協調開発プラットフォーム

CDP（Cooperative Development Platform）

Contents

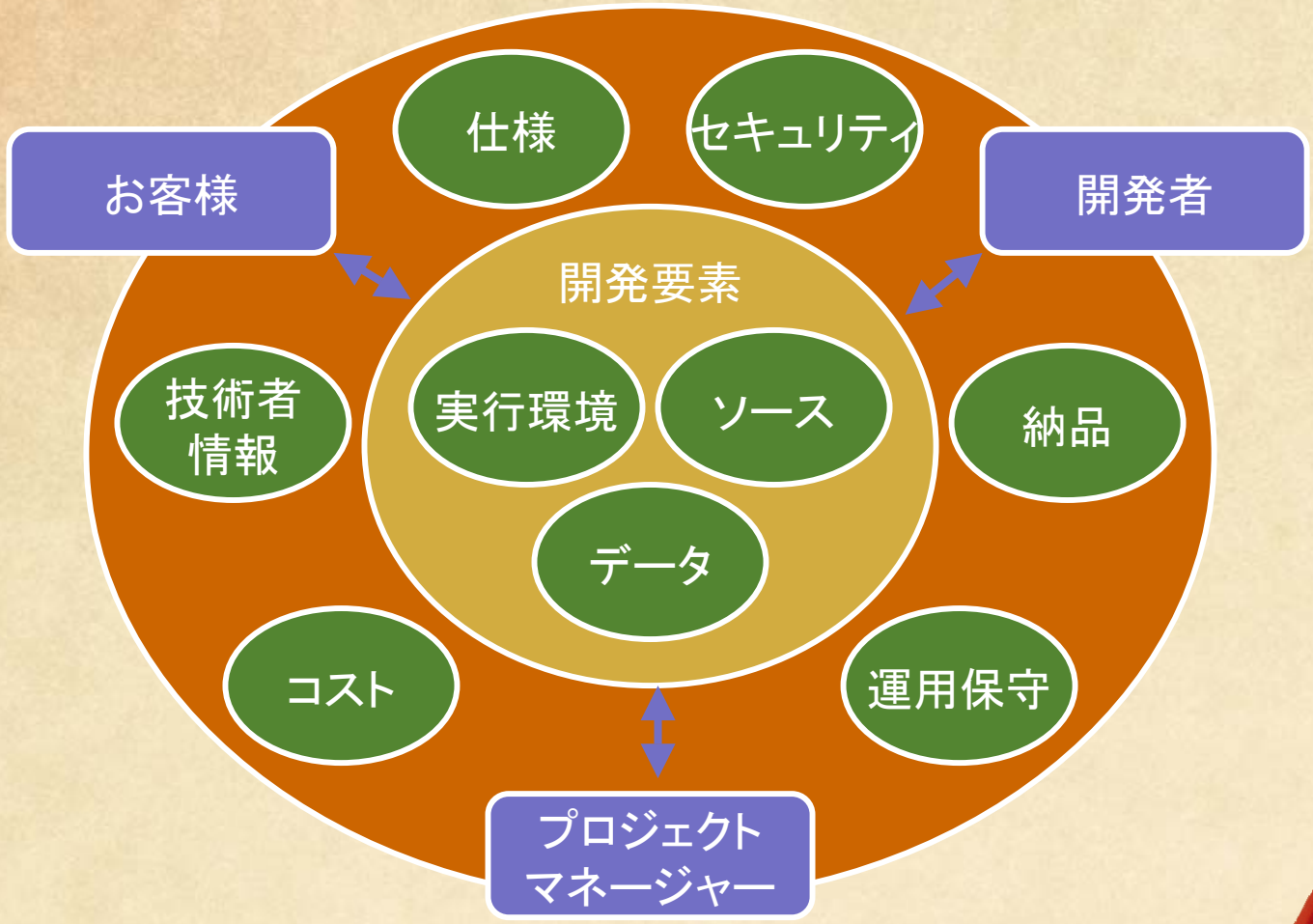
1 開発コミュニケーション環境

2 開発要素

3 協調開発プラットフォーム

4 実装イメージ

開発要素を意識したコミュニケーション環境



機能・品質・納期を守るために・・・

ソースコードの重要性

- ソースコードにはプロジェクトの状況が反映されている
  - ソフトウェア本体ともいえるプログラムのソースコードにはプロジェクトの状況が反映されている。プロジェクトマネジメントにおいてソースコードからダイレクトに情報を収集することにより、報告者の主観等により見えにくい実体を映し出せる。
  - ソースコードマネジメントツールを利用しソースコードを日常的に管理することでプロジェクトの進行状況をリアルタイムに管理し、遅滞のない対策を可能とする。
  - テストで判明する不具合は受身的な解決策であり、不具合が判明したときには既に大部分の製造工程を通過していることが多く、工程間のフィードバックによるオーバヘッドが大きくなる。ソースコードを元に製造工程で積極的な解決を進めることで工程間のフィードバック工数を削減し、またフィードバックの影響による品質劣化防止を期待できる。
  - ソースコードの指摘によるコミュニケーションは、同一コンテクスト上でのコミュニケーションであるため意味の取り違え防止を期待できる。
  - 開発者による報告書作成等のオーバヘッドを軽減する。

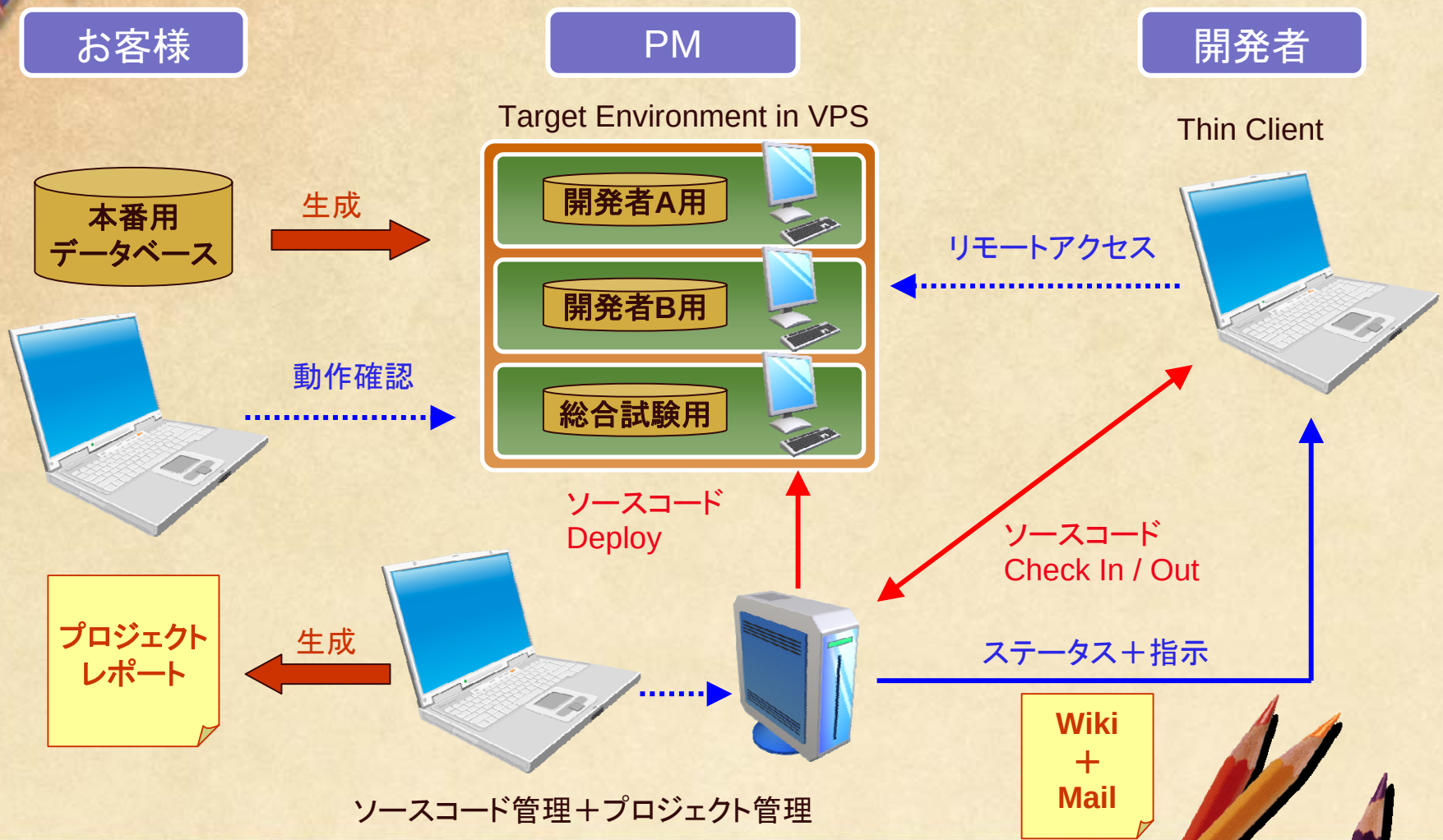
データの重要性

- セキュリティ
  - 顧客データをそのまま使用すれば個人情報等の情報漏えいリスクが高い。
  - 教育による情報セキュリティ意識の向上、あるいは守秘義務による拘束など、いずれも個人に依存するセキュリティ対策であり、時には情報漏洩した場合の口実に過ぎない。
  - 漏洩しても問題ない情報を使用できれば、これらリスクから解放される。
- 品質
  - ソフトウェアの品質を維持するには、良いデータを使用し十分なテストをする必要がある。
  - 単体試験では極限値を持つデータを使用する。
  - 総合試験では実際のデータ相当のテストが望ましいが、セキュリティ・リスクとのトレードオフの問題が発生する。
  - セキュリティと品質を両立するデータが望まれる。

実行環境の重要性

- 開発者単位の実行環境構築
  - 製造および単体試験工程では開発者がストレスなく自由に実行できる環境が望ましい。
  - 開発者毎にデータおよびソースを含めた実行環境を準備できれば開発速度向上が期待できるが、環境構築にかかるコストパフォーマンスに依存。
  - 低コストで簡単に実行環境構築できることが望まれる。
  - 開発者がアクセスできる単位をコントロールし、また個人レベルのアクセス履歴を取得することにより、システムのセキュリティリスク範囲を小さくし、また開発者自身のコンプライアンスを自然に向上する。
- フレキシビリティ
  - お客様の参加を可能とし、お客様～PM～開発者間でリアルな動作確認情報の共有を進めることで円滑なコミュニケーションを促進できる。
  - 開発状況に応じた開発要員の増減へのフレキシブル対応を期待。
  - 国内ローカル要員やオフショア要員の柔軟な参加が望まれる。

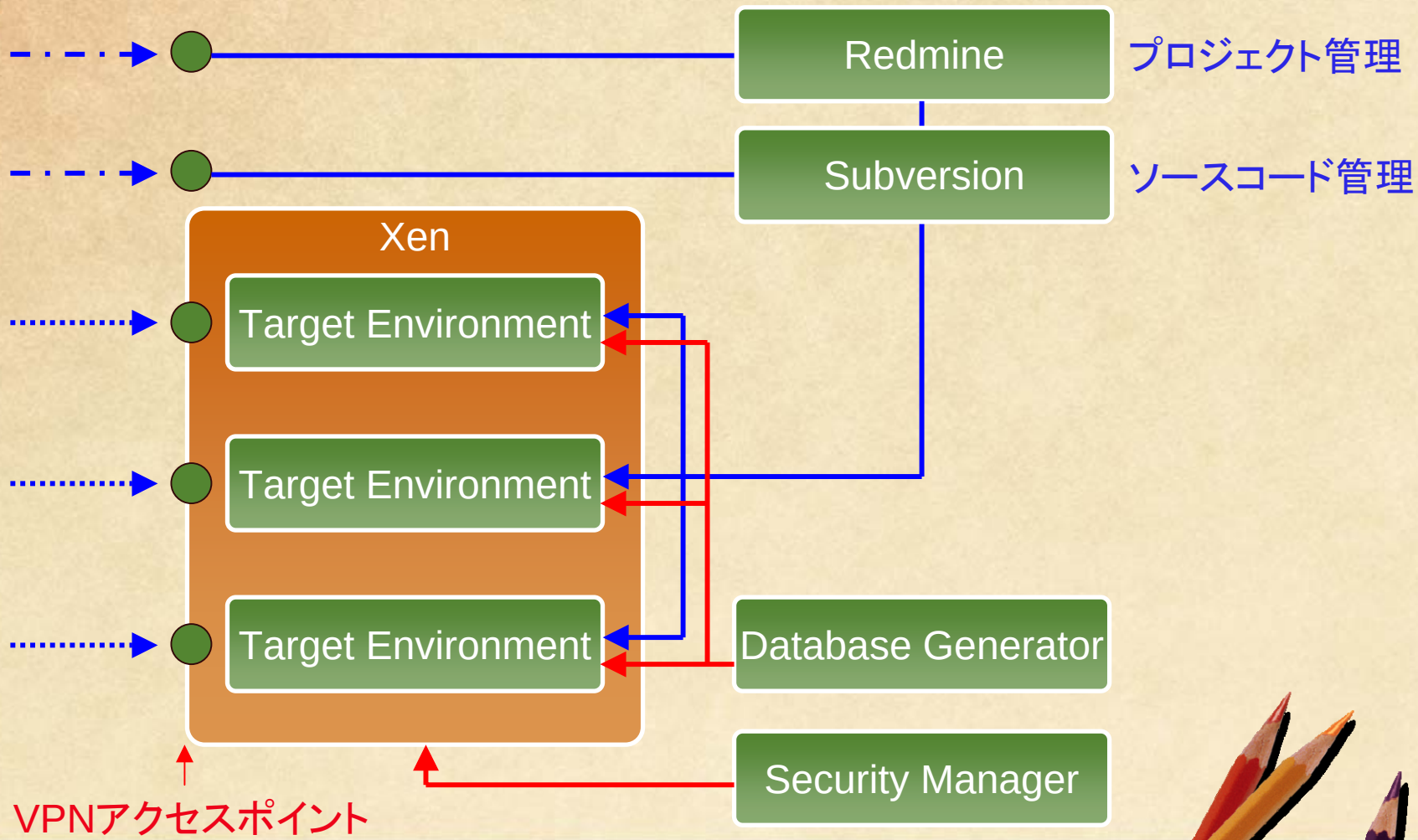
CDP（協調開発プラットフォーム）



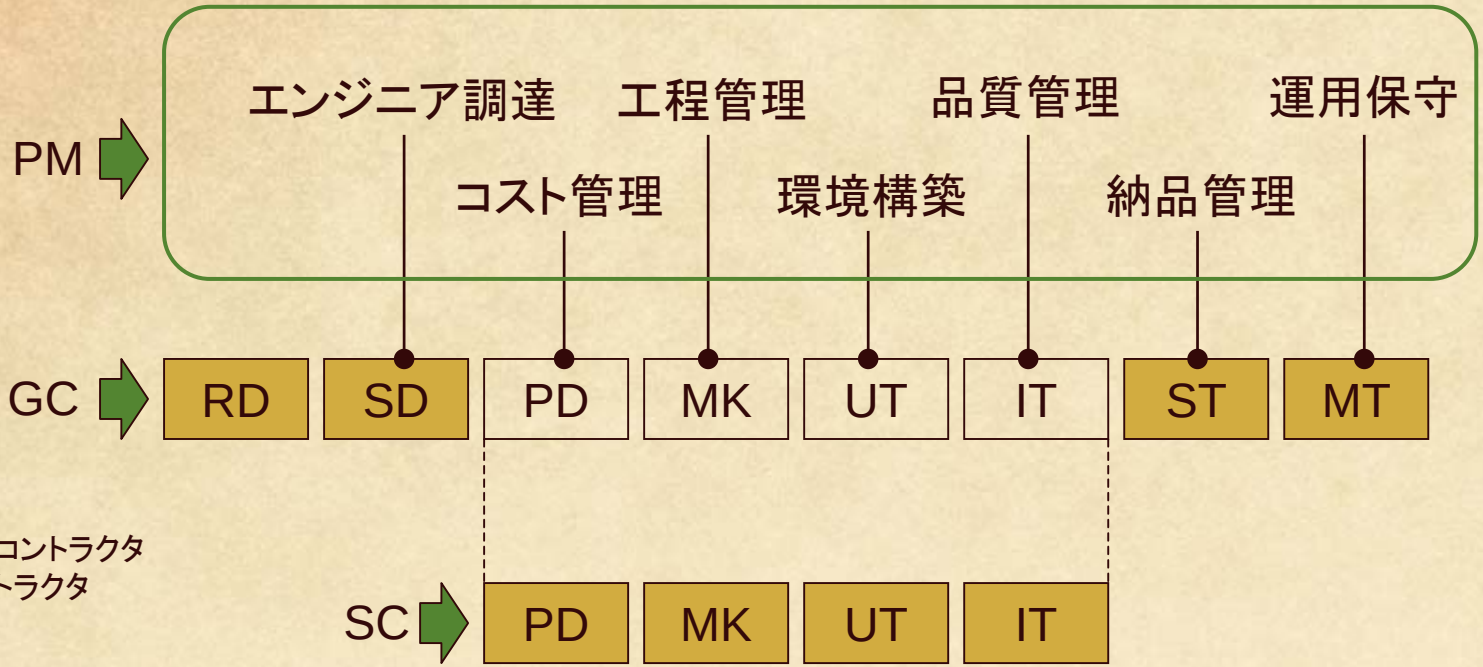
CDPの特徴

- Resource：協調開発環境で最適リソース活用
  - 国内リソース、オフショア・リソースいずれのエンジニアも同様に開発に参加できます。コストパフォーマンスの最適化だけでなく、急な開発要員の増減による開発リスク軽減を実現します。
- Environment：実行環境複製
  - 協調開発を前提としたVPS実行環境により複数の開発者へ同一環境を提供します。ソースコード管理との組合せで単体試験や結合試験環境を自在に構築できます。開発状況把握のタイムラグを軽減します。
- Data：試験用データ生成
  - 本番用データから試験用データを生成し開発することにより、お客様の大事な情報を守ります。
- Source:開発状況レポート生成
  - ソースコードを元にした開発状況レポートにより進捗状況および品質状況をリアルタイムに把握。ソースコードを管理していますのでデグレードへの対策、また納品後のソース管理に威力を発揮します。

CDP実装イメージ



PMの役割(1)



GC:ゼネラルコントラクタ
SC:サブコントラクタ

RD:要求定義
SD:システムデザイン
PD:プログラムデザイン
MK:製造
UT:単体テスト
IT:インタフェーステスト
ST:システムテスト
MT:運用保守

PMの役割（2）

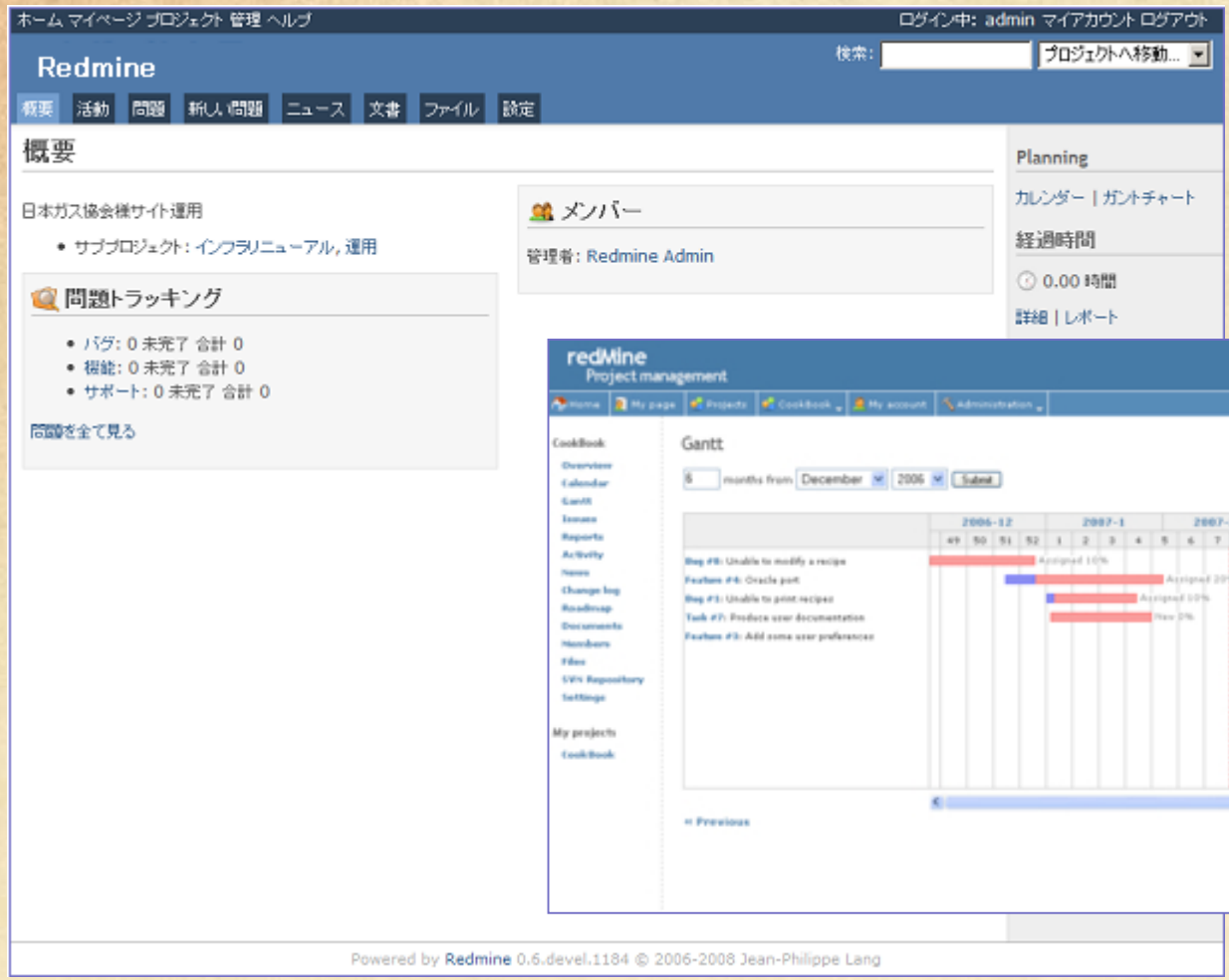
- エンジニア調達
  - エンジニアネットワーク（Online / Offline）
- コスト管理
  - 見積システム化（見積精度および見積レスポンス向上）
- 工程管理
  - ユーザサイドエンジニアによるソースコードレビュー
  - ソースコード数値化による工程の可視化
- 環境構築
  - VPSへセキュアな専用環境構築
  - ユーザサイドエンジニアによる模擬データ構築
- 品質管理
  - ユーザサイドエンジニアによるソースコードレビュー
  - ソースコード数値化による品質の可視化
- 納品管理
  - ソースコードのバージョン管理
- 運用保守
  - エンジニアネットワークによるシステムアドミニストレータの配備

Redmineの特徴

- 複数プロジェクトのサポート
- ロールベースの柔軟なアクセス管理
- バグ追跡システム
- ガントチャート、カレンダー
- ニュース、ドキュメント、ファイル管理
- RSSフィードおよび電子メールによる通知機能
- プロジェクトごとのwiki
- プロジェクトごとのフォーラム
- 工数管理機能
- チケット、プロジェクト、ユーザーに対するカスタムフィールド
- SCMとの連携(SVN, CVS, Mercurial, Bazaar and Darcs)
- LDAP認証
- ユーザー自身による登録機能
- 多言語対応
- 複数のデータベースへの対応

出典：Redmine.JP

Redmineの管理画面



Thank You !

野田啓一