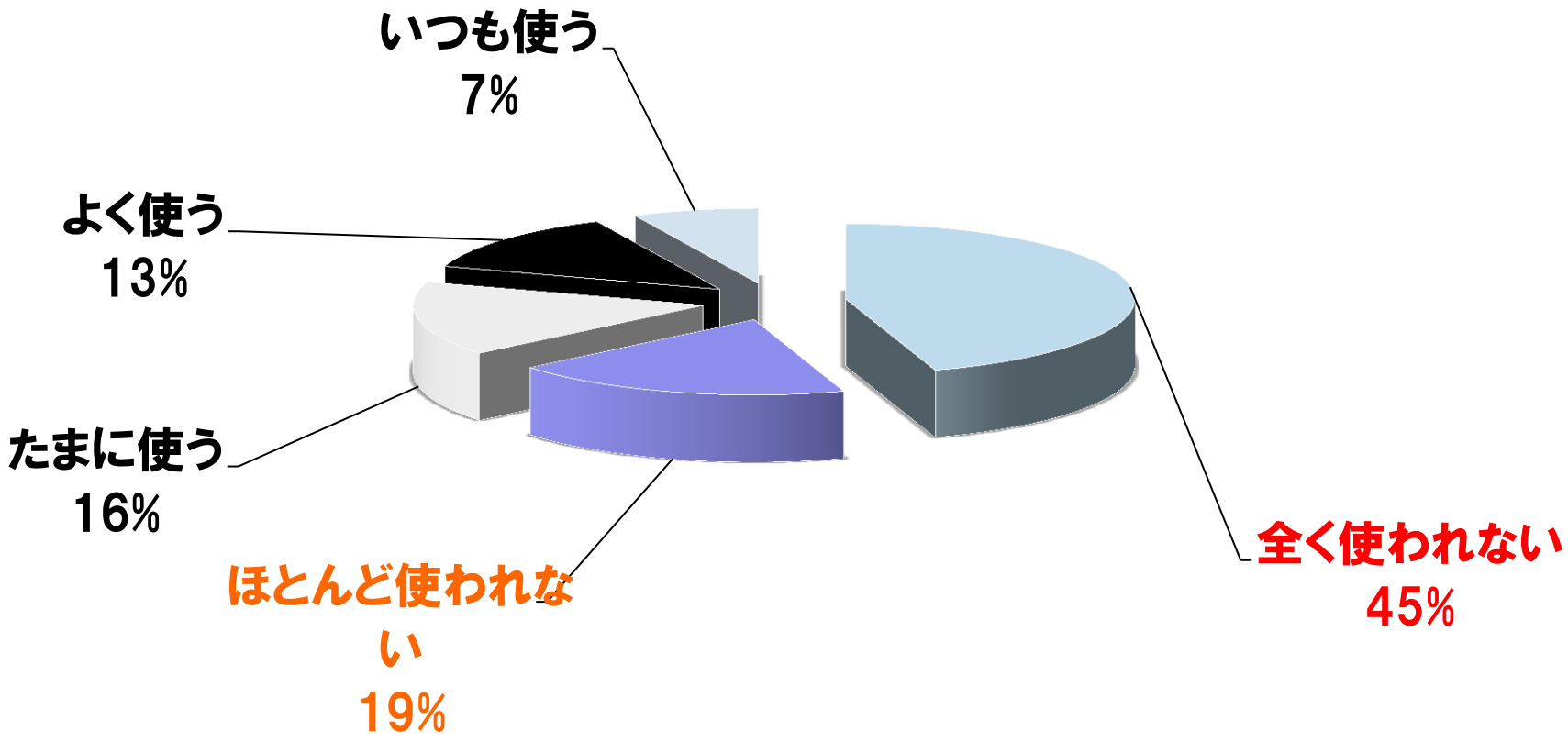


参考

システム機能の利用度（要求の劣化）

SEC Software Reliability Enhancement Center

システムの機能の利用度



<出典> *Standish group study report in 2000 chaos report*
（平鍋健児氏のプレゼン資料掲載）

（2）アジャイル開発の適用領域

［注］
- 適用領域も変化する
- 領域のコンテキストに応じ，適切な開発手法を

アジャイル開発の適用領域・試行領域

SEC
Software Reliability Enhancement Center

アジャイル開発は、
- 「顧客の参画の度合いが強い」
- 「動くソフトウェアを成長させながら作る」
- 「反復・漸進型である」
- 「人と人のコミュニケーション、コラボレーションを重視する」
- 「開発前の、要求の固定を前提としない」

という特徴を持つ。

全てのソフトウェア開発に、これらの特徴を有するアジャイル開発手法を適用できる、あるいはすべきだ、という**立場ではない**。
ビジネスや市場、その他の開発の“コンテキスト”によって、ウォーターフォール型の開発が適している場面もあれば、アジャイル型の開発が適している場面もある。

アジャイル開発の適用領域

SEC
Software Reliability Enhancement Center

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）.

アジャイル開発の試行領域

SEC Software Reliability Enhancement Center

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。

いずれの領域も適用事例が増えている

参考

中・大規模開発の事例一覧（H23年度調査）

SEC Software Reliability Enhancement Center

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

<出典> *http://www.ipa.go.jp/sec/softwareengineering/reports/20120328.html*

適用にあたっての主な工夫（1/3：一覧）

SEC Software Reliability Enhancement Center

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

適用にあたっての主な工夫(2/3)

SEC Software Reliability Enhancement Center

工夫例(1/2)

チーム間ローテーション

チーム間の知識伝播促進のため，メンバのローテーションを行う．一時的に速度は落ちるが，各チームの知識を効率よく伝播でき多能工化が高まる．

段階的朝会

複数チーム間は朝会を段階的に実施．チーム→全体(→チーム)．

漸進的な展開

一度にすべてのチームに広げるのではなく，まずは導入障壁の低いところ，最も必要なところから順次導入し，少しずつ展開．ふりかえりで，次はどこに広げていけばいいかを考える．

コミュニケーション・ツールの活用

TV会議システム，雑記帳システム(SNS，Blog等)

適用にあたっての主な工夫 (3/3)

SEC Software Reliability Enhancement Center

工夫例 (2/2)

アーキテクチャの重視

プロジェクト前半にアーキテクチャを構築する事例が多く，アーキテクチャ専門チームを編成して構築.
構築されたアーキテクチャを組織の共通基盤とし，再利用できるようにしている事例あり.

疎結合な機能分割

疎結合な機能でサブシステム分割を行う.
7割のチームがCI (継続的インテグレーション) を実施.

テスト専用フェーズ

プロジェクト後半で専用のテストフェーズを実施. プログラム開発の反復を停止する事例と，テストのみの反復期間を設ける事例あり.

浮かび上がった問題点

SEC
Software Reliability Enhancement Center

主な問題点

全体計画の把握困難

要求の変化や開発状況に応じて着手する順番や範囲を決めるため，プロジェクト開始時にプロジェクト全体の把握が困難な事例あり．

ビジネス企画側にボトルネック発生

スクラム導入の結果，ビジネス企画者の決定待ち等のボトルネックが発生した事例が多い．中には開発者の信頼をなくした事例もあり

反復リズムとの不適合状態の発生

セキュリティ監査や外部テスト業者，発注者の外部組織や関連組織との関係において，反復リズムと適合せずに問題が発生している事例あり．

アジャイル開発
プラクティス活用リファレンスガイド

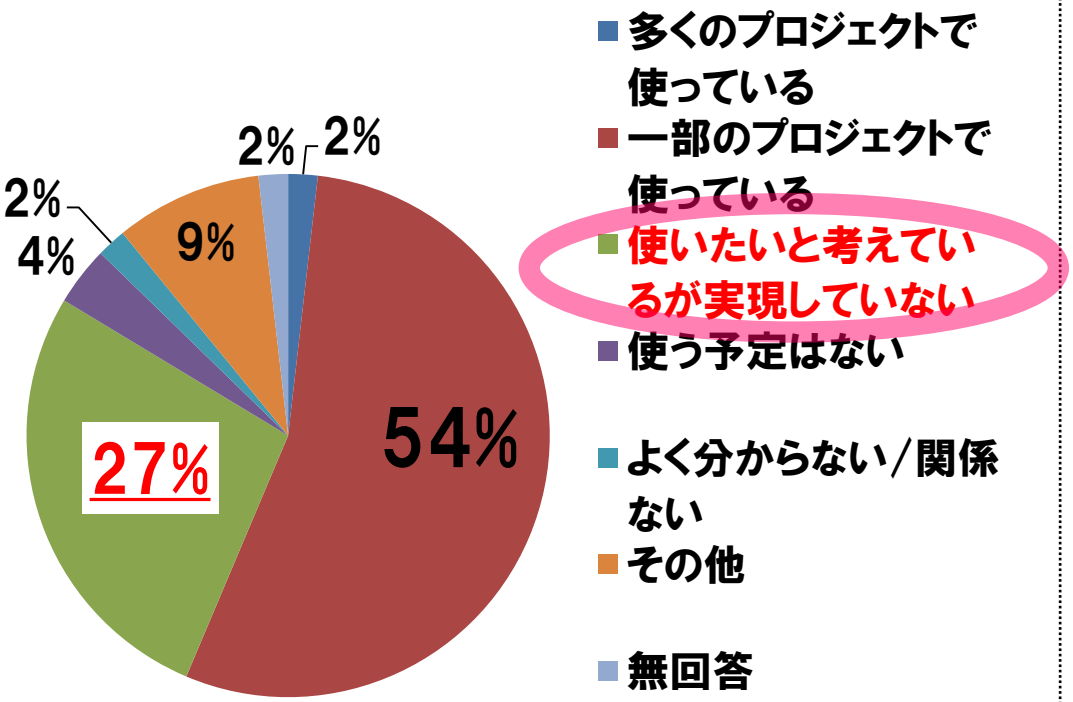
※プラクティス：アジャイル開発を実践する活動項目

http://www.ipa.go.jp/about/press/20130319.html
http://www.ipa.go.jp/sec/softwareengineering/reports/20130319.html

背景:アジャイル開発の適用状況-国内-

SEC Software Reliability Enhancement Center

アジャイル開発を使っていますか？

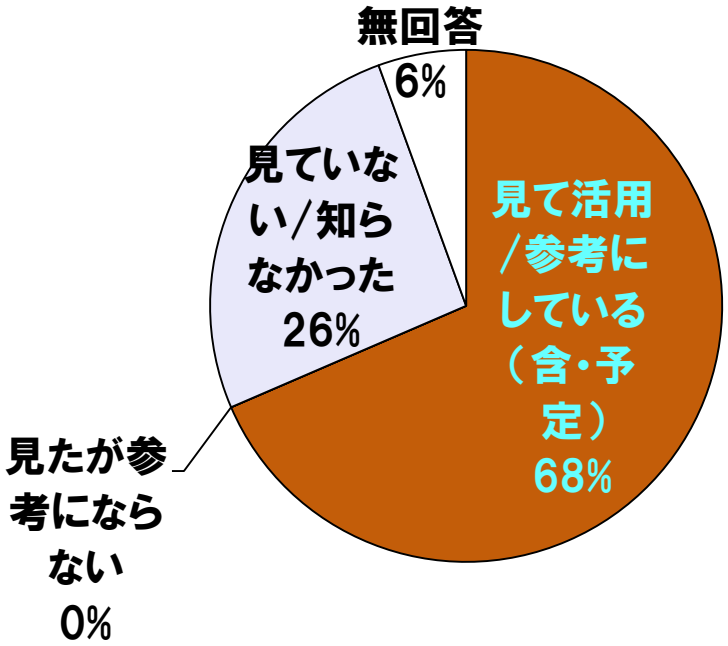


2013年10月30日　東京（55名）
（本日とほぼ同内容のセミナー）

http://sec.ipa.go.jp/seminar/20131030.html

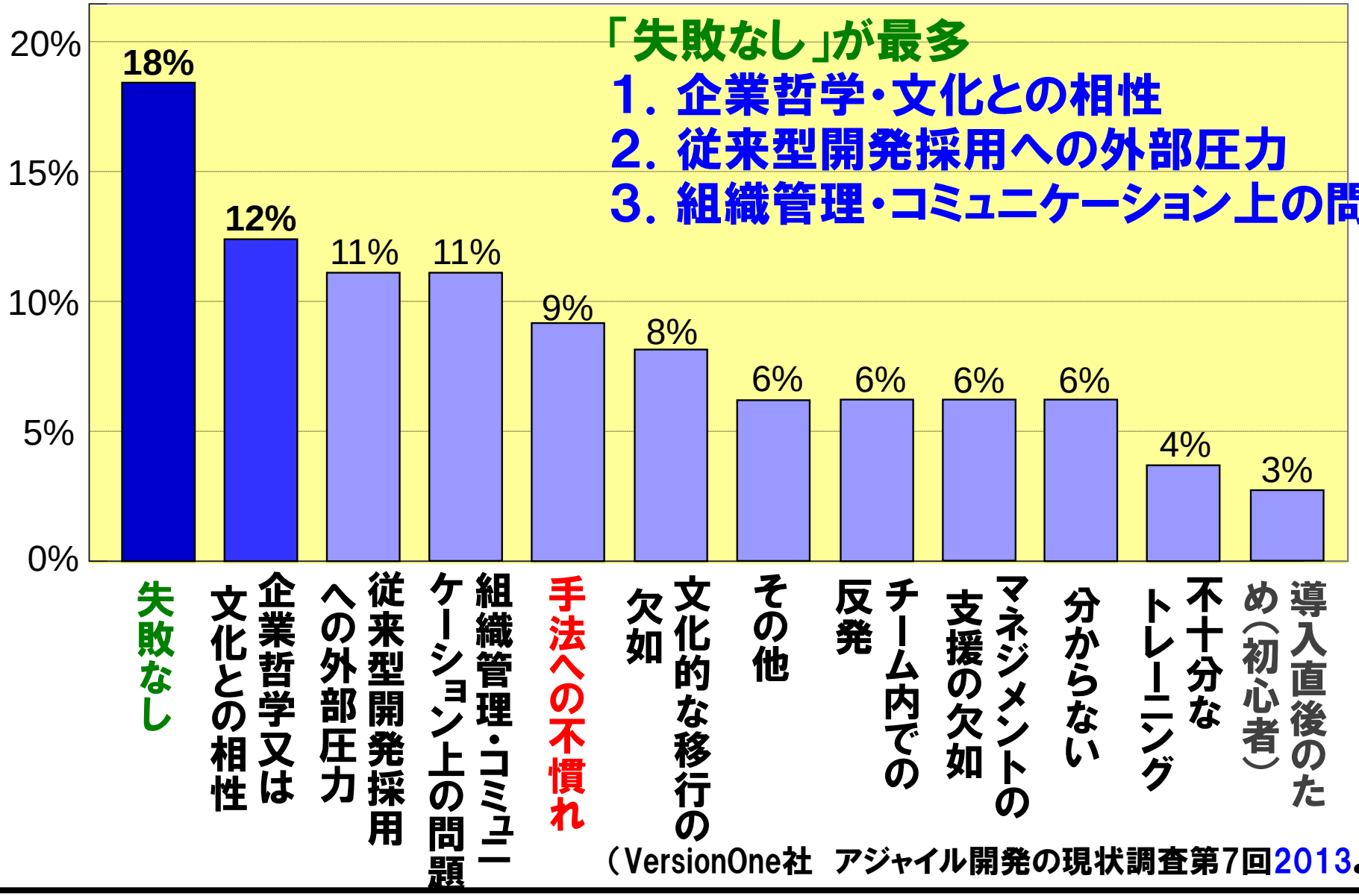
参考

「アジャイル型開発における
プラクティス活用リファレンスガイド」



参考

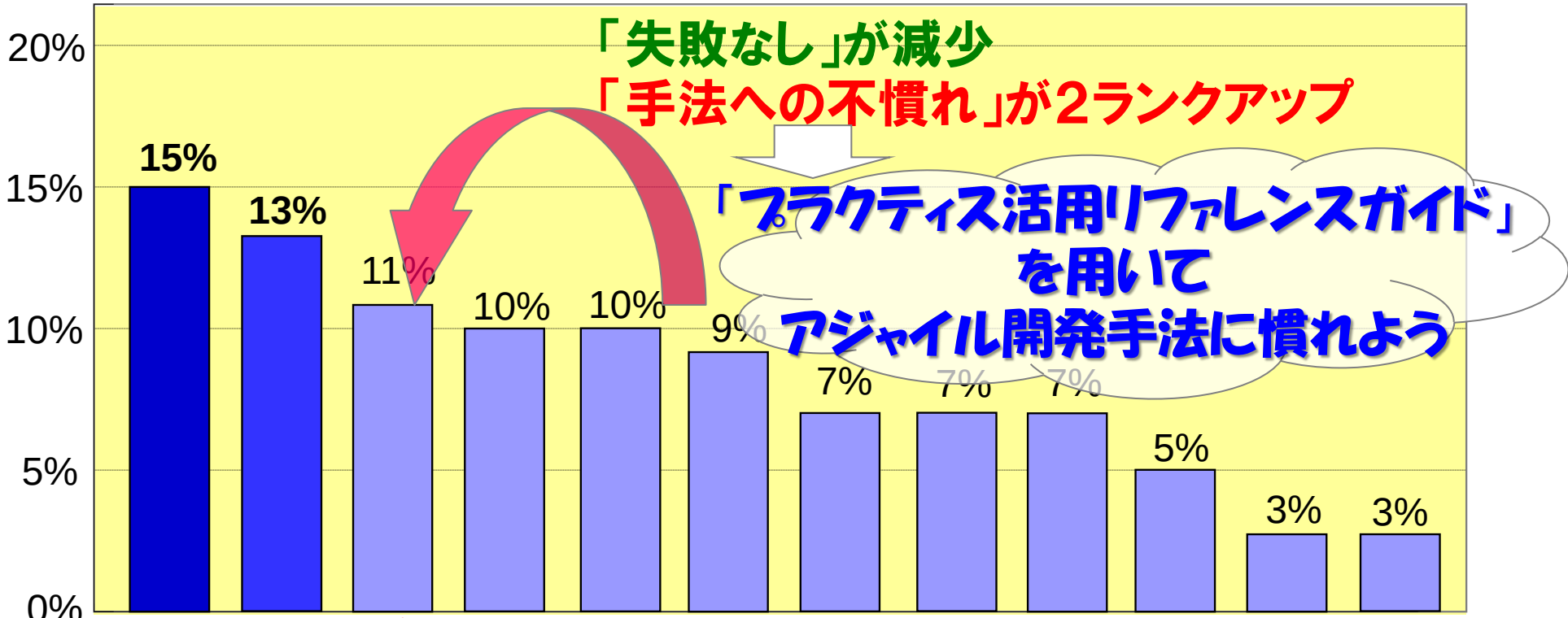
アジャイル型開発プロジェクトの失敗理由（海外）



（VersionOne社　アジャイル開発の現状調査第7回2013より）

参考

アジャイル型開発プロジェクトの失敗理由（海外）



（VersionOne社　アジャイル開発の現状調査第8回2014より）

ガイドの特徴

アジャイル開発を実践する活動項目

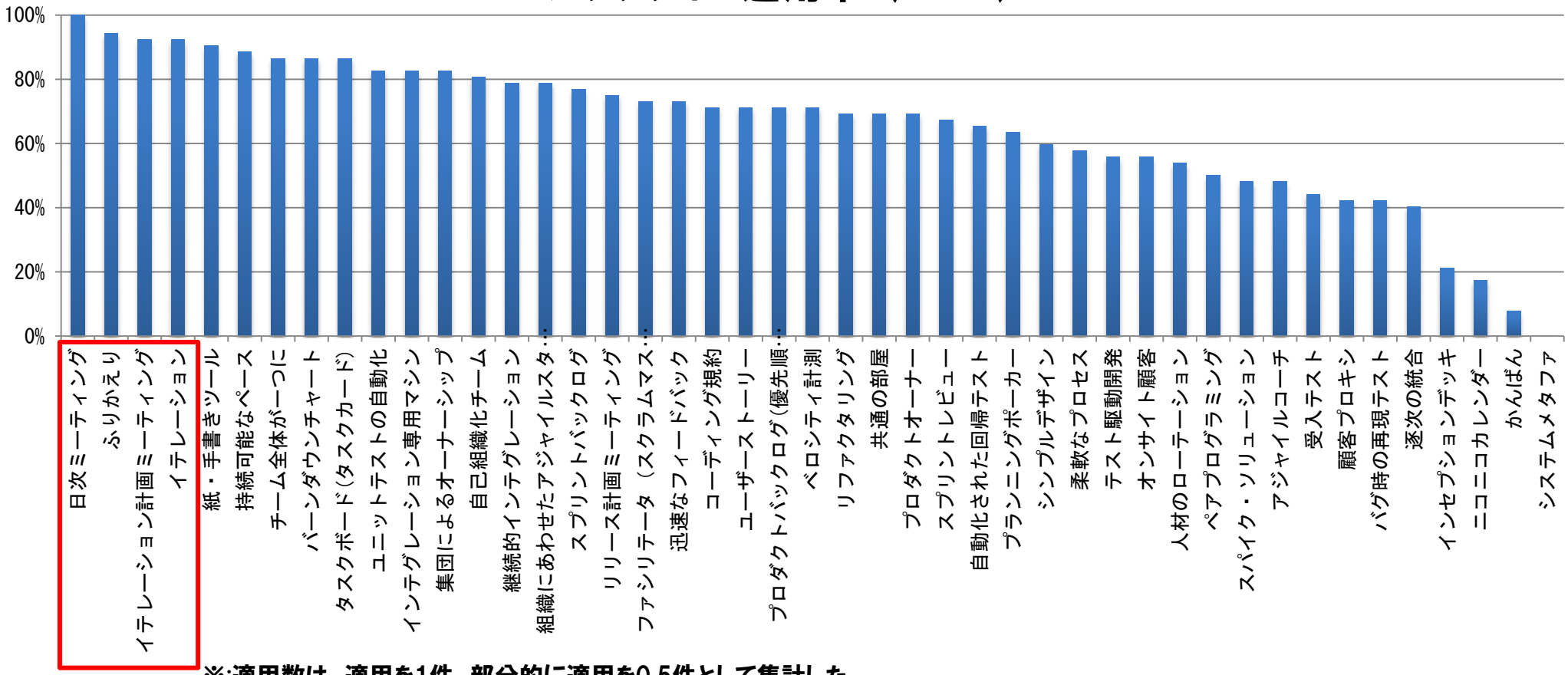
- 55個*のプラクティス，26個の事例，9つの活用ポイント　計224ページ
- 日本国内の開発現場からのヒアリングにより収集した知見を，パターン記述形式で取りまとめ
- MS-Wordファイルを公開し，クリエイティブ・コモンズ・ライセンスの下に，改変自由・営利目的利用可で使用許諾

＊ 類似のものを統合し，最終的には45個

適用プラクティス（全体）

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。

プラクティス例概要 - 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る。

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）。

プラクティス例概要 - ふりかえり

SEC Software Reliability Enhancement Center

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。

プラクティス例概要 - イテレーション計画ミーティング

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(4) 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。

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

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。

活用のポイント (1)

SEC Software Reliability Enhancement Center

(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) スコープの変動が激しい

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))。

活用のポイント (2)

SEC Software Reliability Enhancement Center

(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が一緒に仕事をすることで、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。

活用のポイント (3)

SEC Software Reliability Enhancement Center

(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、ぜひ活用するべきである(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。

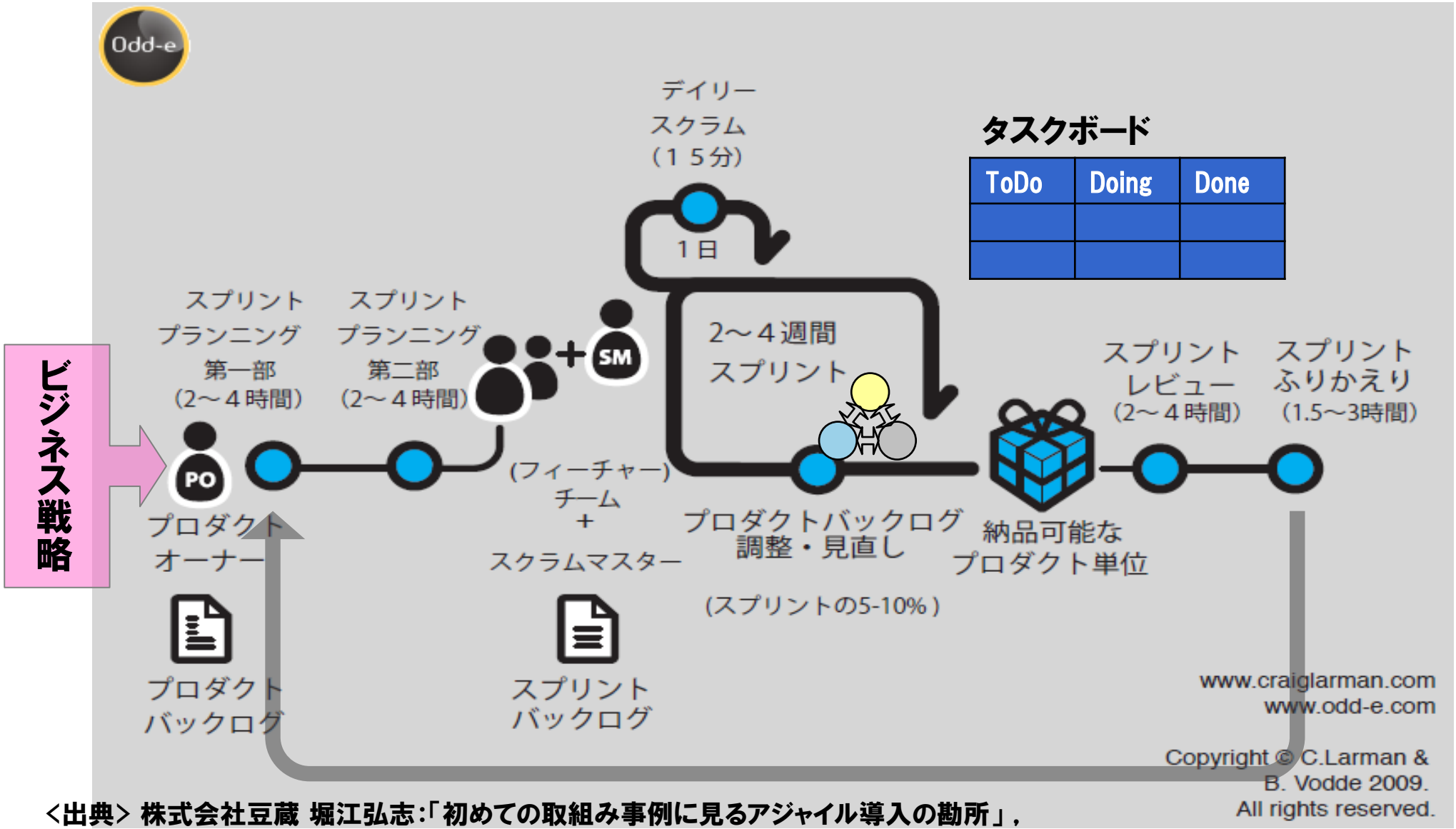
(9) 初めてアジャイル開発に取り組む

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。

アジャイル開発人材

参考

アジャイル開発プロセスの流れ：スクラムの例



<出典> 株式会社豆蔵 堀江弘志：「初めての取組み事例に見るアジャイル導入の勘所」，JASA主催/IPA共催セミナー，2011-11-18. *http://www.ipa.go.jp/sec/softwareengineering/events/20111116.html*

アジャイル開発者に求められるスキル

SEC
Software Reliability Enhancement Center

アジャイル開発における発注者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

アジャイル開発人材の育成の考え方

SEC Software Reliability Enhancement Center

＜アジャイル開発の実際＞

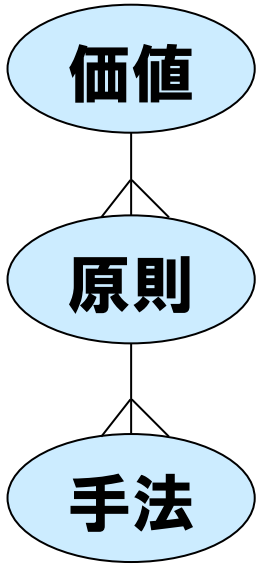
アジャイル開発を実践する活動項目

- 一つのプロジェクトで全てのプラクティスを使う訳ではない
- 各プラクティスに厳格な規範はない
- 様々な方法論・数あるプラクティスから，プロジェクトや組織に適したものを取捨選択し，カスタマイズすることが必要

- （平時）一通りのプラクティスを理解する
- （プロジェクト参画時）使用するプラクティスの習得

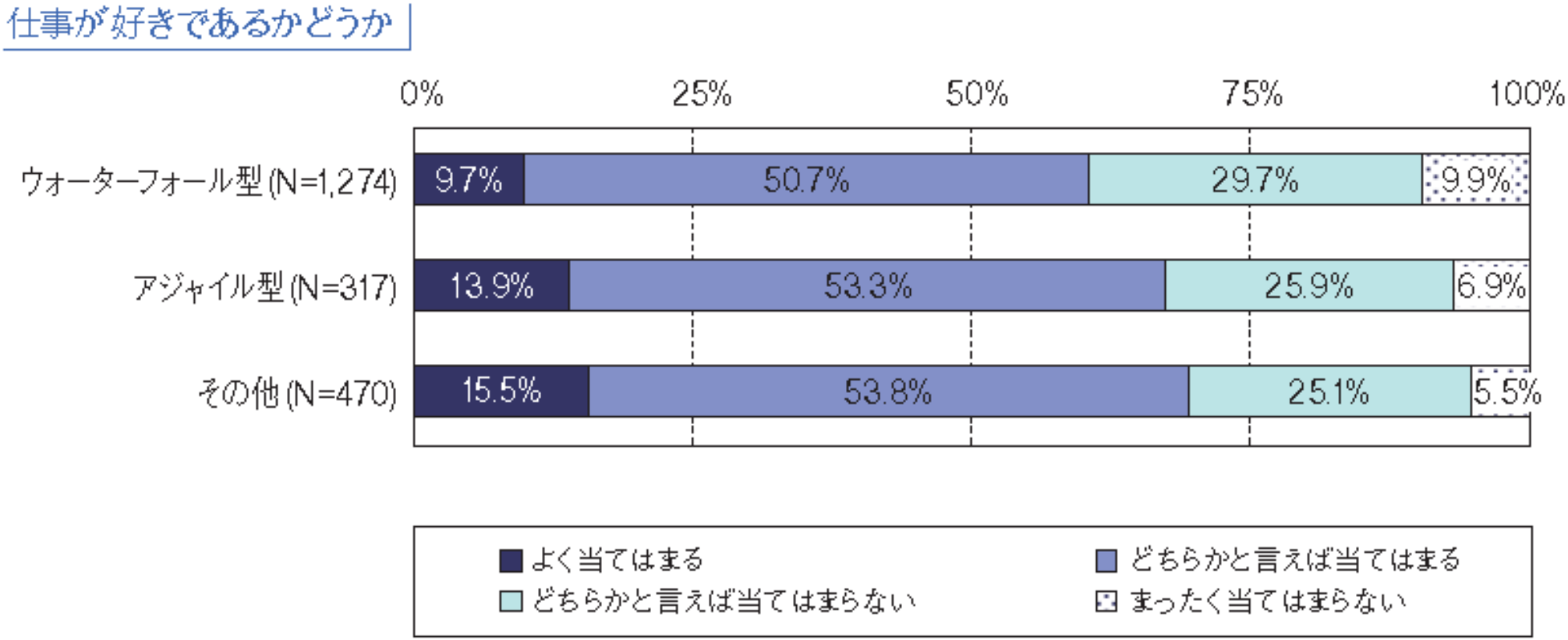
↓

全てを完全に身につけるより，価値に従って行動する習慣を確実に身につけることが重要



参考

アジャイル開発技術者の仕事に対する感じ方・考え方 (1)



アジャイル開発技術者は，WF開発技術者に比べ，仕事が好きな割合が高い

出典:「IT人材白書2014」, IPA, 2014年4月25日.
http://www.ipa.go.jp/jinzai/jigyou/about.html

参考

アジャイル開発技術者の仕事に対する感じ方・考え方 (2)

仕事の充実感・やりがいに対する満足度

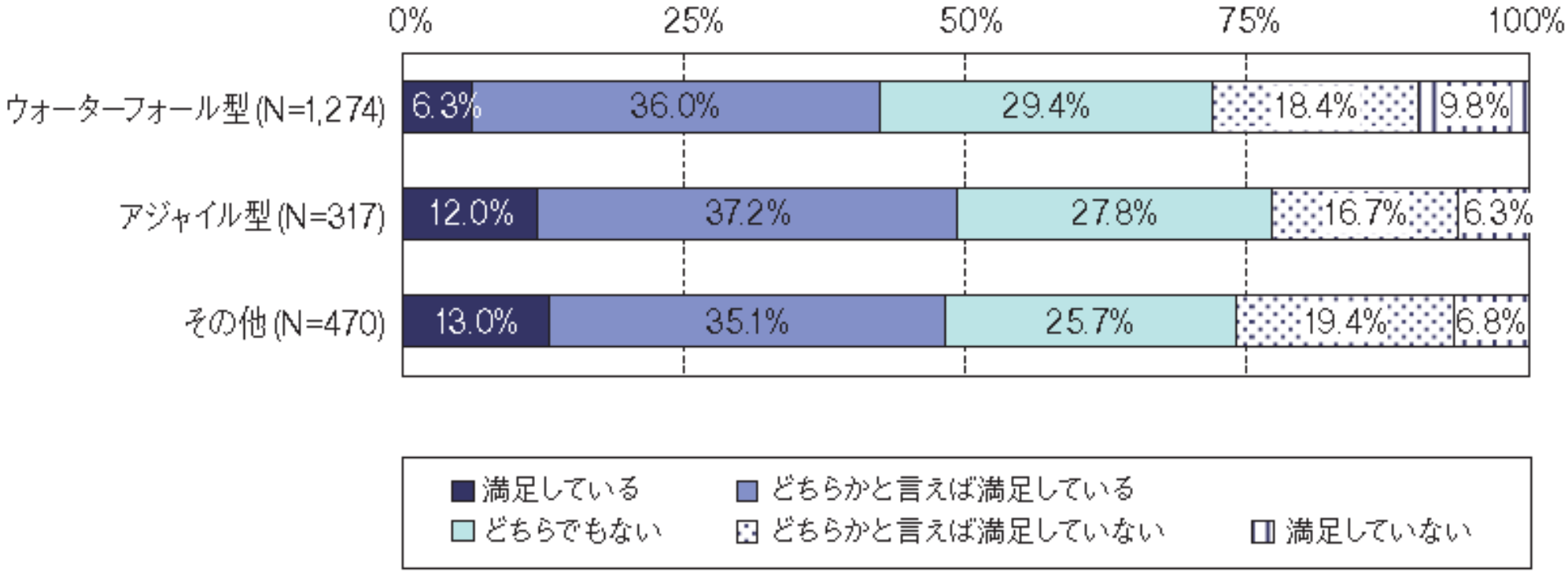


図２－３－19　仕事に対する意識【使用する開発手法別】

アジャイル開発技術者は，WF開発技術者に比べ，仕事に満足している割合が高い

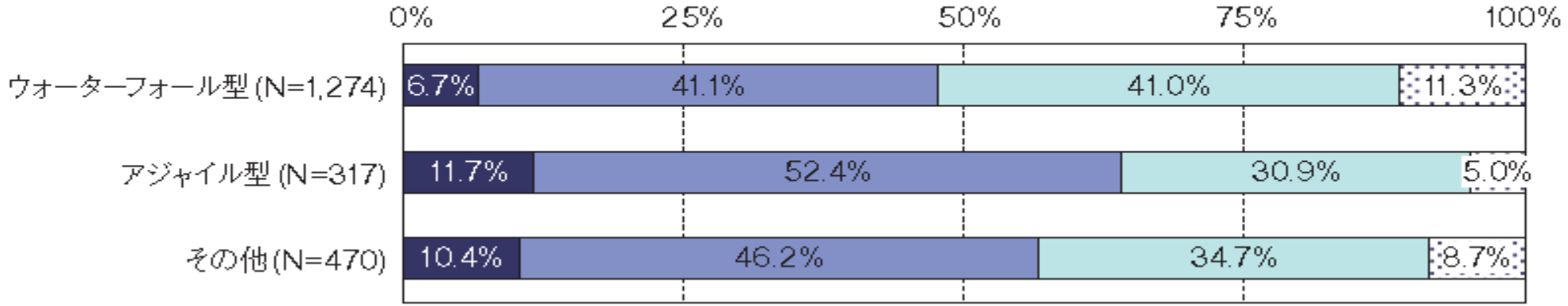
出典:「IT人材白書2014」, IPA, 2014年4月25日.
http://www.ipa.go.jp/jinzai/jigyou/about.html

参考

スキルを学ぶ(1/2)

SEC Software Reliability Enhancement Center

新しい技術やスキルの習得のための勉強に自主的に取り組んでいるかどうか



次にどんな技術やスキルを学ぶべきかよくわかっているかどうか

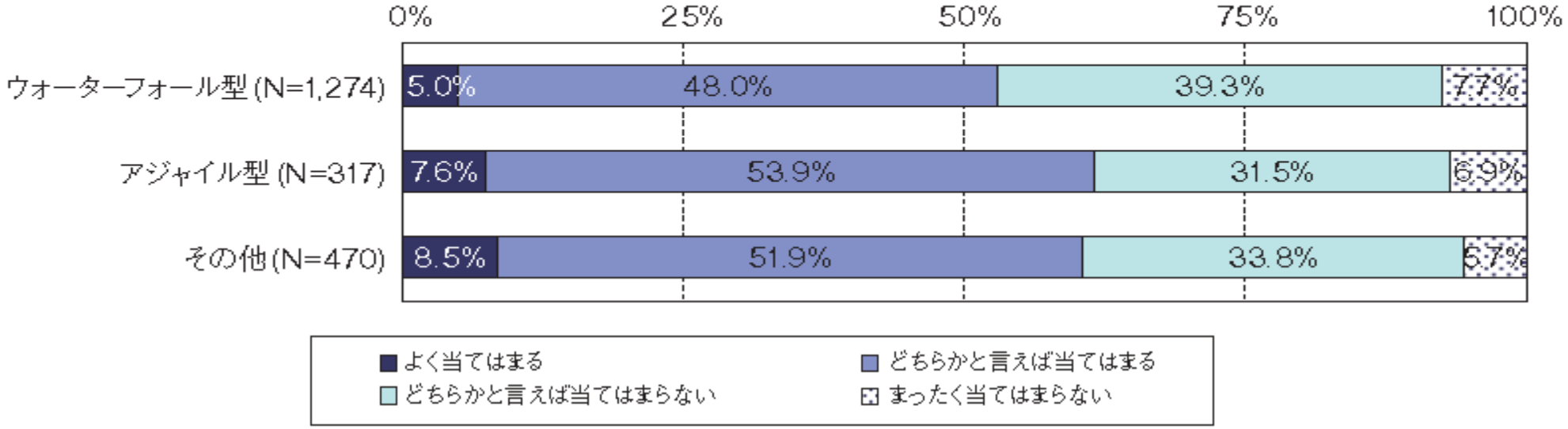


図2－3－20　勉強に対する取り組み姿勢【使用する開発手法別】

出典:「IT人材白書2014」, IPA, 2014年4月25日.

参考

スキルを学ぶ(2/2)

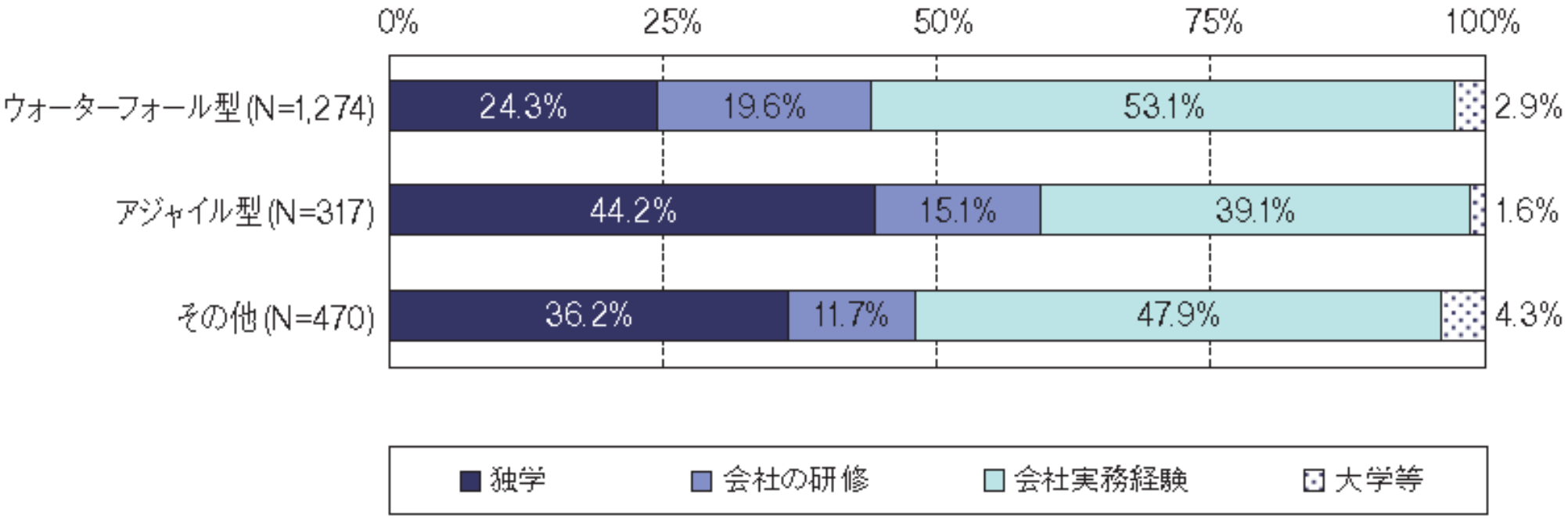


図2－3－21　開発手法を学習した場所【使用する開発手法別】

アジャイル開発技術者は，自主・独学で手法を学んでいる

出典:「IT人材白書2014」, IPA, 2014年4月25日.
http://www.ipa.go.jp/jinzai/jigyou/about.html

参考

モチベーション…科学的実証の結果

SEC Software Reliability Enhancement Center

報酬のインセンティブは，視野を狭め，心を集中させることから，単純な仕事では効果があるが，そうでない創造的な仕事では逆効果．

成果を高めるのは，内的な動機付けに基づくアプローチ．

すなわち，重要だからやる，好きだからやる，面白いからやる，何か重要なことの一部を担っているからやる，というもの．

仕事において重要な要素は次の3つ：

- 自主性…自分の人生の方向は自分で決めたい
- 成長　…何か大切なことについて上達したい
- 目的　…私たち自身よりも大きな何かのためにやりたい

（ある程度の）裁量

スキルアップになる

顧客の”価値”を高める

<出典> Dan Pink on the surprising science of motivation（ダニエル・ピンク「やる気に関する驚きの科学」）
http://www.ted.com/talks/lang/ja/dan_pink_on_motivation.html

上記要素は開発手法とは独立であるが，アプローチのし易さに特徴が反映される．

アジャイル開発の課題

アジャイル開発には、どんな契約がふさわしいのか？

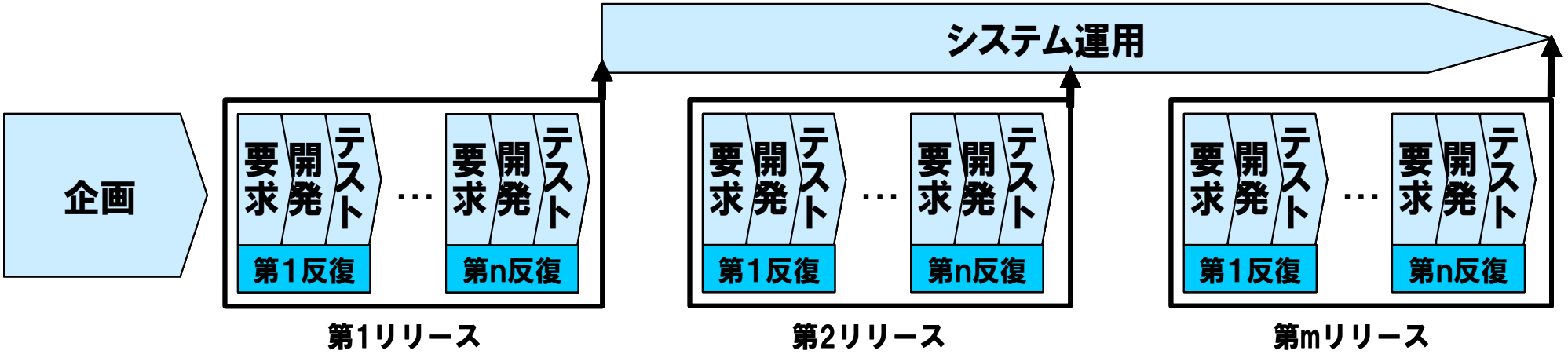
SEC Software Reliability Enhancement Center

- 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。

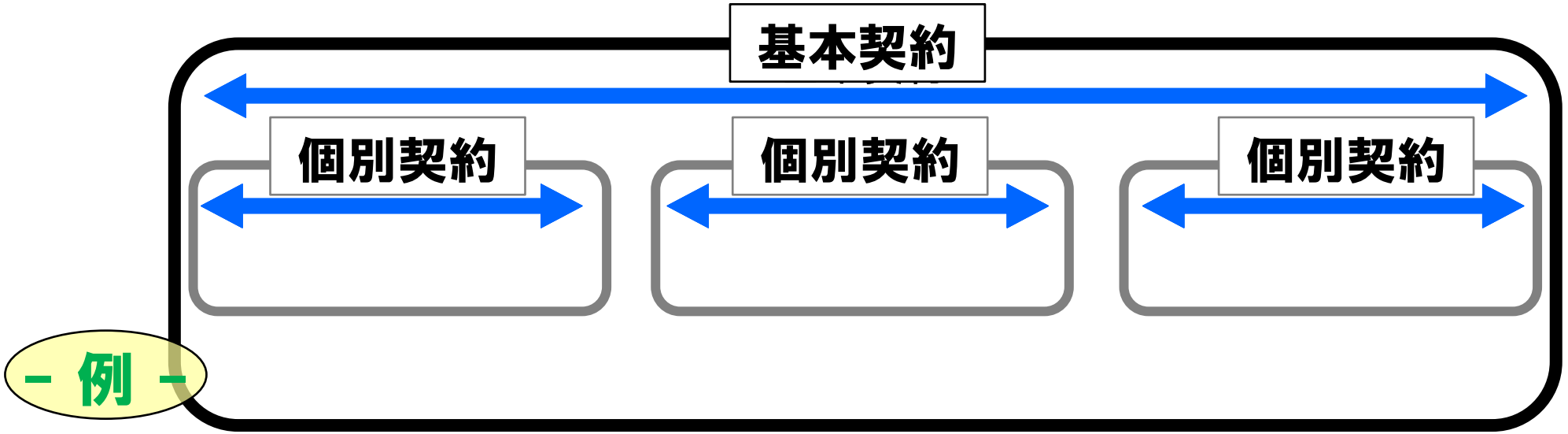
注）ユーザ側での労働者派遣契約に基づく開発チーム構成には，契約上の問題は特にない．
（内製傾向の高まりに伴い増加？）

契約

基本/個別契約モデルの概要



- n=1のケースもあり。



契約

参考

各種開発手法の比較(2/2)

SEC Software Reliability Enhancement Center

図表6-258 3種開発法の比較(参考値)

		WF	アジャイル	xRAD	アジャイル/WF	xRAD/WF
総費用/JFS	平均	112.19	135.45	40.70	1.21	0.36
	係数	28.20	57.65	6.40		
工数/JFS	平均	1.28	2.15	0.48	1.68	0.37
	係数	0.44	1.60	0.26		
工期/JFS	平均	0.31	0.24	0.10	0.77	0.32
	係数	0.04	0.04	0.03		

データ数はウォーターフォール法が337件(総開発費2億円以下)、アジャイルが51件、超高速開発が43件であった。 (xRAD: 超高速開発手法のツールのひとつ)
平均は各開発法のデータの平均値であり、係数はJFSの1単位増加にともなう総費用、工数、工期の増加を示している。係数が大きいと、システム開発規模が増加するにつれて各要因の値の増加はより大きくなることを示している。

アジャイルは,テストを繰り返す等の理由により,工数大

アプリケーションの特性に合わせて、各種開発方法の特徴をつかんでツールの選択をすること。適材適所である。

<出典> ソフトウェアメトリックス調査2014,一般社団法人 日本情報システム・ユーザー協会(JUAS).

適切なツールの活用 (1/2)

SEC Software Reliability Enhancement Center

保守

ソフトウェアメトリックス2014年版(保守調査報告) JUAS

保守環境について(Q7)

図表7-60 テストツールの使用(単位:件, %)

テストツールの使用の有無	件数(件)	割合(%)
1.テストツールを使用している	153	27.0%
2.テストツールを使用していない	414	73.0%
合　計	567	100.0%

■ 保守環境におけるテストツールの使用は少ない
■ 保守作業結果の確認を目視作業に頼る精度には限界があり、常時ツールで前後比較を実施することが望まれる

<保守調査> 44

<出典> ソフトウェアメトリックス調査2014, 一般社団法人 日本情報システム・ユーザー協会 (JUAS).

適切なツールの活用 (2/2)

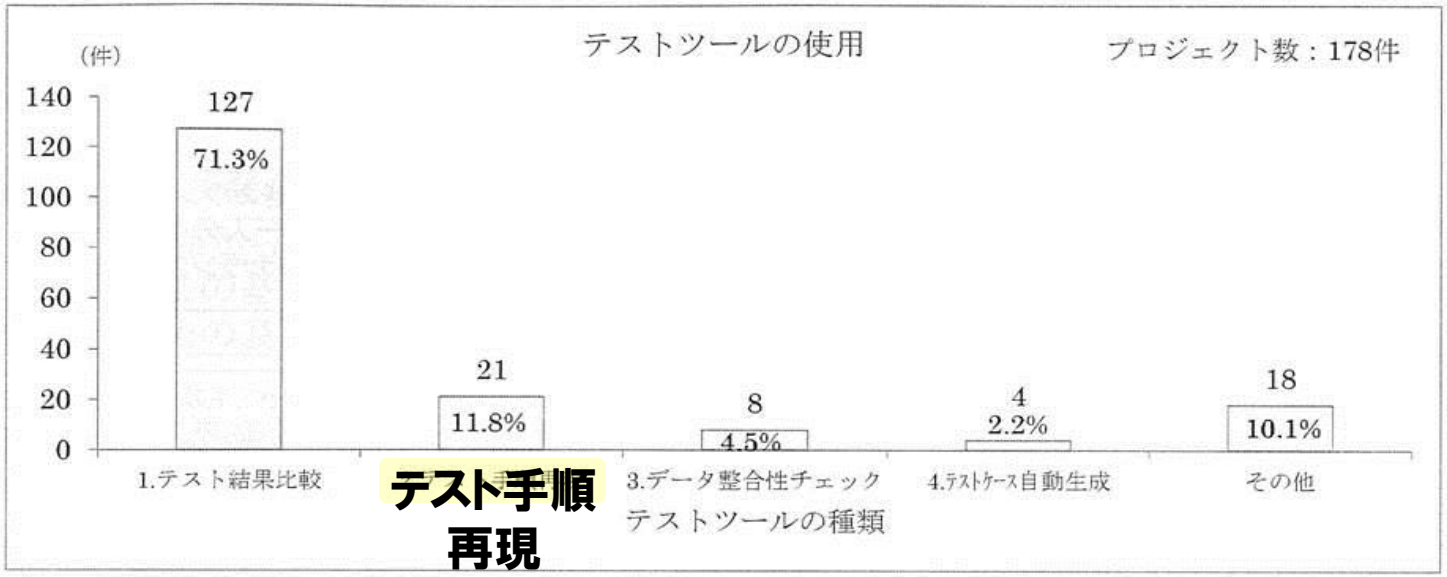
SEC Software Reliability Enhancement Center

ソフトウェアメトリックス2014年版(保守調査報告) JUAS

保守

保守環境について(Q7)

図表7-61 テストツールの使用の分布(単位:件, %)



■ツール使用は「テスト結果比較」が多いが、テスト手順の再現ツールの活用は生産性、品質向上に役立つので、更なる使用拡大が望まれる

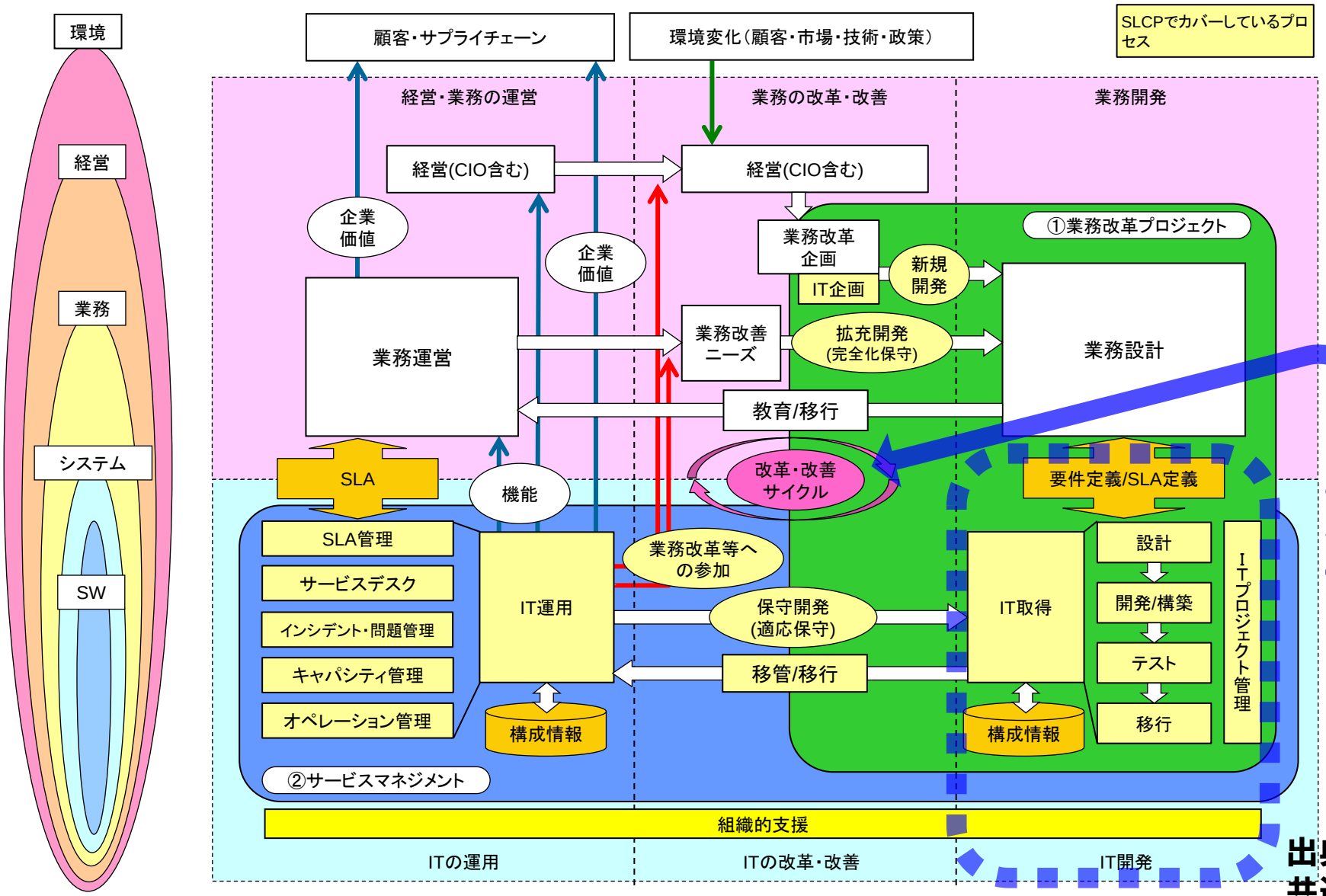
テストツールの活用により, 繰返しテストの効率化を

<保守調査>

45

<出典> ソフトウェアメトリックス調査2014, 一般社団法人 日本情報システム・ユーザー協会 (JUAS).

業務運用とアジャイル開発



全体最適

『“運用→開発→運用”という流れで全体を捉えるべき』

このサイクルを短期間でまわす

出典：共通フレーム2013

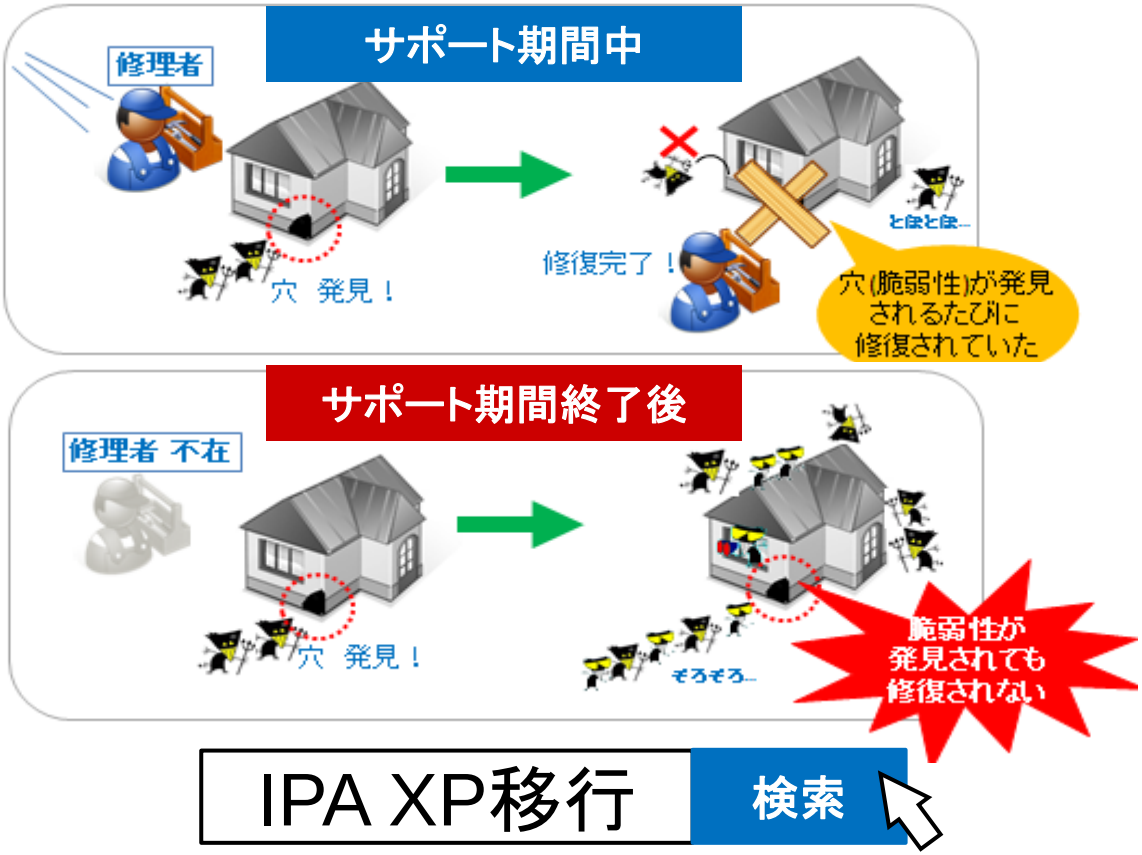
ご清聴，ありがとうございました

http://www.ipa.go.jp/sec/index.html

IPAからのお願い

Windows XPのサポートが、2014年4月9日に終了しました。

まだ移行していない方は、不正アクセス等を回避するためサポートの継続する後継OS、または代替OSへの移行が望まれます。



iパス ITパスポート試験

SEC
Software Reliability Enhancement Center

https://www3.jitec.ipa.go.jp/JitesCbt/

付録

参考

アジャイル宣言における4つの価値

SEC Software Reliability Enhancement Center

私たちは，ソフトウェア開発の実践を手助けする活動を通じて，よりよい開発方法を見つけだそうとしている．

この活動を通して，私たちは以下のことを重視する：

①プロセスやツールよりも，個人と対話を

②包括的なドキュメントよりも，動くソフトウェアを

③契約交渉よりも，顧客との協調を

④計画に従うことよりも，変化への対応を

すなわち，①～④の各文の前者(「よりも」の前の言葉)に価値があることを認めながらも，私たちは後者(「よりも」の後の言葉)の事柄により価値をおく．

アジャイル宣言(Agile Manifesto)
アジャイルな開発手法の提唱者17名が集まり，2001年に発表．
http://agilemanifesto.org/iso/ja/manifesto.html

参考

アジャイル宣言の背後にある12の原則

SEC Software Reliability Enhancement Center

私たちは以下の原則に従う。

①顧客満足を最優先し、価値のあるソフトウェアを早く継続的に提供する。

②要求の変更はたとえ開発の後期であっても歓迎する。
変化を味方につけることによって、顧客の競争力を引き上げる。

③動くソフトウェアを、2-3週間から2-3ヶ月というできるだけ短い時間間隔でリリースする。

④ビジネス側の人と開発者は、プロジェクトを通して日々一緒に働く。

⑤意欲に満ちた人々を集めてプロジェクトを構成する。
環境と支援を与え仕事が無事終わるまで彼らを信頼する。

⑥情報を伝える最も効率的で効果的な方法は、フェイス・トゥ・フェイスで話をすることである。

⑦動くソフトウェアこそが進捗の最も重要な尺度である。

⑧アジャイル・プロセスは持続可能な開発を促進する。
一定のペースを継続的に維持できるようにしなければならない。

⑨技術的卓越性と優れた設計に対する不断の注意が機敏さを高める。

⑩シンプルさ(ムダなく作れる量を最大限にすること)が本質である。

⑪最良のアーキテクチャ・要求・設計は、自己組織的なチームから生み出される。

⑫チームがもっと効率を高めることができるかを定期的に振り返り、それに基づいて自分たちのやり方を最適に調整する。

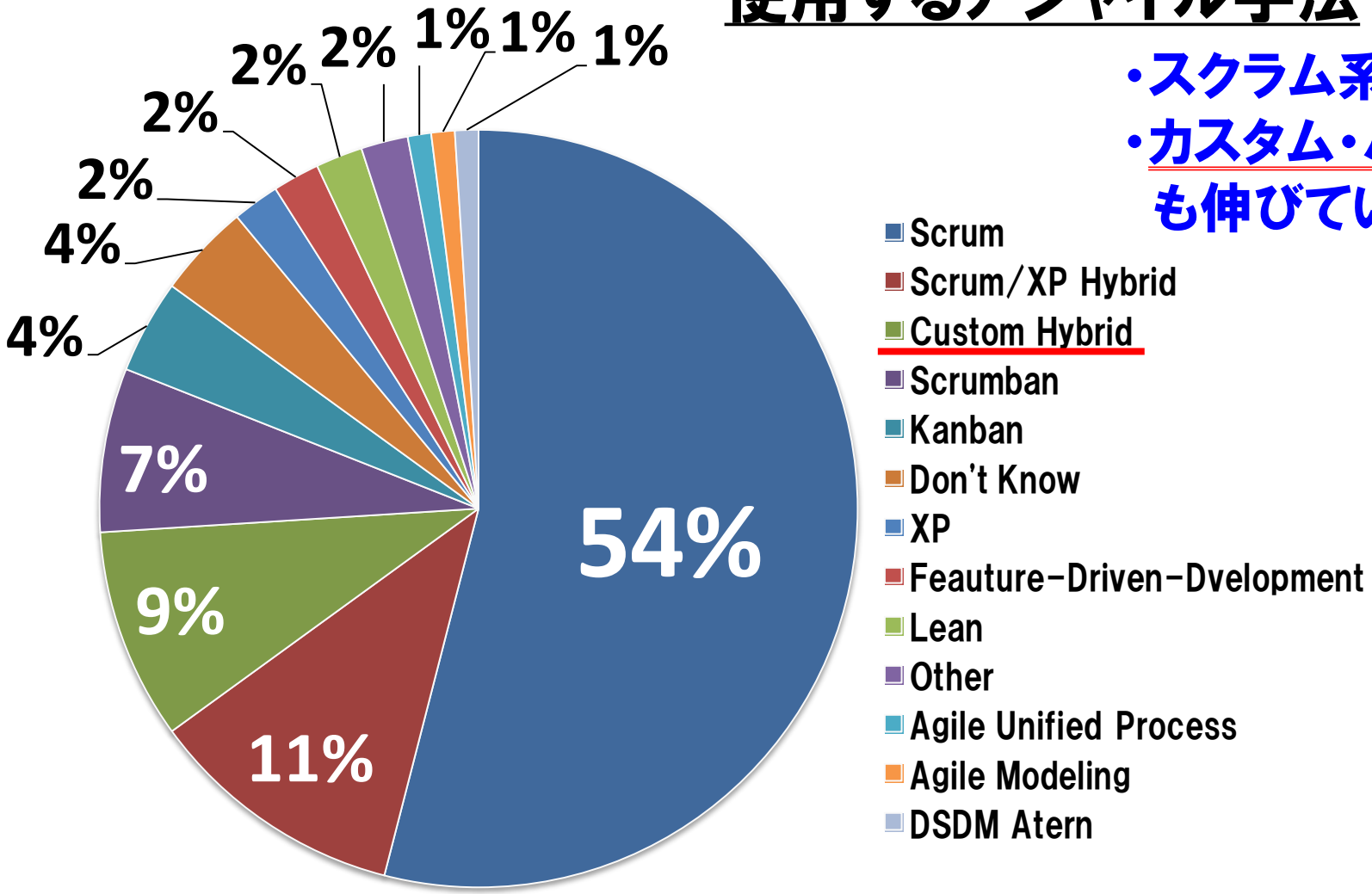
補足情報

参考

ハイブリッド型の適用が進む(１／２)

SEC Software Reliability Enhancement Center

使用するアジャイル手法



- スクラム系が多い
- カスタム・ハイブリッドも伸びている

VERSIONONE®: 7th ANNUAL STATE of AGILE DEVELOPMENT SURVEY, 2013
http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf

参考

ハイブリッド型の適用が進む（２／２）

SEC Software Reliability Enhancement Center

28 percent of 450 software professionals
said they use a hybrid approach.
Another 12 percent use lean software development,
which includes agile processes.

Source: 2011 Agile ALM and Testing Survey,
SearchSoftwareQuality.com

Of 4,770 respondents from 91 countries,
90 percent said they use some form of agile.
Only 27 percent of respondents solely use one type of agile,
while 35 percent mix agile with waterfall,
and 39 percent mix agile with Scrum.

Source: Analysis.Net and VersionOne

Source: PM NETWORK,
January 2012, Vol. 26, No. 1

アジャイル開発プラクティス活用リファレンスガイド 事例一覧（1）

SEC Software Reliability Enhancement Center

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#

アジャイル開発プラクティス活用リファレンスガイド 事例一覧 (2)

SEC Software Reliability Enhancement Center

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

アジャイル開発プラクティス活用リファレンスガイド プラクティス一覧（1）

SEC Software Reliability Enhancement Center

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

アジャイル開発プラクティス活用リファレンスガイド プラクティス一覧（2）

SEC Software Reliability Enhancement Center

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	継続的インデグレーション、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

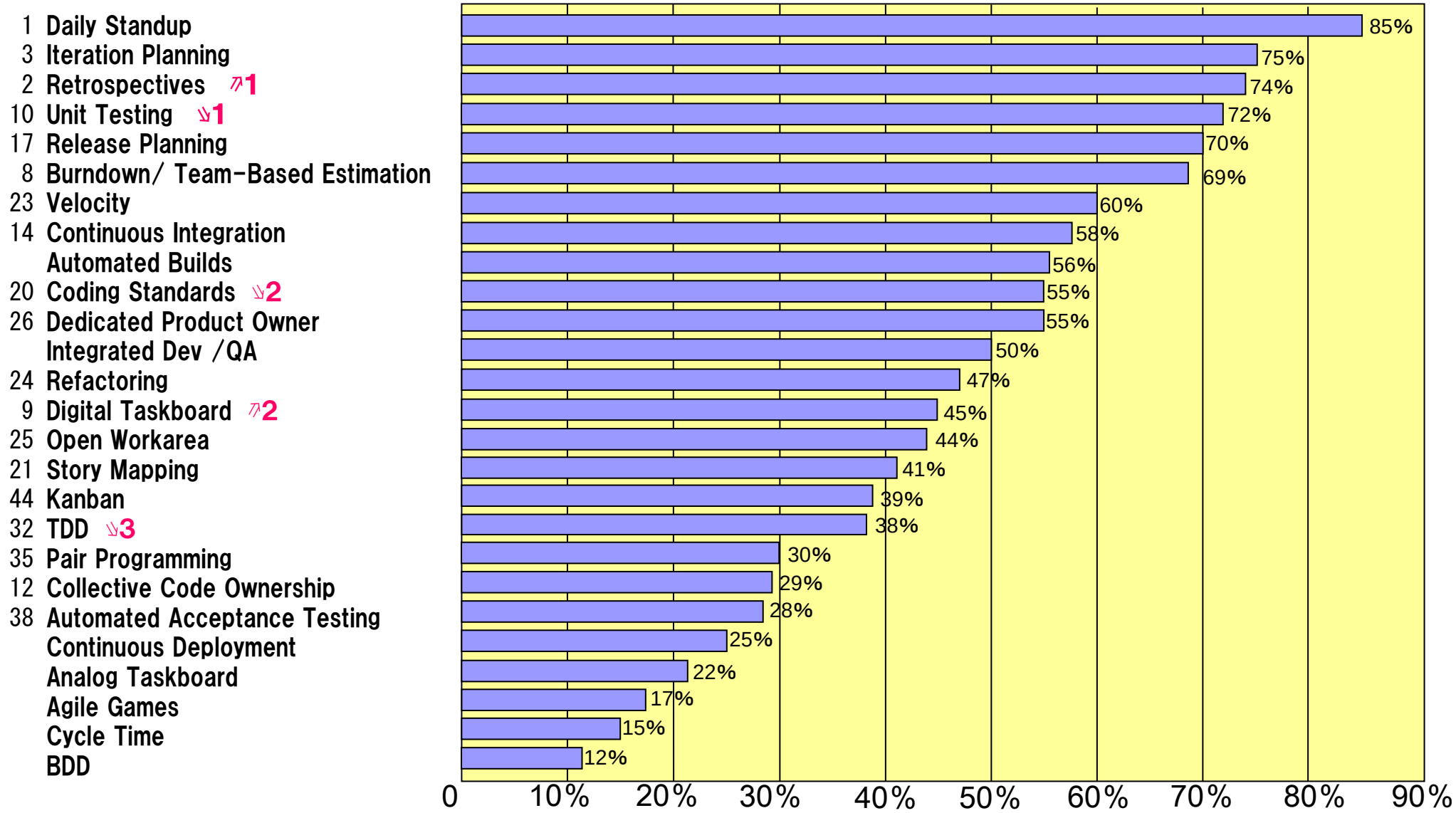
アジャイル開発プラクティス活用リファレンスガイド プラクティス一覧（3）

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

参考

調査事例：活用されているプラクティス（海外）

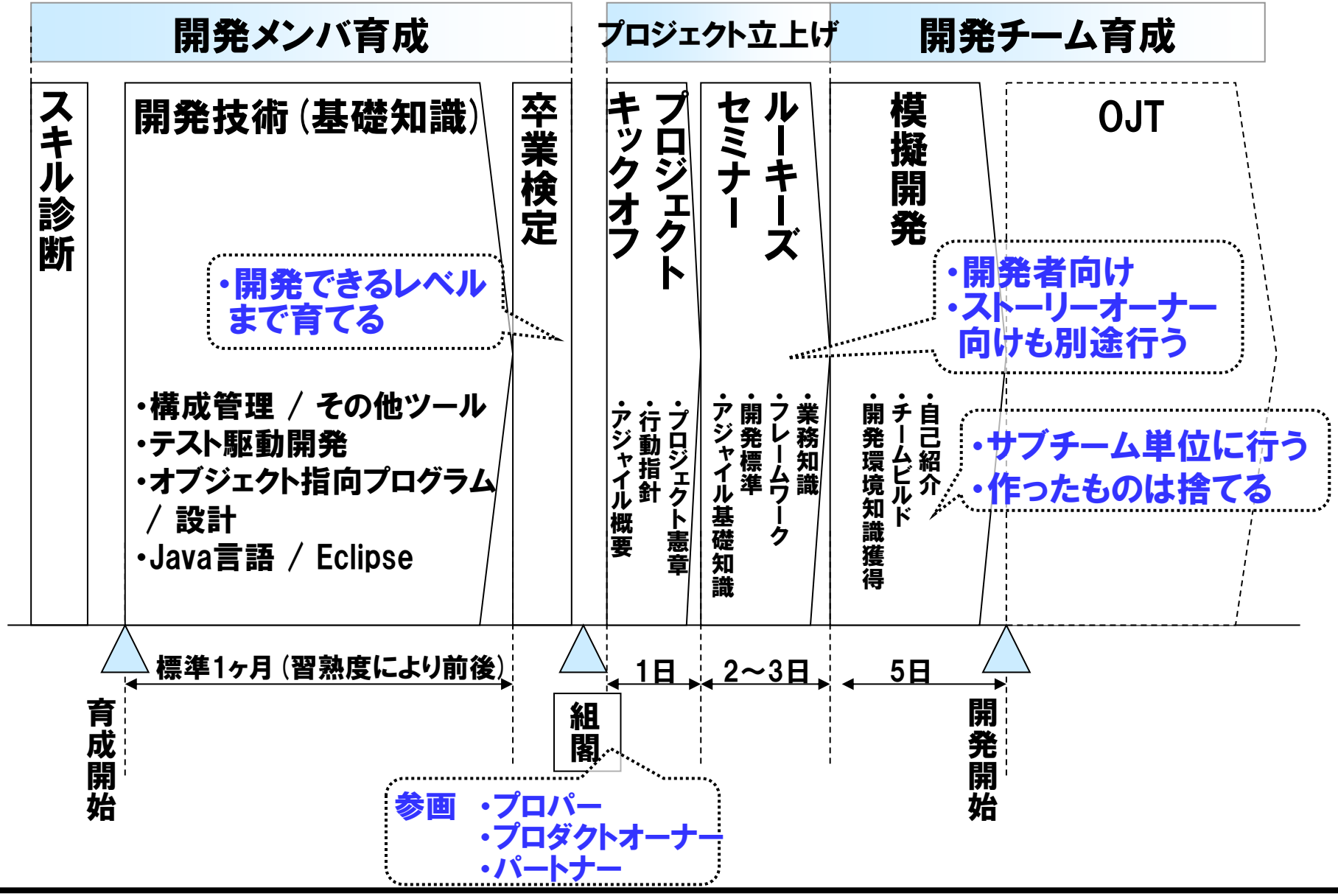
SEC Software Reliability Enhancement Center



↑ 左の数値は，日本の事例調査における順位

（VersionOne社　アジャイル開発の現状調査第8回2014より）

人材育成の事例－スケジュール



人材育成の事例－対象別育成カリキュラム例

SEC
Software Reliability Enhancement Center

	開発チーム	スクラムマスター	顧客／プロダクトオーナー	先行チーム	リーダー	PM	経営者層／購買担当など
アジャイル概要	○	○	○	○	○	○	○
アジャイル基礎知識	○	○	○	○	○	○	
アジャイル擬似体験	○	○	○				
業務知識	○		＊	○			
開発環境	○			＊			
基本アーキテクチャ	○			＊			
業務分析／モデリング	△		△				
開発技術	△			＊			
ファシリテーション概要	○	○	○	○	○	○	
ファシリテーション演習		○			○	○	

アジャイル開発を初めて行う組織を対象

○：立上げ前後の必須教育の領域
△：事前に準備が困難でOJTが必要な領域
＊：内容を組織内で個別に検討する必要がある領域

人材育成の事例－カリキュラム概要

SEC
Software Reliability Enhancement Center

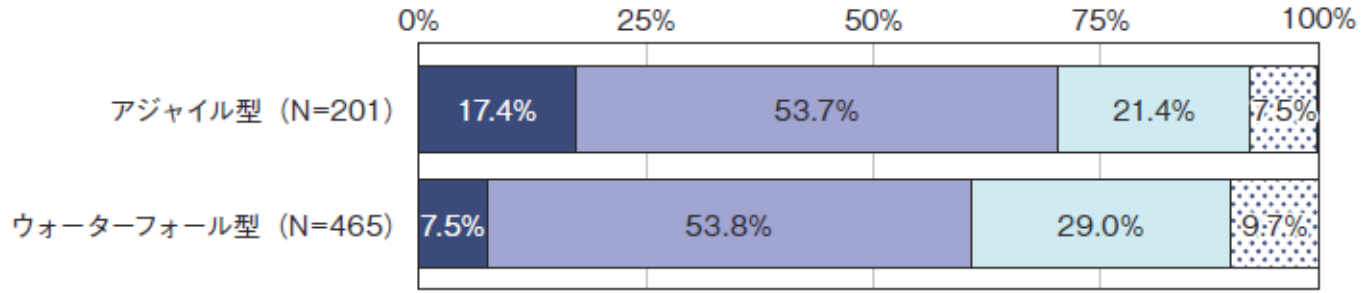
名称	概要
アジャイル概要	アジャイル開発に携わる方向けの基礎知識
アジャイル基礎知識	一般的なプラクティスについての紹介
アジャイル擬似体験	アジャイル開発のプロセスを体験を通して理解する チームビルディング的な狙いもある
業務知識	開発対象の業務を理解する（内容は先行チームと検討）
開発環境	開発に使用するツールなどを理解する（内容は先行チームと検討）
基本アーキテクチャ	開発対象のシステム構成や、利用するフレームワークなどを理解する（内容は先行チームと検討）
業務分析／モデリング	業務を整理し、開発側に伝えるための手法を理解する
開発技術	開発に必要な技術を身につける（必要に応じて）
ファシリテーション概要	ファシリテーションに関する知識を理解する
ファシリテーション演習	ファシリテーションに関する知識を体験を通して理解する

参考

アジャイル開発技術者の仕事に対する感じ方・考え方（1）

SEC Software Reliability Enhancement Center

仕事の内容（希望に合った仕事かどうか）



自分の仕事の成果に対する評価

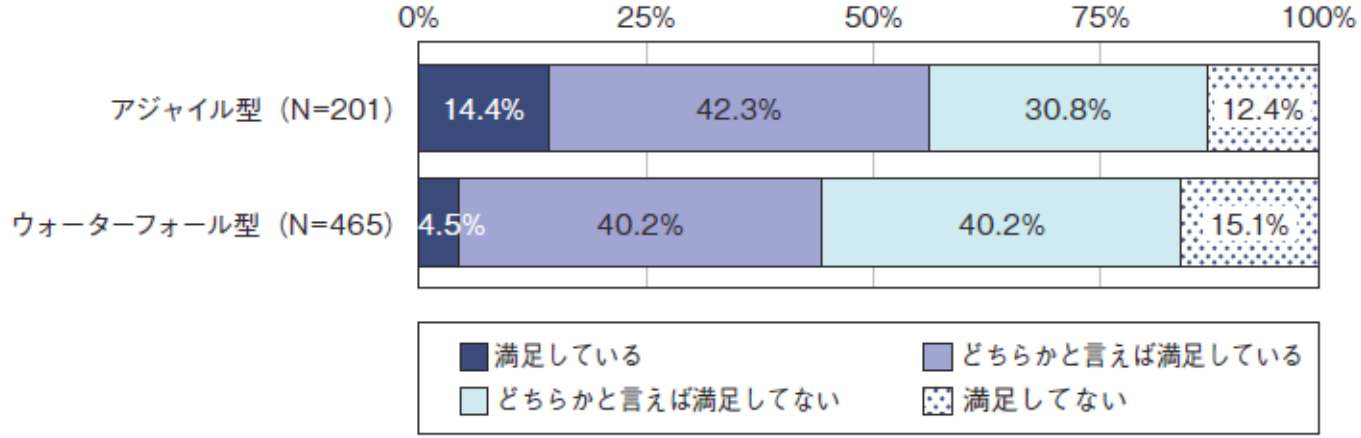


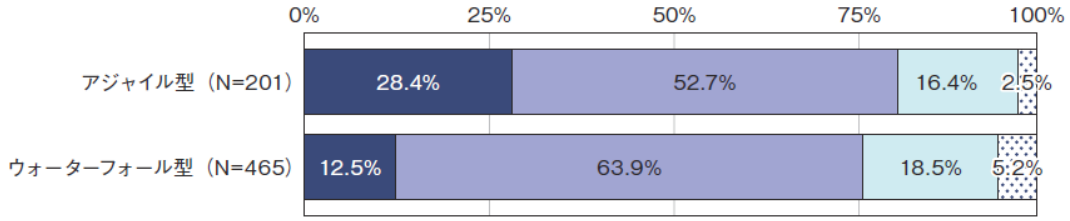
図３－82　開発プロセス別に見たIT人材の仕事に対する満足度の傾向

出典：「IT人材白書2013」, IPA, 2013/3/28.
http://www.ipa.go.jp/jinzai/jigyou/about.html

参考

アジャイル開発技術者の仕事に対する感じ方・考え方（2）

今の仕事に一生懸命取り組んでいる

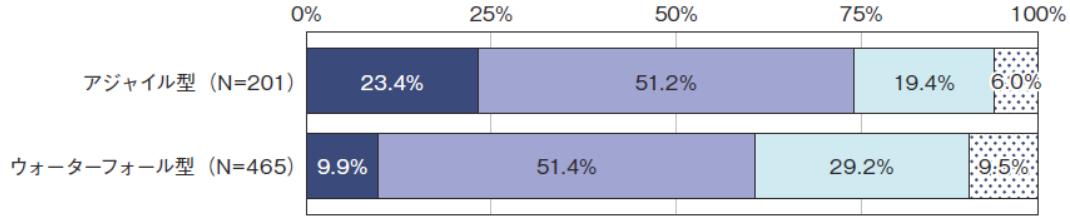


Q. 今の仕事に一生懸命取り組んでいる

アジャイル型　：28.4%

ウォーターフォール型：12.5%

仕事が好きである

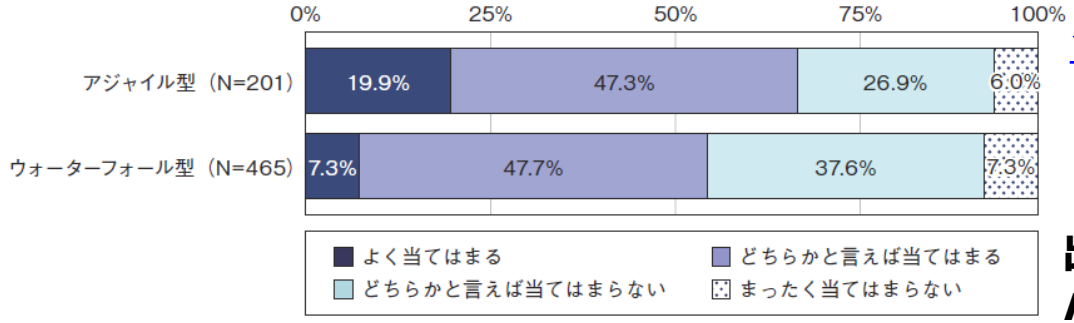


Q. 仕事が好きである

アジャイル型　：23.4%

ウォーターフォール型：　9.9%

この仕事をしていることを誇りに思う



Q. この仕事をしていることを誇りに思う

アジャイル型　：19.9%

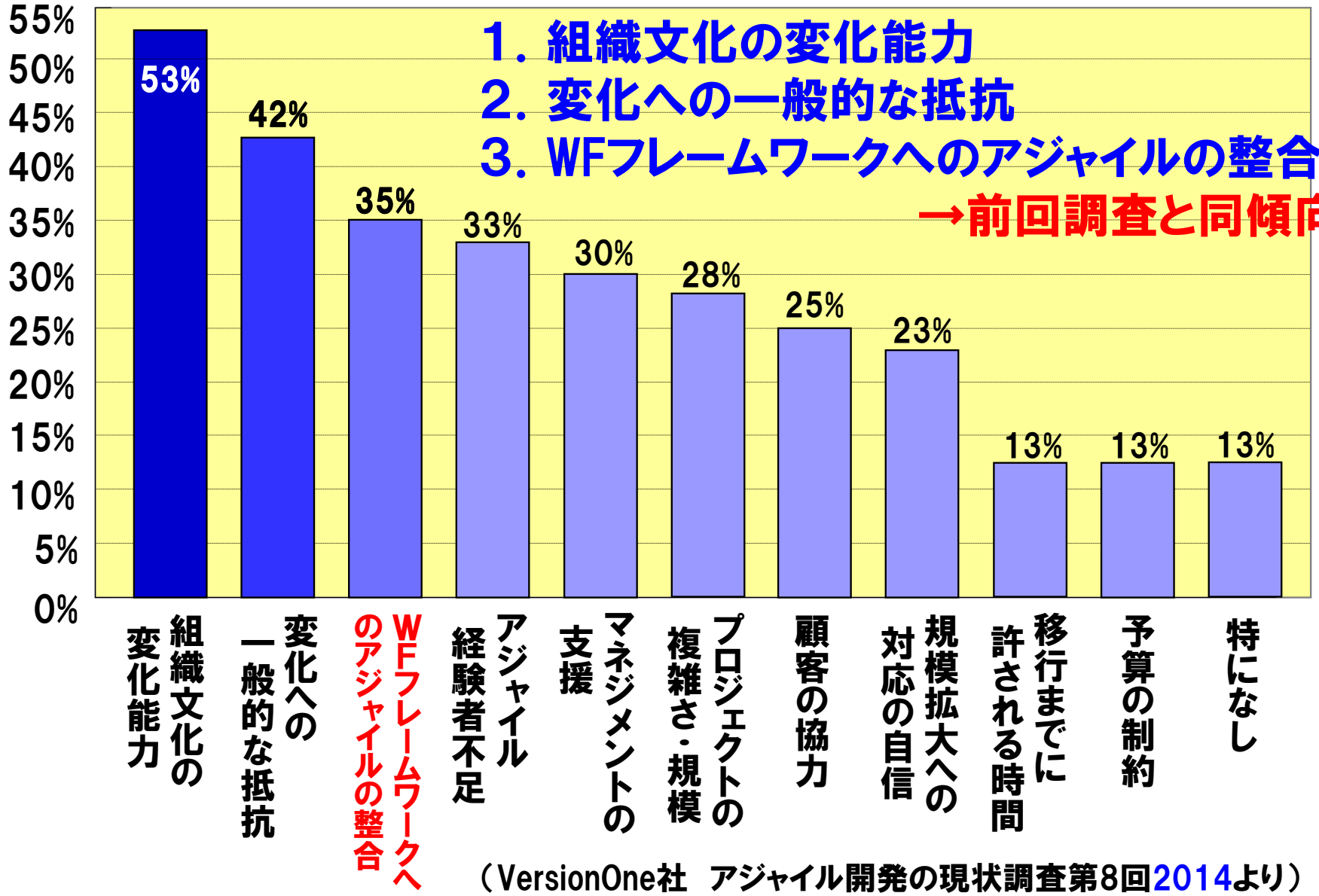
ウォーターフォール型：　7.3%

<回答=「よく当てはまる」の割合>

出典：「IT人材白書2013」, IPA, 2013/3/28.
http://www.ipa.go.jp/jinzai/jigyou/about.html

参考

アジャイル型開発手法の導入拡大の障壁（海外）



（VersionOne社　アジャイル開発の現状調査第8回2014より）

WF型開発と比較したアジャイル開発の特徴

SEC
Software Reliability Enhancement Center

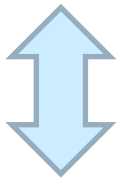
- ユーザとベンダの**緊密な協力体制**が必須
  - 相手方の問合せへの迅速な応答
  - 担当作業の迅速な実施
  - ユーザ/ベンダ間の責任分担が不明確になりがち
- ユーザ要求の詳細が**契約時点では未確定**
  - 何を作るか決まっていない(成果物未定)
  - 性能・品質等が不明確
  - 工数見積りが困難(コスト未定)
- 開発途中での**ユーザ要求の変化**を柔軟に受け入れる必要
  - 決定した事項も変更されることがある

契約

アジャイル開発と契約の対立

SEC
Software Reliability Enhancement Center

■ **契約**
合意内容を固定して、当事者を法的に拘束する



■ **アジャイル開発**
変化に対応すべく、合意内容の変更を柔軟に認め、当事者をなるべく拘束しない

⇒アジャイル開発にはふさわしい契約とは？

契約

システム開発において主に使われる契約類型

SEC
Software Reliability Enhancement Center

請負契約(民法632条～642条)

一方が仕事を完成させることを請負い、その相手方が完成した仕事に対して報酬を支払うことを約束する契約。

⇒契約時点で、ベンダが完成すべき仕事の内容を明確にしておく必要。

準委任契約(民法643条～656条)

事務処理を目的とする契約であり、仕事の完成を目的としない。ベンダは、善良な管理者の注意をもって、委任された事務を処理する義務(善良管理者注意義務)を負う。

⇒ベンダに完成義務がなく、ユーザにとって不安。

契約

参考

アジャイル開発の推進に向けて

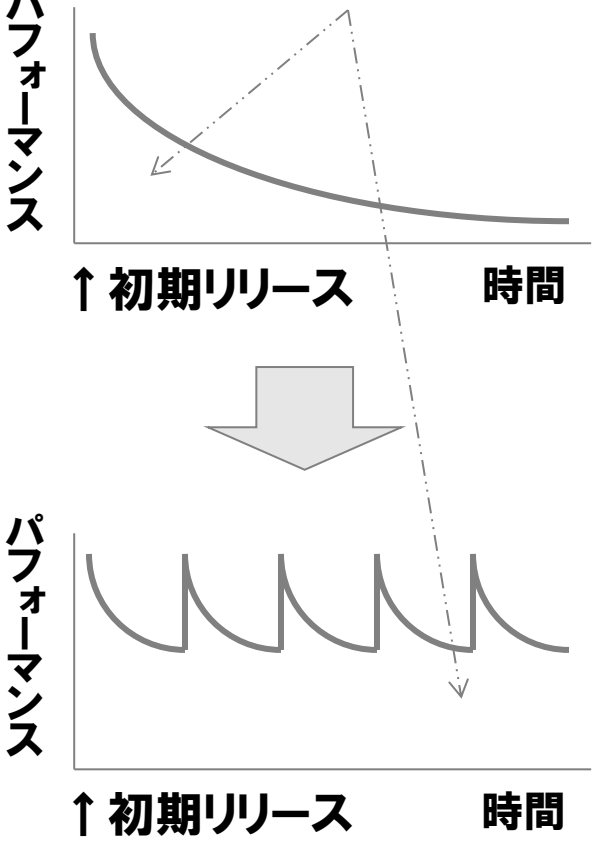
少し触れます

トップダウン：ビジョン

ボトムアップ：トレーニング

本日の主テーマ

「価値」（スループット）は，面積で表される



- コミュニケーションが最も重要
- 変化に対する抵抗が最大の障害

<出典>
河合太郎（ヤフー）：大規模開発におけるリーン・スタートアップ，Innovate 2013, 2013.10.28.

継続的デリバリーでスループットを大きく

開発手法とマインド（私見）

SEC Software Reliability Enhancement Center

＜アジャイル開発＞再掲

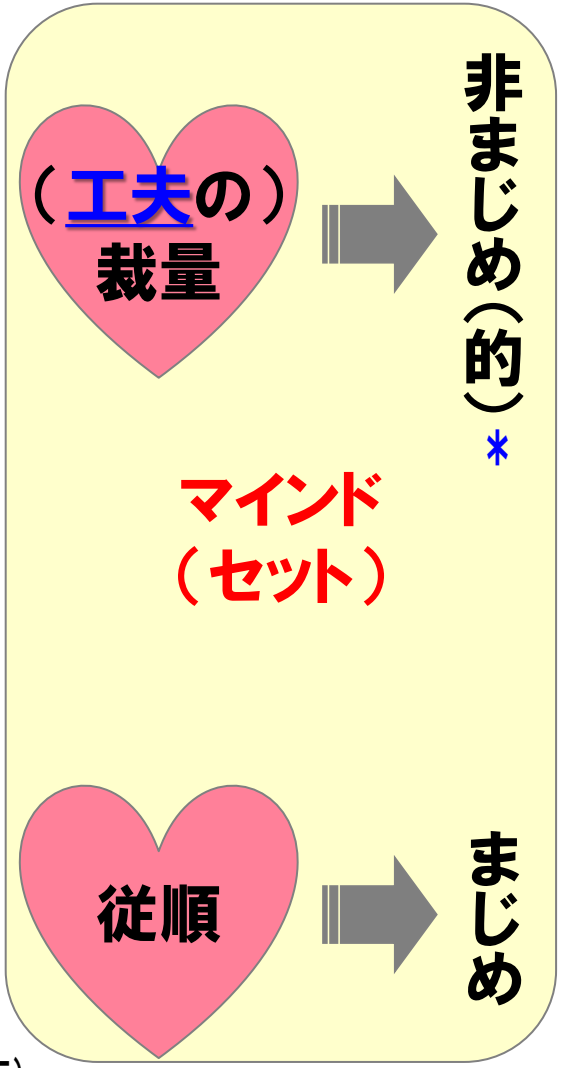
- 一つのプロジェクトで全てのプラクティスを使う訳ではない
- （今のところ）各プラクティスに厳格な規範はない
- 様々な方法論・数あるプラクティスから，プロジェクトや組織に適したものを取捨選択し，カスタマイズすることが必要

＜ハイブリッド型開発＞

文化的相違

＜ウォーターフォール型開発＞

- 開発標準・作業標準



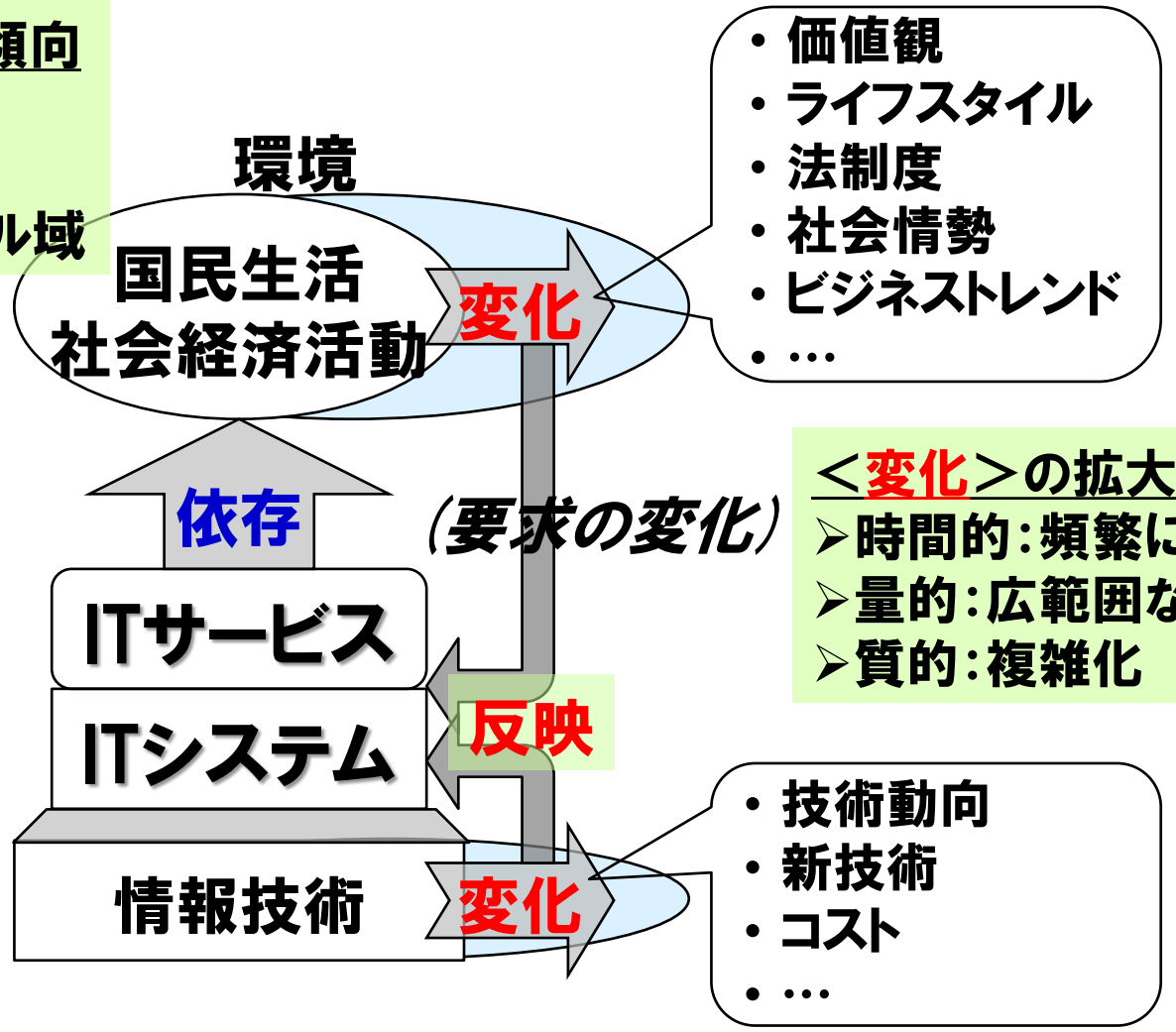
＊『非まじめ』のすすめ（森政弘著，講談社文庫，1977年）

ビジネス動向とアジャイル開発

ITシステムへの依存と，システム環境の変化

SEC
Software Reliability Enhancement Center

＜依存＞の拡大傾向
- ➢時間的：常時化
- ➢量的：広範囲化
- ➢質的：クリティカル域



＜変化＞の拡大傾向
- ➢時間的：頻繁に
- ➢量的：広範囲な影響
- ➢質的：複雑化

安心 ↑ 変化の俊敏な反映

安全 ↑ 高信頼化

⇒このような傾向を考慮した，ITシステムの開発・運用

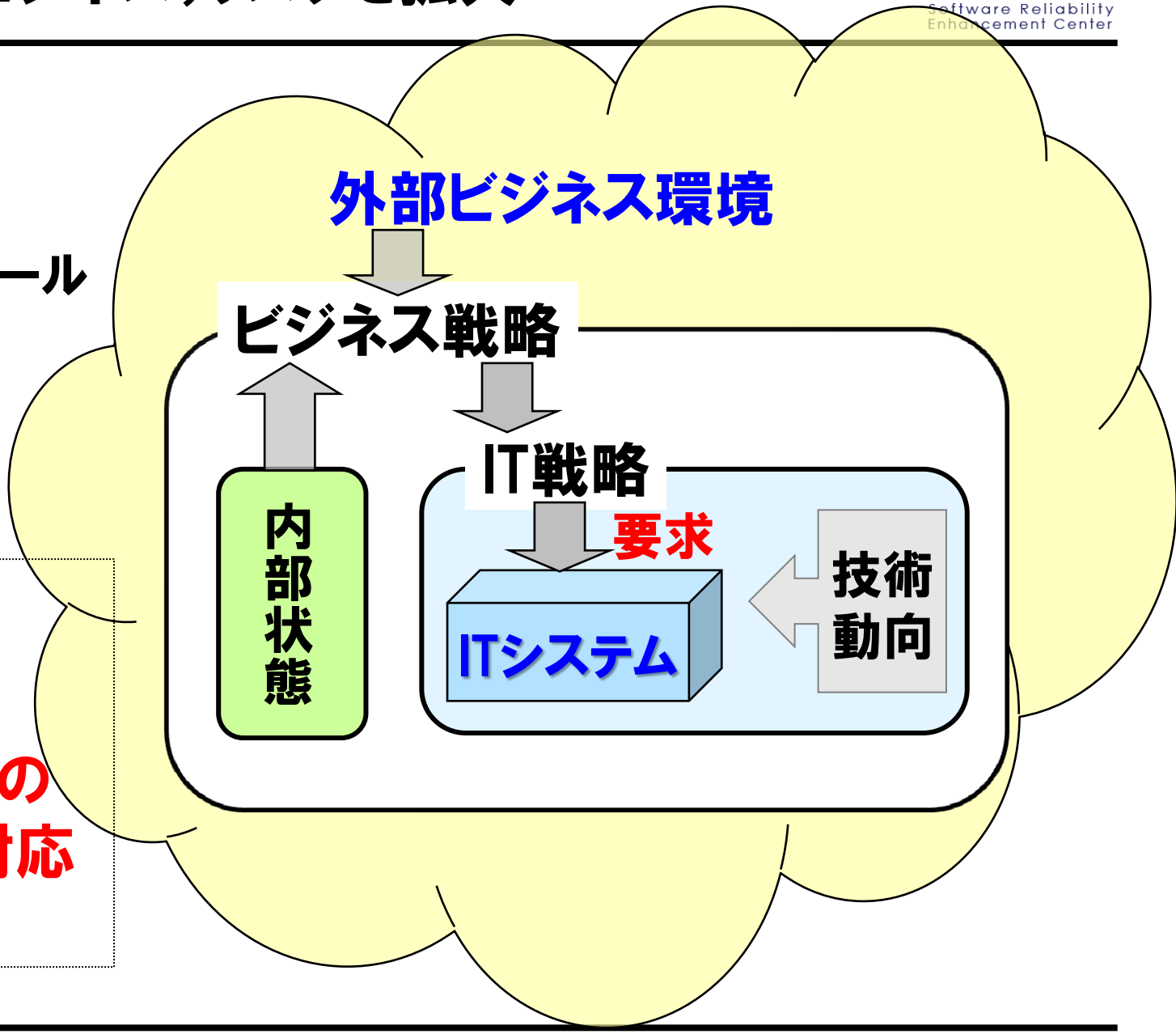
要求の固定が(ビジネス)リスクを拡大

SEC Software Reliability Enhancement Center

要求が固定されない

↓リスク

システム開発スケジュールの遅延



要求を固定化

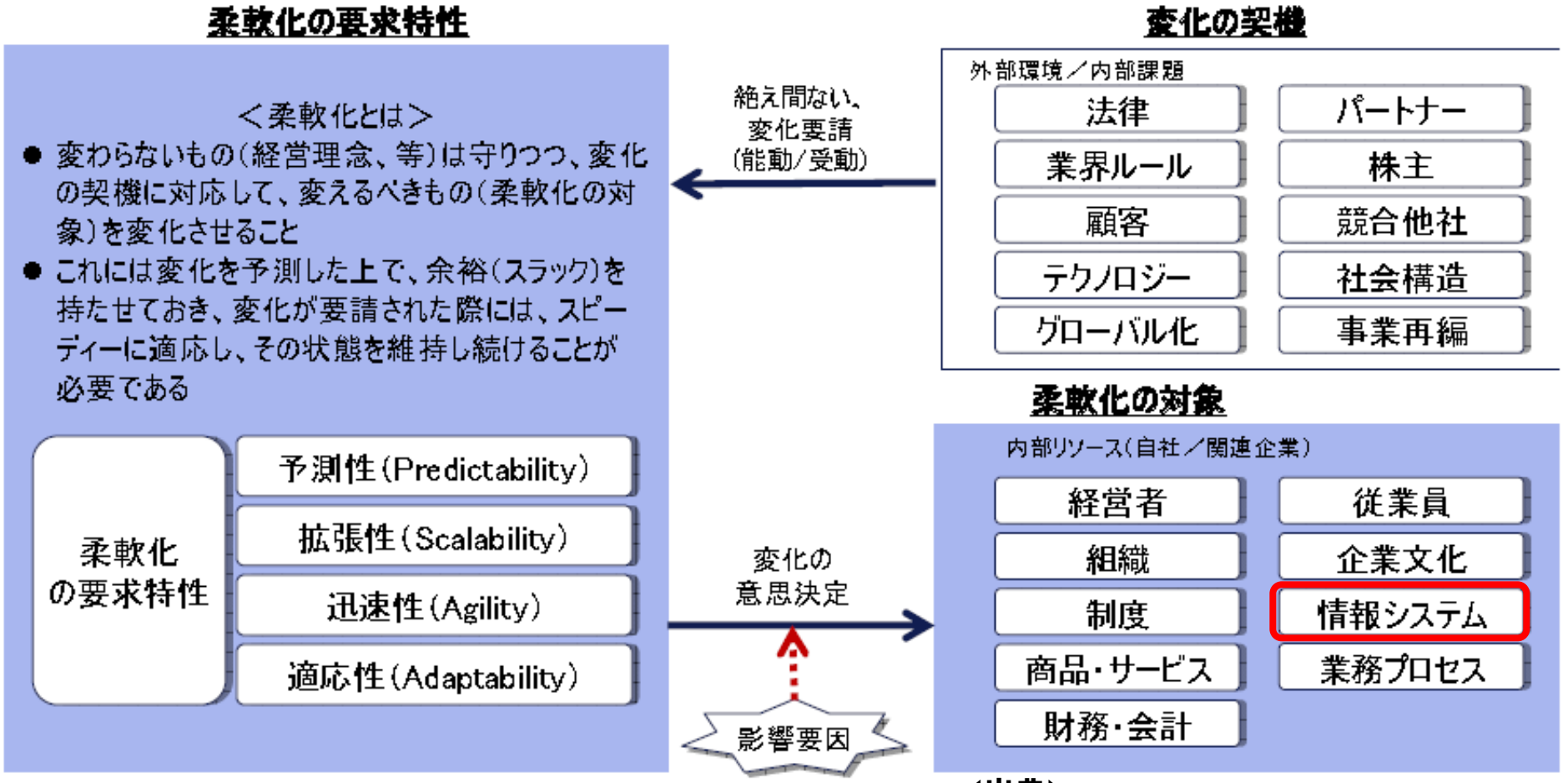
↓リスク

外部ビジネス環境の変化への迅速な対応の遅れ

参考

環境変化に即応できるための経営の「柔軟性」

SEC Software Reliability Enhancement Center

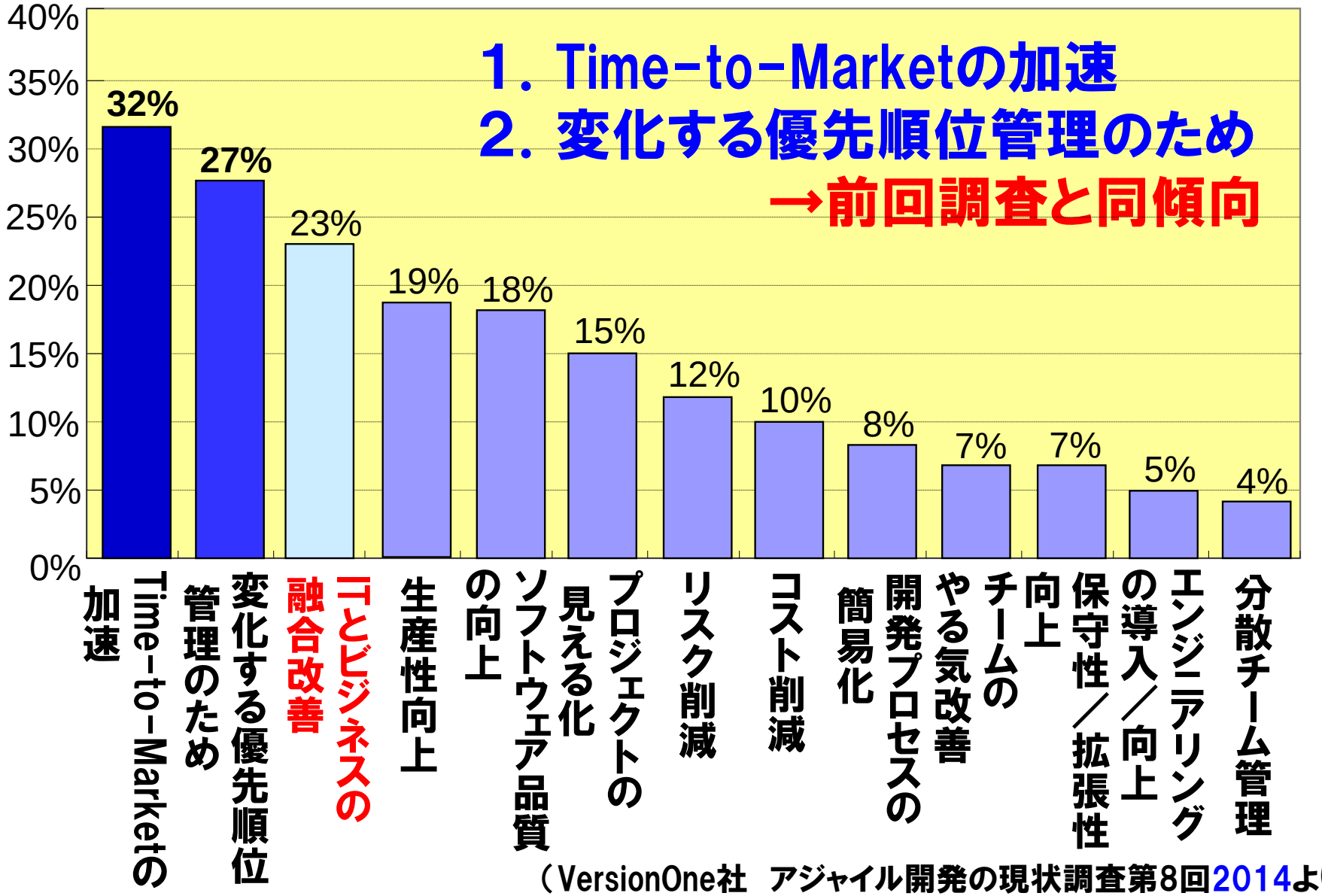


予測性：可変要素が将来変化をする予兆を事前にとらえること
拡張性：既存のリソース（人，モノ，カネ，情報等）に将来の可変要素を想定した余裕を持たせておくこと
迅速性：起きた変化／起こすべき変化に対して，すぐに対応できること
適用性：これまでと違った環境、シチュエーションに，うまく対応できること

＜出典＞
平成22年度経済産業省委託調査：「IT 経営普及促進に向けた調査研究」報告書，社団法人日本情報システム・ユーザー協会，平成23年2月，p.76.
http://www.meti.go.jp/meti_lib/report/2011fy/0022948.pdf

参考

アジャイル型開発手法の導入理由（海外_2014年）



（VersionOne社　アジャイル開発の現状調査第8回2014より）

参考

パフォーマンスの高い組織ほど，俊敏性に重きを置く

SEC Software Reliability Enhancement Center

High-performing organizations are more likely than their low-performing counterparts to focus on agility

そして，そのことが，
最優先業務の成功に好影響を及ぼし，
プロジェクトのパフォーマンス向上を導く

An organization's focus on agility and strategic alignment not only impacts the success of its highest priority initiatives, it also leads to better project performance overall; 89 percent of projects at high-performing organizations meet original goals and business intent, compared with just 36 percent at low-performing organizations. And high-performing organizations lose 12 times less money than low performers (US$20 million versus US$230 million for every US$1 billion spent on projects).

<出典> *PMI's 2014 Pulse of the Profession®*
http://www.pmi.org/Knowledge-Center/Pulse.aspx#

参考

組織の俊敏性とは？ What Defines Organizational Agility?

SEC Software Reliability Enhancement Center

Here's how respondents described the practices or characteristics of organizational agility:

素早い応答

75%
Quick response to strategic opportunities

短サイクル

64%
Shorter decision/ production/review cycles

変更管理

59%
Focus on change management

顧客の声を聞く

54%
Integrating voice of the customer

リスク管理

53%
Focus on risk management

多様なチーム

53%
Interdisciplinary project teams

サイロ化防止

53%
Elimination of organization silos

非常事対策

51%
Contingency planning

反復プロセス

50%
Use of iterative project management practices

技術の活用

46%
Leveraging technology

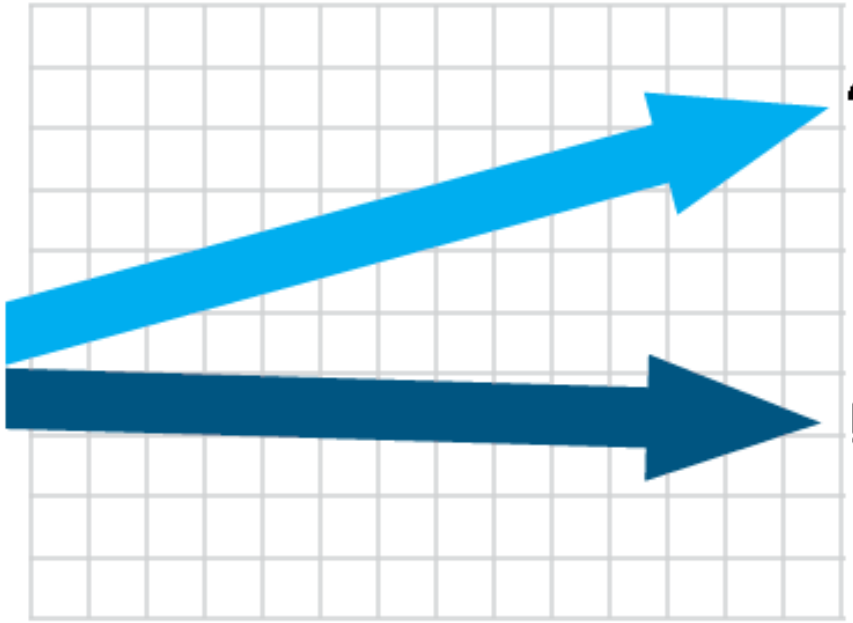
<出典> *PMI Pulse of the Profession In-Depth Report: Organizational Agility, 2012*

参考

組織の俊敏性が大きいほど，パフォーマンスはよくなる

SEC Software Reliability Enhancement Center

Success with new initiatives over the past 2-3 years



Greater organizational agility leads to better performance—providing organizations with a powerful edge on the competition.

成功率が増大している組織ほど，高い俊敏性を有する

Those organizations that are successful report higher levels of organizational agility—giving them a powerful edge on the competition.

	HIGH AGILITY	LOW AGILITY
Increased Success	60%	27%

<出典> *PMI Pulse of the Profession In-Depth Report: Organizational Agility, 2012*

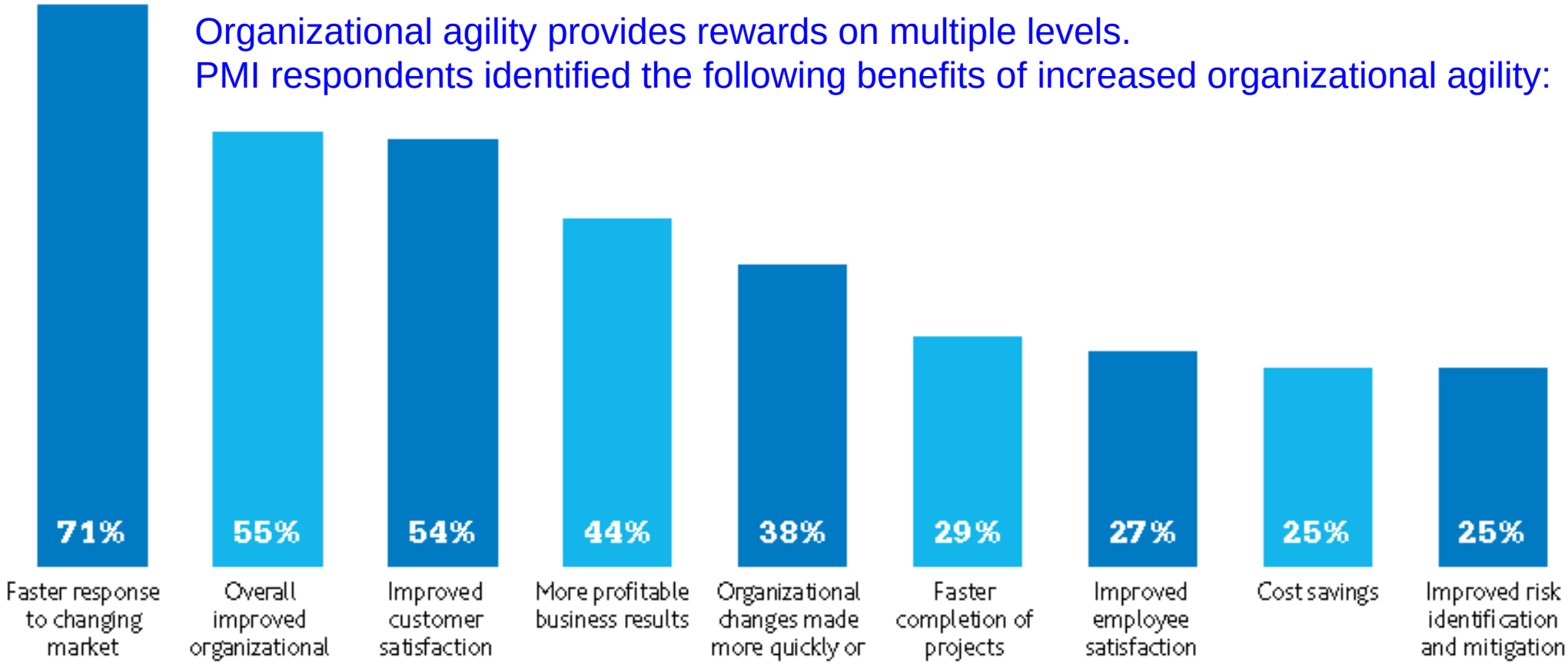
参考

組織の俊敏性が増大することによる利点

SEC Software Reliability Enhancement Center

Benefits of increased organizational agility

Organizational agility provides rewards on multiple levels.
PMI respondents identified the following benefits of increased organizational agility:



顧客満足向上
早期プロジェクト完了
リスク低減
組織の効率向上
迅速で効率的な組織変化
コスト削減
変化への迅速な対応
ビジネス増益
従業員満足向上

<出典> *PMI Pulse of the Profession In-Depth Report: Organizational Agility, 2012*

参考

標準化されたプラクティスを有する組織の俊敏性大

SEC Software Reliability Enhancement Center

Organizations that use standardized practices are more agile.

	Standardized practices through all departments	Standardized practices not through all departments
High Agility	22%	7%
Moderate Agility	60%	51%
Low Agility	18%	42%

26% Apply iterative project management concepts to portfolio management
23% Iterative techniques*
22% Project task simplification*
22% Apply iterative project management concepts to program management
18% Portfolio management*
18% Interdisciplinary project teams*
17% Integrate voice of the customer*
17% Risk management*
17% Change management*
17% Resource management*
17% Change process used within and outside of projects
16% Program management*
15% Risk process used within and outside of projects
14% Formal risk process
14% Formal change process
12% Study Average

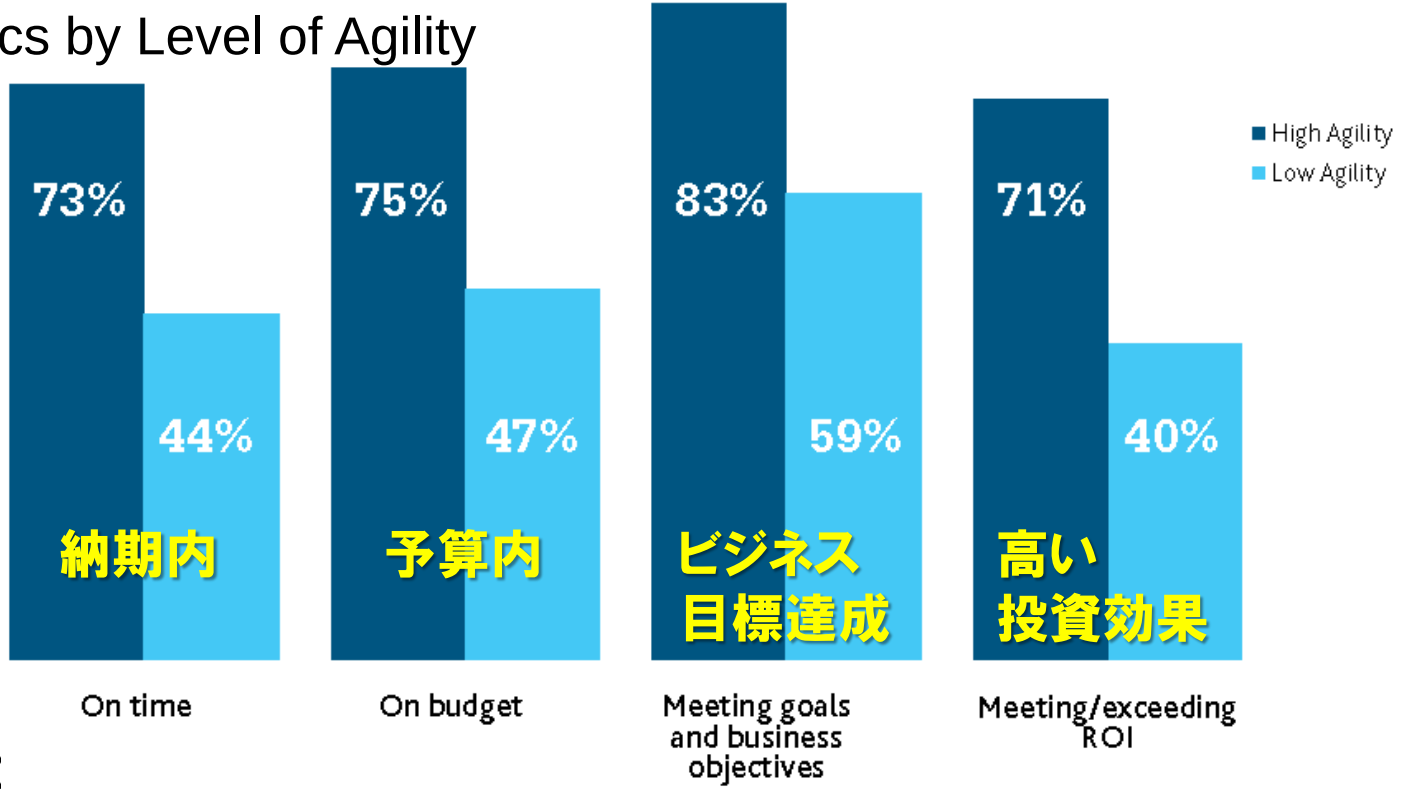
<出典> *PMI Pulse of the Profession In-Depth Report: Organizational Agility, 2012*

参考

組織の俊敏性の度合いとプロジェクト成功率

SEC Software Reliability Enhancement Center

Project Success Metrics by Level of Agility



The End Result

高いパフォーマンス

Greater Organizational Agility = Better Performance = Improved Competitive Advantage

俊敏性の大きな組織　　競争力強化

<出典> *PMI Pulse of the Profession In-Depth Report: Organizational Agility, 2012*

組込み機器・システムとアジャイル開発

再掲

アジャイル開発の試行領域

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。

組込み製品開発の特徴

顧客(エンドユーザ)が見えない

- 経験と想像に基づく要求設定
- 未来予測の要求への反映
- 多くの関係者(関連部署)との協力による開発推進
- 市場からのフィードバックを迅速に行う仕組み

短いサイクルで機能を積み上げ，評価しつつ，
製品の価値を高めていく

参考

ビジネス・ステージと開発手法

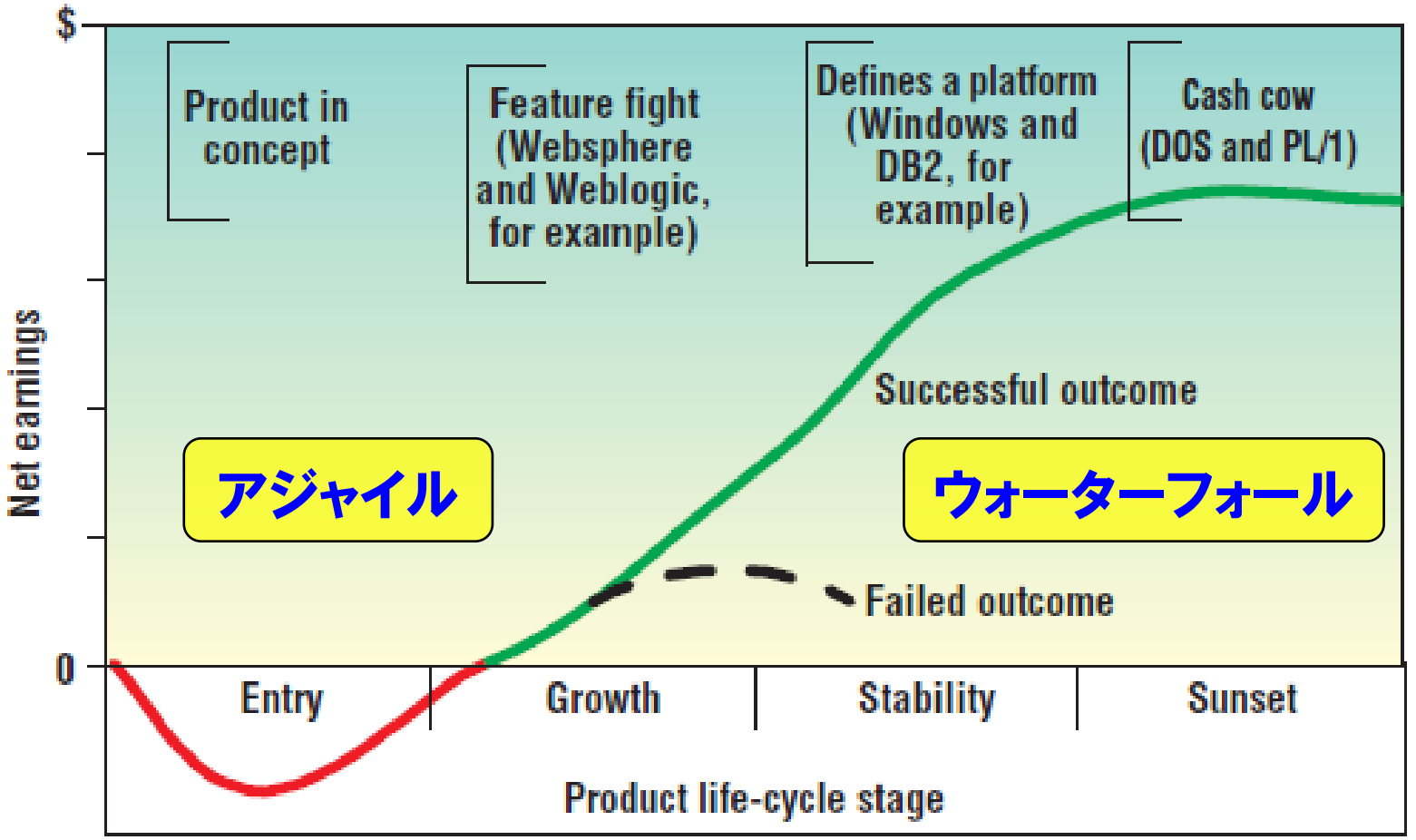


Figure 1. A financial model of software product development.

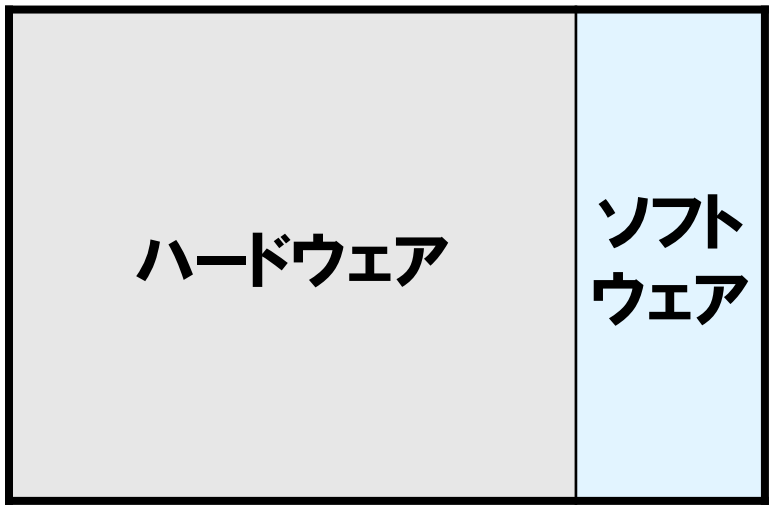
ソフトウェア製品のライフサイクル・モデル例と開発手法

<出典> Ram Chillarege: The Marriage of Business Dynamics and Software Engineering, IEEE SOFTWARE, November/December 2002.

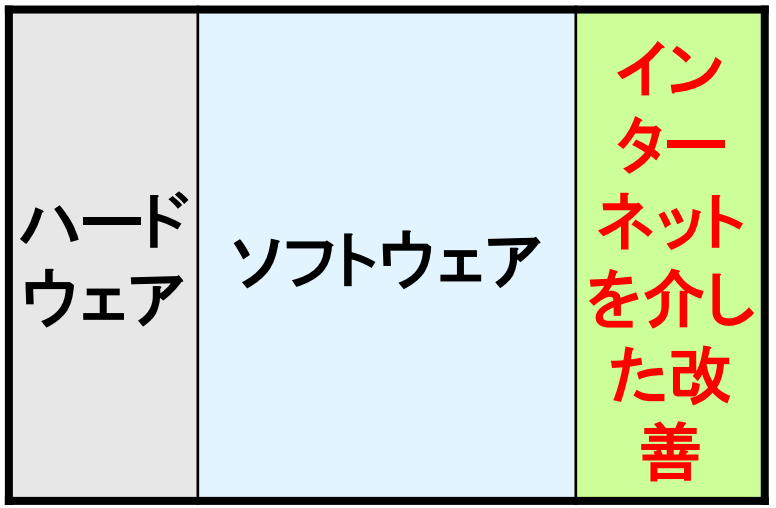
日米における今日のデバイスの比較

SEC
Software Reliability Enhancement Center

日本



米国(シリコンバレー)



＜出典＞
クリストファー・テイト「イノベーションを生み出す日本へ、再び
～ソフトウェアとハードウェアの対話が、日本に強さもたらす～」
ET-West 2013 ヒートアップセッションHU-5講演，2013年6月14日，大阪.

組込み機器・システムとアジャイル開発

SEC
Software Reliability Enhancement Center

利用者の購入後に
インターネットを介して
定期的に機能拡張＊

クラウド

アジャイル開発

エンドユーザ側
データの収集の
仕組みがある

真の要求

＊ エンドユーザからのフィードバック対応を含む

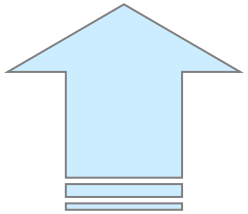
組込み系とエンタプライズ系の技術者間の協働

SEC
Software Reliability Enhancement Center

機器・システムだけを見ていては不十分
（イノベーションに結びつきにくい）

エンドユーザ側データの収集の仕組みと運用

- **機能拡張の仕組み（サーバ／バックヤード／クラウド側）の理解**
- **機能拡張項目選定のトリガ（利用者の声を捉える仕組み）の理解**



組込み系とエンタプライズ系の協働／両スキルの獲得