SPI Japan 2015

HITACHI
Inspire the Next

Rubyを「やりたい」から「やった」にするまでの道のり

2015/10/22

株式会社 日立ソリューションズ
技術開発本部　生産技術部

細美 彰宏、牧 俊男

HITACHI
Inspire the Next

Contents

1. Rubyを始めた背景
2.「やりたい」から「やった」に変えるまでの道のり
3. 振り返り

HITACHI
Inspire the Next

1. Rubyを始めた背景

1-1 Rubyを始めた背景

HITACHI
Inspire the Next

世の中

- 2005年：「Ruby on Rails」登場
  - 「15分でブログアプリケーションを作る」という動画を作者が公開
    「Ruby on Rails」の持つ生産性の高さが世界中で話題に
  - 日本国内でも、「10分で作るRailsアプリ」を公開
- 2007年：日経BP技術大賞「プログラミング言語Ruby」
  - それまでは、一部のマニアが知っている言語だったが、経営幹部が「Ruby」に対して興味を持ち始めたキッカケ

結果

- 2008年：Rubyの社内検証をスタート
  - 社内の技術者が片手間に動向をウォッチしていたが、生産技術部門として、本格的にRubyの検証をスタート

1-2 社内検証フェーズ

HITACHI
Inspire the Next

社内SNSをRubyでリプレース

- 「OpenPNE」で構築された社内SNSをRubyでリプレースして、評価。
- アジャイルを意識し、利用者の声を聞きながら、改良とリリースを繰り返す。

画面数	99 [画面]
ステップ数	8,119 [step]
開発工数	638 [時間]

（1画面平均82step、6.4h）

■ RubyWorld Conference 2009
「コミュニティシステム開発を通してRuby on Railsの実力を検証」で成果を発表

「Rubyでいける」という実感を得て、本格的に社内で普及させる活動を開始

1-3 Rubyセンタ発足

HITACHI
Inspire the Next

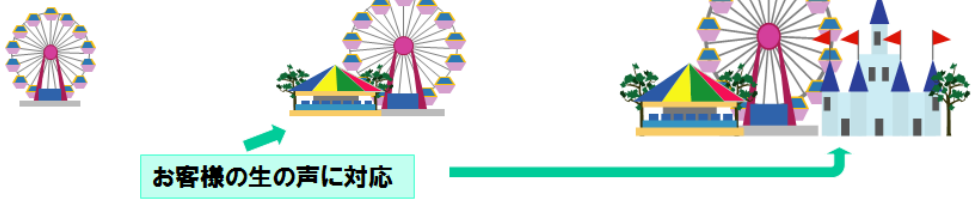
Ruby専門組織の立ち上げ

- 2009年12月、Ruby専門組織「Rubyセンタ」を立ち上げ
- Rubyを普及させる取り組み（研究開発、技術整備、活用推進）を開始。

■ まずは作って、随時機能を追加・改良する開発スタイル

- ▶『動く仕様書』で使い勝手やイメージを確認
- ▶ お客様の真のニーズを引き出す

- ▶ お客様の生の声に応える機能追加で満足度向上
- ▶ システムを利用しながら改良する開発状況の見える化



当初のターゲットは、
「中小案件向けソリューション」

≒ アジャイル開発

大きな案件の下で、まだ拾えていない大量の小型の案件をカバーしたい。
潜在的に「やりたい人」はいるはず。

HITACHI
Inspire the Next

2.「やりたい」から「やった」に変えるまでの道のり

2-1 Rubyセンタ設立初期の活動（2009年～2010年）

HITACHI
Inspire the Next

Rubyを社内に普及させるための取り組み

分類	課題	対策
開発環境	環境構築が難しい	開発用VM提供 手順書の作成・推奨ツールの紹介
教育	Ruby技術者がいない	社内の教育部署と連携し、 教育コースの立ち上げ
見積もり	工数見積もりが難しい	検証時に得られたデータを公開
受注支援	お客様への説明が難しい	営業に同行して、説明
技術サポート	新技術採用に対する不安がある	社内からの問い合わせ窓口の開設 社内向けRuby情報サイトの開設

この頃、開発していた案件

- 大学向けeラーニングシステム
- 水族館向け館内案内システム

Rubyアソシエーション（http://www.ruby.or.jp/ja/）で事例を紹介。

2-2 行き詰りと方針転換

HITACHI
Inspire the Next

行き詰まり

- 2年ほど活動したが、適用プロジェクト数が伸びない。
- 一度Rubyを使って上手くできたプロジェクトは継続してくれるが、そもそも、「やりたい人」から「やった人」にするには壁がある。

原因

- 中小案件は適用できても、大規模には使えないという社内評価。
- Rubyという新技術採用に対するリスク。
- リスクを取りたい人はいない。リスクを乗り越えて、やりたがる人は少ない。

方針転換

- 元々、中小規模向けソリューションを対象に拡大を狙っていたが、普及のためには、大規模案件でも使えるという社内実績が必要。
- Rubyがアジャイルだけでなく、汎用的に使える技術だと証明する。

2-3 Rubyセンタの活動（2010年～2011年）

HITACHI
Inspire the Next

大規模案件でRubyを使うための取り組み

分類	課題	対策
開発プロセス	大人数で開発するための取り決めが必要	コーディング規約や開発のルールを整備
技術者確保	多くの技術者が必要	声をかけられる会社を増やす
ロビー活動	顧客向けのPRが足りない	Ruby推進団体への加盟 Ruby事業の発展に注力
	個人が作った言語への懸念	「JIS X 3017」「ISO/IEC 30170」の策定委員として標準化に協力

大規模案件でのRuby適用

- Rubyをやってくれそうなプロジェクト管理者を直接説得。
- 顧客との長年の付き合いがあり、提案を信頼してもらえる下地。
- Rubyの訴求ポイントとして、素早い改修が可能を強調。

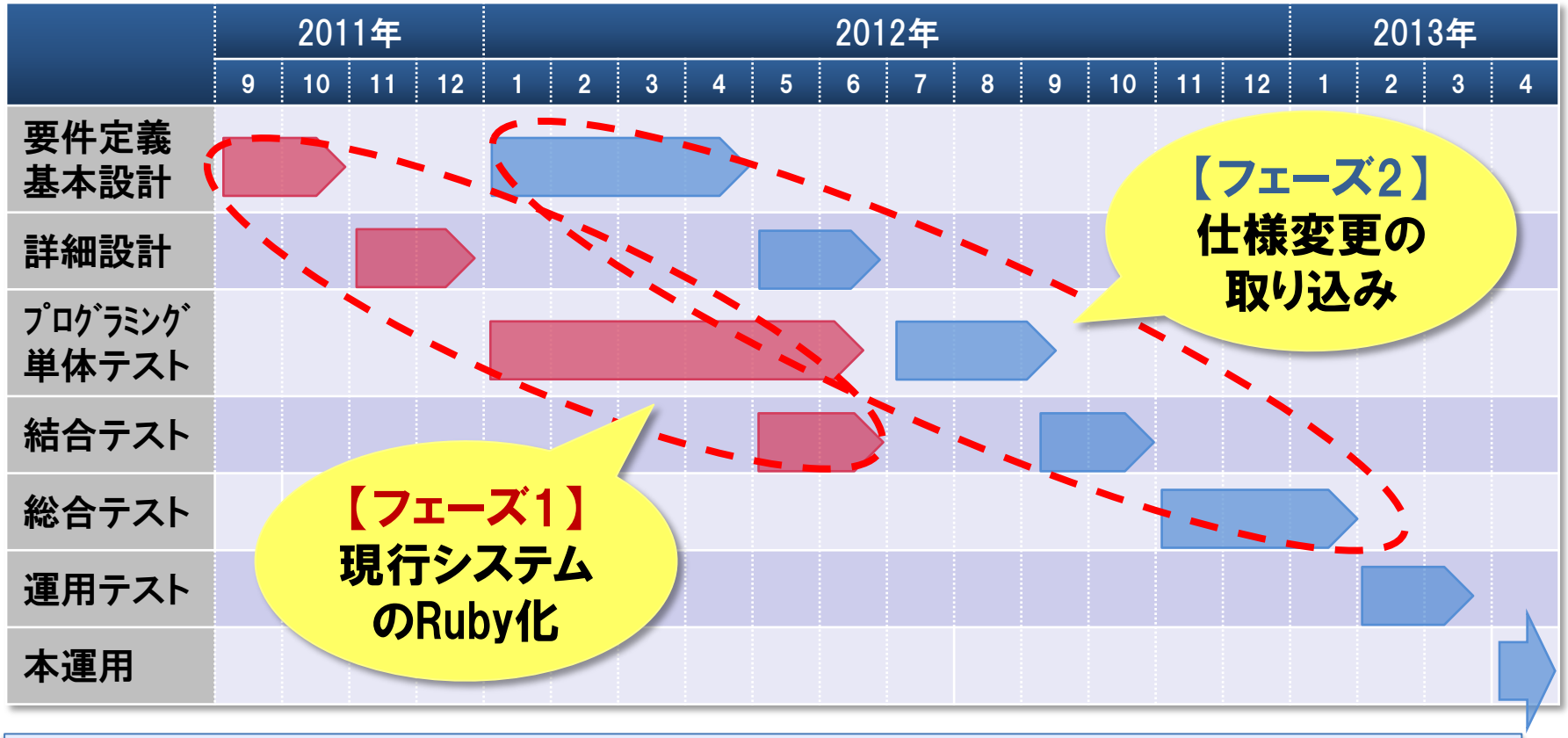
これらの活動が功を奏し、初の大規模案件のRubyプロジェクトがスタート

2-4 初のRuby大規模案件への挑戦（2011年～2013年）

HITACHI
Inspire the Next

社内初の大型Rubyプロジェクト

- 公共システムのリプレース。
- JavaでかかれたWebシステム(430Ks)とC言語で書かれたバッチ(70Ks)。

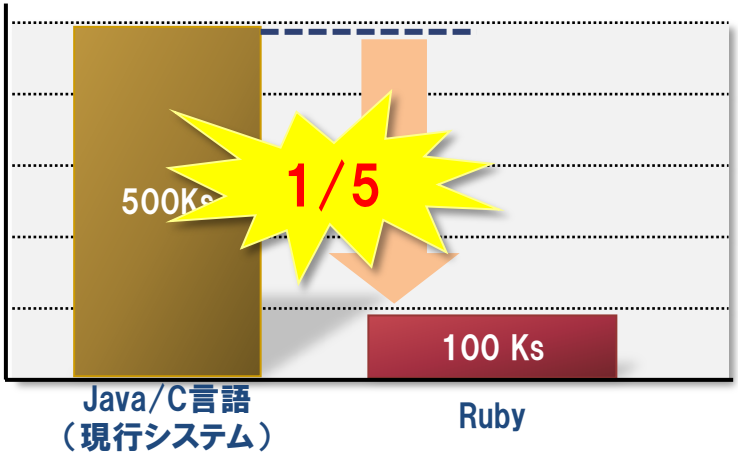


開発期間1年半、最大時30人以上がプロジェクトに参画

2-5 社内初のRubyの大規模案件の成功

HITACHI
Inspire the Next

大きな問題なく、予定通りリリース



- 第1フェーズ終了時点で、開発規模は1/5。
- 第2フェーズでは、変更に強いRubyの特性を十分に発揮できた。
- 当初計画より前倒しできたため、残った時間を品質向上や性能向上に充てた。

Rubyが大規模案件にも使えるという知見を得た

- C言語で書かれたバッチと同等の速度にチューニングするためのノウハウ。
- 小規模の案件では出てこないような細かい機能要件。
- 全く同じもの（現行のRuby化）を作って比較したため、Rubyのコード圧縮率の高さが目立った。

2-6 大型不採算プロジェクトで採用（2012年～2013年）

HITACHI
Inspire the Next

背景

- ECサイトのリプレース案件。
- 当初はパッケージ提案がったが、要件が合わず、スクラッチ開発に変更。
- 要件定義・基本設計のスコープ確定しないまま、開発が進行。
- 仕様変更も重なり、プロジェクトの完了が難しいと判断し、中断。

その後

- 再調整の結果、納期を1年延期してもらい、再開。
- 再見積りの結果、Javaで500Ks超え(後にさらに増大)。
- Javaでは納期に間に合わないと判断し、プロトタイプ開発に使っていたRubyを本番にも適用することに。

大型不採算プロジェクトをRubyでやり直し

2-7 大型不採算プロジェクトにおけるRuby開発

HITACHI
Inspire the Next

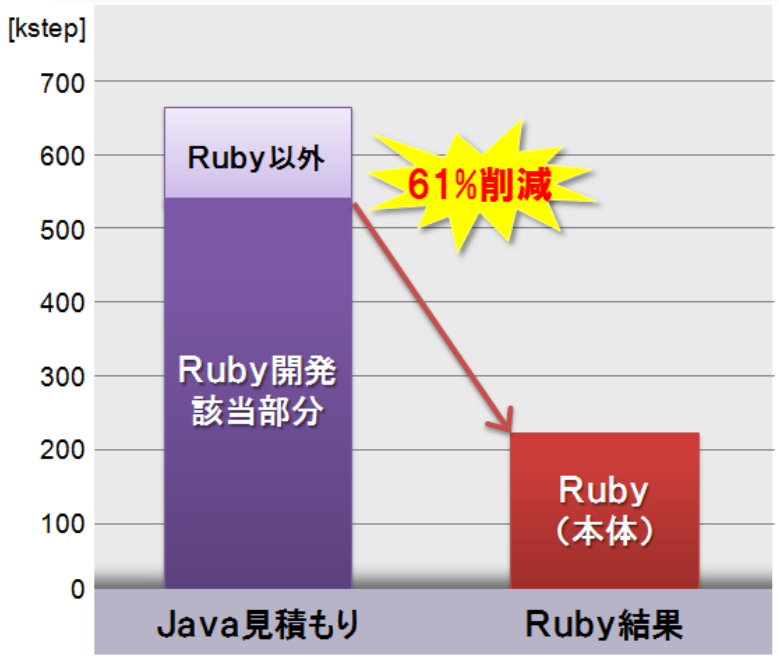
最大140名の開発体制

- プログラマ最大時50名。Ruby経験者は2割程度で、経験者がRuby/Railsを教育しながら開発チームを構築。
- セール時、30,000pv/minのアクセスが性能要件。

2-8 大型不採算プロジェクトの収束

HITACHI
Inspire the Next

ぎりぎり、間に合った



- 仕様調整と設計の遅れ(2ヶ月)をプログラミングとテスト工程で取り戻す。仕様調整が遅れた部分は見込みで作り、後で差分修正。
- 1ヶ月で、1,740件の仕様変更と不具合修正を解決。
- コード量はJava見積りと比較して61%減。

社内で注目されていただけに、収束したときの反響は大きかった

反省事項

- 不採算案件のため、重厚な管理プロセスが組まれ、Rubyらしさが薄かった。
- 設計と開発が分断しており、Ruby/Railsの考え方の共有に時間がかかった。
- Javaと同じ感覚で設計したため、終盤で性能問題が多発した。

2-9 その後の活動（～現在）

HITACHI
Inspire the Next

2つの大型案件の成功により、社内でのRuby知名度が高まった

- 特に、経営幹部が注目して、事業部に展開したのが大きい。
- Rubyに、あまり興味を持っていなかった人が、「やりたい人」に変化。

その後の活動

分類	活動内容
事例収集	過去の事例をまとめ、展開できる資料として文書化
開発支援	Rubyのプロジェクトの進め方を文書化 大型案件に活用したフレームワークをライブラリ化して、社内公開
広報活動	社内セミナを定期的に開催し、事例の紹介や最新動向を紹介
支援活動	声を掛けてくれたプロジェクトに参画し、スタートアップを支援 プロジェクトの立ち上がりを支援する過程で、公開したドキュメントやライブラリの使い方を説明
技術サポート	Rubyサポートサービスを開設、お客様向けに販売

導入に成功した部署では、継続的にRubyを使用してくれるようになり、
少しずつ、社内にRubyを「やった人」が増えていった。

HITACHI
Inspire the Next

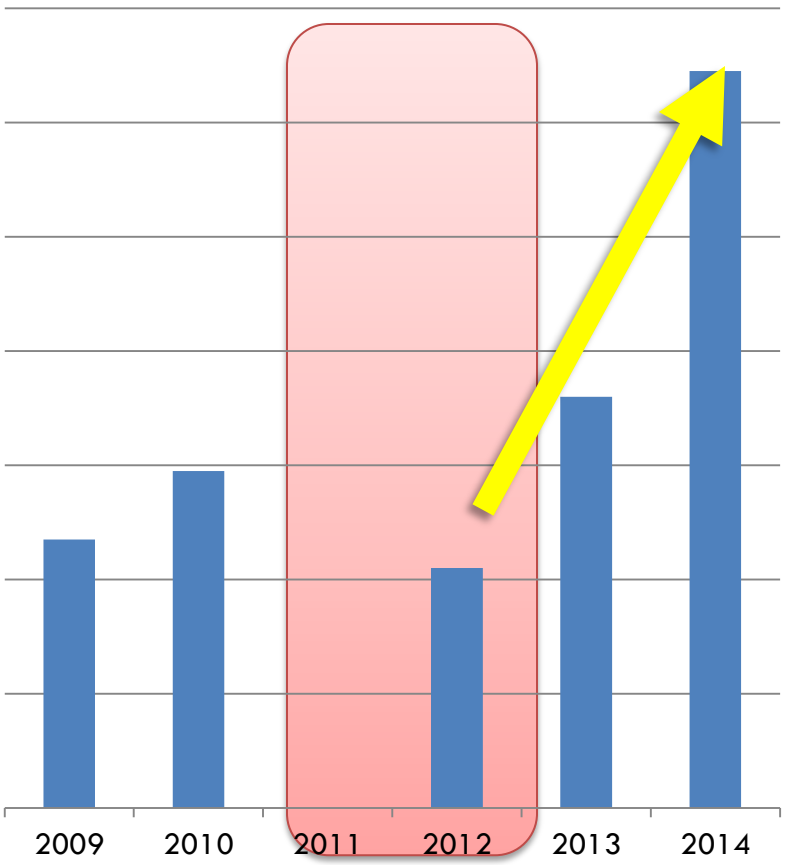
3. 振り返り

3-1　評価(１)「やりたい人」の推移

HITACHI
Inspire the Next

社内で開催していたRuby技術セミナーの参加人数を集計
Rubyに興味がある人≒「やりたい人」と想定

「やりたい人」の推移



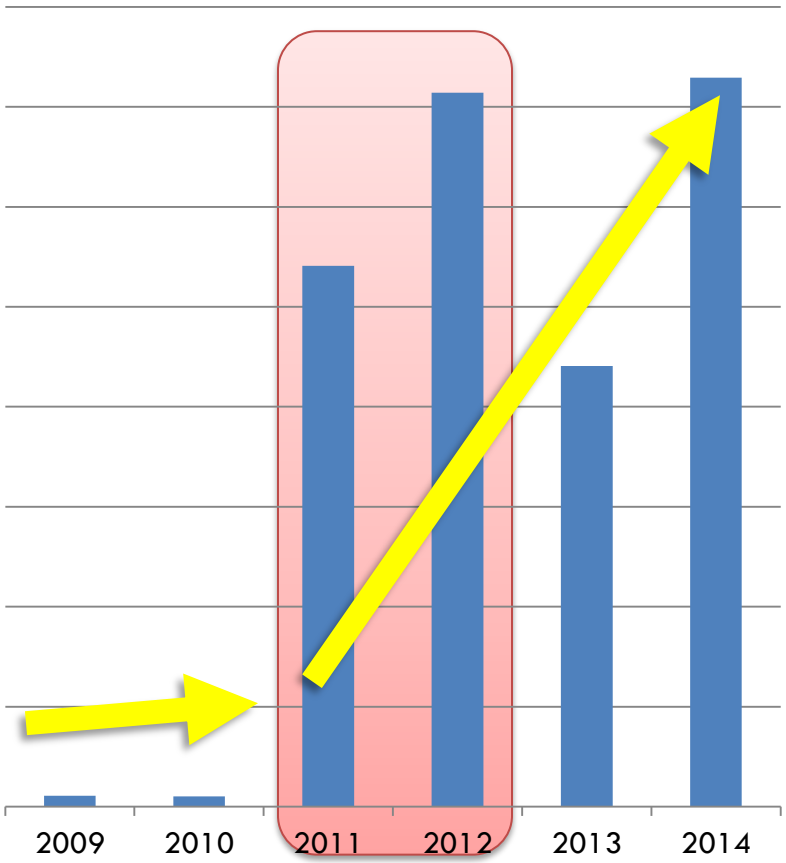
- 大型事例ができた2013年以降、**新技術採用への安心感が高まり、「やりたい人」が増えた。**
- セミナーのアンケート結果では、2010年までは「内容に興味がある」という人が多かったが、2013年以降は**「採用を検討している」という前向きな回答が多くなった。**

3-2　評価(２)「やった人」の推移

HITACHI
Inspire the Next

規模を反映するため、Ruby案件の売上高を集計
社内外の係わった人≒「やった人」と想定

「やった人」の推移



- 2011年～2013年で大型事例を作ることに注力したため、一時的に「やった人」が増えた。
- 2013年以降は、大型の成功事例によるPRから、「やりたい人」が適用に積極的になり、「やった人」が増えた。
- 今後も「やった人」の成功事例を基に、「やりたい人」を「やった人」にする活動を継続する。

3-3 今後の課題・展望

HITACHI
Inspire the Next

大規模案件で、Rubyを積極的に適用するには、まだ課題がある

課題	理由
Ruby技術者の確保	まだJavaに比べて技術者人口が少ない。 大量に受注すると、身内での取り合いになる。
いきなり性能の壁にぶつかる	本来なら、スモールスタートで、初めてスケールする過程で徐々に問題となり、解決していくものが、 大規模開発では、初めから問題になることがある。 性能問題の解決には、ある程度の知見が必要。
プロセスの問題	これまでの開発をJavaからRubyに置き換えることで改善されるのは、プログラミング・テストの領域だけ。 大規模開発のボトルネックは別にある。

もう一度、「中小案件向けソリューション」に立ち戻る時

- Rubyは、スモールスタートし、ビジネスの成長に合わせて、方向転換するやり方が適合しやすい。
- Ruby採用の敷居が低くなった今、本来やりたかったことにチャレンジ。

HITACHI
Inspire the Next

END

Rubyを「やりたい」から「やった」にするまでの道のり

2015/10/22

株式会社 日立ソリューションズ
技術開発本部 生産技術部

細美 彰宏、牧 俊男