#

JaSST鹿児島

アジャイル・Ruby・クラウド（ARC）を活用したビジネスにおける テストの実践

～ソニックガーデンの事例～

倉 貫 義 人　http://kuranuki.sonicgarden.jp　@kuranuki

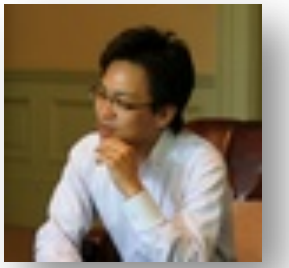
本日のテーマ

「テストの理想を現場視点で実現する」

アジャイル・Ruby・クラウド（ARC）を活用したビジネスにおけるテストの実践

はじめに: 自己紹介

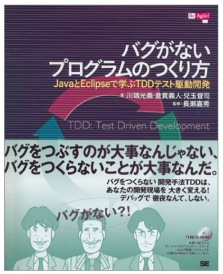
自己紹介



倉貫 義人　Yoshihito Kuranuki

昭和49年生まれ　38歳　京都府出身

株式会社ソニックガーデン 代表取締役社長



Twitter: @kuranuki

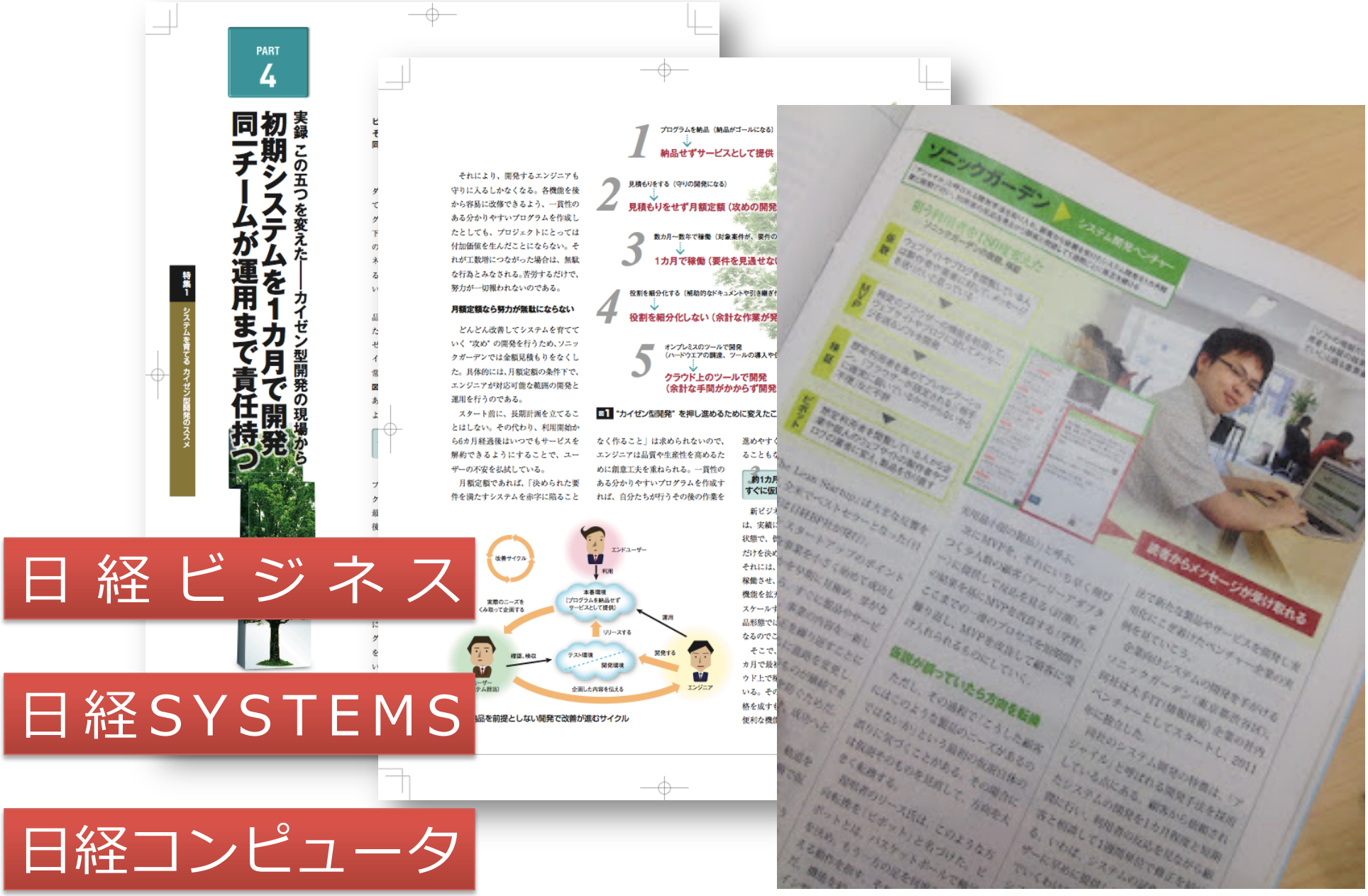
ブログ： Social Change!

http://kuranuki.sonicgarden.jp

「心はプログラマ、仕事は経営者」

http://www.sonicgarden.jp/

日経BP社さんにはお世話になってます



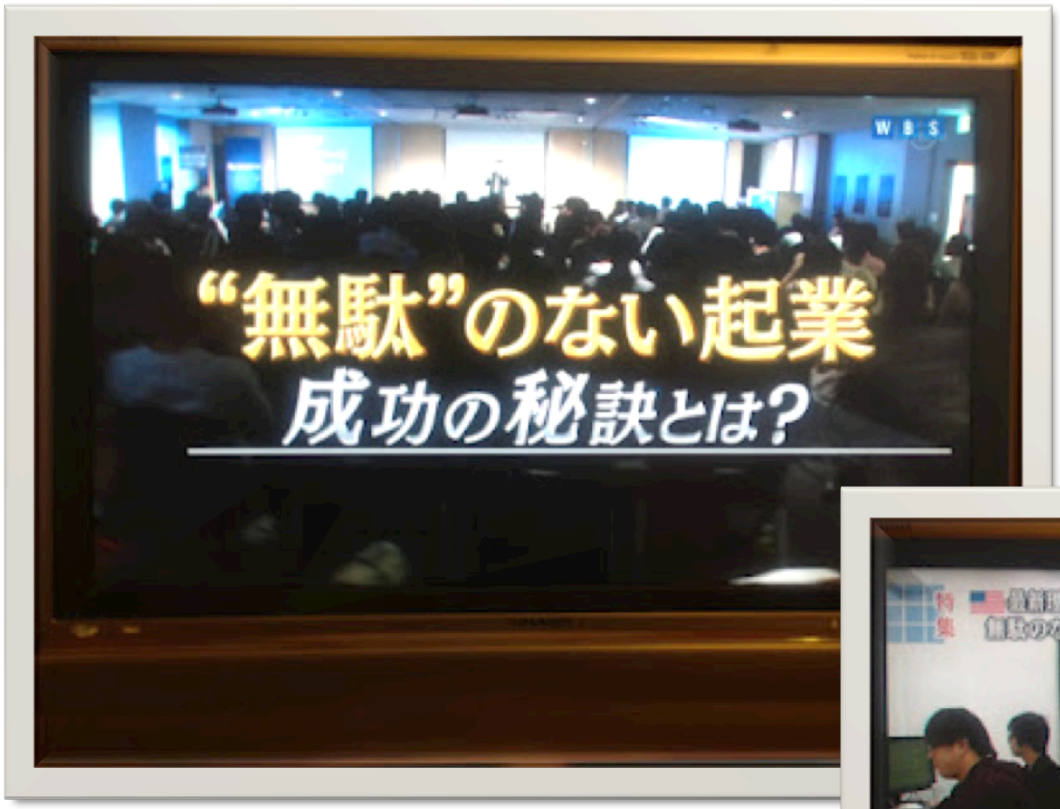
日経ビジネス

日経SYSTEMS

日経コンピュータ

http://www.sonicgarden.jp/

テレビ東京系列「ワールドビジネスサテライト」に出演！



自分戦略 @IT研究所

「エンジニアのための
リーンスタートアップ入門」連載

http://www.sonicgarden.jp/

これまでのキャリア

2011年**MBO**による株式会社ソニックガーデン設立

2009年TIS**社内ベンチャー**としてSonicGarden立ち上げ

2008年SKIP**オープンソース化**

2005年**Ruby**で**社内SNS：SKIP**開発

2004年ADC2004受賞　　書籍執筆　＆　**日本XPユーザグループ**代表就任

2003年TISにて基盤技術センター立ち上げ

1999年立命館大学院卒業　＆　TIS（旧：東洋情報システム）入社

株式会社ソニックガーデン

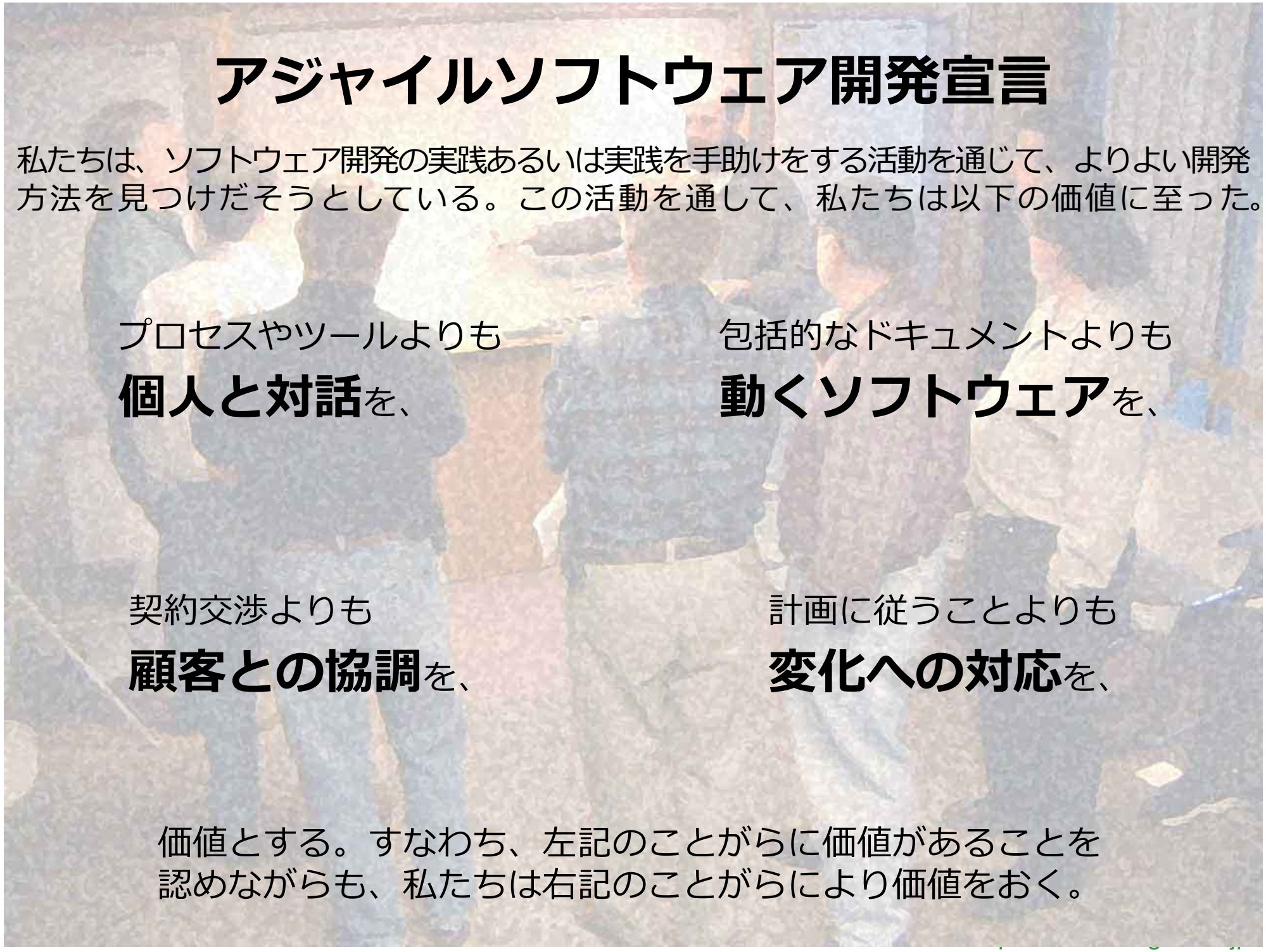
SonicGarden

「IT投資に対するソフトウェアの価値を最大化すること」

「習慣を変えることのできるソフトウェアをつくること」

に取り組んでいるソフトウェア企業が **SonicGarden** です。

アジャイル × Ruby × クラウド



アジャイルソフトウェア開発宣言

私たちは、ソフトウェア開発の実践あるいは実践を手助けをする活動を通じて、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、私たちは以下の価値に至った。

プロセスやツールよりも
個人と対話を、

包括的なドキュメントよりも
動くソフトウェアを、

契約交渉よりも
顧客との協調を、

計画に従うことよりも
変化への対応を、

価値とする。すなわち、左記のことがらに価値があることを
認めながらも、私たちは右記のことがらにより価値をおく。

PROGRAMMING
Language

DRY

Don’t Repeat Yourself.

保守性

高い生産性を維持継続

amazon web services™

仮想化

Infrastructure / Platform as a Service

スモールスタート
&
スケールアウト

ソニックガーデンで開発しているソフトウェアの特徴

- 完成して販売ではなく、継続的にサービス提供
- 自動的にバージョンアップがずっと続いていく
- 不具合が無いからといって、儲かる訳ではない

社内SNSの企画・構築・活用を
トータルサポートするSKIP

http://www.sonicgarden.jp/

株式会社ソニックガーデン
SonicGarden

ソニックガーデンでは、自社企画も受託開発も、

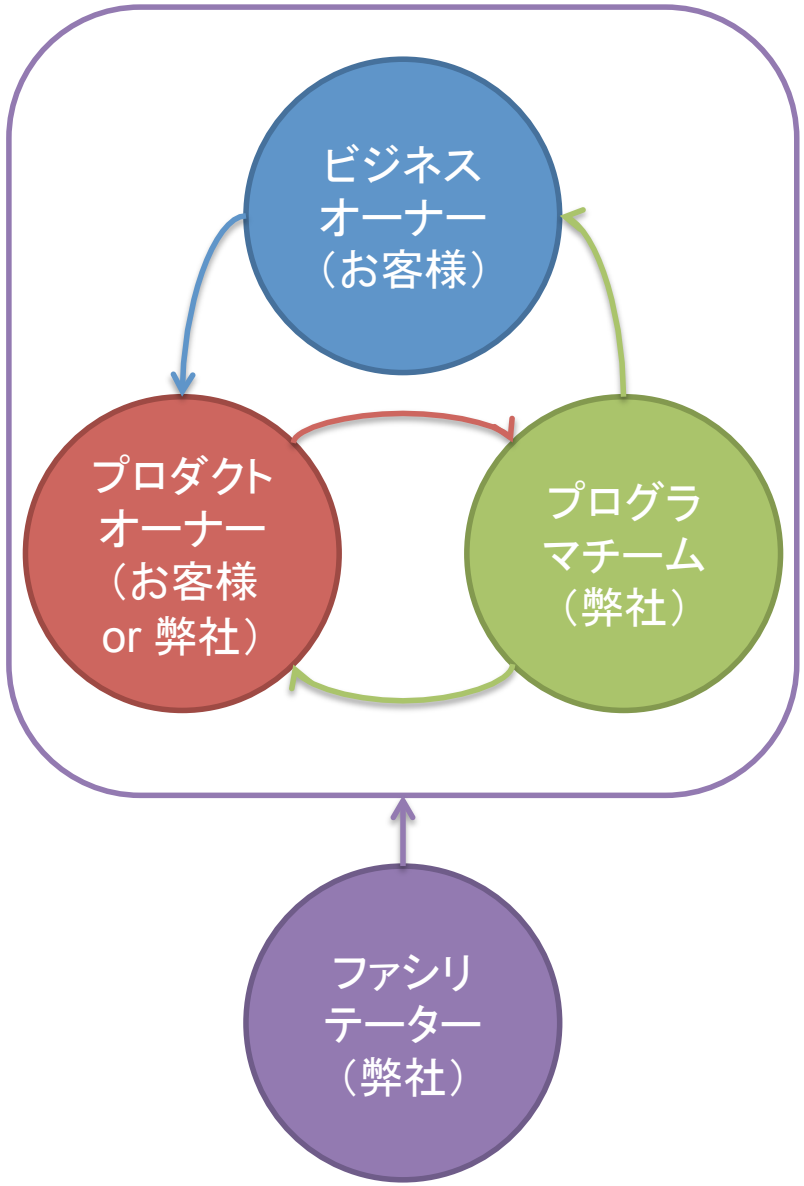
「ソフトウェアそのもの」を売るのではなく、

「ソフトウェアが使えること」を売っています

アジャイル・Ruby・クラウド（ARC）を活用したビジネスにおけるテストの実践

開発プロセスとテスト

役割分担



ビジネスオーナーの役割

- ソフトウェアで解決したい問題の設定
- ゴールおよびビジョンの設定
- 企画・マーケティング・プロモーション

プロダクトオーナーの役割

- ソフトウェアの機能およびUI/UXの設計
- 何を作れば正解か（＝仕様）の決定
- 決定権の混乱を防ぐため1名と限定します

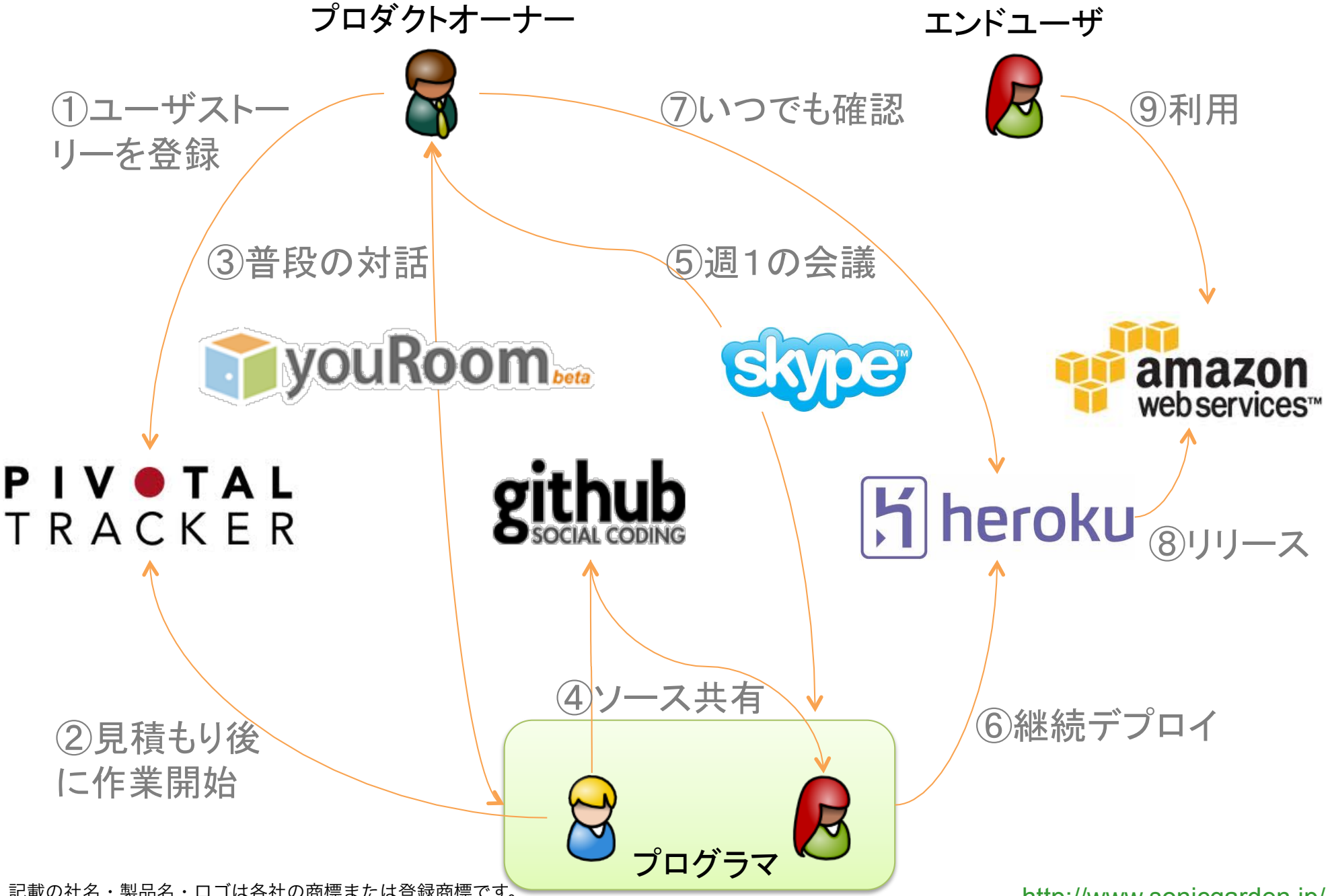
プログラマチームの役割

- プロダクトオーナーの設計の支援
- プログラミングとクラウドでの運用
- メインとサポートの2名体制で対応します

ファシリテーターの役割

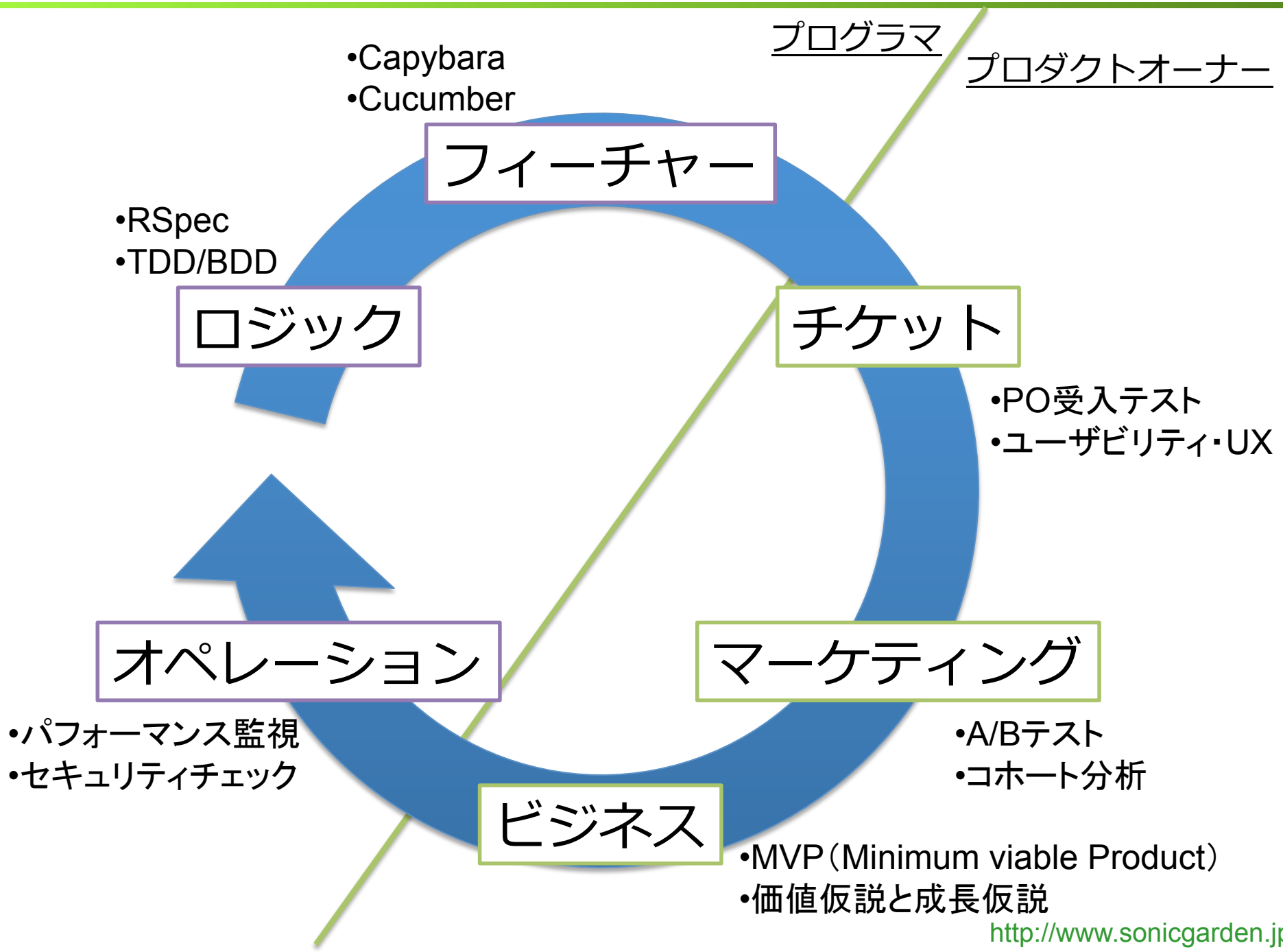
- プロジェクト進行のサポート
- 契約内容に関するご相談（営業担当）

ツールとプロセス



記載の社名・製品名・ロゴは各社の商標または登録商標です。

ソニックガーデンにおけるテスト



http://www.sonicgarden.jp/

RSpecのサンプルコード

```ruby
require 'youroom_assistant/youroom_bot_available'
require 'youroom_assistant/loggable'

class ReminderSetting < ActiveRecord::Base
  include YouroomAssistant::YouroomBotAvailable
  include YouroomAssistant::Loggable

  attr_accessible :group_id, :message, :scheduled_day
  validates :group_id, presence: true
  validates :message, presence:true, length: {maximum: 280}
  validates :scheduled_day, presence:true, inclusion: 1..31

  def self.post_reminder_for_all
    logger.info 'start post_reminder_for_all ==============================='
    settings = self.where('scheduled_day in (?)', target_days(Date.today))
    settings.each do |setting|
      setting.post_reminder_to_youroom
    end
    logger.info 'completed post_reminder_for_all ==============================
  end

  def self.target_days(today)
    today.end_of_month == today ? (today.day..31).to_a : [today.day]
  end

  def post_reminder_to_youroom
    logger.info 'start post_reminder_to_youroom ==============================
    body_hash = { 'entry[content]' => self.message }
    youroom_bot.post_entry self.group_id, body_hash
    logger.info 'completed post_reminder_to_youroom ==========================
  rescue => e
    log_error e
  end
end
```

```ruby
require 'spec_helper'

describe ReminderSetting do
  let(:youroom_bot) { double(:youroom_bot) }
  let(:setting) { create :reminder_setting }

  before do
    youroom_bot.stub(:post_entry)
    setting.stub(:youroom_bot).and_return(youroom_bot)
    setting.stub(:log_error)
  end

  describe '#post_reminder_to_youroom' do
    it 'should post message to youRoom' do
      youroom_bot.should_receive(:post_entry).once
      setting.post_reminder_to_youroom
    end

    context 'when unknown error occurred' do
      before do
        youroom_bot.stub(:post_entry).and_raise(StandardError)
      end

      it 'should write error log' do
        setting.should_receive(:log_error).once
        setting.post_reminder_to_youroom
      end
    end
  end

  describe 'self#post_reminder_for_all' do
    before do
      setting.stub(:post_reminder_to_youroom)
      ReminderSetting.stub(:where).and_return([setting])
    end

    it 'should post reminder to youRoom' do
      setting.should_receive(:post_reminder_to_youroom).once
      ReminderSetting.post_reminder_for_all
    end
  end

  describe 'self#target_days' do
    context '1st day' do
      let(:today) { Date.civil 2012, 1, 1 }
      subject { ReminderSetting.target_days(today) }
      it { should == [1] }
    end
```

RSpecからの自動生成

```
/Users/mat/dev/skip% spec -c spec/controllers/application_spec.rb
..................

Finished in 0.935321 seconds

18 examples, 0 failures
/Users/mat/dev/skip%
```

通常の出力結果
（色付きテキスト）

仕様書として出力
（HTML）

Capybaraのサンプルソース

```ruby
require 'spec_helper'

describe 'ReminderSettings' do
  before do
    visit_group_page
  end

  subject { page }

  describe 'GET new' do
    before do
      click_link 'Create Reminder'
    end

    it { should have_content('Setting on Reminder') }
  end

  def create_reminder
    select('1st', from: 'Scheduled day')
    fill_in 'Message', with: 'Hello'
    click_button 'Create Reminder setting'
  end

  describe 'POST create' do
    before { click_link 'Create Reminder' }
    context 'when inputs are valid' do
      before do
        create_reminder
      end

      it { should have_content('Setting Was Successfully Created') }
    end

    context 'when inputs are invalid' do
      before { click_button 'Create Reminder setting' }

      it { should have_content('Some errors were found') }
    end
  end
```

http://www.sonicgarden.jp/

アジャイル・Ruby・クラウド（ARC）を活用したビジネスにおけるテストの実践

品質に対するパラダイムシフト

「完成すること」から「持続すること」へ

「完成指向」から「持続可能」へのパラダイムシフト

- バグをなるべく出さないようにする

  → **バグが出てもすぐに直せるようにする**

- サーバは落ちないようにする

  → **落ちてもすぐに復旧できるようにする**

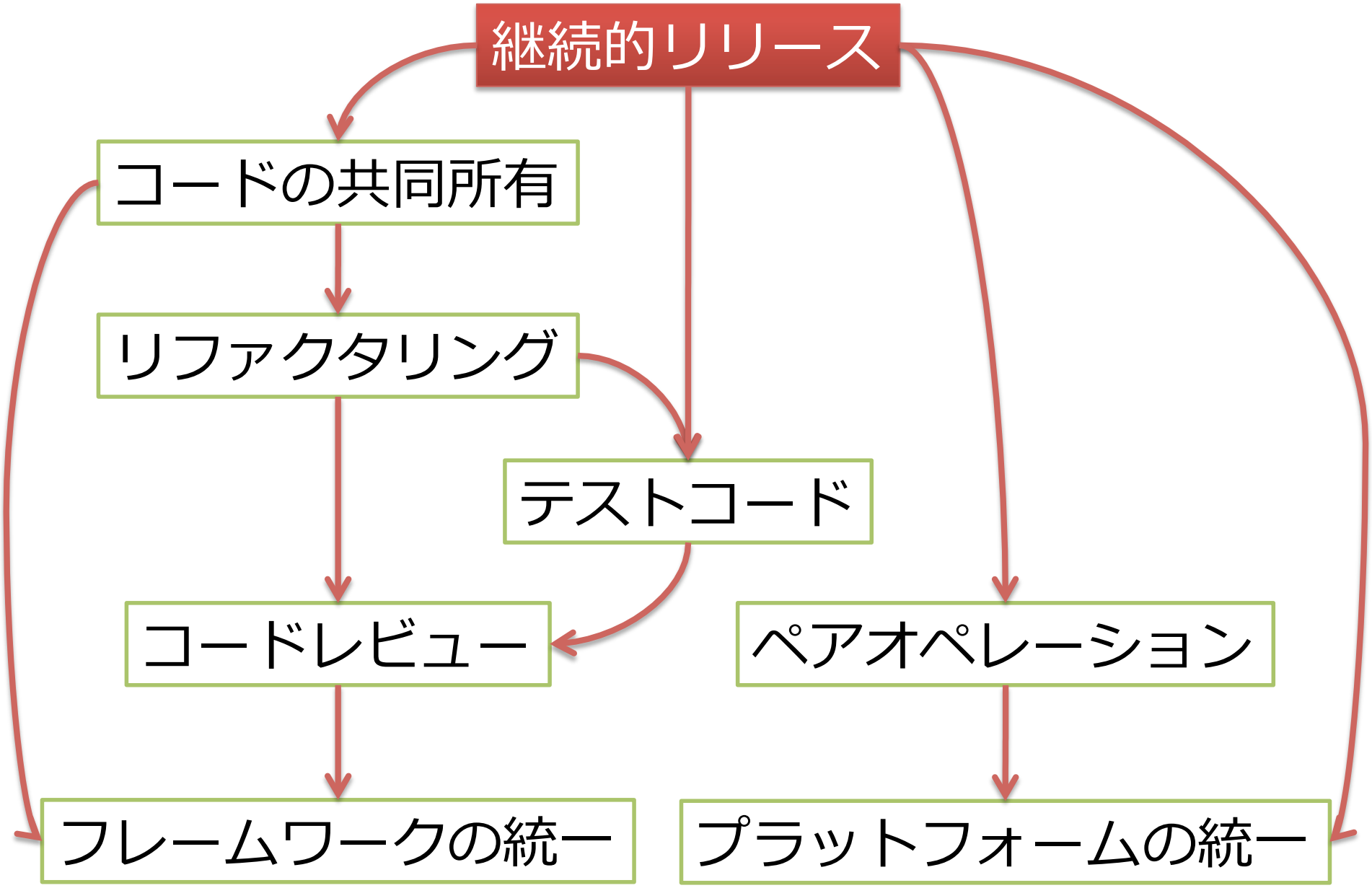
- データ変更ないように設計する

  → **データ変更しても対応できるようにする**

- ビジネスのプランを重視する

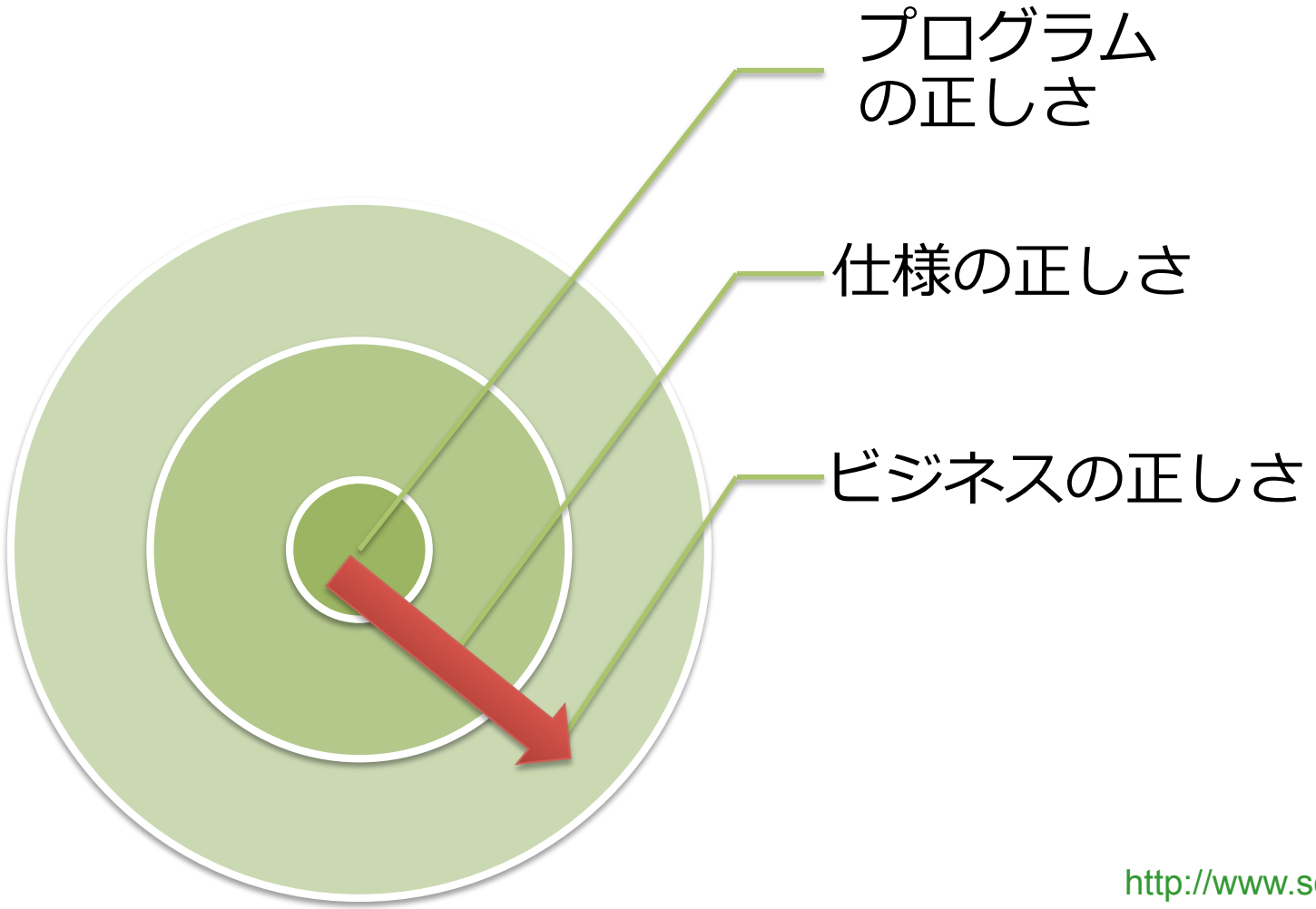
  → **ユーザのフィードバックで製品を変える**

「バグが出てもすぐに直せるようにする」ためには？



ソニックガーデンの考える「品質」とは？

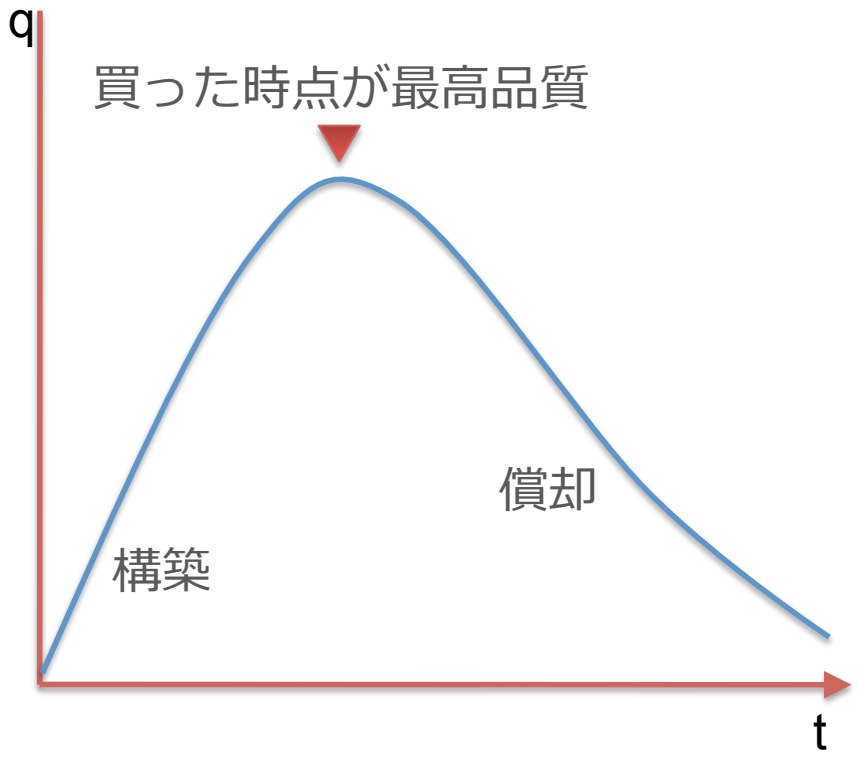
- 「誰が顧客なのかがわからなければ、何が品質なのかもわからない。」（リーンスタートアップより）
- そのテストの目的はなにか？何を売っているのか？



「Point of Sales」から「Point of Use」へ

Point of Sales

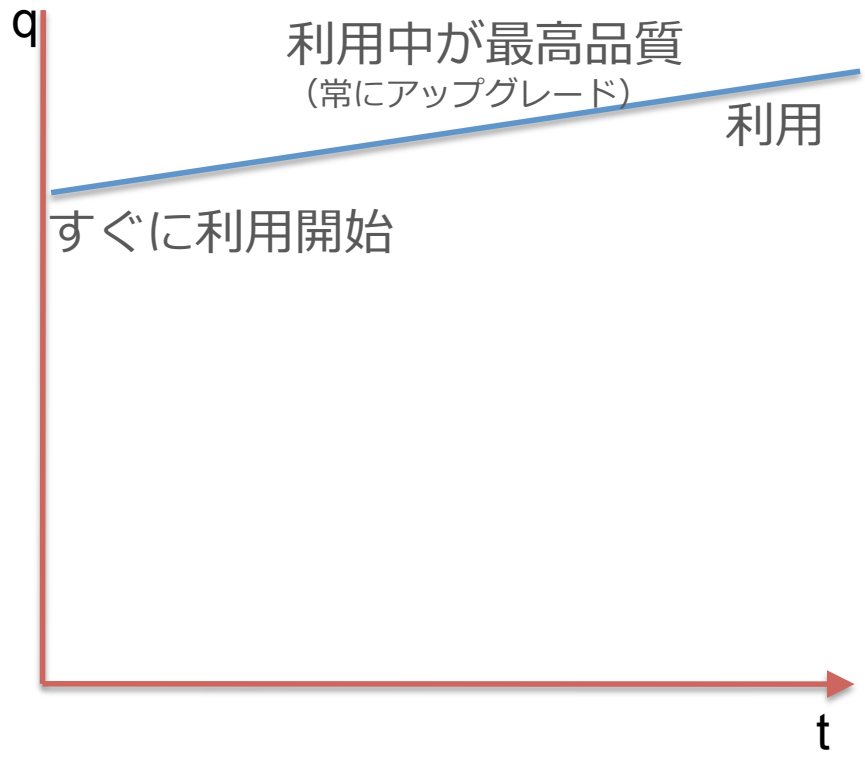
買った時点が最高で、そこから陳腐化が始まるもの



製造業

Point of Use

常に使っている時点で最高、最新のものを利用できる



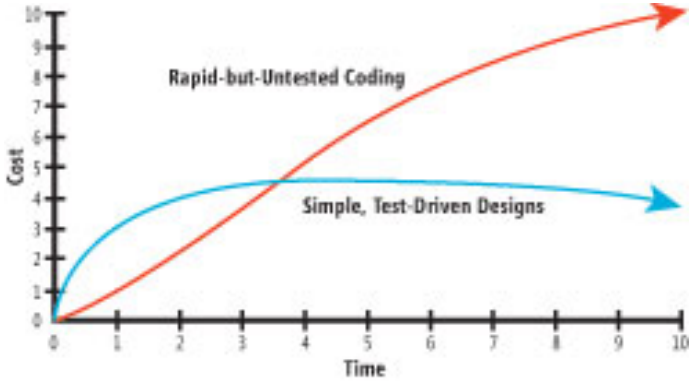
サービス業

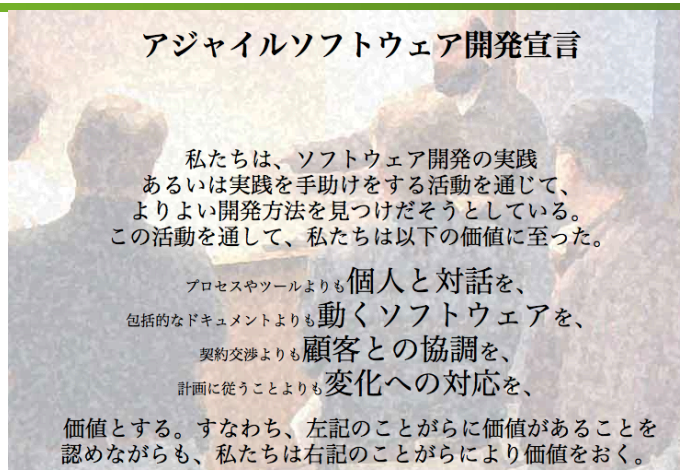
「Point of Use」で大事なこと

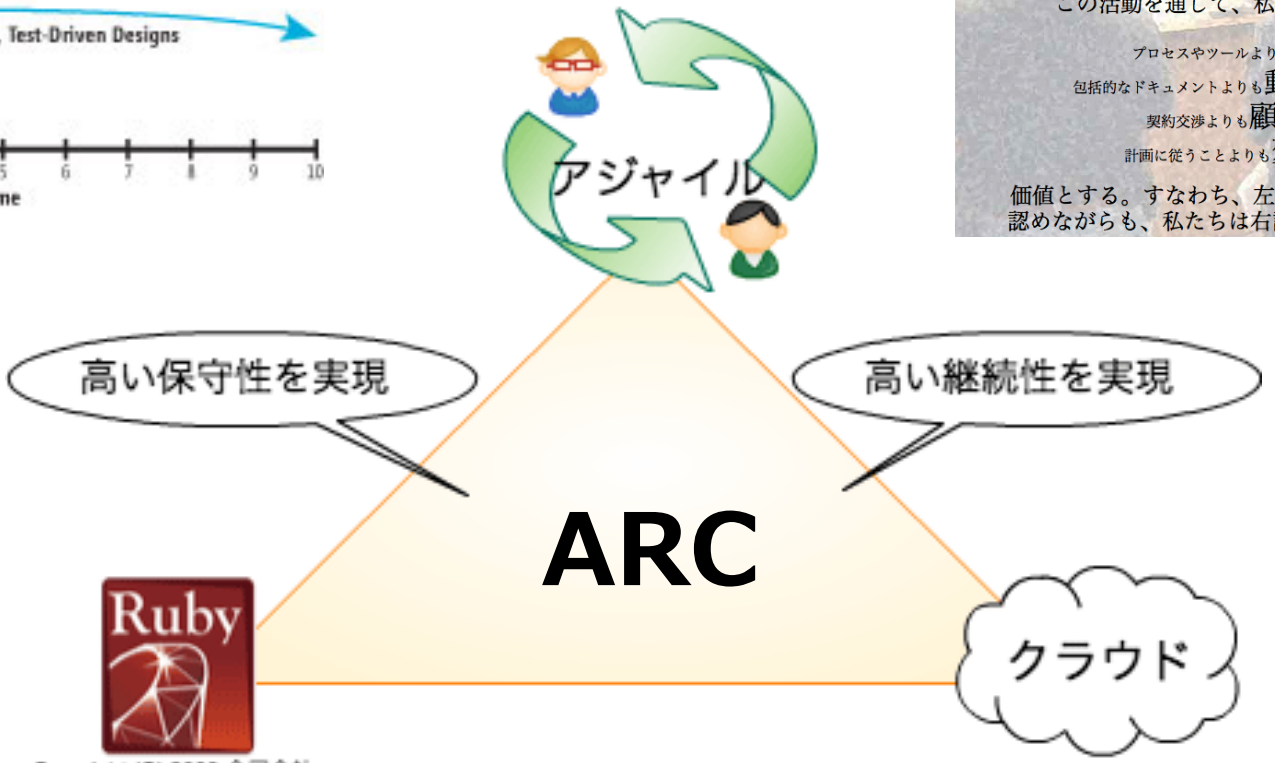
継続性

保守性

サービス型の受託開発を支える技術







Copyright (C) 2008 合同会社
Rubyアソシエーション

「DRY（Don‘t Repeat Yourself）」
「CoC（Convention over Configuration）」

http://www.sonicgarden.jp/

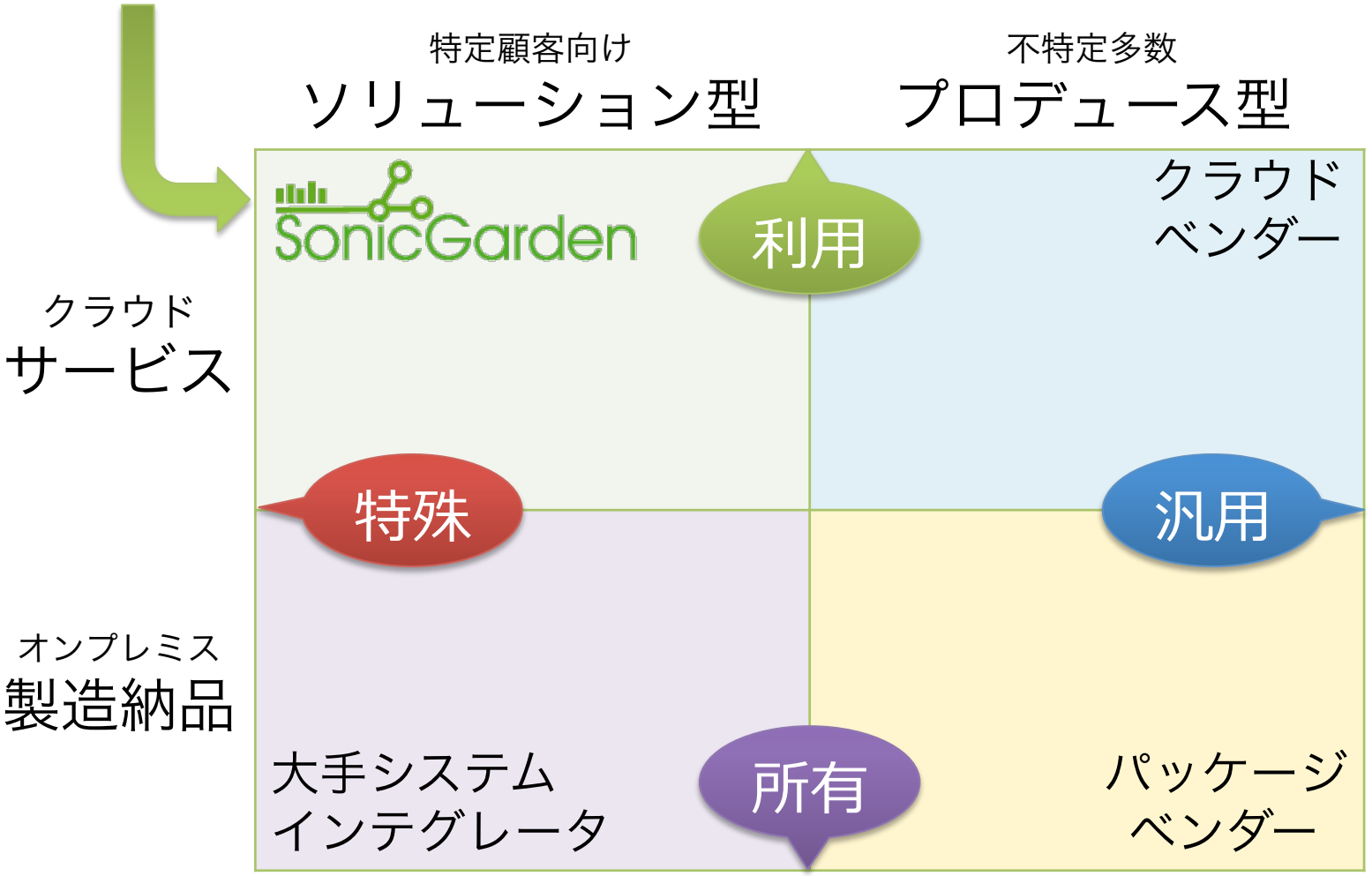
アジャイル・Ruby・クラウド（ARC）を活用したビジネスにおけるテストの実践

さいごに:ビジネスモデルを変える

ソフトウェアビジネスの分類

開発受託を行いますが、人月による見積の納品型の受託は行いません。
SonicGardenのビジネスモデルは、

納品しない受託開発（サービス型の受託開発）です。



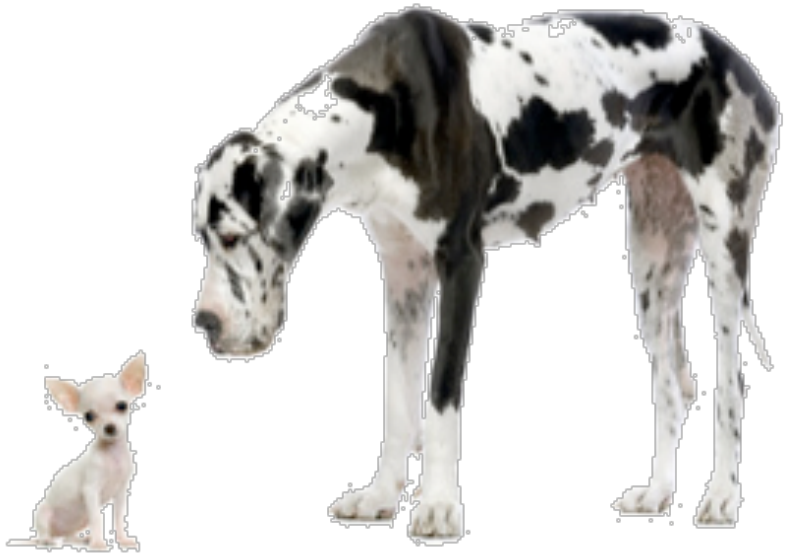
http://www.sonicgarden.jp/

ディフェンシブな開発

アジャイルは悪くないが、ビジネスモデルが問題
　予め決まった金額と要件の中で納品と検収を目指す
　利益を出すにはリスクを取らずに「守り」に入る
システムで価値を産むこと　＜　要件通りに作ること

ディフェンシブな開発がもたらす問題

- SIビジネスの本質は保険屋
  - お金さえ出せば赤字だろうと最後まで責任を持ってくれる
  - リスクの丸抱えをするだけの企業体力のみが優位性になる



- 技術者からみたSIビジネス
  - 「人月による見積もり」と「成果物の完成責任」の捻れ？
  - プログラマの生産性が上がれば上がるほど売上が下がる？

「ディフェンシブな開発」から脱却を

- IT投資のコストパフォーマンスの悪さ
  - 1）ベンダの見積りにおける過重なバッファ
  - 2）開発と運用で別の業者を利用していること
  - 3）最大時を想定したハードウェアを購入していること

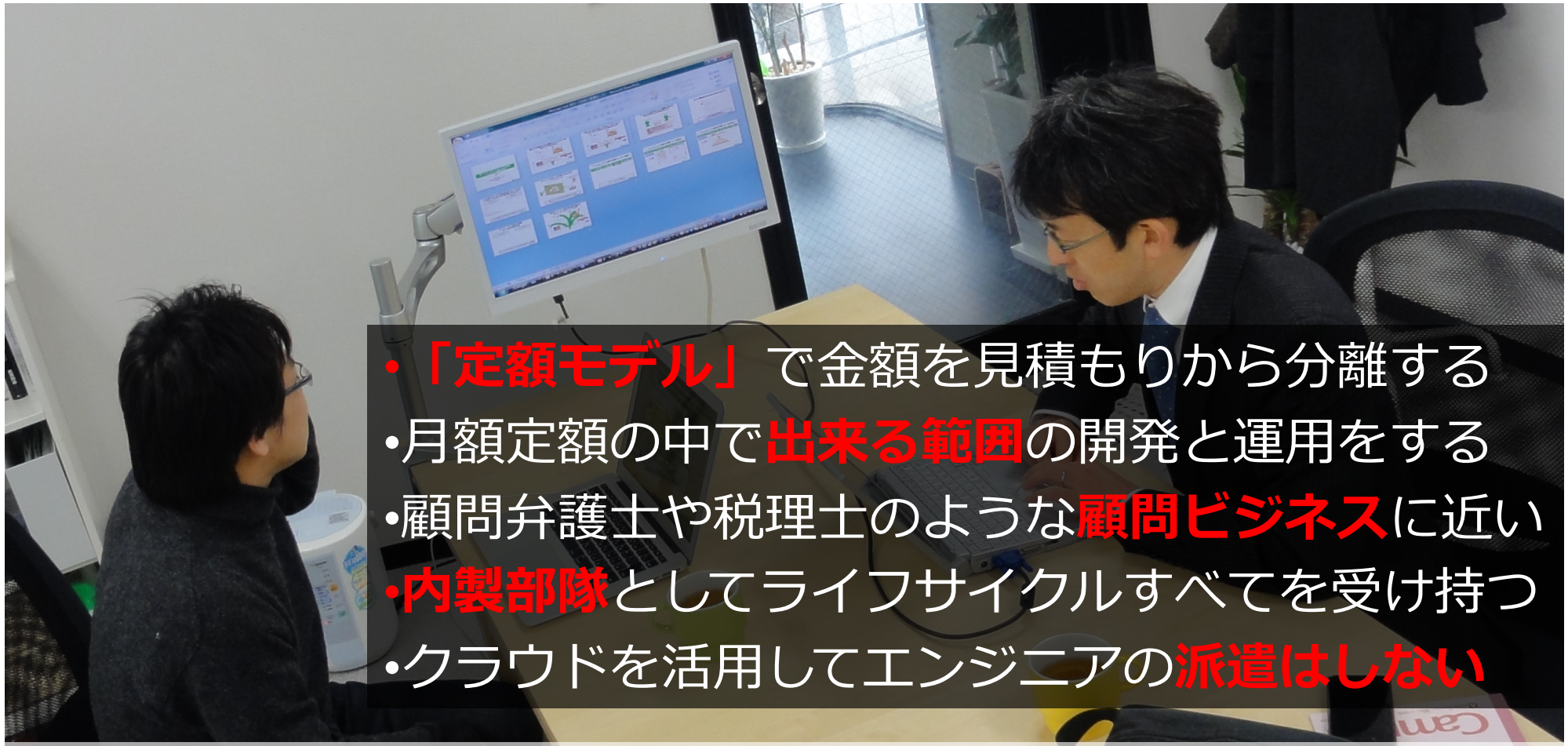
「一括での発注・請負」が原因

>>　仮説「納品しない」

ソフトウェアパートナーシップモデル

「オーダーメイドの受託」と

「納品しないこと」と「派遣しないこと」の両立

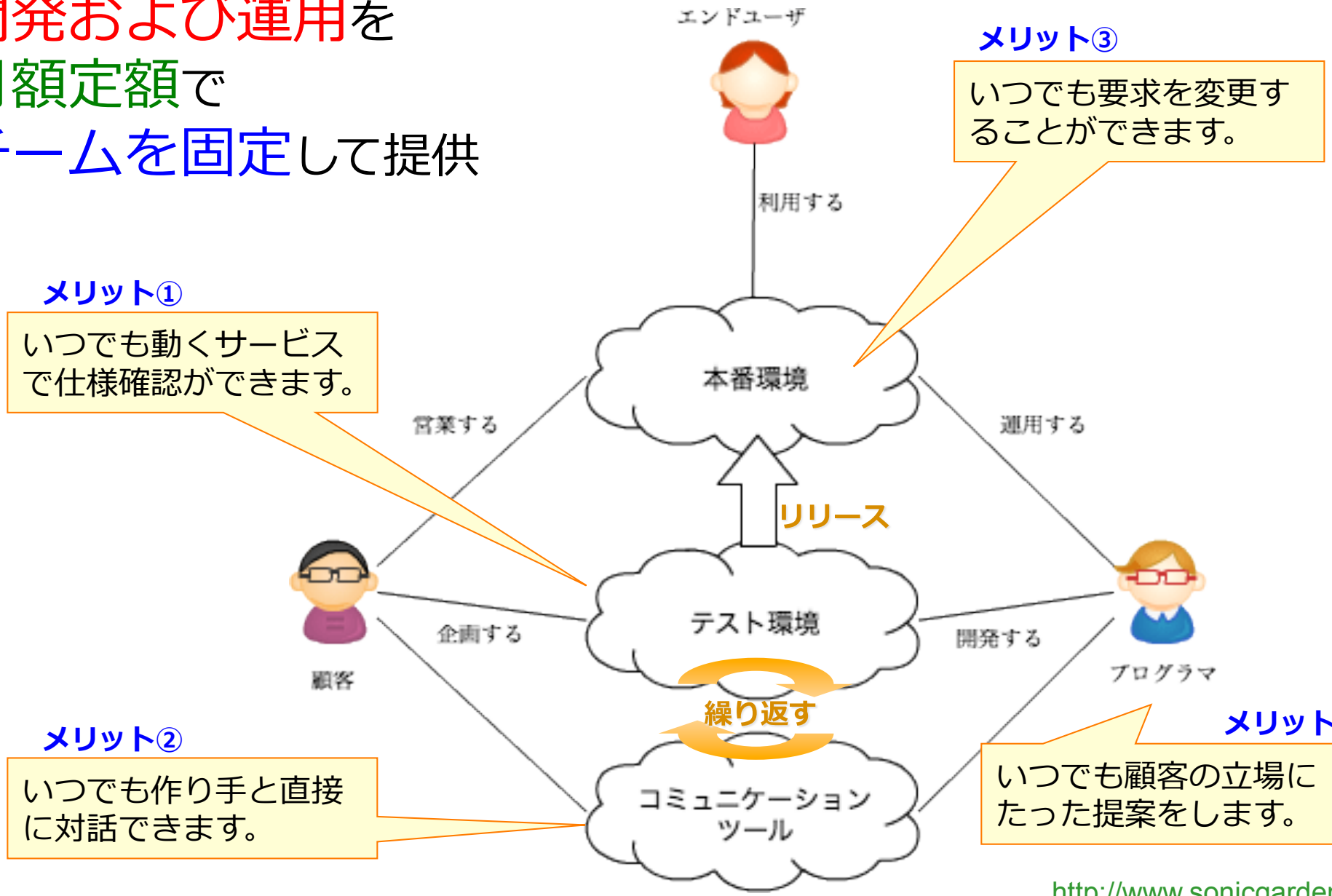


- **「定額モデル」**で金額を見積もりから分離する
- 月額定額の中で**出来る範囲**の開発と運用をする
- 顧問弁護士や税理士のような**顧問ビジネス**に近い
- **内製部隊**としてライフサイクルすべてを受け持つ
- クラウドを活用してエンジニアの**派遣はしない**

目指す姿は、リスクを理解した**お客様**と、**プログラマ**が直接対話し、お客様のために
オーダーメイドで、**価値を産む**ソフトウェアを提供するスタイル

サービス型の受託開発とは

クラウドで使えるソフトウェアの
開発および運用を
月額定額で
チームを固定して提供



メリット①
いつでも動くサービスで仕様確認ができます。

メリット②
いつでも作り手と直接に対話できます。

メリット③
いつでも要求を変更することができます。

メリット④
いつでも顧客の立場にたった提案をします。

http://www.sonicgarden.jp/

本日のテーマ

「テストの理想を現場視点で実現する」

「テストの目的をビジネス視点で考える」

さいごに:ソニックガーデンのミッションとビジョン

Embrace Change

変化を受け入れ変わっていくこと

Fearless Change

変化を恐れず自ら変えていくこと

Social Change

自らの変化を周囲に広げていくこと

ありがとうございました

続きはブログで…

http://kuranuki.sonicgarden.jp

株式会社ソニックガーデン
SonicGarden

kuranuki@sonicgarden.jp