

# Development Baseline

Archway Inc.  
Consulting Service

<http://www.archway.co.jp/>

# Self Introduction

- 中西 庸文(Tsunefumi Nakanishi)
  - 株式会社アークウェイ大阪事業所 所長
    - コンサルではなくて**メンター**を目指している。
    - 一歩前から業界を引っ張って行くのは優秀な方々におまかせして、僕は一歩後から業界を**底上げ**していきたい。
    - .NETが初めてこの世に出る前からAgile開発を実践してきた経験を活かして、**現場**の方々とカイゼンを行っていきたい。

# Agile VS Waterfall

アジャイル 対 ウォーターフォール

# What The Fuck Is That?

- AgileはWaterfallを敵にしないと正当性を証明できないほど**無力**なのだろうか？
- Waterfallの**歴史**から学ぶべきところはないのだろうか？
- プロジェクトがうまく回らないことに対する**言い訳**に開発方法論を持ち出していないだろうか？
- とは言うものの、そもそも開発方法論の**適用**がそんなに大事なことなのだろうか？

# Domain Specific Methodology

ドメイン特化方法論

# In Your Culture

- 本当に大事なものは、**私たちの開発チーム**というドメインに特化した方法論。
- ベースの方法論はAgileになるかもしれないし、そうじゃないかもしれない。
- 重要なのはベースの方法論ではなく、自分たちで**ふりかえり**を行い、**KAIZEN**した結果として最適化された**持続可能なプラクティス**の集合があるかどうか。
- 自分たちの**文化**に適合させなければ継続することはできない。



# Believe Yourself

もっと自信を

# Do The Right Thing

- 自分たちのチームにとってベストプラクティスならそれは**正解**。
- **やるべきこと**をきっちりこなしていれば、自信は後からついてくる。
- 自信があれば、他のやり方を**非難すべき理由**などあるはずもない。
- ただし自分をいつもしっかりと**コントロール**していなければ自信は逃げていってしまう。

# Manage By Ourselves

自分たちのために管理する

# Change Your Mind

- 実は他人に管理**されている**方が楽。
- でも本当に管理すべきなのは**自分自身**。
  - 自分自身の**スキル**を管理せよ。
  - 自分自身の**コミュニケーション能力**を管理せよ。
  - 自分自身の**無駄**を管理せよ。
  - 自分自身の**姿勢**を管理せよ。
- 自分たちの管理もできていないのに**ソフトウェア開発**を管理できるのか？

# Self Management Skill

自己管理スキル

# Five Skill Sets

- プロジェクトを管理するスキル
- ソフトウェアの構成を管理するスキル
- 自動化によって無駄を排除するスキル
- すぐれた設計を行うスキル
- 開発の基盤を構築するスキル

# **It's Old School, It Ain't New School**

目新しいことなど何もない

# Look At You!

- ソフトウェア開発において、やらなければいけない**あたりまえ**のこと。
- あたりまえのことが**きっちり**とできていますか？
- 本当の気づきとは既知であると思っていた**基本を見つめなおす**作業の過程で得られるものである。
- うまくいかない理由に対する言い訳を探すよりもまず**地に足のついた開発**が行えているかどうかをふりかえるべき。

# Make A Decision First

変化するために、まず決める

# Just Do It

- 何かを**決定**しなければ、変化することはできない。
- 決めて実行すれば、変えなければいけない**本質**もおのずと見えてくる。
- スタートからゴールまでの間にやるべきこと(**目的**)を明確にし、その実現方法(**手段**)を導き出す。

# Development Baseline Overview

デベロップメント ベースラインの全貌

# What To How

- Development Baselineとは、ソフトウェア開発のライフサイクルにおいて真にやるべきこと(**What**)から最適な実現方法(**How**)を決定すること。
- ただしツールにこだわりすぎてはいけない。手段に縛られずに**目的に縛られるべき**。
- 時として、目的に合わなくなったお気に入りのツールを**捨てる勇気**も必要。

# Five Backbones

- プロジェクト管理
  - **P**roject **M**anagement
- ソフトウェア構成管理
  - **S**oftware **C**onfiguration **M**anagement
- 自動化
  - **P**roject **A**utomation
- テスト駆動開発
  - **T**est **D**riven **D**evelopment
- ゼロ機能リリース
  - **Z**ero **F**eature **R**elease

# Project Management

- プロジェクト管理でやるべきこと
  - **タイムボックス管理**
    - フィーチャ プランニング
    - リリース プランニング
    - イテレーション プランニング
  - **進捗のトラッキング**
    - タスク
    - バグ
    - 要望
  - **情報の共有**
    - プロジェクトに関する情報は一個所に

# Software Configuration Management

- ソフトウェア構成管理でやるべきこと
  - **リポジトリ**
    - メインライン
    - リリースライン
    - サンドボックス
  - **プライベート ワークスペース**
    - 自分の作業場所を分離して変更を管理
    - マージを受け入れる勇気
  - **タスクレベルのコミット**
    - 粒度の細かいタスクが完了するたびに、こまめにコミットする癖をつける

# Project Automation-Part1

- 自動化(自動ビルド)でやるべきこと
  - 1ステップビルド
    - 毎回1ステップでソフトウェアをゼロからビルド
  - 移植可能なビルド
    - どのワークステーションにおいても同じ結果となるビルドの生成
  - 結合に対する自信
    - 成功するビルドを簡単に生成できることで、結合時の不安を解消
  - ビルド手順の明確化
    - 手作業のビルドで発生しがちな単純なミスを解消

# Project Automation-Part2

- 自動化(継続的インテグレーション)でやるべきこと
  - 定期ビルド
    - リポジトリの変更を検出して自動ビルド
    - スケジューリングによる自動ビルド
  - 詳細なレポート
    - ビルド結果
    - テスト結果
    - コードカバレッジ
    - コーディング規約
    - コードの重複
    - etc..
  - ビルド結果通知
    - Emailによる通知
    - Extreme Feedback Devicesによる通知

# Test Driven Development-Part1

- テスト駆動開発(カスタマー テスト駆動)でやるべきこと
  - ソフトウェアの内部構造と分離されたテスト
    - 顧客要求の実現方法とソフトウェアの内部構造は分離すべき
  - 要求ドリブンによる進捗管理
    - 失敗するカスタマー テストを先に作成し、成功したテストの数がそのまま進捗に
    - 顧客のビジネス価値に注目した進捗管理

# Test Driven Development-Part2

- テスト駆動開発(デベロッパー テスト駆動)でやるべきこと
  - 開発のハートビート
    - レッド
      - 新しい機能に対してテストを書く
    - グリーン
      - テストが成功するような最低限の実装を行う
    - リファクタリング
      - テストが通る状態のまま、内部構造の改善を行う
  - 設計手法の確立
    - テスト駆動開発はテスト技法ではなく設計手法
    - 目標は、動作するきれいなコード
  - 自己テストコードとしての再利用
    - テスト駆動開発において作成されたテスト群は自動ビルド時に優れた自己テストコードとなる

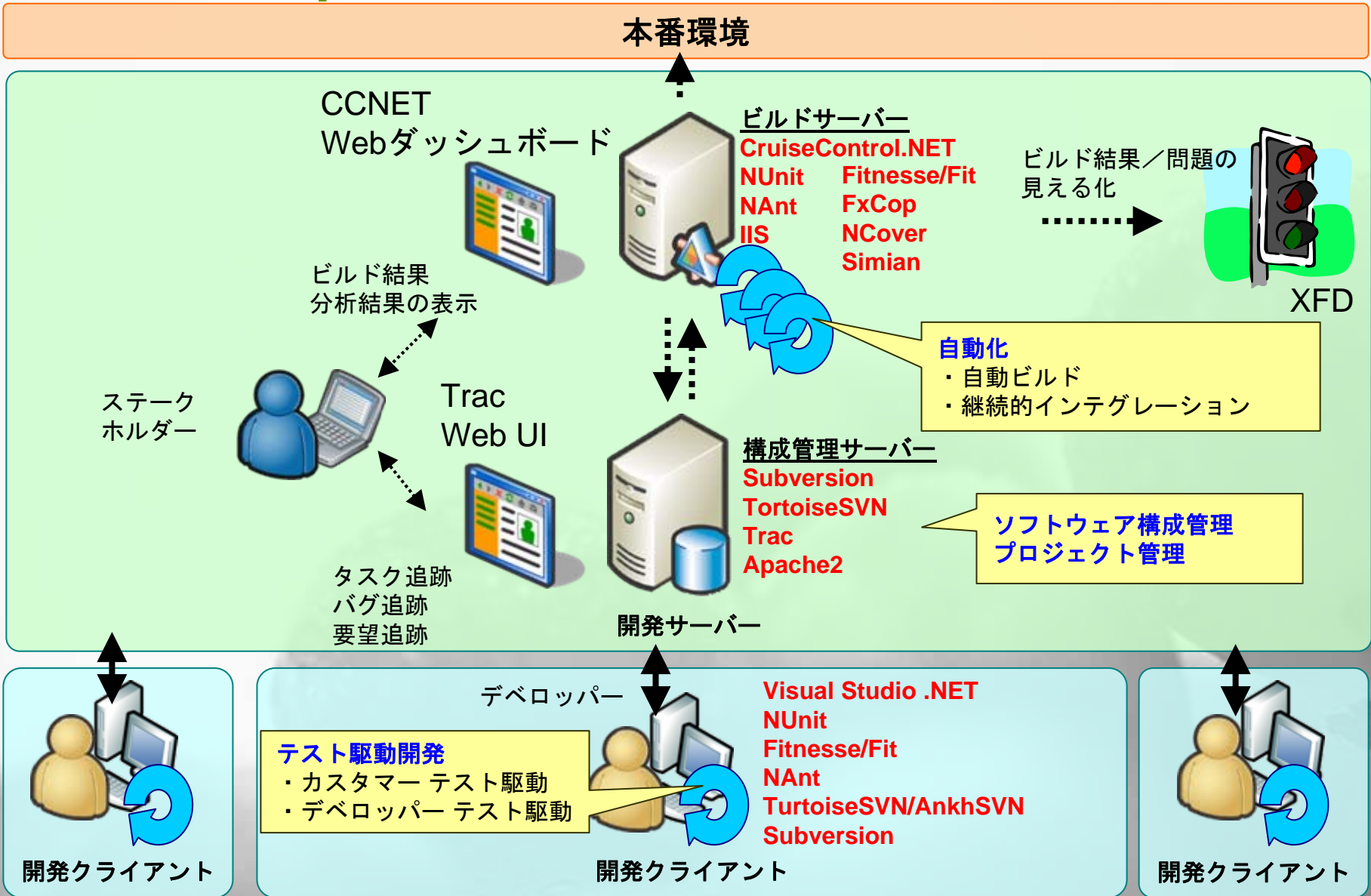
# Zero Feature Release

- ゼロ機能リリースでやるべきこと
  - **基盤の構築**
    - テスト基盤
    - 自動化基盤
  - **フィーチャの抽出**
    - メインフィーチャ
    - ハイリスクフィーチャ
  - **アーキテクチャ プロトタイプの作成**
    - メインフィーチャからプロトタイプを作成
    - ハイリスクフィーチャを検証
  - **デベロッパー トレーニング**
    - 躰(自分たちのことは自分たちで)

# Development Baseline Implementation

デベロップメント ベースラインの実装

# Implementation in .NET



# Development Baseline Is The Best Thing To Do, You Know What I'm Sayin'?

デベロップメント ベースラインこそが  
今まさにやるべきこと

**ご清聴**

**ありがとうございました。**