



ファッションと実践

druby.org

関将俊



今日の話

- 私について
- ファッションの話
- 実践のふりかえり
- メニューたくさん
- まとめ

●●● 私について

- 自称アーティスト
- サラリーマン

○●● 自称アーティスト

- 自分のためのプログラミング
- アーティストとしてのプログラマ
- わかって欲しい人にウケるため
- オープンソースへの愛や正義には興味がない

○ ● ● Rubyと私

- 20世紀の終わり頃から
- ERB, **dRuby**, Rinda, RWiki,
- 2000年のPerl/RubyConference
- 書籍・雑誌



重要

- 2005年夏
- まだ初刷買えます!
- 初刷5周年!!





今年もRubyKaigi

- 2006 dRuby, Again.
- 2007 Answering dRuby and Rinda
- 2008 erbを偲んで
- 2009 偉大なBigTableとぼくのおもちゃ
- **2010 RWikiと怠惰な私の10年間**

○●● ちゃんとした会社員

- 製品のためのプログラミング
 - お仕事用のコード
- eXtreme Programmingだったもの
- プロセス・品質系イベント
 - 多くの場合、悪役としてご招待

悪役として

- JaSST'04 - 忍者式テスト
- XP祭り04 - XPと計画ゲーム
- デブサミ05 - 複数バージョンを同時に扱う
- JaSST'06 - テストレート
- ソフトウェア品質シンポジウム06 - 自分たちのもの宣言
- JaSST'07 - 規模
- デブサミ'08 - 片山さんとのコラボ
- SQiP'08 - テスト計画
- デブサミ'09 - テストに関する雑談

実践の報告ばかり

- 具体的な問題に対する解決ばかり
- 地味で慎ましい生活
 - 上京する旅費にも困窮
 - 講演に呼ばれたお金で講演を聴く
 - 自転車操業

● ● ● まとめ1

- 初刷5周年である
- 自転車操業である



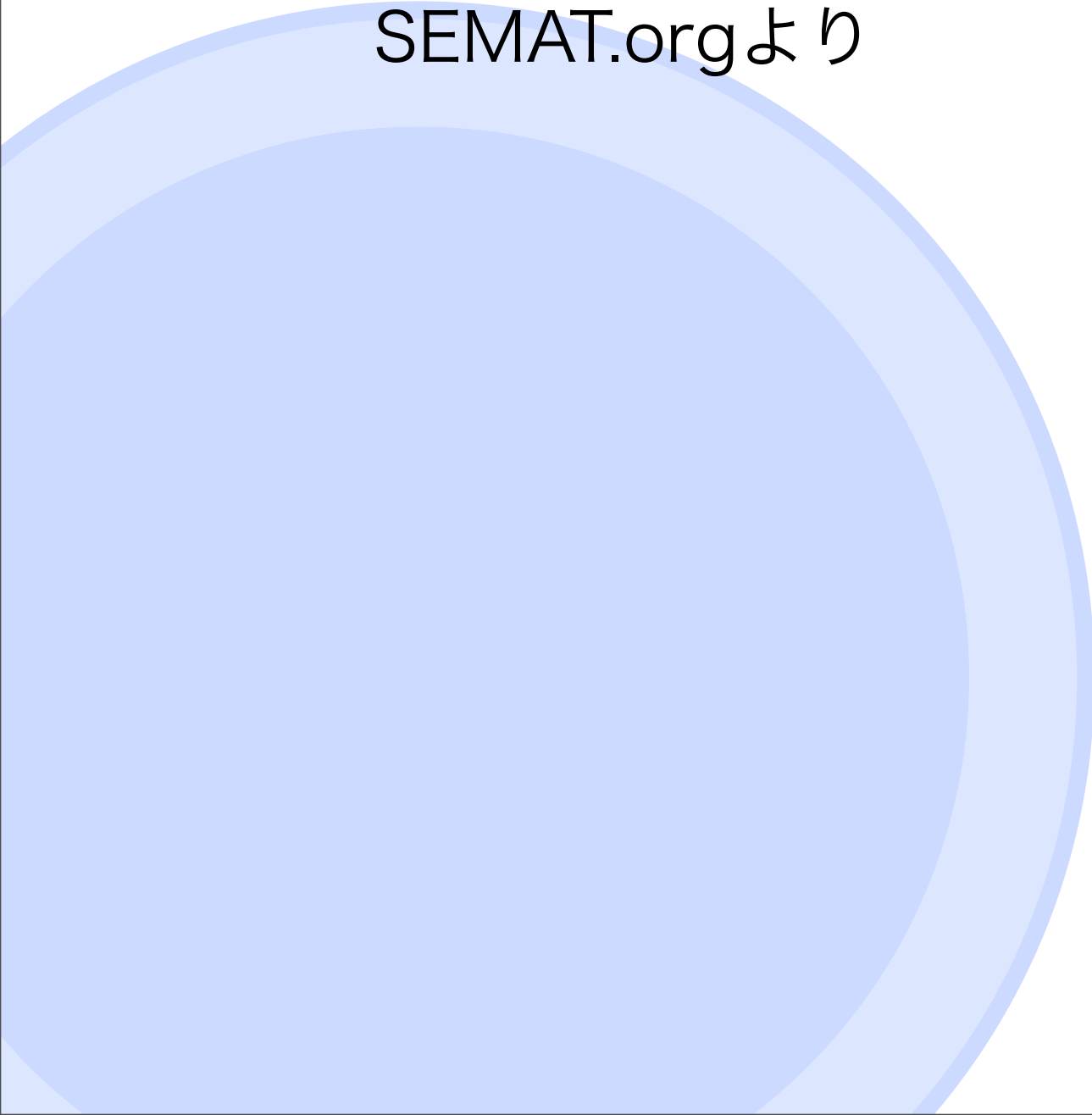
しばらくお待ちください

- ✓ 初刷5周年
- ✓ 貧乏をアピール
- ✓ 自称アーティスト
- ✓ +02分



ファッションの話

SEMAT.orgより



○ ● ● SEMAT.org

- ソフトウェア工学の再建
- 平鍋さんのblog - An Agile Way

ソフトウェア工学の方法

論と理論

ソフトウェア工学の方法論と理論 (SOFTWARE ENGINEERING METHOD AND THEORY – A VISION STATEMENT)

By Ivar, Bertrand and Richard
(日本語翻訳¹ : 平鍋健児)

1. 目的とスコープ(Purpose and scope)

Semat の最初の Call for Action では、今後 Semat initiative が扱おうとしている問題の概要を定義する。

Call for Action

ソフトウェア工学(Software Engineering)は未成熟なプラクティス(immature practices)によって、重大な障害(gravely hampered)を今日受けている。例えば、具体的には以下のように:

- 言葉の流行が、工学の一分野というよりファッション業界のようだ。
- しっかりした広く受け入れられた、理論的基礎の欠如。
- 非常に多くの方法論(methods)とその派生。またそれらの違いがほとんど理解されずに作為的に強調されている。

●●● 難しい!

- すみません。ほとんどわかりませんでした…
- 俺のやり方で再定義する!?
- キャッチーなコピーはよくわかる

●●● キャッチーなコピー

- 今日のソフトウェア工学は未成熟なプラクティスによって重大に阻害されている。
- 言葉の流行が多く、工学というより**ファッション業界**のようだ。
- すごい! 誰が言ってんの?



署名した人々

Pekka Abrahamsson

Scott Ambler

Victor Basili

Jean Bézivin

Dines Bjorner

Barry Boehm

Alan W. Brown

Alistair Cockburn

Larry Constantine

Bill Curtis

Donald Firesmith

Erich Gamma

Carlo Ghezzi

Tom Gilb

Ellen Gottesdiener

Sam Guckenheimer

David Harel

Brian Henderson-Sellers

Watts Humphrey

Capers Jones

Martin Griss

Ivar Jacobson

Philippe Kruchten

Robert Martin

Stephen Mellor

Bertrand Meyer

James Odell

Meilir Page-Jones

Dieter Rombach

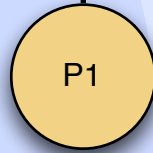
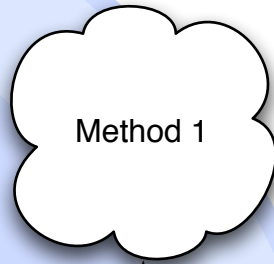
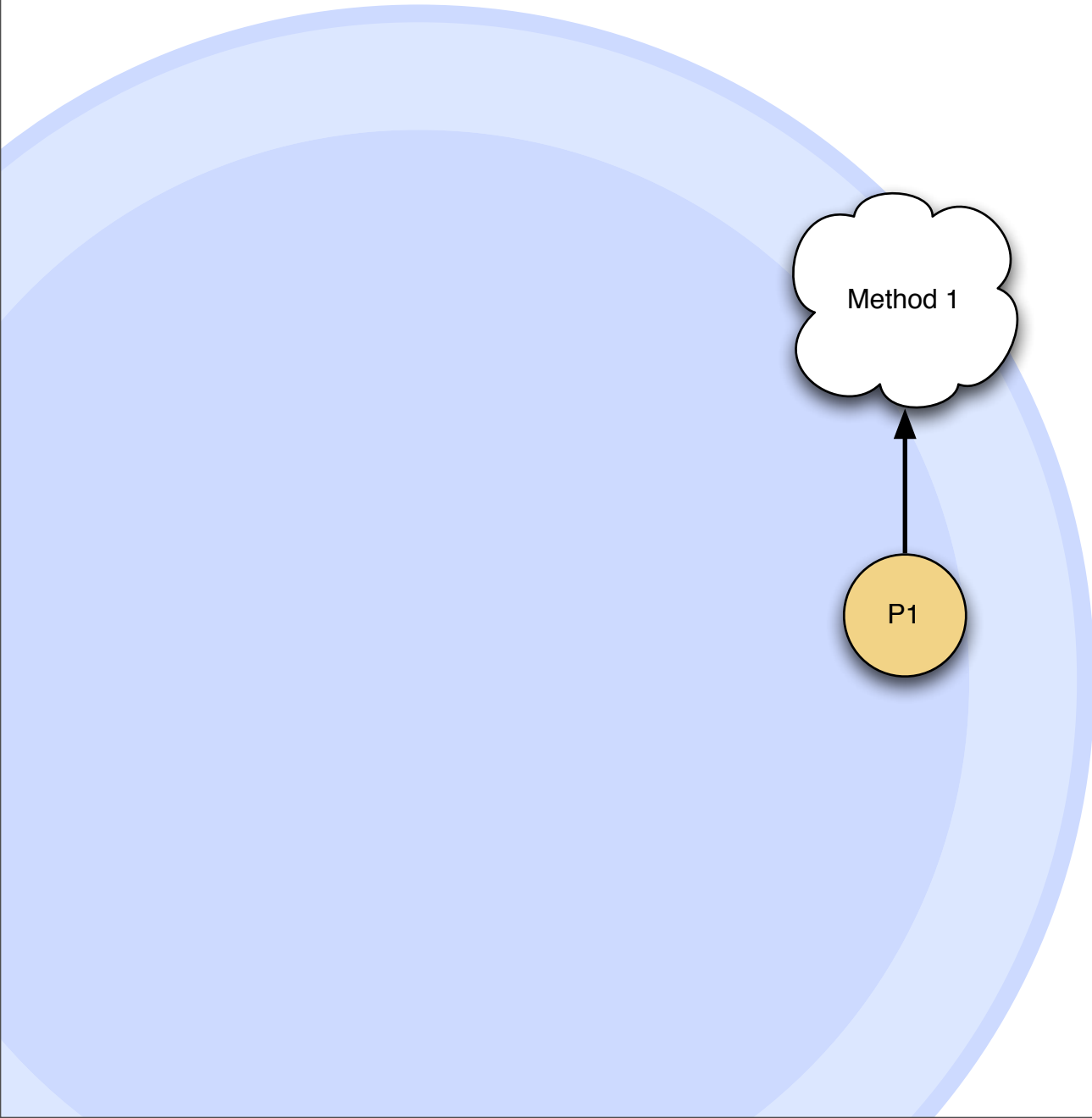
Ken Schwaber

Alec Sharp

Richard Soley

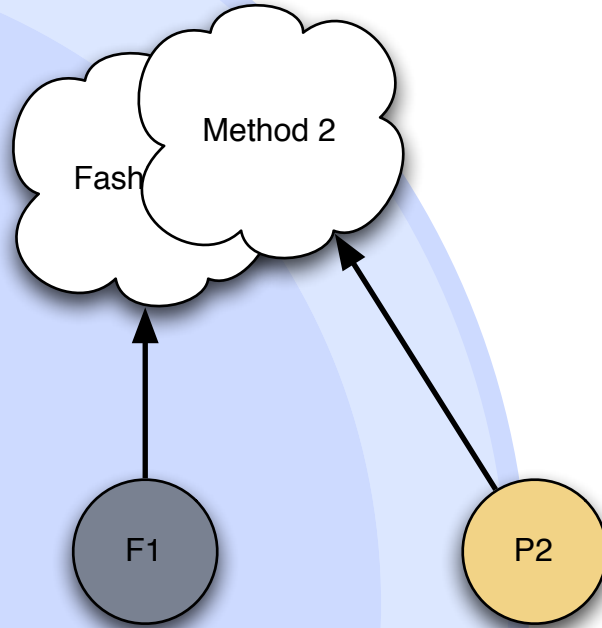
ファッションリーダー

- なんかおかしい
- ファッションリーダーの再帰的定義
- 「あいつはファッションリーダー。俺は違うぜ。」と言った人が次世代の人にファッションリーダーと呼ばれる



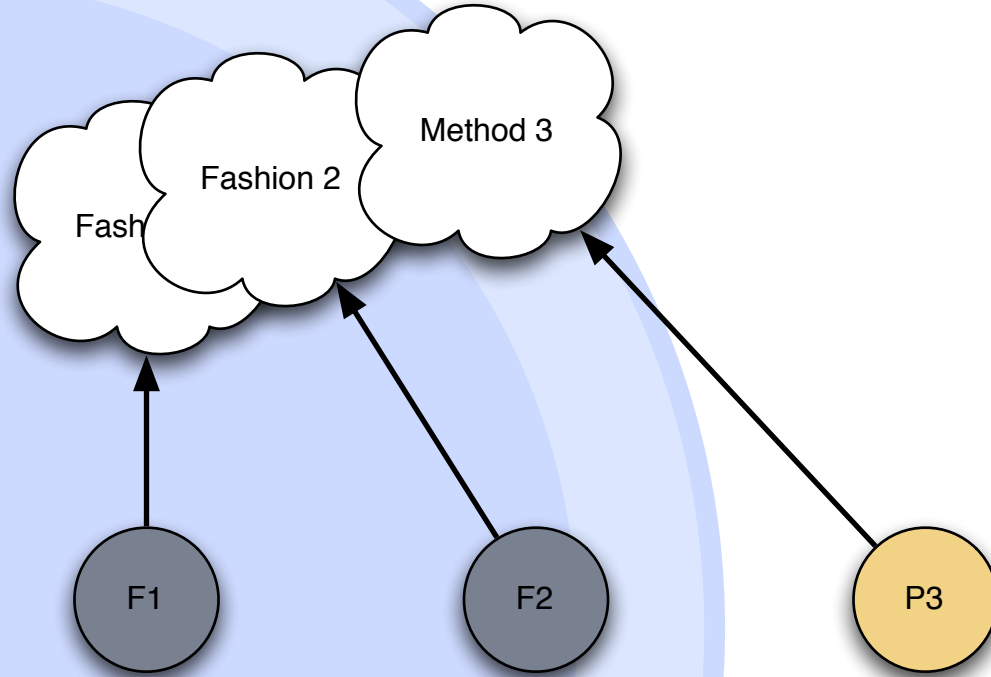


奴らはファックションだ



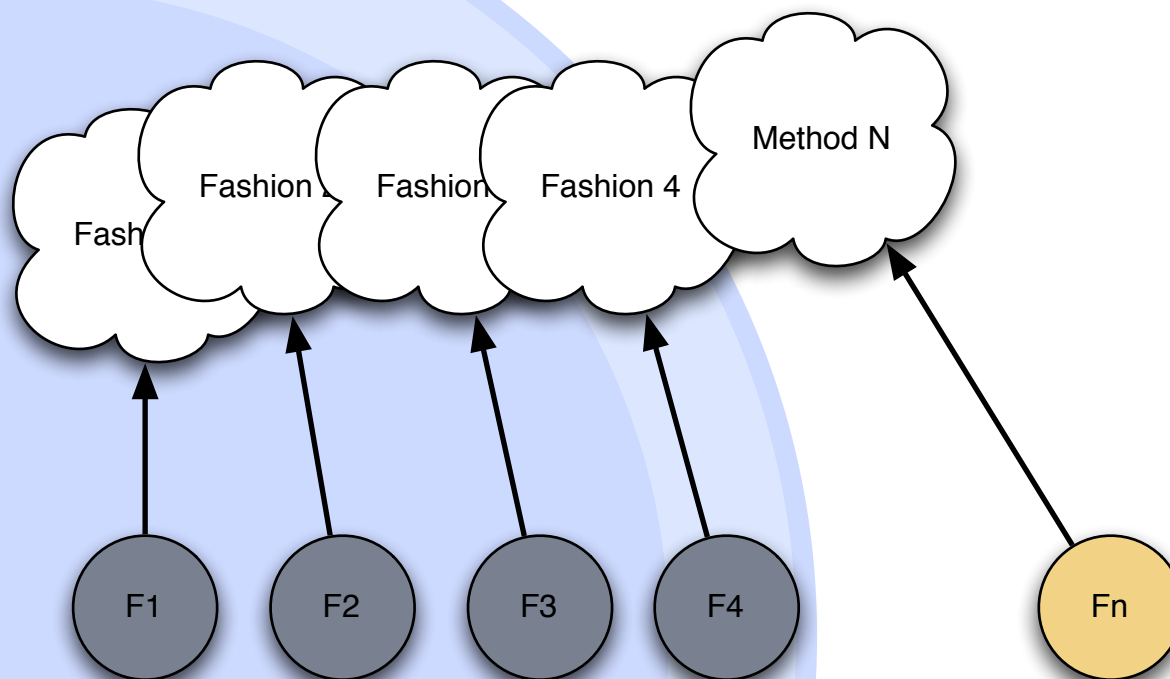


お前もファッションだ





再帰的構造



● ● ● まとめ2

- ソフトウェア工学はファッション業界
- 再建しようとする人々がいる
 - 素人にはファッションリーダー達に見えるよ…
- 再帰的な構造を持つ



しばらくお待ちください

- ✓ B Mayerは好き
- ✓ 署名者のファンに注意
- ✓ バカだと思われないように
- ✓ +07分

実践のふりかえり

実践を概念化してファッションリーダーになる
試み

● ● ● これまでの失敗

- あるチームの具体的な事例ばかり
- 問題に依存している解決策であった
- どこでもインストール可能な商品としてのモデルが必要
- モデルを売って儲けるべき

○●● モデル

- 本物ではない
- 模型
- ある視点における本質
- 捨象と抽象

○●● 方法論屋さんの悩み

- 具体的な問題を具体的に解決
- 成功をある視点でモデルにする
- モデルで説明する
- このモデルでやれば成功するぜ!!とつい
言ってしまう
- モデルの商品化、コンサルの誘惑

● ● ● 今日の作戦

- 幸い事例報告はたくさんしてる
- 事例をモデル化して稼ぐ
- これまでの報告をふりかえり、丁寧に説明しますので信者になってね



メニュー

- 忍者式テスト
- 計画ゲーム
- テストのレート
- メタプロセス
- ロールプレイング

○ ● ● 忍者式テスト

JaSST'04

成長する受け入れ試験とその実施

○●● 忍者式テスト

- TDDの自然な拡張
- TDDは知っていますね？



TDD

- テストに駆動された開発
- プログラミングの様式
- テストファースト
- テストスイート
- xUnitによる自動化されたテスト



TDD

- いまやることをテストとして記述
- そのテストをパスする実装を書く
- これまでのテストを全部やり直して壊れてないことを確認

とちぎテストの会議

- Twitterで行われたTDDにまつわるいろんな議論を再燃させる有料イベント
- 参加費 50円
- 交通費別。最寄駅は西那須野

○●● 受け入れテストで

- TDDをやったらどうなるのか
- とりあえずヒトがやる
 - 自動化？ どうでもいいよ



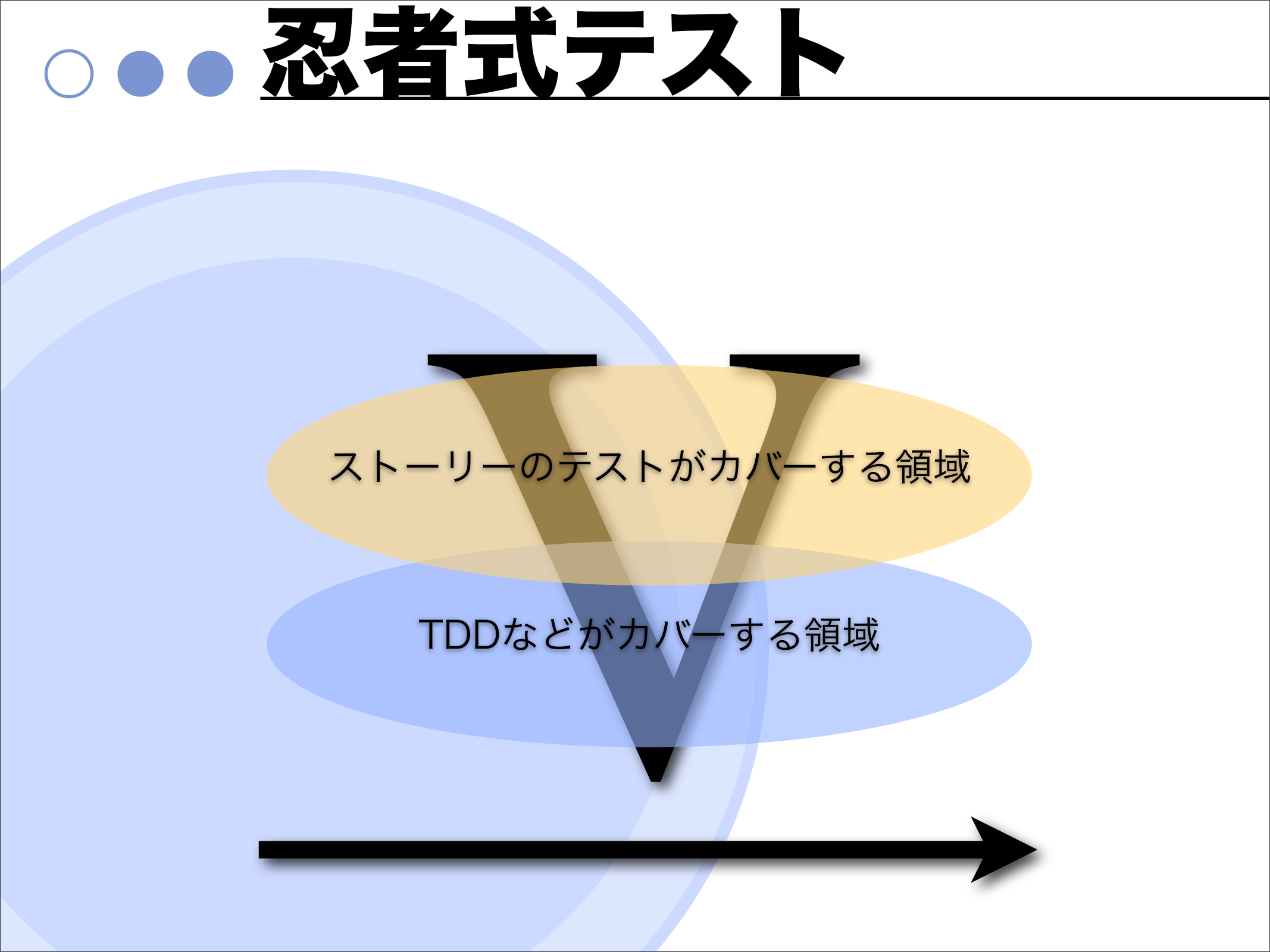
準備

- 問題を小さな要求単位（ストーリー）に分割
- ストーリーはテスト可能なものに
- ストーリーはこの先のプラクティスで何度も使うので大事だよ!

●●● 忍者式テスト

- ストーリーのゴールをテストで記述
- 作ったらテストで確認する
- 受け入れ試験を全部やりなおして壊れてないことを確認

忍者式テスト



The diagram features a large, light blue, semi-transparent shape on the left side, resembling a stylized 'L' or a large bracket. In the center, there is a large, dark blue, downward-pointing arrow. Two smaller, dark blue, downward-pointing arrows are positioned above the main arrow, one on each side. A thick black horizontal arrow points to the right at the bottom of the diagram. Two overlapping, semi-transparent ellipses are centered horizontally: a yellow one on top and a blue one on the bottom. The text is placed within these ellipses.

ストーリーのテストがカバーする領域

TDDなどがカバーする領域

忍者式テスト

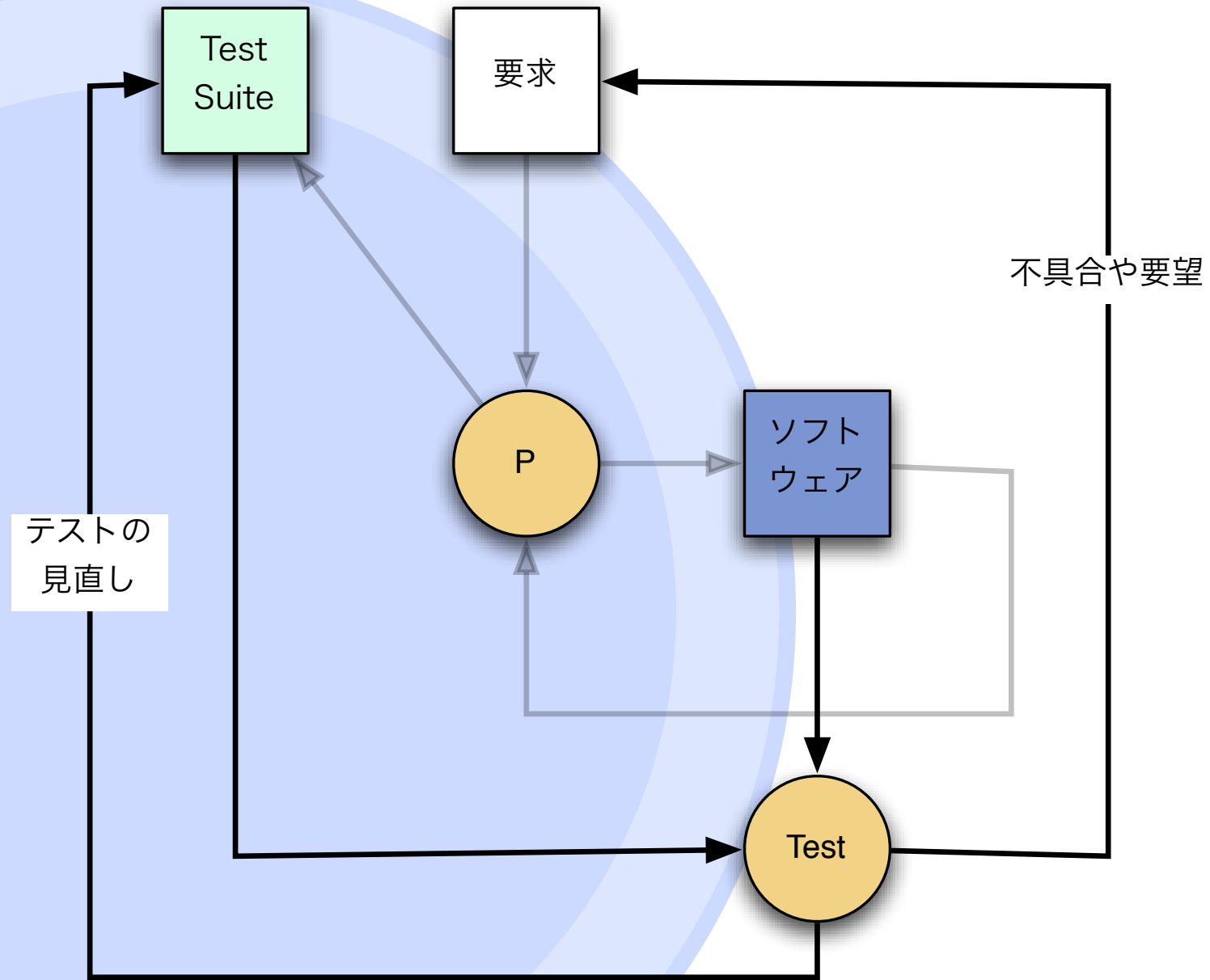
- ペアテスト
- ストーリーの意図を考えテストケースをアレンジ
- 実施時に内容を見直す

○●● 忍者式テスト

- 不具合はストーリーと同様に追加
- 対策後には確認のためのテストが追加される



忍者式テスト



忍者式テスト

- 毎日成長するテストを飛び続ける
- **Ninja tree-jumping analogy - M.F**
 - ちょっとウケた

●●● テスト時間の捻出

- リーダーが全部一人でやる作戦
- 開発者全員が毎日1時間ずつやる作戦

○ ● ● 効果

- すぐに気づく
 - 作り壊し
 - 新機能の評価
 - 矛盾する外部仕様
- テストによる開発の支援

○ ● ● 効果

- 攻めたい部分のテストの強化
 - 顧客の興味のある部分
 - 実装の弱い部分
- 問題の見つかるペースが一定になる
 - 開発のピークがない

○●● 忍者式テストのまとめ

- まさにテスト駆動開発
- 毎日受け入れテストを全部やるだけ
- 超簡単！



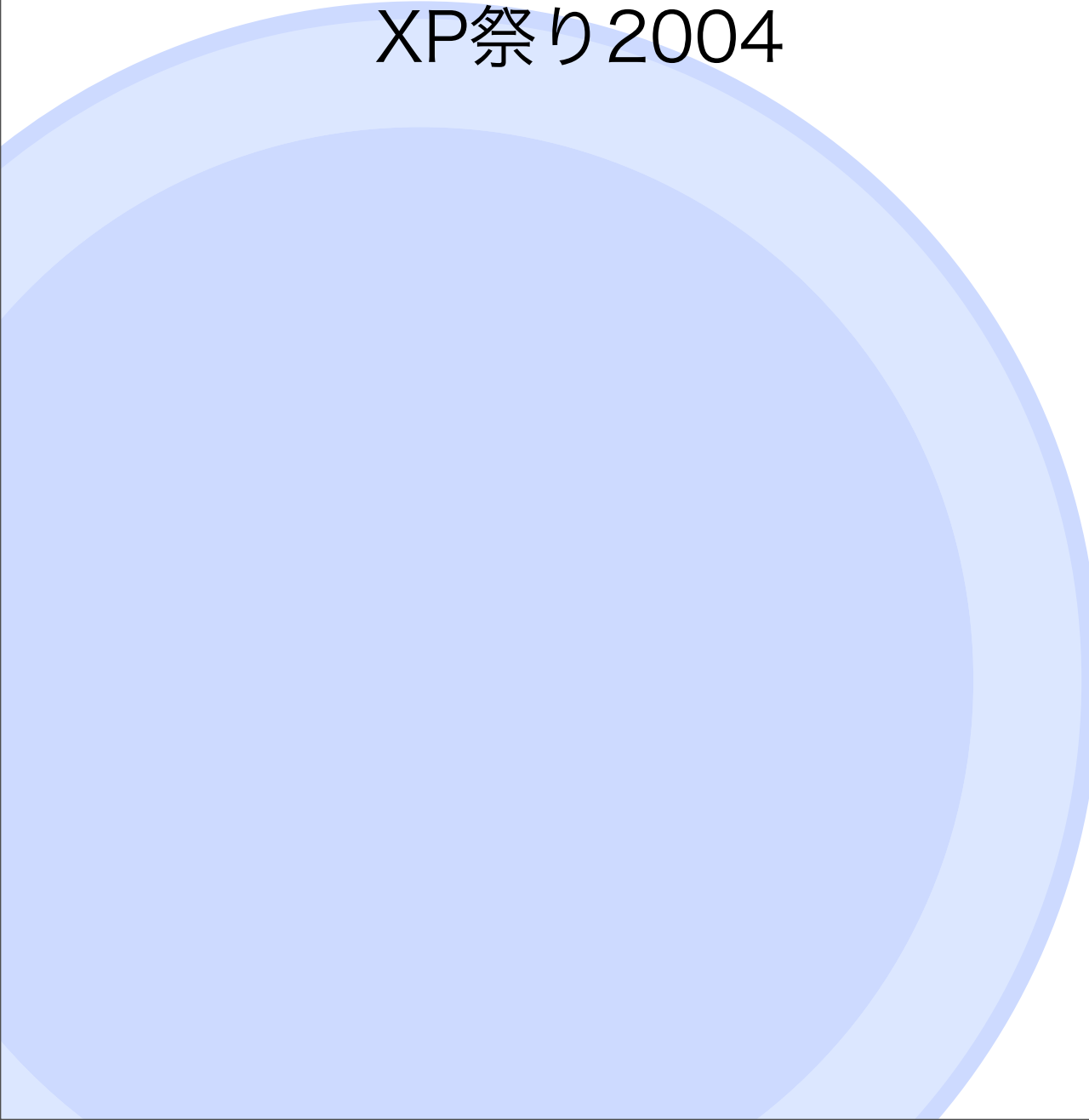
しばらくお待ちください

- ✓ 簡単さを強調
- ✓ とちぎテストの会議50円
- ✓ +14分



計画ゲーム

XP祭り2004

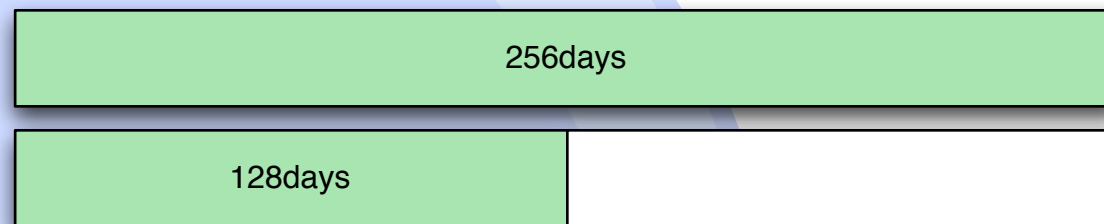


思考実験

- 大きな開発を分割する
- 半分の期間で二回開発するのを想像

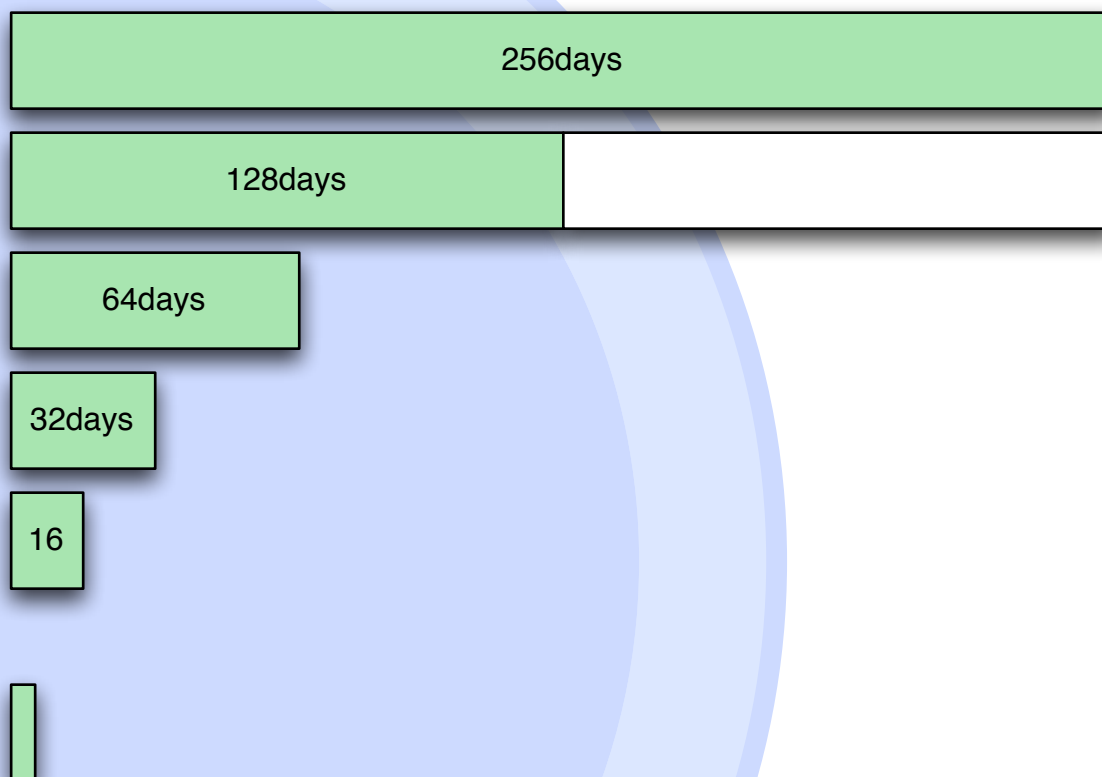


半分は想像しやすい





どこまで小さくできる？



○●● 計画ゲーム

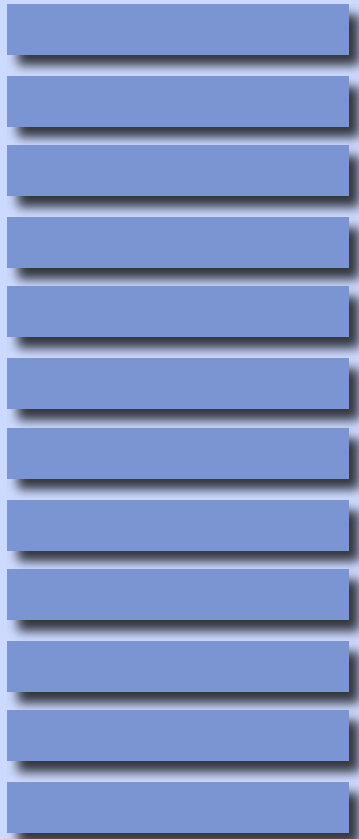
- ストーリーは小さな要求、機能の単位
- 数日で完了する程度の規模
- ストーリーの束を使って計画

○●● 計画ゲーム

- ストーリーがいつ手に入るかを考える
- ビジネス的な価値、実装の難しさ、嫌な予感などで優先度づけ
- 早く実現するストーリーは確実に手に入る

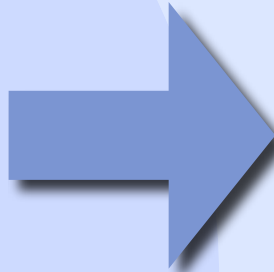


計画ゲーム



要求

プログラマによる
見積り

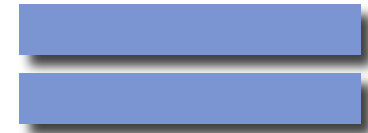


計画ゲーム

顧客による
優先度付け



t4



t3



t2



t1

タイムボックス

- 一度に気を配れる量には上限がある
 - 遠い未来まで並べることはない
 - 近くの単位時間だけ考えれば充分

● ● ● 計画ゲーム

- バグもストーリーとして扱うと簡単
- 変更？バグ？とかどうでもいい議論がなくなる
 - 理想との差分があることが重要
- 変更もストーリーの追加で表現する

優先度付け

● 今やるモノを選び、今やらないモノを選ぶ

○ 重要な機能

○ 難しい実装

○ 心配事

○●● 計画ゲームまとめ

- 欲しいものから確実に手に入る
- ある日突然失敗しない
- 超簡単!!!!!!
- 優先度をつけ続けるだけ!!



しばらくお待ちください

- ✓ 簡単さを強調
- ✓ 思考実験をフォロー
- ✓ 優先度がミソ
- ✓ +21

○ ● ● テストのレート

JaSST'06

反復開発に適応したテストスイート生成と継続的テストの実施

● ● ● テストのレート

- 忍者式テストを思い出して
- ストーリーの数が増加するとテストが一日で回らなくなる
- 毎日全体を覆うようなテストをするにはどうしたら…

● ● ● テストのレート

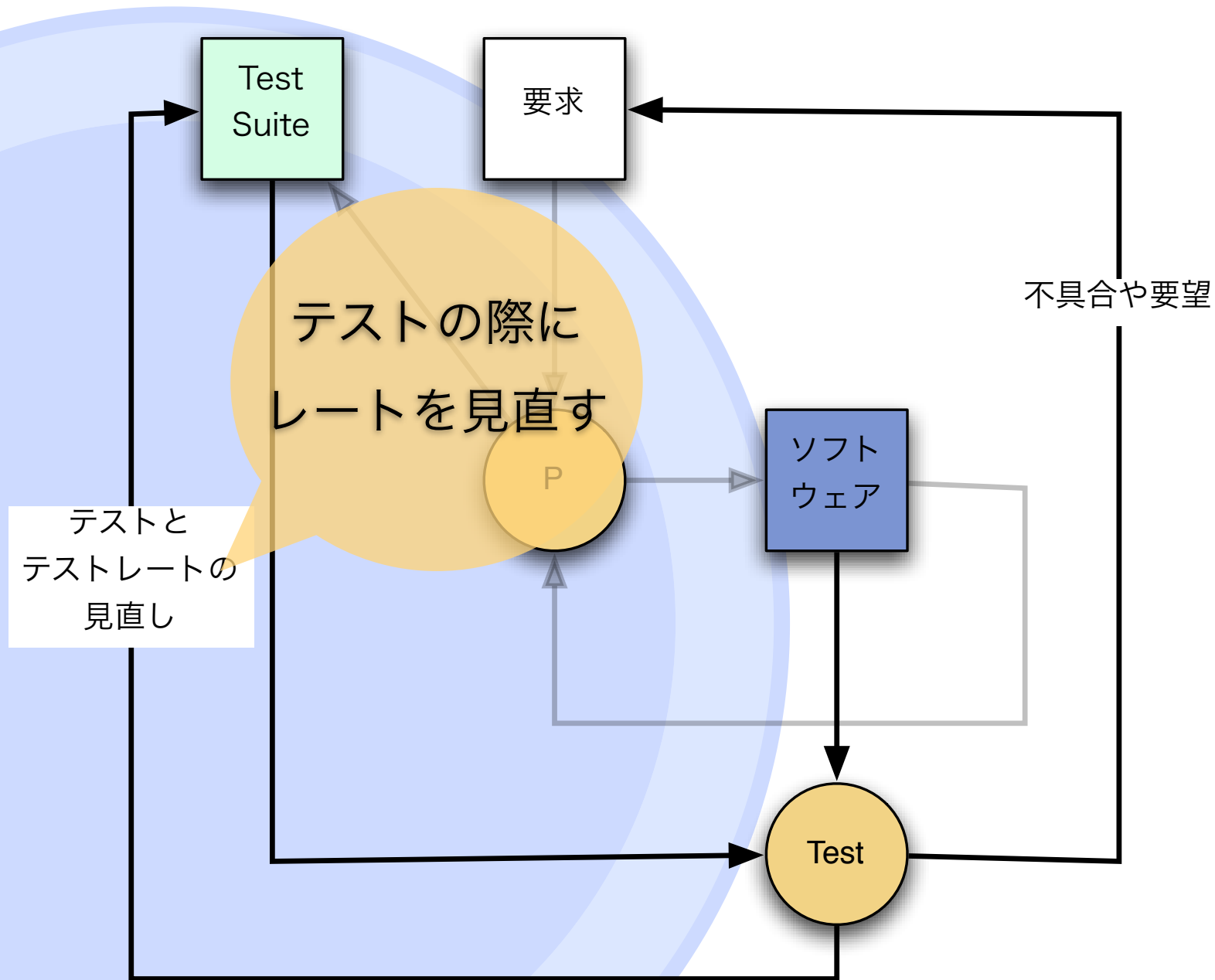
- 頻度の調整を基本に
- 重要でないテストを薄く
- 大事な項目を頻繁に
- ビジネス的 / 技術的なリスク

●●● テストスイートの生成

- iTunesのスマートプレイリストを真似
- レート・鮮度・最後の成績から今日のおすすめテストスイートを自動生成
- 量に圧倒されないように適度に隠す



レートの管理



○●● テストのレポートの管理

- テスト実施のたびにレポート見直す
- 一度に全て見直そうとしてもムリ

●●● テストのレポートまとめ

- テストスイート生成ツール**RWiki**(無料)
- 毎日テストレポートを管理する
- そんなだけ! わりと簡単!!!!!!!!!!!!!!



しばらくお待ちください

- ✓ RubyKaigiでRWikiネタ
- ✓ リスクベースドテスト
- ✓ +28分



プロセスのプロセス

JaSST'07

規模と向き合う

○ ● ● 規模の大きさを味方に

- 大きな集合を扱うには
- たくさんのストーリーによる開発
- プロセスを調整するプロセス

○ ● ● MapReduce

- バッチ処理システム
- 集合の扱い方の戦略のちがい
 - 集合の演算 → SQL
 - 集合の走査 → MapReduce
- 巨大な集合は操作に工夫が必要

○ ● ● MapReduce

- キーで並んでいる集合を走査
- 巨大な集合を一度に扱わない

○●● 大きな集合を扱うには

- 大きな集合のまま扱うには広大なメモリが必要
- 順次処理すると楽ちん
- どんな構造の集合も要素を順に辿るという操作で考えると線形にも見える

○●● ストーリーの集合

- これまで小さなストーリーの大きな集合について説明してきた
- 大きな集合を扱う問題とよく似てる
- たくさんの小さなストーリーを繰り返して処理していく

量を味方につける

- 全てを一度に処理しない
- 今日のテスト・ストーリーに集中
- 身軽な計画
- 絶えず見直すための、見直せる量

○●● プロセスの調整

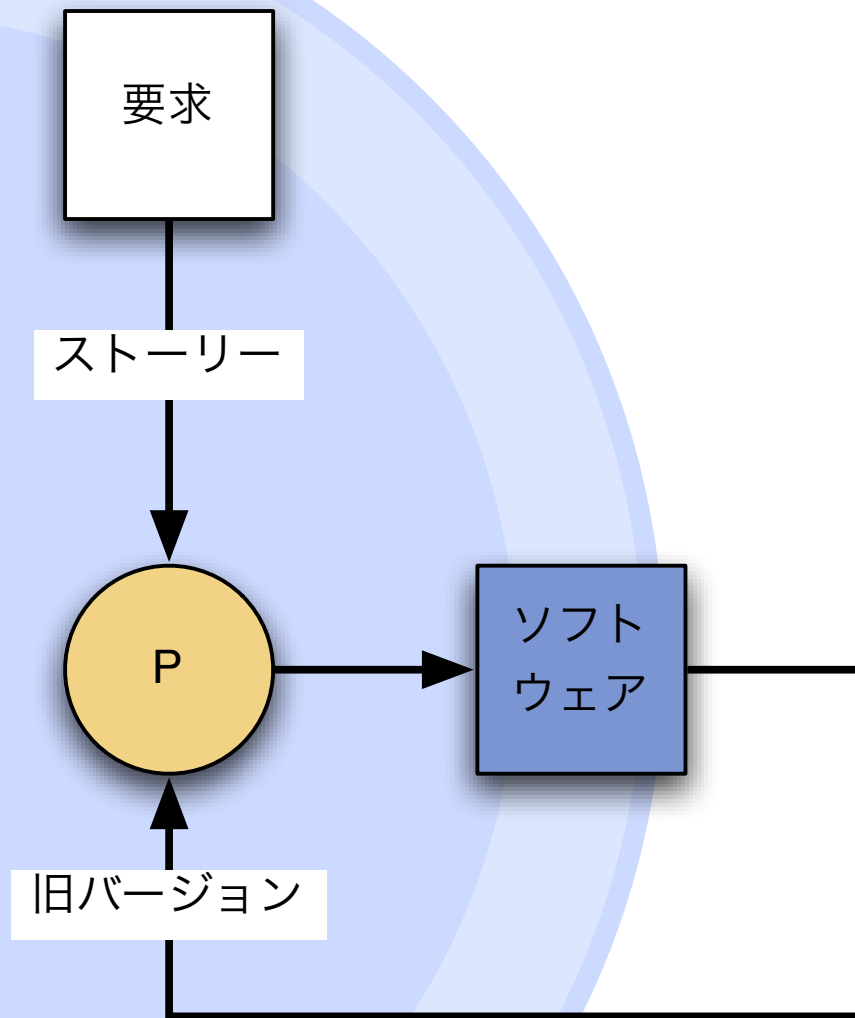
- バッチ処理のプロセスは変わらない
- チームは変わることができる

○●● プロセスの調整

- ループの中で徐々にチーム自身を調整
- よい製品を出力してる？
- 自分たちは正常？

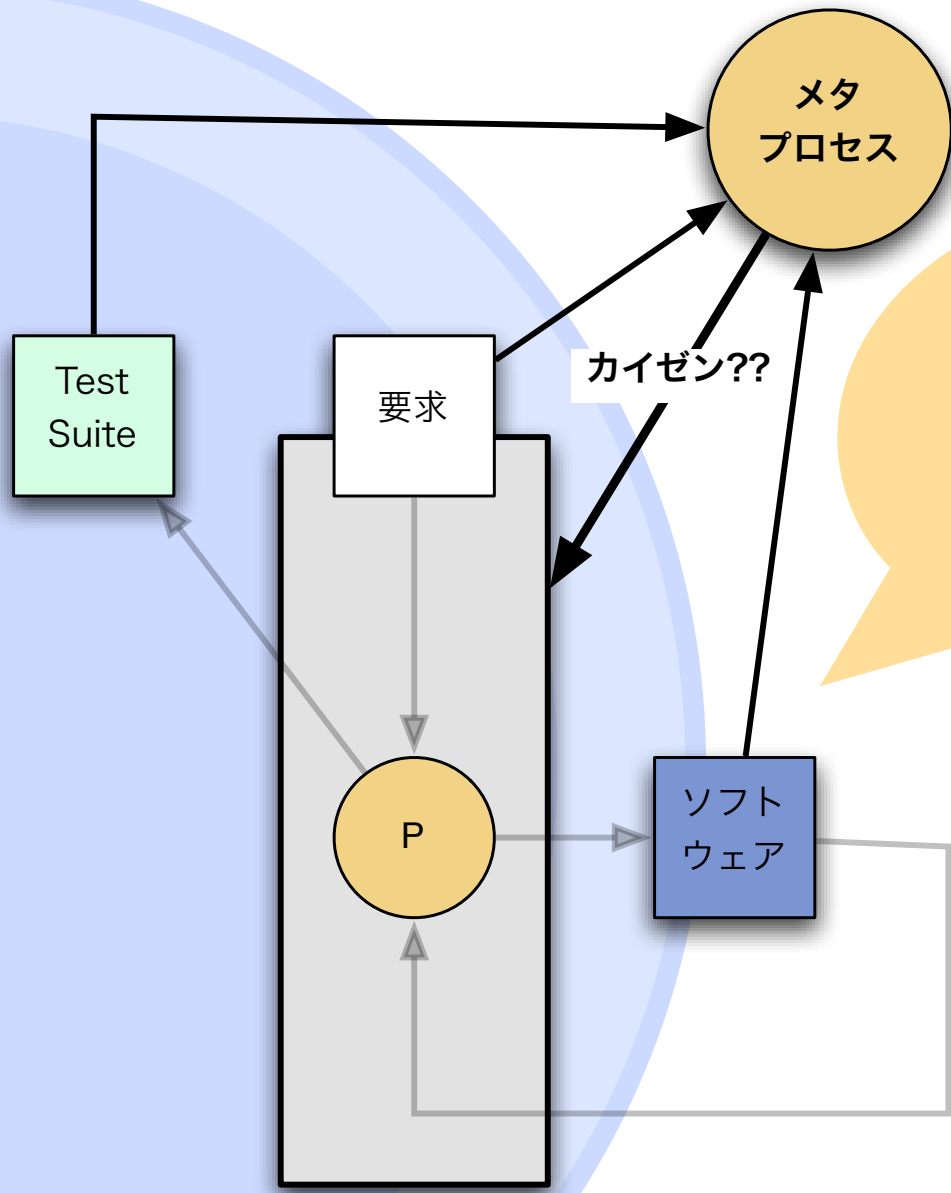


フィードバック





Pを変化させるP



出力を観察して
変わって行く



調整

- プロセスを調整するプロセス
 - 出力(製品)のできばえから
 - 自分たちの様子から

○ ● ● ストーリーの数とはなにか

- ループの回数
- ビジネス側が製品を調整できた回数
 - 機能・作り込み
- ハンドルを切れる頻度

○ ● ● プロセスのプロセスまとめ

- プロセスを調整し続けるためのループと調整するしくみ、そして十分に大きな問題があればよい
- PDCAと一緒に!!! 超簡単!!
- PDCA? これ売れないんじゃない?



しばらくお待ちください

- ✓ 二つネタがあるので整理
- ✓ 量を利用する
- ✓ メタプロセス
- ✓ やや儲からない感じ
- ✓ +38分



ロールプレイング

オブジェクト倶楽部2010夏イベント

ファッションと実践

●●● ロールプレイング

- プロセスの調節器をメンテナンス
- みんなで理想の開発者を演じてみよう

プロセスの調節器

- 良い出力(製品)は想像しやすい
- 自分たちが正常ってなんたる
- 良い自分たち?

理想の開発者

- 自分が想像する理想の開発者
 - 誰かが与えるのではなく自分で
- 場にかもされた価値観などから



場から学ぶ

- 価値観をかもす

- 日々の活動における、無数の状況と判断の組合せを通じて

● ● ● 演じてみて

- 自分が変わったり誰かを変えたりするのはなんかいやだ。
- 我々は単なる仕事上のおつきあい
- お仕事なので仕事でだけそういう役を演じてみよう

●●● だって仕事でしょ

- 個人の信条はいろいろあると思う
 - 自称アーティストとかネット弁慶とか
 - そこまで立ち入れない
- 9時から18時だけ自分の理想とする開発者の役を演じてくださいませ

○●● インストール難しい

- タネとなる誰か
- 維持する人々

○●● タネとなる誰か

- たぶんすでにその場に居ると思う
 - あなた
- あるいは私をインストール

理想の維持

- 判断へのフィードバック
- おかしな判断へのつつこみ
 - 「なんでそんなことすんの？」
- チームへの関心

○●● チームへの関心

- 一緒に開発している人々はチームか？
- 組織上のグループ？

●●● ロールプレイングまとめ

- 仕事中は理想の開発者を演じる、だけ
- 簡単!!!!!!!!!!!!!!



しばらくお待ちください

- ✓ 儲からないのはバレた？
- ✓ 置いてけぼりの人に配慮
- ✓ 私についてのリフレイン
- ✓ +50分

● ● ● まとめ3

- これまでの実践をふりかえりモデル化
- すぐにインストール可能？
- どこでも使えるモデルはメタすぎる
- 商品化は難しそう

●●● 今日話したこと

- 重要なこと
- ファッションの話
- ファッションリーダーになれなかった話

初刷5周年

- 2005年夏
- まだ初刷買えます!



● ● ● まとめ

- 私は実践が好きである
- 方法論は実践のモデルである
- 買ってきてインストールしたくなる方法論は提案できなかった
 - ごめんなさい
 - 続きはまた明日



しばらくお待ちください

- ✓ 初刷5周年
- ✓ SEMATファンに愛想よく
- ✓ ロールプレイングしてね
- ✓ 明日 とちぎテストの会議
- ✓ 質疑へつなぐ



なにかご質問は

