Java, Ruby そして Agile

株式会社永和システムマネジメント
株式会社チェンジビジョン
平鍋健児

Ruby
Agile

概要

- 「Ruby on Rails」で Web 開発に衝撃を与えたRuby。そして、すばやい繰り返し開発をテストで駆動するAgile。JavaとRubyの違いは何か。双方の優位性は？そして、Agile開発との関係は？これまで10年のソフトウェア開発の歴史を踏まえ、2007年から2010年のソフトウェア開発の現場で起こるパラダイムシフトについての予測します。

Ruby
Agile

自己紹介

- 株式会社永和システムマネジメント
  - 金融･医療･オブジェクト指向技術を使ったシステム開発、教育、コンサルティング
  - オブジェクト倶楽部
  - アジャイル開発の知識と実践
  - Ruby 開発者の数(30人)と実績(13プロジェクト)
- 株式会社チェンジビジョン
  - 本社は東京都新宿、JUDE開発部は福井
  - JUDE と TRICHORD で見える化
- 平鍋健児
  - リアルタイム, CAD、オブジェクト指向の実践
  - UMLエディタJUDEの開発
  - オブジェクト倶楽部主宰
  - アジャイルプロセス協議会、副会長
  - 翻訳、XP関連書籍、『リーンソフトウェア開発』、『アジャイルプロジェクトマネジメント』

Ruby
Agile

アジェンダ

- ビジネス背景（繰り返せ！漸進的に！）
- Agile開発の本質
- Java と Ruby
- まとめ

Ruby
Agile

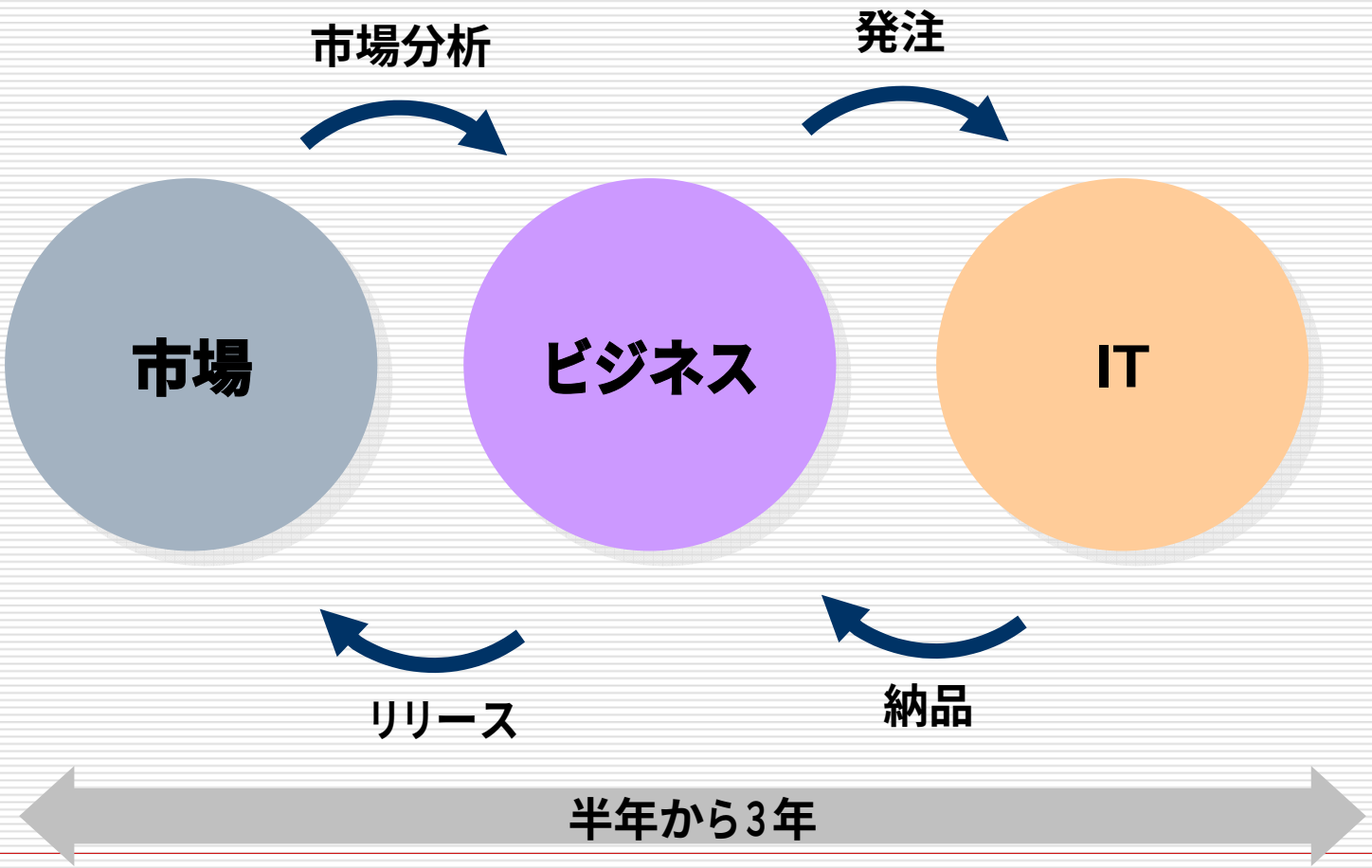
ビジネス背景 (繰り返せ！漸進的に！)

繰り返す = Iterative

漸進的に = Incremental

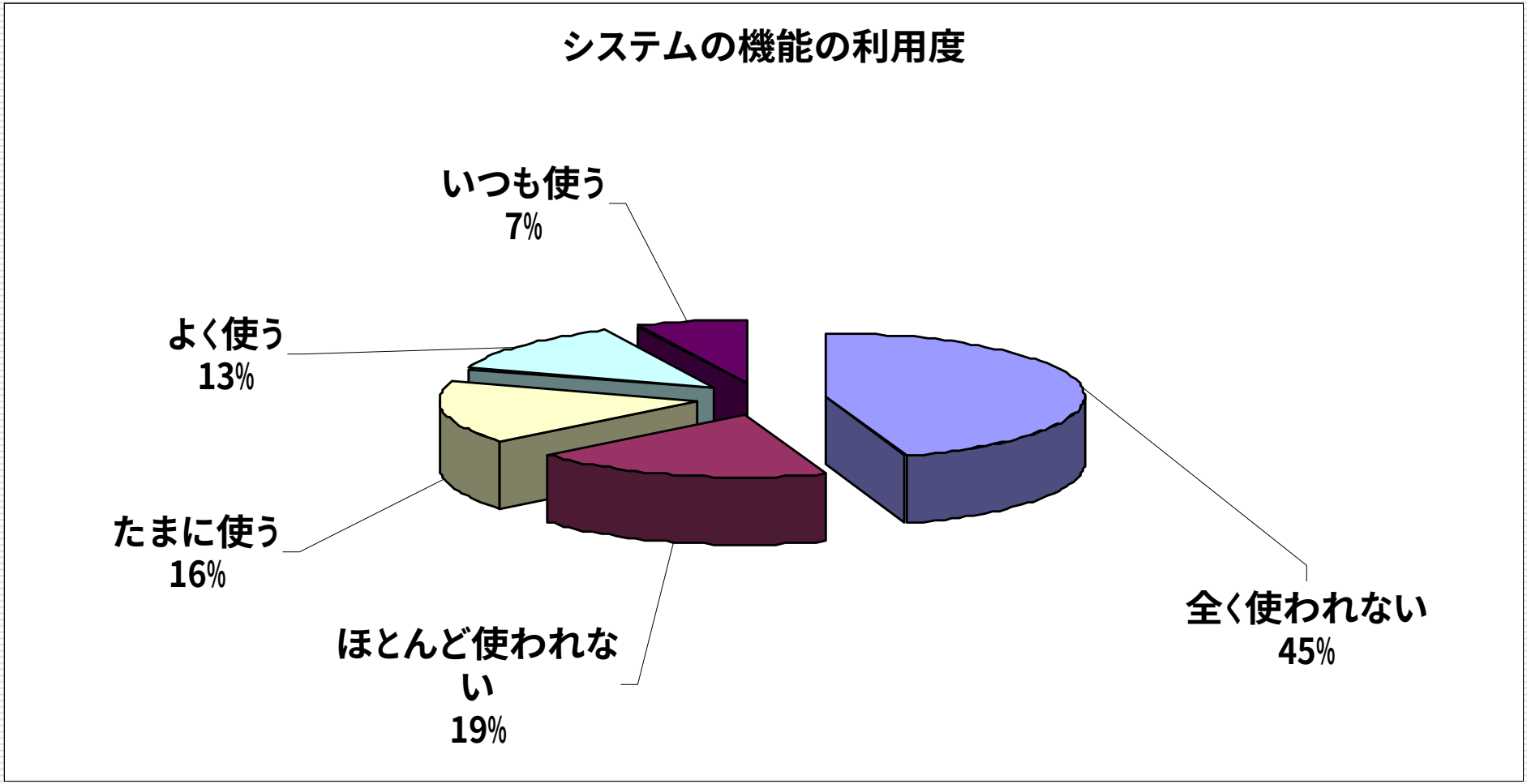
Ruby Agile

1 ミッション・リスク分割型ビジネスとウォーターフォール型開発



Ruby
Agile

要求の劣化

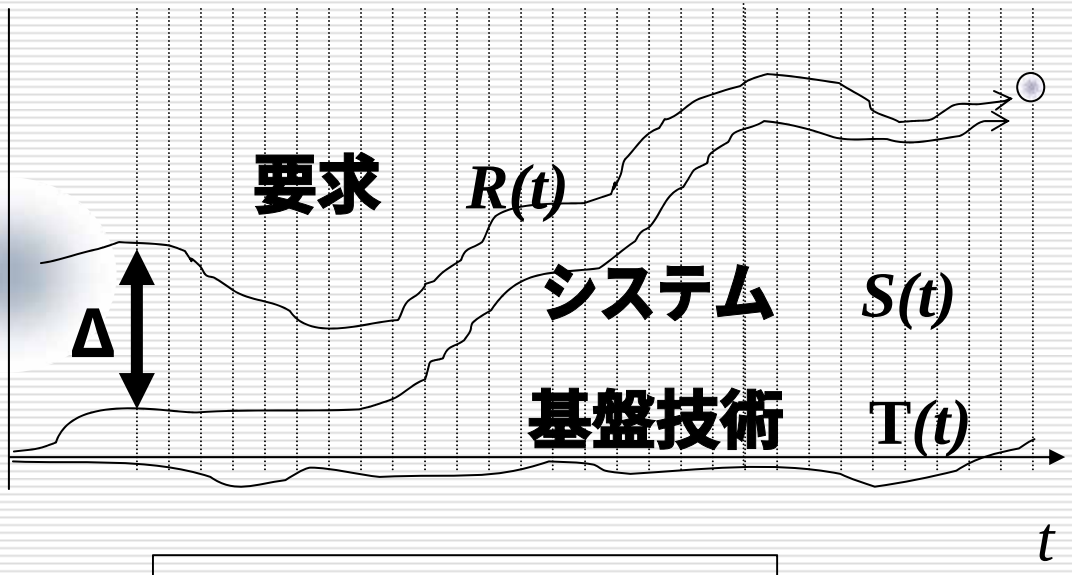


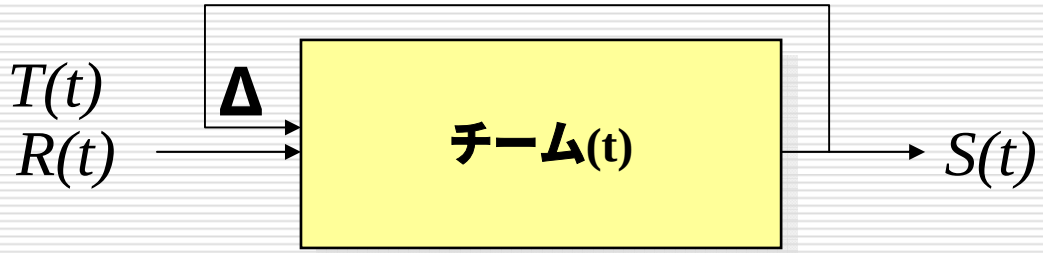
Standish group study report in 2000 chaos report

プロジェクトの成功は、
Moving Target

Ruby Agile

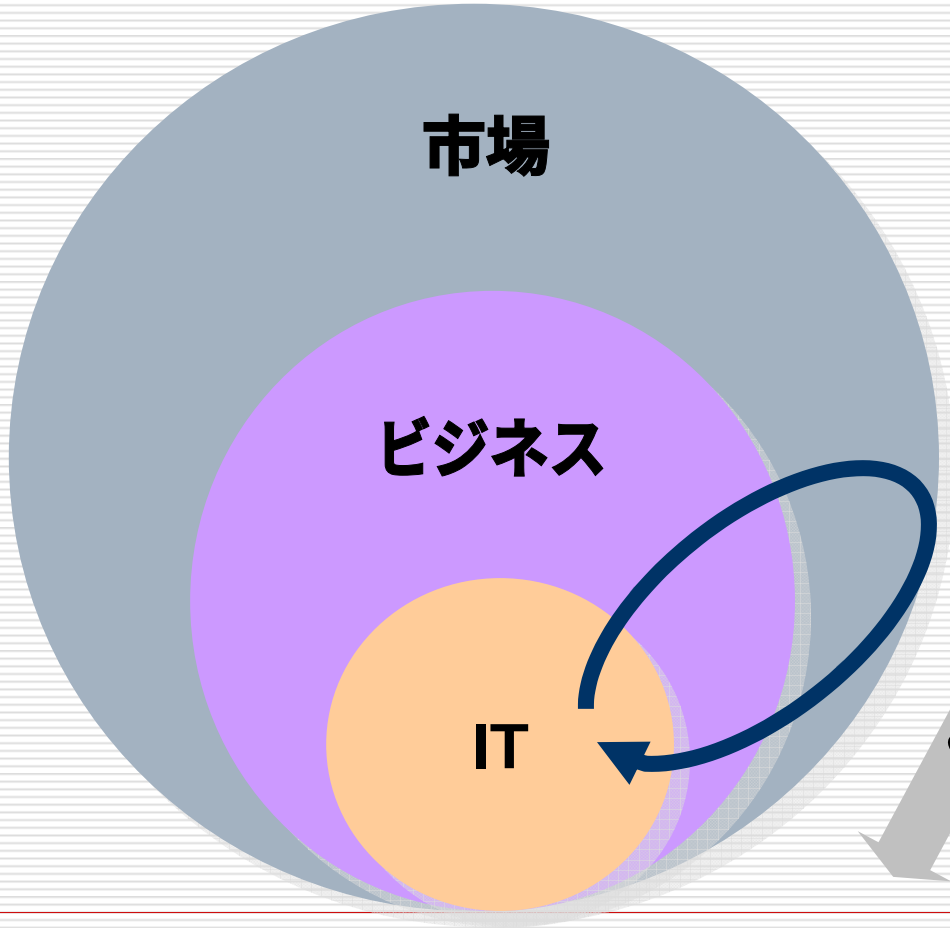
不明確かつ不安定な要求。





Ruby
Agile

2 ミッション・リスク共有型ビジネスとAgile型開発

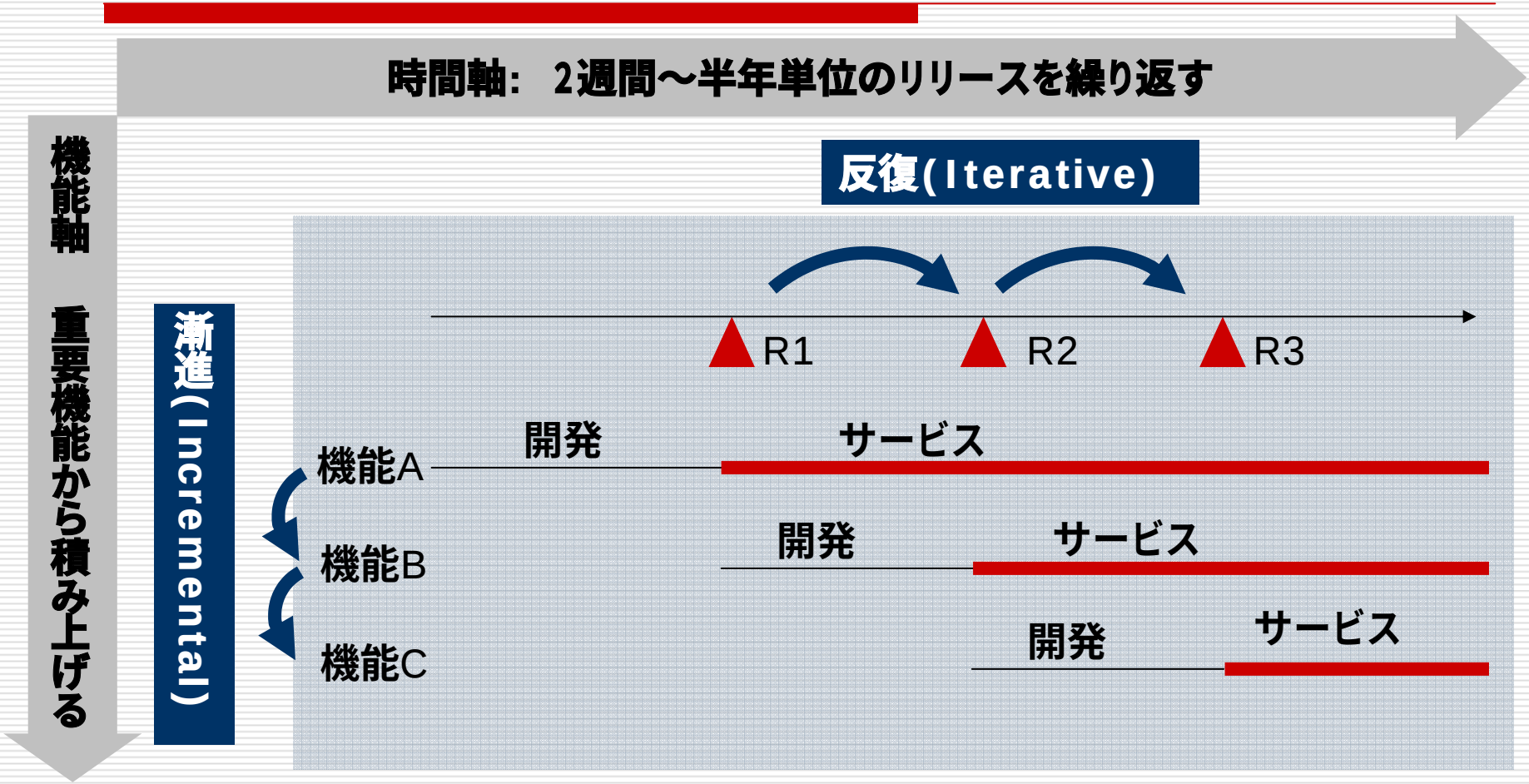


ビジネスとITが一体になった「OneTeam」を作り、ミッションとリスクを共有する。
やってみて、結果から戦略を作りながら進む。

2週間から半年

Web2.0型

WebビジネスとAgile型開発 (1/2)



Ruby
Agile

Webビジネスとagile型開発(2/2)

- 機能軸： 積み上げる (漸進)
  - ビジネス重要度の高いものから開発する。
- 時間軸： 繰り返す (反復)
  - 2週間～3ヶ月～半年のリリースを繰り返す。

ビジネスの利点

早期にサービスを開始し、ビジネス効果を見ながら、投資を決めていく。

ITの利点

不慣れなテクノロジに徐々に慣れていく。開発に不慣れな人材を教育する。

ビジネスとITがともに「学習」しながら、育っていく。

Ruby
Agile

高速な繰り返しを可能にする技術

- Agile 開発プロセス
  - XP、Scrum、プロジェクトファシリテーションをベースとしたAgile型の開発プロセス。
- Rails フレームワーク
  - Webアプリケーションを素早く、見栄えよく作れるアプリケーションフレームワーク。
- Ruby 言語
  - 読みやすく、変化を受け入れる言語。

Ruby
Agile

Agile **開発の本質**

人間とチームがもつ、学ぶ力

Ruby
Agile

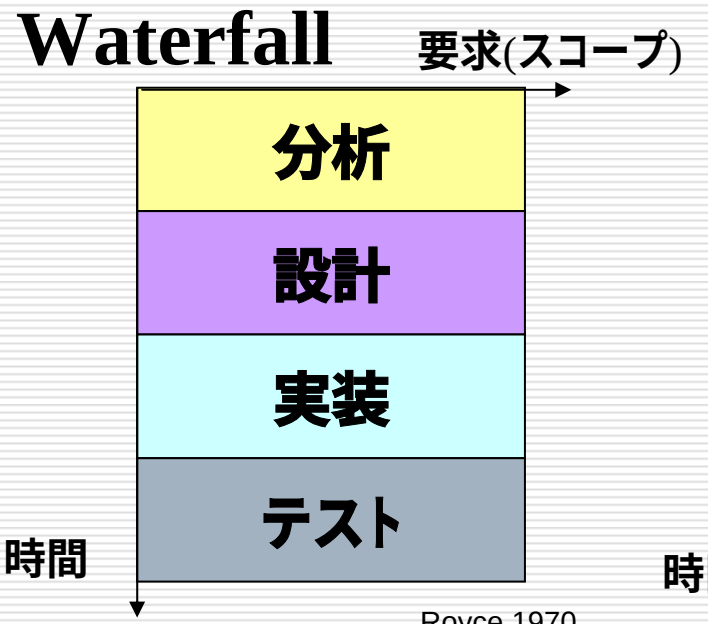
Agile関連の重要書籍

- 『XP(エクストリーム・プログラミング)』(Kent Beck)
  - 「コミュニケーション」、「シンプルさ」、「フィードバック」、「勇気」、「敬意」
  - テスト駆動開発
- 『リーンソフトウェア開発』(Mary Poppendieck)
  - トヨタ生産方式をソフトウェア開発へ
  - もの作りはチーム作り
  - バーンダウン、かんばん、7つのムダ
- 『Crystal Clear』(Alistair Cockburn)
  - プロジェクトを「安全地帯」へ導くチームづくり
  - プロセスからピープルへ
- 『アジャイルプロジェクトマネジメント(APM)』(Jim Highsmith)
  - 変化に対応するチームづくり
  - 「コマンドーコントロール」⇒「リーダシップーコラボレーション」
  - 「Plan-Do」⇒「Envision-Explore]
- 『達人プログラマー』(David Thomas, Andrew Hunt, Mike Clark)

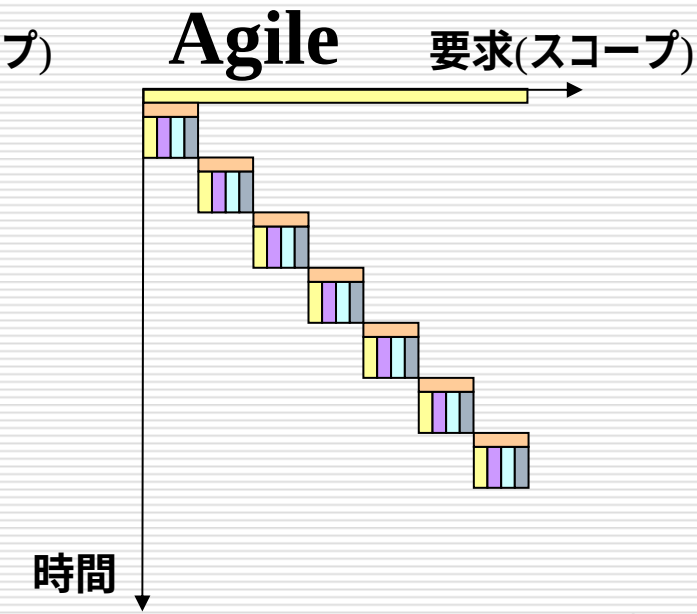
Ruby
Agile

プロセスとしてのAgile

- 短いサイクルで、分析、設計、実装、テストを並列に行う
- 進化型開発



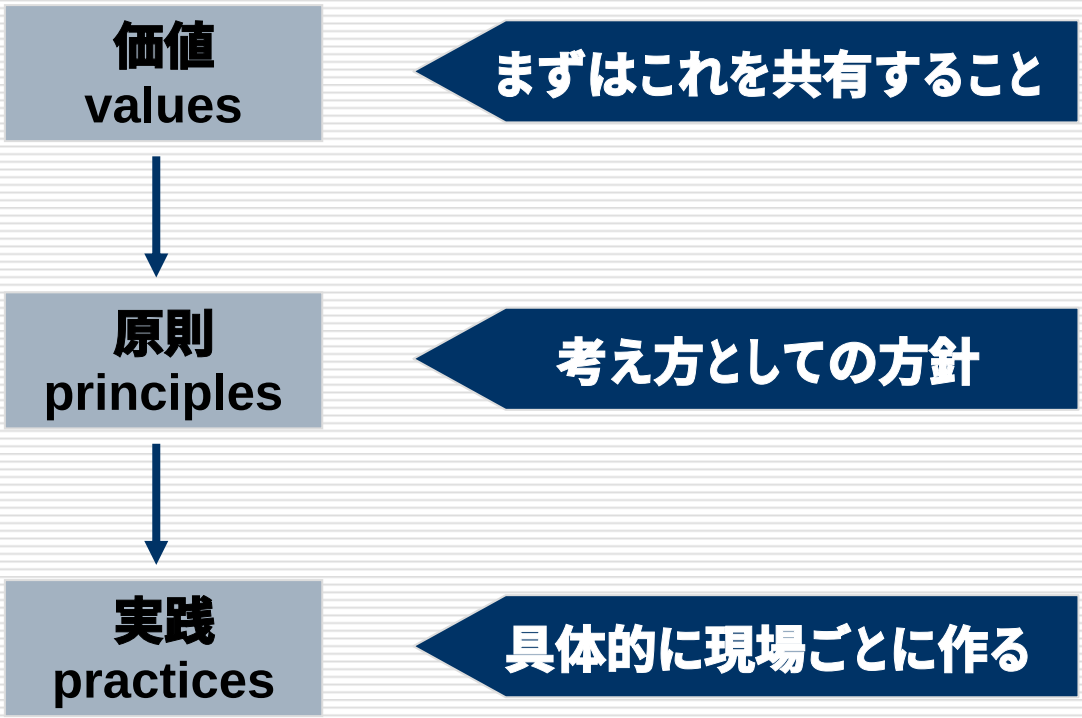
Royce 1970



Beck 2000

Ruby
Agile

Agileの価値、原則、実践



Ruby
Agile

Agileの価値

私たちは，

プロセスとツールよりも ..... 個人と対話に．
包括的なドキュメントよりも ..... 動くソフトウェアに．
契約交渉よりも ..... 顧客との協調に．
計画に沿うことよりも ..... 変化に対応することに．

価値をおく．

出展：アジャイル宣言(agilemanifesto.org)

Ruby
Agile

Agileの原則(1/3)

- **顧客価値の優先**
  価値のあるソフトウェアをできるだけ早い段階から継続的に納品することによって顧客満足度を最優先します。
- **変化に対応**
  要件の変更はたとえ開発の後期であっても受け入れます。変化を味方につけることによってお客様の競争力を引き上げます。
- **短期のリリース**
  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します。
- **全員同席**
  ビジネスをする人と開発者はプロジェクトを通して日々一緒に働かなければなりません。

Ruby
Agile

Agileの原則(2/3)

- **モチベーションと信頼**
  意欲に満ちた人々を集めてプロジェクトを構成します。環境と支援を与え仕事が無事終わるまで彼らを信頼してください。
- **会話**
  情報を伝えるもっと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す。
- **動くソフトウェア**
  動いているソフトウェアこそが進捗の最も重要な尺度です。
- **持続可能なペース**
  アジャイル・プロセスは持続可能な開発を促進します。一定のペースで永続的に保守できるようにしなければなりません。

Ruby
Agile

Agileの原則(3/3)

- **技術**
  卓越した技術と優れた設計に対する不断の注意こそが機敏さを高めます。
- **シンプル**
  シンプルさ – ムダなく作れる量を最大限にすること - が本質です。
- **自己組織的チーム**
  最良のアーキテクチャ、要件、設計は自己組織的なチームから生み出されます。
- **ふりかえりと改善**
  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します。

Ruby
Agile

Agileの実践（例　XP）(1/2)

- **計画ゲーム**
  ビジネス優先度と技術的見積により次回リリースの範囲を早急に決める。現実が計画と変わったら、計画を更新する。
- **小規模リリース**
  シンプルなシステムを早急に生産に投入する、それから新バージョンを非常に短いサイクルでリリースしていく。
- **メタファー**
  どの様に全体のシステムが機能するかを示すシンプルな メタファーを共有することで全ての開発を導く（ガイドする）。
- **シンプルデザイン**
  いつでもシステムは出来る限りシンプルに設計されるべきだ。余分な複雑さは見つけ次第取り除かれる。
- **テスティング**
  プログラマは継続的にユニットテストを書く。顧客は、機能の開発が終わったことを示す受け入れテストを書く。
- **リファクタリング**
  2重コードを取り去り、単純化し、柔軟性を加えるために、プログラマは、システムの動作を換えることなくシステムを再構成する。

Ruby
Agile

Agileの実践（例　XP）(2/2)

- **ペアプログラミング**
  全てのコードは2人のプログラマにより一台のマシンで書かれる。
- **共同所有権**
  誰でも、どのコードでも、どこででも、いつでも、プログラマはコードを修正できる。
- **継続的インテグレーション**
  システムを一日に何回もインテグレードしビルドし、テストを 100% パスさせる.
- **週40時間**
  週40時間以上仕事をしてはいけないのがルール。一日8時間を燃焼する。
- **オンサイト顧客**
  現実のユーザをチームに加えて、フルタイムで質問に答えられるようにする。
- **コーディング標準**
  プログラマは、コーディング標準に従って全てのコードを書く

Ruby
Agile

Agile開発の本質

- 単に繰り返すだけでなく、
  「人」に焦点を当てている。

- コミュニケーションを大事にする
- モチベーションを大事にする
- 成長を促す
- 「人が学ぶ」、ことを最大限に活かす

Ruby
Agile

Java と Ruby

Java, Ruby, Agileの歴史

Ruby
Agile

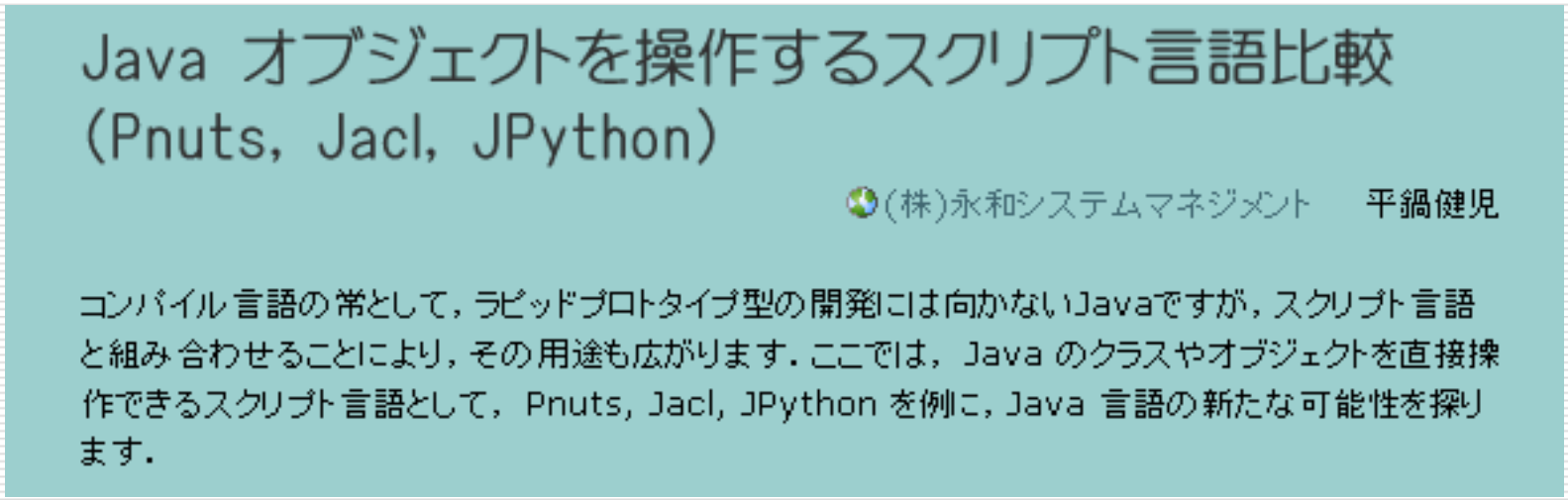
- **Javaの年表**
  - * 1996年：JDK 1.0リリース
  - * 1999年：J2EE 1.0リリース
  - * 2004年：Martin FowlerがDependency Injectionパターンを提唱
  - * 2006年：EJB 3.0リリース
- **Rubyの年表**
  - * 1993年：Ruby誕生
  - * 1995年：Ruby 0.95 リリース
  - * 1999年：「オブジェクト指向スクリプト言語 Ruby」出版
  - * 2000年：Perl/Ruby Conference開催(京都)、「プログラミング言語Ruby 第2版 言語編」出版
  - * 2001年：RubyConf 2001開催
  - * 2005年：「Railsによるアジャイルwebアプリケーション開発」出版
  - * 2006年：日本Rubyカンファレンス2006開催、第2次Rubyブームの予感
- **Agile開発の年表**
  - * 1999年：「エクストリーム・プログラミング入門」出版
  - * 2000年：「プログラミングRuby 達人プログラマーガイド」出版
  - * 2001年：アジャイルマニフェスト制定
  - * 2004年：「エクストリーム・プログラミング入門 第2版」出版
  - * 2005年：「Railsによるアジャイルwebアプリケーション開発」出版
  - * 2007年：「Agile Software Development 2nd Edition」がJolt Awards受賞

タイムライン：http://timeline.nifty.com/portal/show/2242

Ruby
Agile

平鍋とJavaの関わり(1)

- 生涯はじめての記事(DDJ日本版 1998年5月号)
- 『Javaオブジェクトを操作するスクリプト言語』
  - www.objectclub.jp/technicaldoc/java/scripting-java



Java オブジェクトを操作するスクリプト言語比較
(Pnuts, Jacl, JPython)

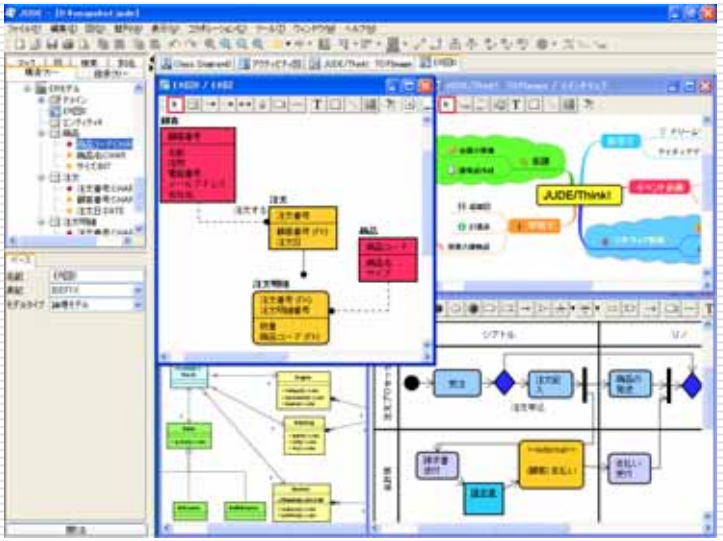
(株)永和システムマネジメント　平鍋健児

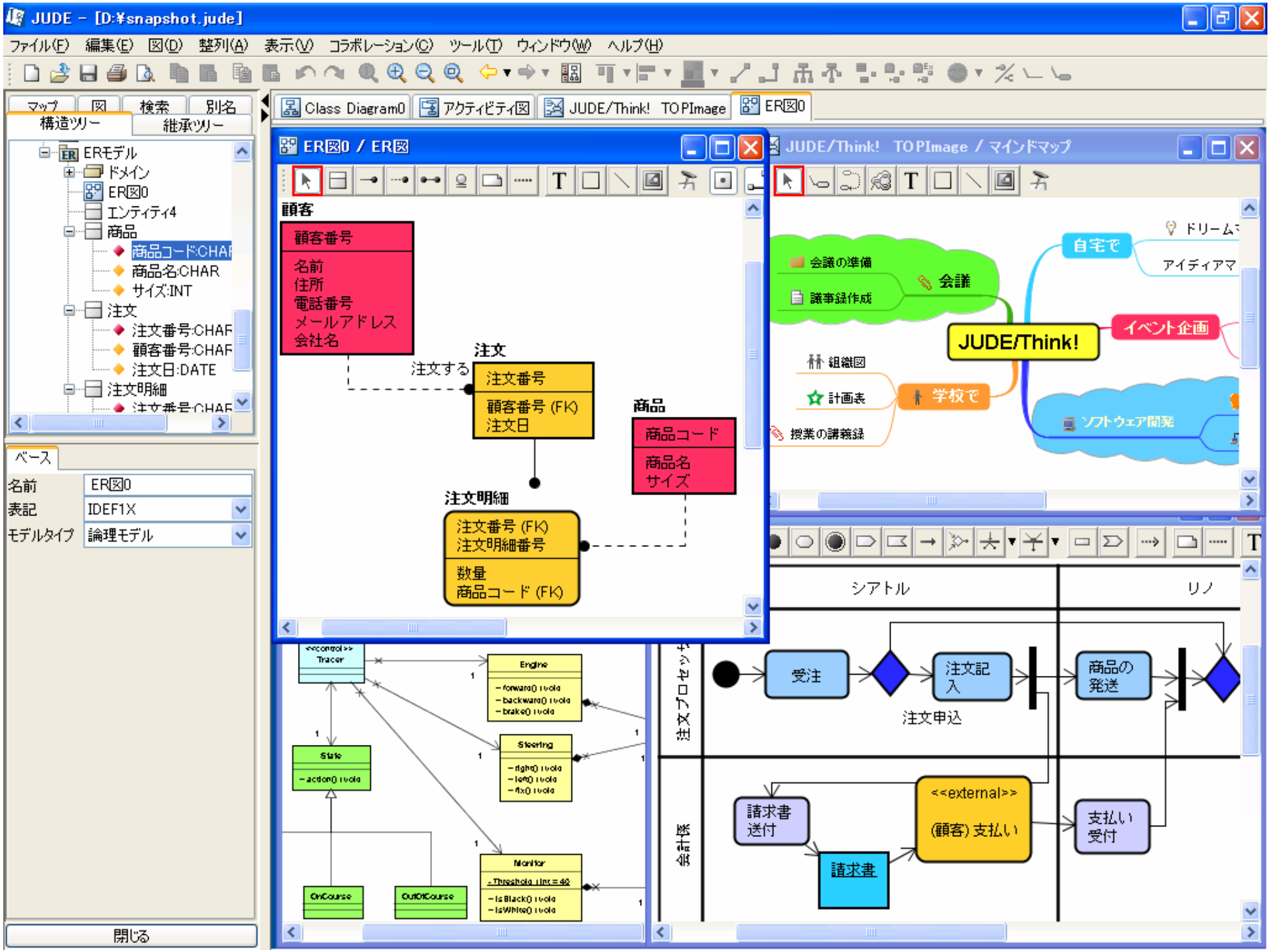
コンパイル言語の常として，ラピッドプロトタイプ型の開発には向かないJavaですが，スクリプト言語と組み合わせることにより，その用途も広がります．ここでは，Java のクラスやオブジェクトを直接操作できるスクリプト言語として，Pnuts, Jacl, JPython を例に，Java 言語の新たな可能性を探ります．

Ruby Agile

平鍋とJavaの関わり(2)

- Jude梅開発開始 1997年12月
- 2007年 10 月 200,000ユーザ達成
- jude.change-vision.com
  - UML　エディタ
  - Mind Map
  - ERD
  - フローチャート
  - DFD
  - CRUD

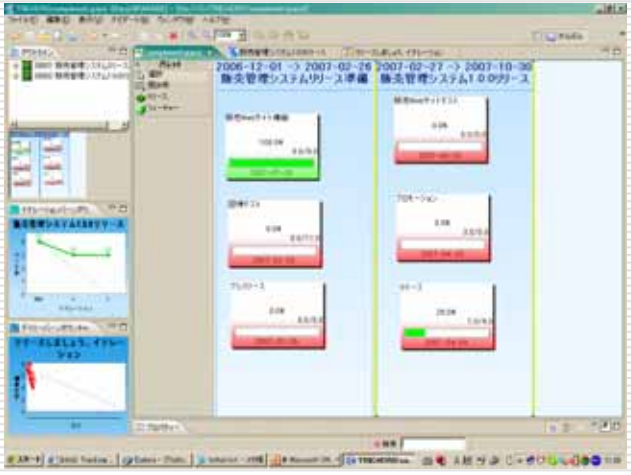
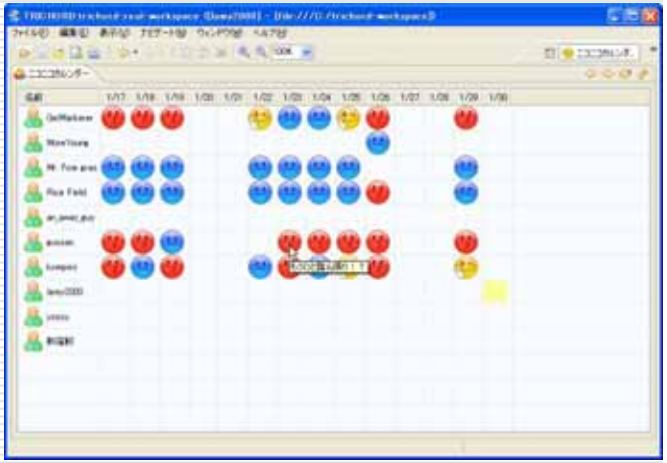


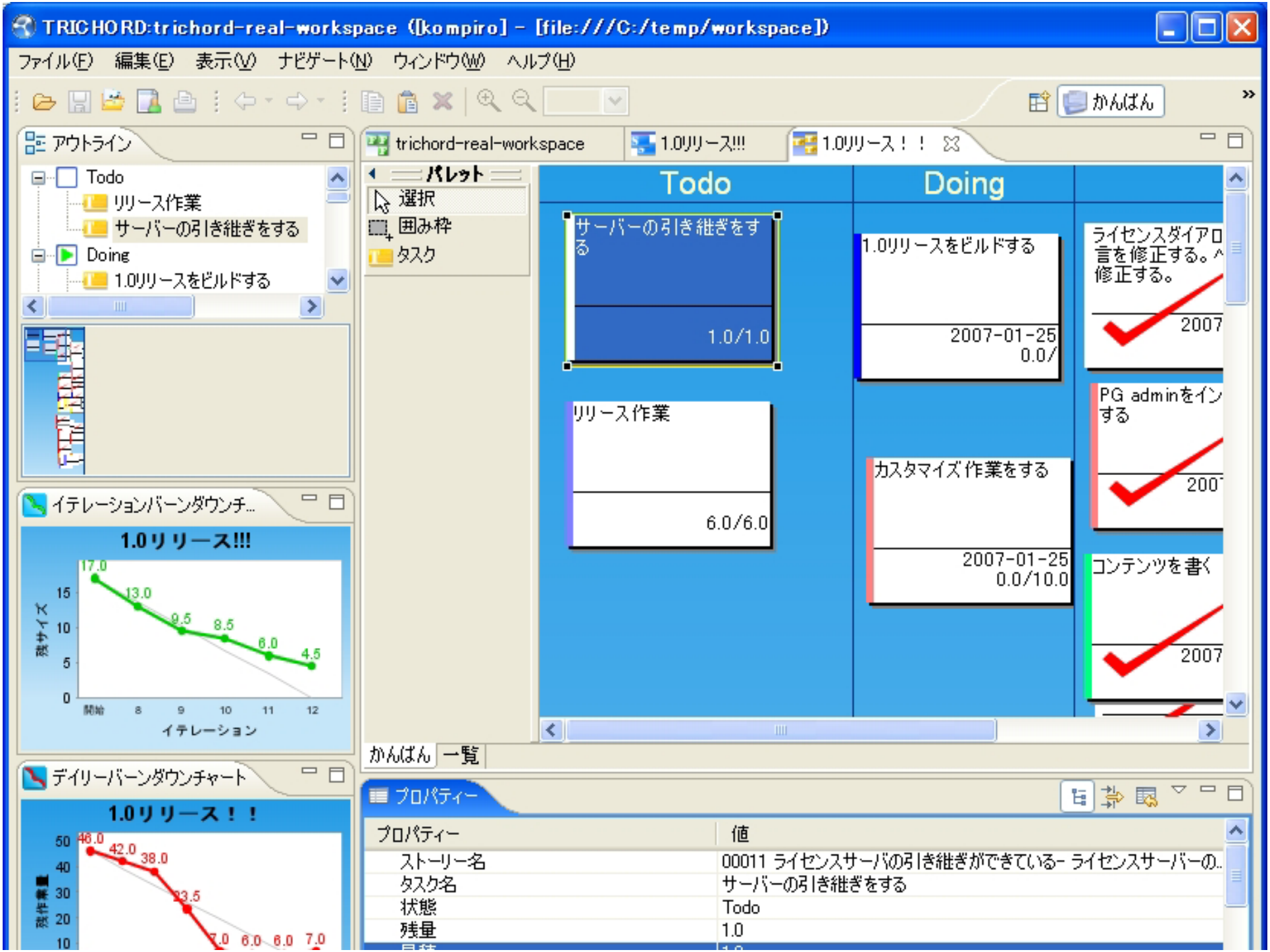

JUDE - [D:¥snapshot.jude]
ファイル(F) 編集(E) 図(D) 整列(A) 表示(V) コラボレーション(C) ツール(T) ウィンドウ(W) ヘルプ(H)
マップ 図 検索 別名
構造ツリー 継承ツリー
ERモデル
ドメイン
ER図0
エンティティ4
商品
商品コード:CHAR
商品名:CHAR
サイズ:INT
注文
注文番号:CHAR
顧客番号:CHAR
注文日:DATE
注文明細
ベース
名前 ER図0
表記 IDEF1X
モデルタイプ 論理モデル
閉じる
Class Diagram0
アクティビティ図
JUDE/Think! TOPImage
ER図0
ER図0 / ER図
顧客
顧客番号
名前
住所
電話番号
メールアドレス
会社名
注文
注文する
注文番号
顧客番号 (FK)
注文日
商品
商品コード
商品名
サイズ
注文明細
注文番号 (FK)
注文明細番号
数量
商品コード (FK)
JUDE/Think! TOPImage / マインドマップ
JUDE/Think!
会議
会議の準備
議事録作成
自宅で
アイディアマ
イベント企画
学校で
組織図
計画表
授業の講義録
ソフトウェア開発
Engine
Steering
State
Monitor
OnCourse
OutOfCourse
シアトル
リノ
注文プロセッ
会計係
受注
注文記入
注文申込
商品の発送
請求書送付
<<external>>
(顧客) 支払い
支払い受付
請求書

Ruby Agile

平鍋とJavaの関わり(3)

- TRICHORD: Agile なチーム開発サポート
- trichord.change-vision.com


TRICHORD:trichord-real-workspace ([kompiro] - [file:///C:/temp/workspace])
ファイル(F) 編集(E) 表示(V) ナビゲート(N) ウィンドウ(W) ヘルプ(H)
かんばん
アウトライン
Todo
リリース作業
サーバーの引き継ぎをする
Doing
1.0リリースをビルドする
イテレーションバーンダウンチ...
1.0リリース!!!
残サイズ
イテレーション
17.0
13.0
9.5
8.5
6.0
4.5
デイリーバーンダウンチャート
1.0リリース！！
残作業量
46.0
42.0
38.0
23.5
7.0
6.0
6.0
7.0
trichord-real-workspace
1.0リリース!!!
1.0リリース！！
パレット
選択
囲み枠
タスク
Todo
Doing
サーバーの引き継ぎをする
1.0/1.0
1.0リリースをビルドする
2007-01-25
0.0/
ライセンスダイアロ
言を修正する。^
修正する。
2007
リリース作業
6.0/6.0
カスタマイズ作業をする
2007-01-25
0.0/10.0
PG adminをイン
する
200
コンテンツを書く
2007
かんばん
一覧
プロパティー
プロパティー
値
ストーリー名
00011 ライセンスサーバの引き継ぎができている- ライセンスサーバーの..
タスク名
サーバーの引き継ぎをする
状態
Todo
残量
1.0

Ruby
Agile

今後5年で起こること

- Java は1995年当初、遅くて使えない言語だった。
- 10年かかってWebアプリケーションの標準言語になったのは、CPUとネット速度の向上による。
- 今後5年で、Rubyで同じことが起こる。
- CPUとネットが安くなるため、さらに「人にやさしい言語」、「変更に対応できるプロセス」、「人からチベーションを引き出す言語とプロセス」が、ビジネスでも実際に使われるようになる。

Ruby
Agile

Rubyの優位性

- ソースコードの読みやすさ
- プログラミングの楽しさ
- 習熟度にあった学習曲線
- オープンクラス

Java から Ruby へ(1/3)

Java

```
for (int i = 0; i < 10; i++)
    System.out.println(i);
```

Ruby

```
10.times {|i| puts i }
```

Java から Ruby へ(2/3)

Java

```
new Date(new Date().getTime()- 20 * 60 * 1000);
```

Ruby

```
20.minutes.ago
```

出展：http://haacked.com/

Java から Ruby へ(3/3)

Java

```
public ActionForward edit(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response) throws
    Exception {
  PersonForm personForm = (PersonForm) form;
  if (personForm.getId() != null) {
    PersonManager mgr = (PersonManager)getBean("personManager");
    Person person = mgr.getPerson(personForm.getId());
    personForm = (PersonForm) convert(person);
    updateFormBean(mapping, request, personForm);
  }
  return mapping.findForward("edit");
}
```

Ruby

```
def edit @person = Person.find(params[:id]) end
```

出展：http://www.relevancellc.com/

A better idea is to create a flexible language and let human creativity flow.

Fluent Interface
(流れるようなインターフェース)

Ruby Agile

両方Javaですが...

```
private void makeNormal(Customer customer) {
 Order o1 = new Order();
 customer.addOrder(o1);
 OrderLine line1 = new OrderLine(6, Product.find("TAL"));
 o1.addLine(line1);
 OrderLine line2 = new OrderLine(5, Product.find("HPK"));
 o1.addLine(line2);
 OrderLine line3 = new OrderLine(3, Product.find("LGV"));
 o1.addLine(line3);
 line2.setSkippable(true);
 o1.setRush(true);
}
```

```
private void makeFluent(Customer customer) {
     customer.newOrder()
           .with(6, "TAL")
           .with(5, "HPK").skippable()
           .with(3, "LGV")
           .priorityRush();
  }
```

http://www.martinfowler.com/bliki/FluentInterface.html

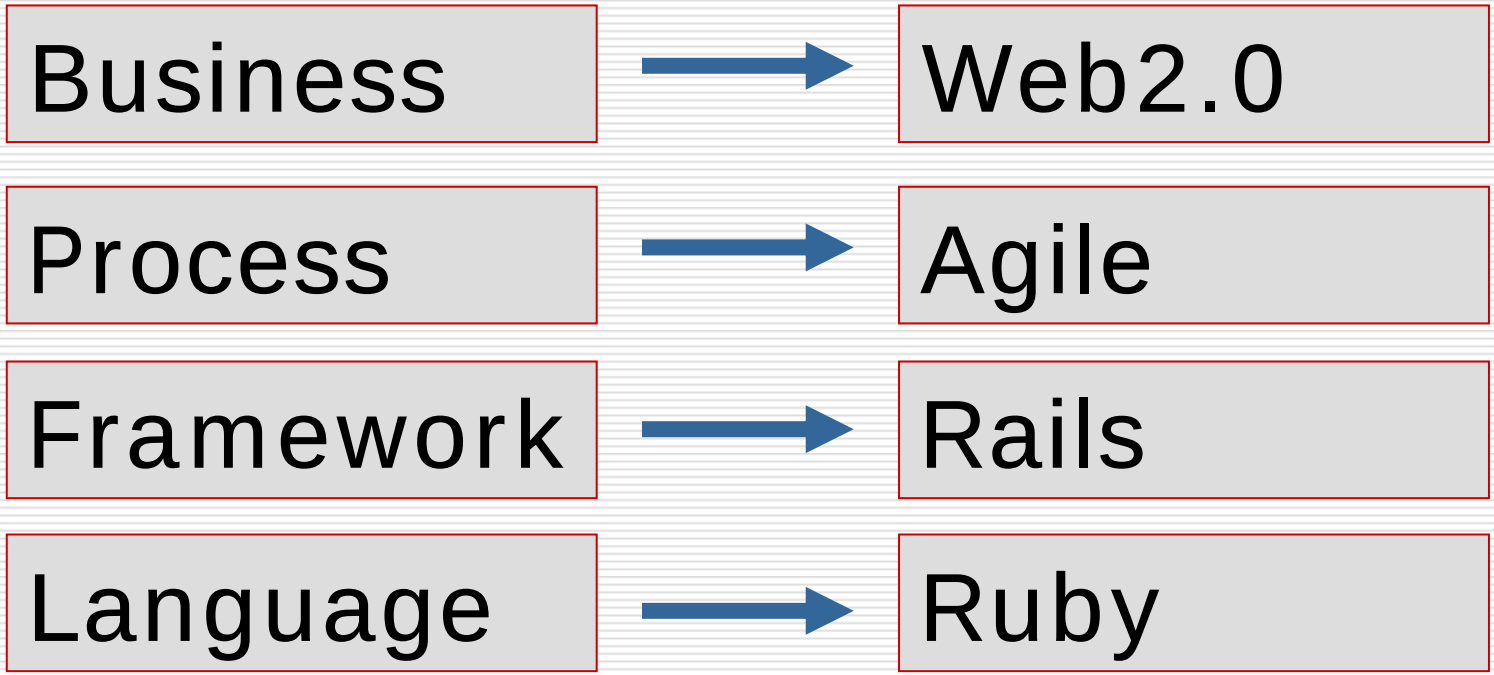
Ruby
Agile

まとめ

- 変化が早いビジネス領域では、「すべての要求を固めてドキュメント化し、それを一年かけて開発する」やり方では要求が変化・劣化する。
- Agile開発はすばやくリリースを反復し、漸進的に機能を提供し、変化に対応する。
- Agile開発は、人とソフトウェアを成長させることで、変化に対応する。
- Ruby, Rails は、マシンではなく、人の思考を自然に表現することに長けた言語とフレームワークである。
- Ruby + Agile は、ビジネスの変化に対応し、かつ、「ソフトウェア」と「人」を育てる、現在最も有望な組み合わせである。

Ruby
Agile

1つのインスタンス化



Javaがんばれ!

Diversity is good (**多様性は善**)