

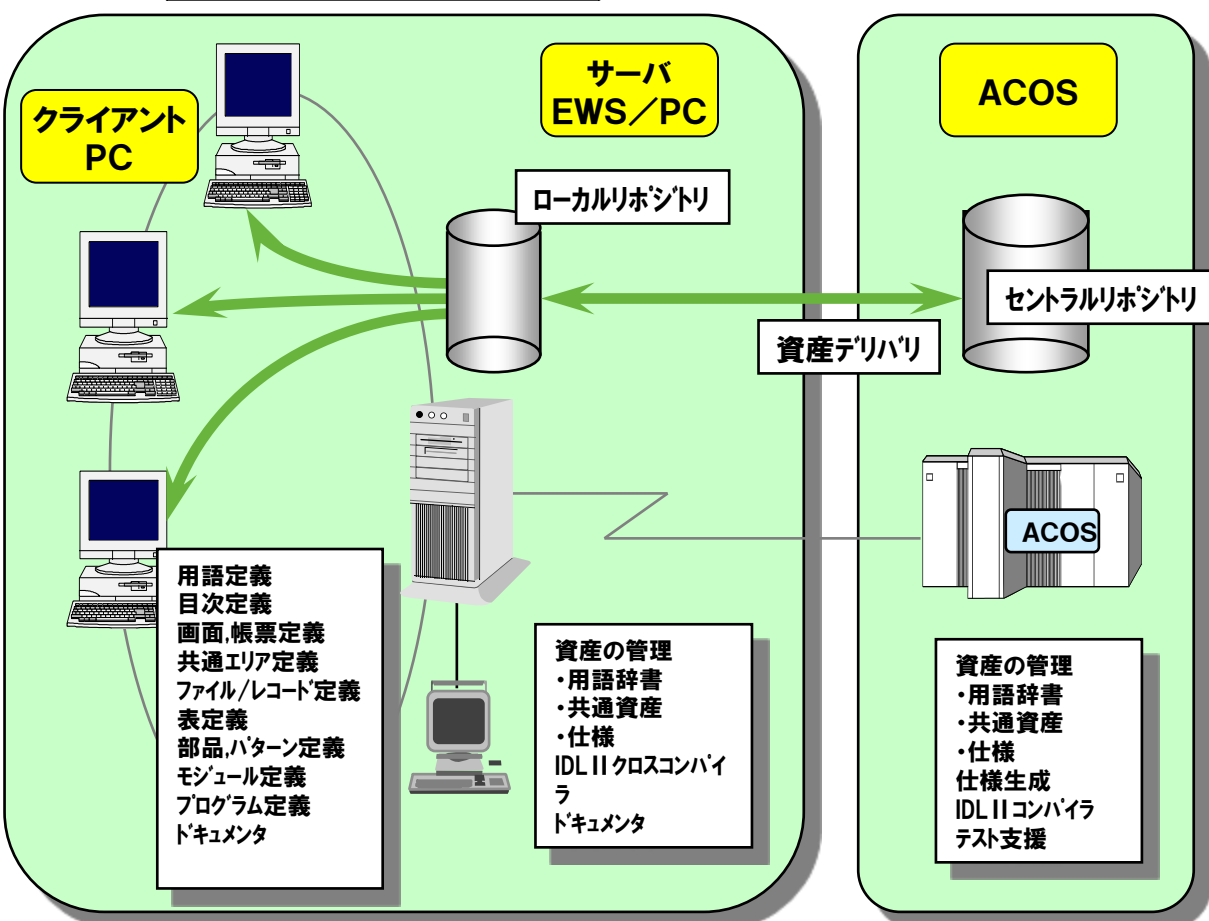
# ソフトウェア開発環境に関する NECの取り組み

日本電気株式会社

# メインフレームの時代

## 大規模開発向けに、重厚長大なオールインワン型CASEを実現

### CASEWORLD



### 【生産性向上】

- 既存ソースコードの流用
- 雛形コードの横展開
- 部品化
- マクロ化
- 第四代言語
- ノウハウDB

### 【保守性向上】

- 標準化(開発方法論)
- ソース・仕様書一致管理
- 用語辞書
- 版管理・構成管理
- 関係管理

# CASEWORLDの製品コンセプト

## ■ ミッションクリティカル

- ・大規模基幹システム向け統合開発環境
- ・設計・開発・作業の標準化を推進、高信頼システム開発基盤

## ■ 上流から下流までを一貫支援

- ・システム化計画から保守までの一貫した統合開発環境

## ■ 開発資産の統合一元管理(リポジトリ)

- ・大規模資産および資産間の関係情報をリポジトリで一元管理

## ■ 日本語設計書ベースの開発

- ・設計書の電子化
- ・設計書からのプログラム自動生成

## ■ データ中心型設計(STEPS II)

- ・データ中心型設計技法とウォーターフォール型開発技法の長所を取り入れた開発方法論に準拠

The diagram illustrates the relationships between various system documentation types, organized into three main vertical sections:

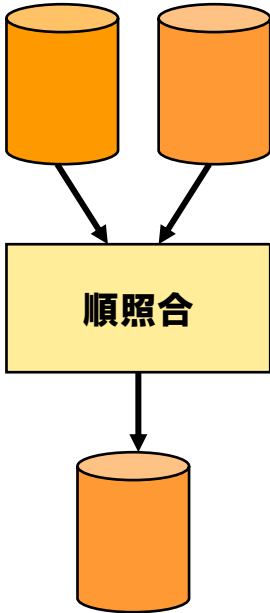
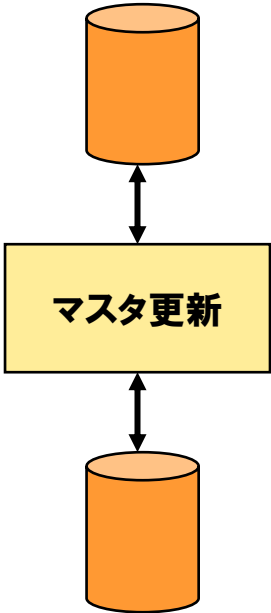
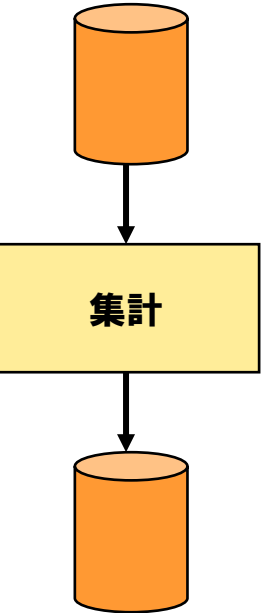
- Left Section (System Overview & Flow):**
  - エンティティ関係図 (Entity Relationship Diagram)
  - 機能階層図 (Function Hierarchy Diagram)
  - 機能情報関連図 (Function Information Related Diagram)
  - 業務フロー図 (Business Flow Diagram)
- Middle Section (Detailed Specifications & Definitions):**
  - 用語 (Terms)
  - データベースファイル (Database File)
  - 帳票 (Forms)
  - 画面 (Screens)
  - 目次 (Table of Contents)
  - プログラム仕様 (Program Specification)
  - DOチャート (DO Chart)
  - パターン (Patterns)
  - 手続き定義 (Procedure Definition)
- Right Section (Development & Maintenance):**
  - 対話型デバッグ (Dialog-type Debugging)
  - ファイルプログラム関連図表 (File Program Related Diagrams)
  - ファイル一覧 (File List)
  - 表一覧 (Table List)
  - プログラム (Program)

Arrows indicate the flow and relationships between these documents, such as from 'エンティティ関係図' to '用語', and from 'データベースファイル' to '帳票' and '画面'.

## リポジトリ(仕様、用語、オブジェクト、プロダクト情報)

# 第四世代言語 IDL II

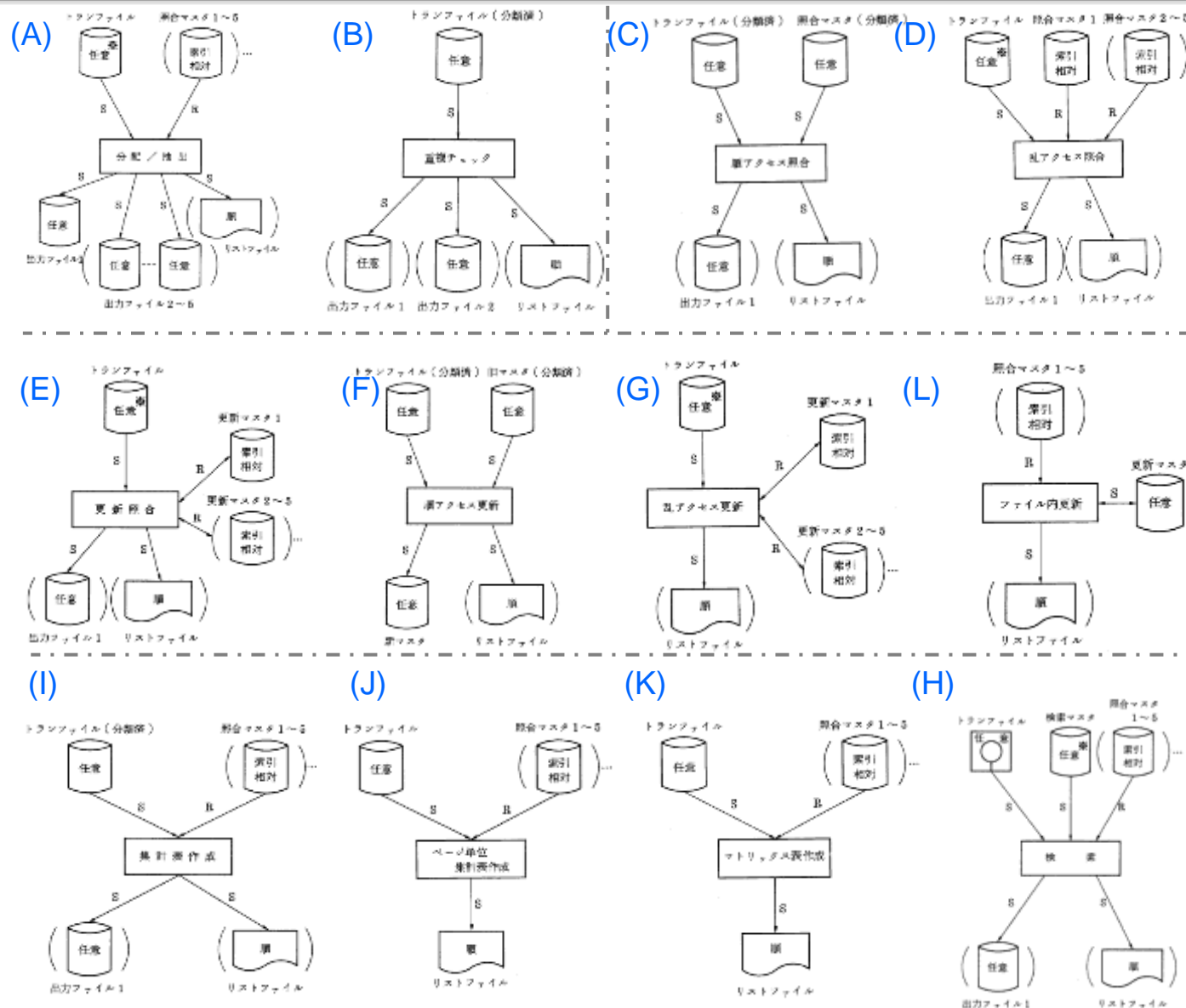
■ 多数のアプリケーションを分析した結果、業務処理に依存しないパターンが存在することを確認。その部分をコンパイラが自動生成する機能を持つ第四世代言語(IDL II)を開発

業務アプリのパターン			
分類	2つのトランザクションの照合、マージ処理等	伝票によるマスタ更新と伝票消込み等	制御切れ検出による集計処理等
生成	ファイルOPEN/CLOSE、入力トランザクション順読み出しループ、照合処理、初期化/EOD処理、制御切れ処理、コミットメント制御、エラー時処理。。		

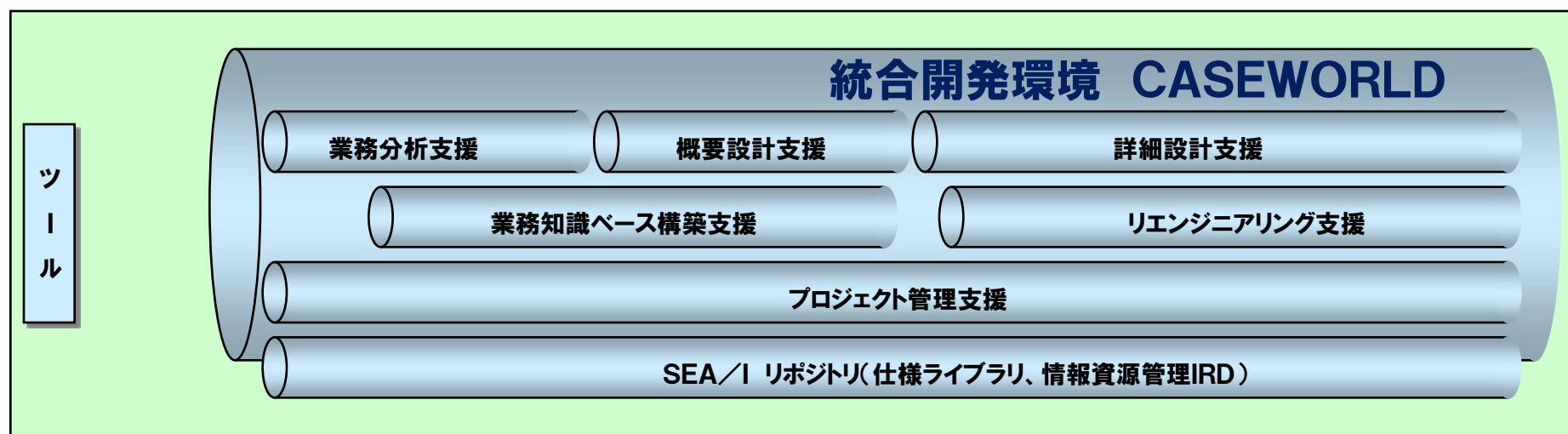
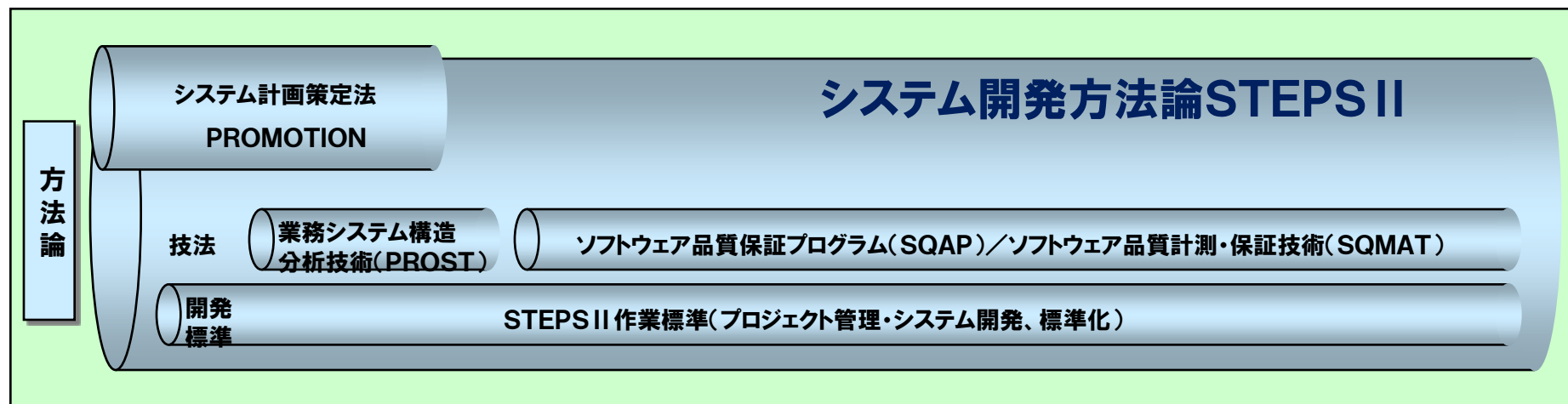


# 業務処理のパターン (ACOS標準パターン)

分類	ID	パターン機能
抽出系	A	分配／抽出
	B	重複チェック
照合系	C	順アクセス照合
	D	乱アクセス照合
更新系	E	更新照合
	F	順アクセス更新
	G	乱アクセス更新
	L	ファイル内更新
分析系	I	集計表作成
	J	ページ単位集計表作成
	K	マトリックス表作成
	H	検索



# 開発方法論 – STEPS II –

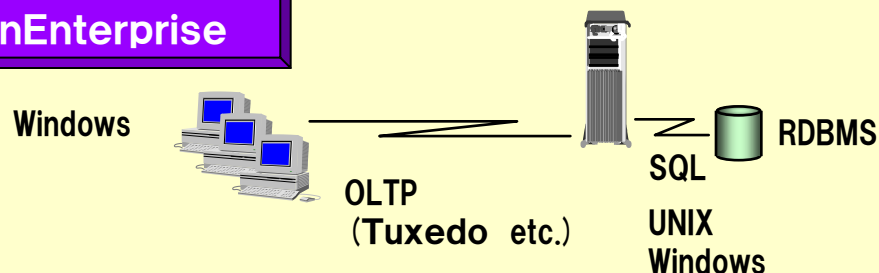




# オープンシステムの時代

## オールインワン型CASEのコンセプトを継承しつつ、メジャーなオープン製品と連携を強化

### HolonEnterprise



- ◆GUI定義機能: 画面／帳票ビジュアルレイアウト
- ◆業務記述能力: COBOL中核機能＋オープン系特有機能
- ◆部品化機能: DLL/SL生成機能、マクロ、画面部品
- ◆大規模開発支援: 用語辞書、版・構成管理、関係管理
- ◆ドキュメント作成支援: 仕様書自動生成、ドキュメント一元管理
- ◆コンポーネント作成支援: CORBAインタフェース生成、ActiveX生成、ミドルウェアシミュレーション
- ◆Web開発支援: HTML操作ライブラリ、シンクライアント・マイグレーション
- ◆他ツール連携: Rational、Erwin、Xupper、Purify

### 高移植性・拡張性を提供するマルチプラットフォーム

- ◆OS: Windows95/98/2000/NT、UX/4800、HP-UX、Linux
- ◆DB: Oracle、Informix、SQL Server
- ◆ミドルウェア: WebOTX、Tuxedo、DCOM/MTS、ACOS/OLF
- ◆プリンタ: PSプリンタ、NPDLプリンタ、Windows接続プリンタ(GDI/IF)

### 【生産性向上】

- 既存ソースコードの流用
- 雛形コードの横展開→フレームワーク
- 部品化、マクロ化→コンポーネント
- 第四代言語→インタプリタ言語
- ノウハウDB→Webによる情報共有
- テスト自動化

### 【保守性向上】

- 標準化(開発方法論)
- ソース・仕様書一致管理→ドキュメント生成
- 用語辞書
- 版管理・構成管理
- 関係管理→モデリング



# HolonEnterpriseの製品コンセプト

## ■ハイスケーラビリティ／ミッションクリティカル

- ・全社基幹システムから部門システムまでの開発をカバー
- ・設計・開発・作業の標準化を推進、高性能・高信頼性システム開発をサポート

## ■上流から下流までを一貫支援

- ・システム化計画から保守までの一貫した統合開発環境

## ■プラットフォームフリー

- ・Windows,UNIXなどのプラットフォームに依存しない開発／実行環境のサポート
- ・データベース, ミドルウェアに依存しない開発／実行環境のサポート

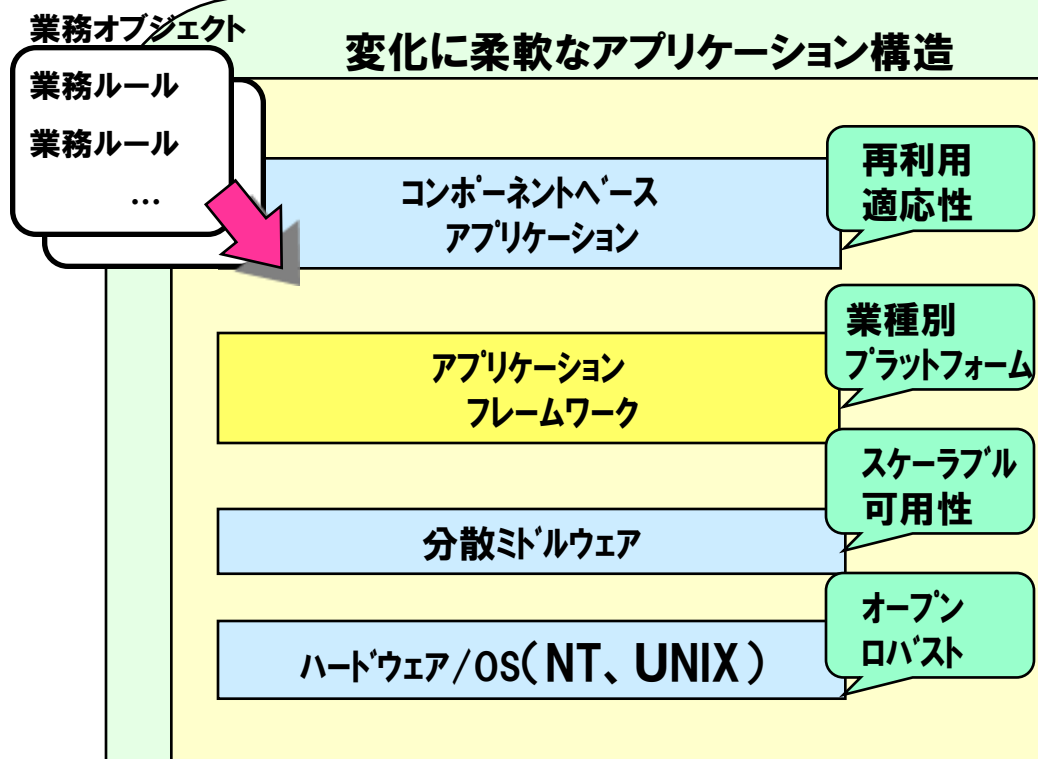
## ■ビジネスルールベース設計(SDUP)

- ・ビジネスルールに着目したRAD型の開発方法論に準拠

## ■コンポーネント開発

- ・ビジネスルールに基づくActiveX、CORBAコンポーネントの開発を支援

# オープン系開発標準 – フレームワーク、コンポーネント、業務ルール –

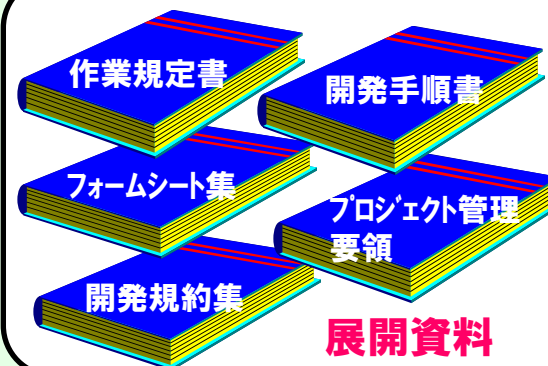


**開発環境**

• **HolonEnterprise**

• 各種自動生成ツール  
(バッチAP、DBアクセス...)

## BrainMethod



### 業務テンプレート集 (HolonEnterprise版)



業務テンプレート集 (VB版)  
(C版)  
(PowerBuilder版)

# オブジェクト指向開発への取り組み

**SystemDirector/UnifiedProcess**  
**オブジェクト指向開発**  
継承,多相性等を駆使し抽象度を高度化

技術者不足、スキル不足、  
経験不足…

従来の考え方、  
やり方では  
壁が高すぎる

**オブジェクト  
指向開発**

従来のやり方  
でも何とか  
次のステップには  
行けそうだ

**構造化手法**

**データ中心型  
開発 (DOA)**

**BrainMethod**のアプローチ  
「DOA+ビジネスルール」による開発

- ・ TopDown&BottomUp
- ・ [業務データ+ビジネスルール]の  
カプセル化で独立性を確保
- ・ 分析~製造までのコンポーネント

# 業務ルールベース開発方法論 –BrainMethod–

- ① 業務AP開発者が業務要件の実現に専念できる  
…未知のIT基盤技術の習得を、フレームワーク整備担当者に特定し、業務AP開発者は標準I/Fでの業務要件実現に専念できる
- ② コンポーネントの修正をエンドユーザと協調してできる  
…ビジネスルールは熟知しているエンドユーザが定義・変更し、システム部門は実装に専念できる
- ③ 変更箇所が特定なコンポーネントに局所化、コンポーネントの組合せで新サービスが速やかに提供できる  
…業務データ毎の関連するビジネスルールが把握でき、修正や再利用が容易にできる

## ビジネスルールベース開発

- ・ ビジネスオブジェクトをコンポーネント化して実装  
= 共通のインターフェースを持たせることで、再利用性UP
- ・ ビジネスルールをユーザーに理解しやすい表現形式で定義  
= 要件定義と乖離しない成果物で保守性をUP

## フレームワークベース開発

- ・ 業務処理や制御処理、実装依存部分を標準化
  - ・ 複雑な制御処理や実装インタフェースを隠蔽化
- 生産性・保守性・品質をUP**

## データ中心型開発

# ビジネスルール設計

## データ分析手順

### ◆属性に対するビジネスルールを抽出

#### 初期値制約

アトリビュートの初期値を規定するルール  
例: 申込書に申込日が記入されていない場合は申込書を受理した日付を申込日とする。

#### 導出制約

アトリビュートの値を導出するルール  
例: 受注金額 = 受注単価 × 受注数量

### ◆ドメインに対するビジネスルールを抽出

#### 値域制約

ドメインを持つデータの値の範囲を規定するルール  
例: 月は1～12の整数でなければならない

#### 初期値制約

ドメインを持つデータの初期値を規定するルール  
例: 年月日には本日の日付を設定する

### ◆エンティティに対するビジネスルールを抽出

#### 参照整合性制約

エンティティの外部キー値が、関連するエンティティの主キーの値として存在することを保証するルール  
例: エンティティ「受注明細」の「受注番号」

#### 多重度制約

エンティティ間の対応関係が1:N:Mの場合M・Nの最小数・最大数を制限するルール  
例: 「受注」に対して「受注明細」は必ず1つ存在しなければならない

データ項目の抽出

ビジネスルール抽出

用語一覧

データの標準化

データグループの把握

データの正規化

ビジネスルール抽出

エンティティ一覧

ビジネス  
オブジェクト  
仕様書

論理データモデル図の作成

ビジネスルール抽出

ビジネスルール  
仕様書

エンティティライフサイクルの検証

ビジネスルール抽出

エンティティの詳細定義

### ◆エンティティに対するビジネスルールを抽出

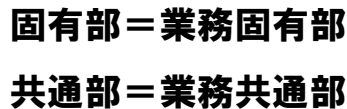
#### 更新制約

更新時におけるデータの値によって、別のエンティティの生成、更新、消滅を引き起こすルール  
例: エンティティ「在庫」の在庫数が安全在庫数を下回ったらエンティティ「発注」を登録する

#### 処理順序制約

データの状態遷移規則(生成・参照・更新・削除まで)を保証するルール  
例: 社員の退職10年経過後、社員情報を削除する

**業務の処理骨格や制御処理を標準化** 生産性、保守性、品質をUP



# 統合CASEアプローチでの課題

## 生産性向上対策は成功したか？

- 既存システムの流用

盛んに行われたがカスタマイズの手間で効果小

- AP雛型に基づく量産

雛型自体にバグがあった場合悲惨なことに

- 部品化、マクロ化

日付処理など単純なものに限り効果あり

- ノウハウデータベース

インセンティブ、鮮度維持のしくみがなく陳腐化  
技術者の自前志向のため普及せず

- 第四世代言語

ソフトハウス要員の募集に苦労

## 保守性向上対策は成功したか？

- 標準化(命名規則、作業標準...)

読解性の向上に貢献。守らせることが難しい

- 仕様書管理

システム改版が進む毎に実体と仕様書の不一致が拡大

- データディクショナリ、リポジトリ

しかけが大掛かりで一部システム以外普及せず

- 開発方法論(データ中心、オブジェクト指向)

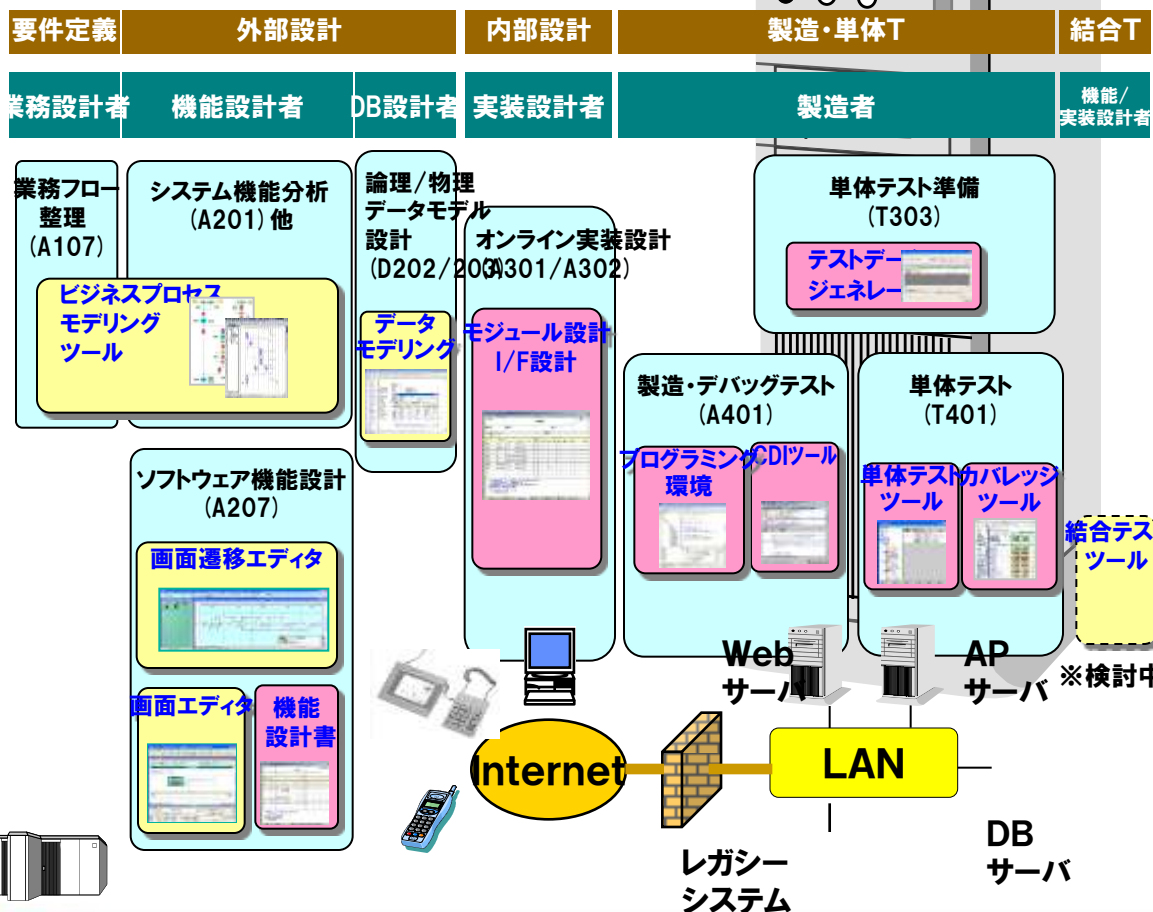
要員育成に難。依然として構造化分析・設計が主流



# インターネットの時代

アプリケーションモデルと開発プロセスをコアとして、各種ツール・ジェネレータ・テンプレート・コンポーネントからなるオープンな開発環境を実現

## System Director Enterprise



### 【生産性向上】

- フレームワーク → アプリケーションモデル
- コンポーネント
- インタプリタ言語 → Java/C#
- Webによる情報共有
- テスト自動化
- ソースコード検査

### 【保守性向上】

- 標準化(開発方法論)
- ドキュメント生成
- 用語辞書
- 版管理・構成管理
- モデリング → ビジネスプロセス・モデリング
- メトリクス検査

# 最新開発環境の動向

開発環境は単体機能から設計～テストのライフサイクル全般をカバーする方向へシフト

## 高シェア製品の評価

- 全体の2/3を超える利用者がEclipseかVisual Studioで開発している
- 気に入った点
  - ー 利用する上で必要な情報を入手しやすい
  - ー ソースコードを入力している途中で入力した文字列を補完するなど効率化の機能がある
- 不満な点
  - ー 上流工程やGUI部分の開発効率支援のための機能が少ない
  - ー プラグインが多く、使いこなせない
  - ー ハードウェア要求が高い

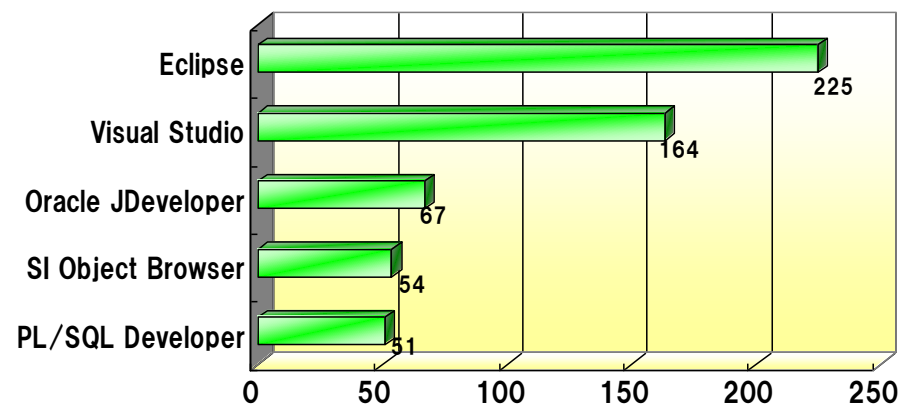
## 業界動向

プログラミングだけではなく、前工程の設計フェーズ、後工程のテストフェーズも含めたALM(\*)を強化した製品の販売、サポートによる収益拡大の動きが活発化

※ALM・・・Application Lifecycle Managementの略

## 他社動向

## 使ってみたい開発ツール(プログラミング分野)



(出展) 日経システムズ2009年6月号「開発支援ツールの利用実態」

## 各社動向

- Microsoft・・・VisualStudio2010を2010年4月に発売。  
コード分析や単体テストツールのみならず構成管理や障害管理機能を提供している。
- 富士通・・・Interstage Studio V9ではEclipseを採用し、UML対応により、設計支援機能を提供
- 日立・・・uCosminexusDeveloperではMyEclipseを同梱し、自社製品と組み合わせている。これにより、設計からコーディング、デバッグまでカバーしている。

# 開発環境に関するNECの取り組み

SystemDirector Enterpriseはユーザのニーズを使い慣れた操作性で提供

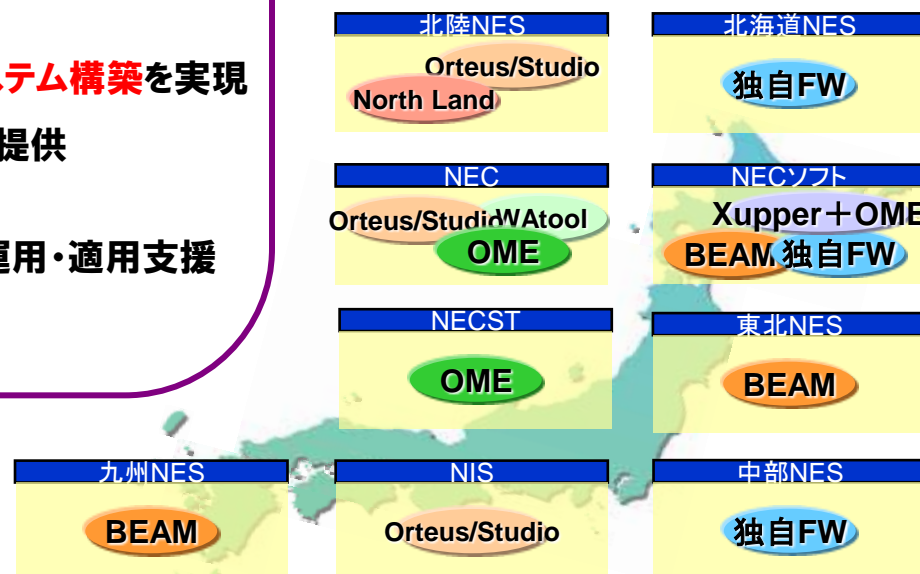
## NECの取組み

- **NECグループSI開発標準確立**を目指し、散在していたFWを統一
- 2006年5月SystemDirector Enterpriseを出荷開始  
(社内向けには2005年11月より提供、2006年5月より一般外販を開始)
- 統合開発環境として**要件定義～テスト**をカバーした各種ツール、開発基盤を提供
- **Java版はEclipseに.NET版はVisual Studio**をベースに、多くのユーザに慣れ親しんだGUI環境で開発できるように拡張機能をアドオン
- NECのSIノウハウを結集した**開発方法論を製品に加え**、競合他社よりも実践的な製品として**短納期・高品質なシステム構築**を実現
- NECグループ社員による開発作業の際には無償で製品を提供
- ITサービスBUの新規開発案件は、特別な事情が無い限り、**SDEを全面適用**するガバナンス運用・適用支援をシステム技術統括本部にて推進  
→NECグループ内の急速な普及を目指す

SystemDirector  
Enterprise

- ◇ **要員や技術ノウハウの効率的活用**  
開発環境の集約による集中化効果
- ◇ **開発環境の先進技術対応の加速**  
開発環境の集約により対応効率化

ノウハウを集約



# SystemDirector製品体系

最新システム向け**オープン言語**と、大規模基幹システムで多数の実績を持つ**NEC製言語**の開発環境をカバーする製品群とソリューションをご提供します

**SystemDirector**

## ソフトウェア開発環境・開発方法論（製品）

スマートデバイス、最新オープンPFに対応する  
Java、C#ベースの開発環境

**SystemDirector Enterprise (SDE)**

あらゆるPFに対応する  
世界標準の業務開発言語環境

**COBOL**

メインフレームのシステム開発に適した  
NEC製4GLベースの開発環境

**CASEWORLD**

C/Sシステム開発に適した  
NEC製4GLベースの開発環境

**HolonEnterprise**

帳票やテスト支援など  
あらゆるPFに対応したデファクトツール群

**開発ツール**

## ソリューション

製品技術サポート

開発自動化

品質エンジニアリング

モダナイゼーション

# SystemDirector Enterprise 概要

**業務システム構築基盤 SDE はオープン開発言語に対応したインターネット・クラウドシステムの構築を支援するソフトウェア開発環境です**

## メリット

ムダを排除し最大限効率化された開発プロセスを標準化  
オープン技術のメリットを享受しつつ、独自技術により革新的な高生産性、高品質を実現

### ● 開発方法論

- ✓ 各種業界標準 (SLCP2007やISO15408等) と整合した開発方法論を提供
- ✓ NECが長年培った技術やノウハウを概説書、手順書、ドキュメントサンプル等の体系的なガイドとして提供

### ● 開発環境

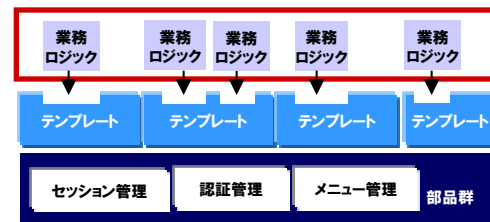
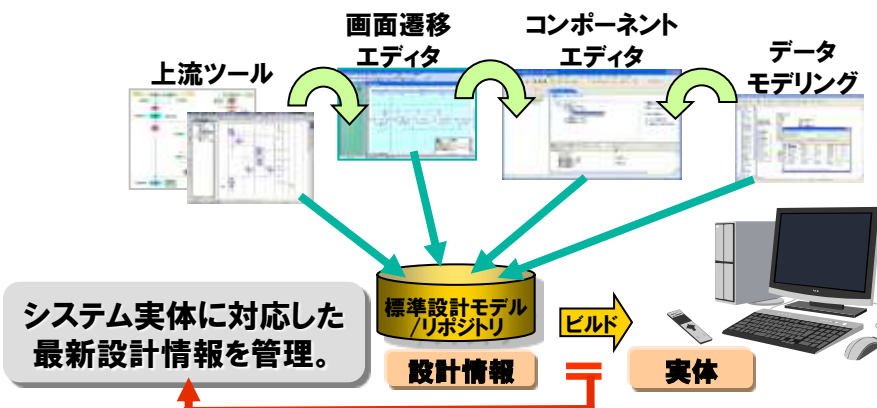
- ✓ フレームワークとGUIベースのツール群を提供、高度な技術をラッピングし高品質な開発を実現
- ✓ 設計情報を管理しコードを自動生成するジェネレータ機構により高生産性を実現

### ● 対応プラットフォーム

対応OS : Linux、Windows、Android、iOS 他

対応言語: Java、C#、C(※1)、PL/SQL(※1) 他

※1: バッチシステムのみ



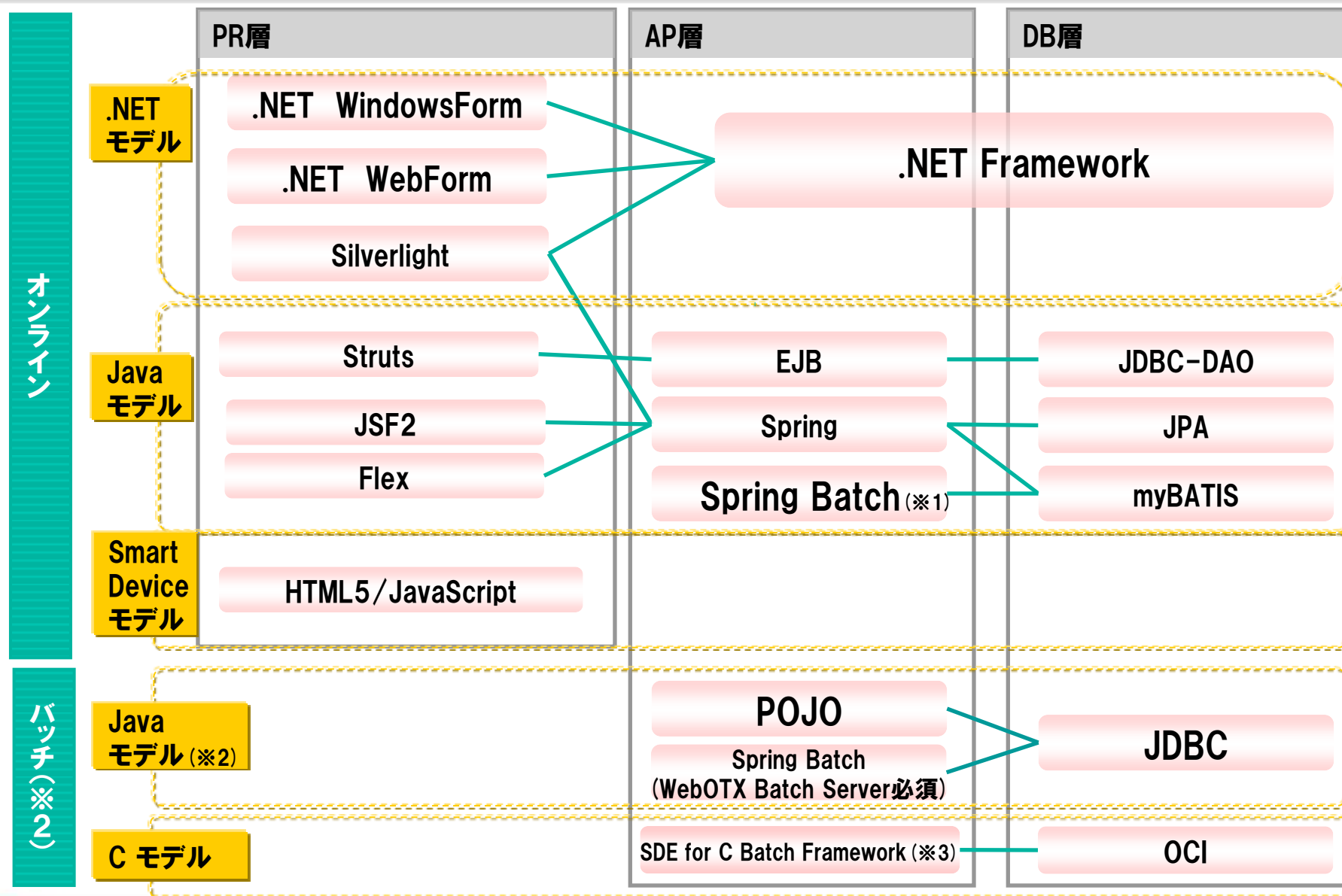
開発者がコーディング  
ジェネレータによる自動生成  
共通機能はあらかじめ準備

開発者は業務ロジックの開発のみに集中することが可能

# SystemDirector Enterprise

## 標準アプリケーションモデル

※1 小規模簡易バッチAP開発向け  
 ※2 大・中規模高速バッチAP開発向け  
 ※3 NEC製フレームワーク

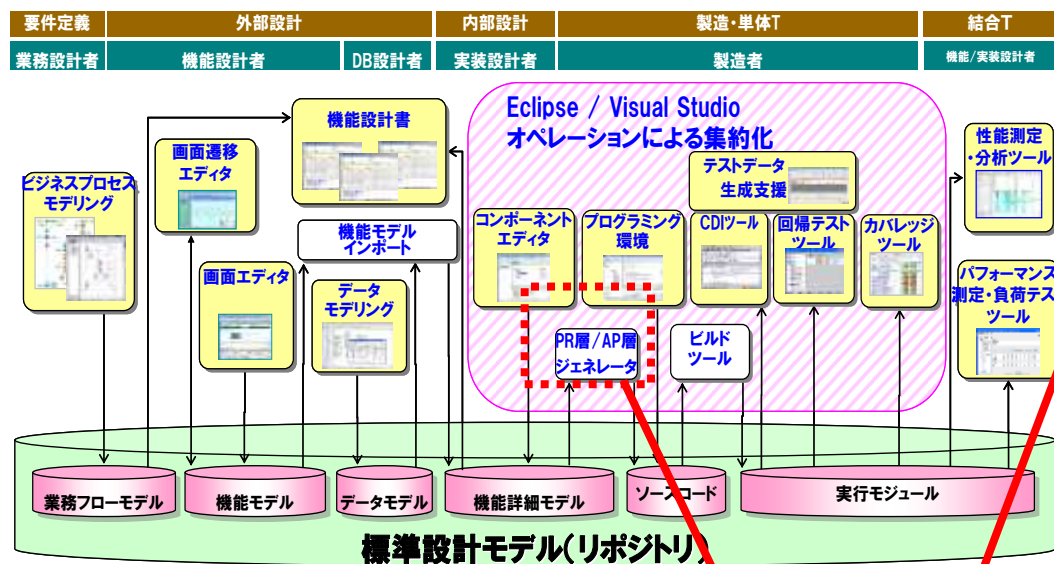




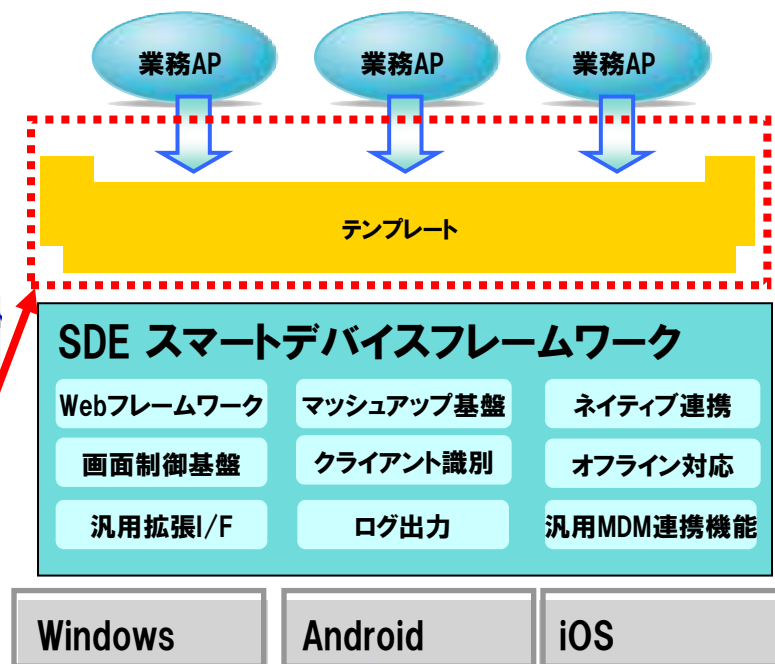
# スマートデバイス開発環境

開発方法論に沿った形で、要件定義～単体テスト工程までの各作業を効率化するための開発ツール、業界標準に対応した実装フレームワークにより、高品質なシステムの早期実現を支援します

## 開発ツール



## 実装フレームワーク



point 1

上流から下流まで一貫した、各種ツール群により、上工程からの品質の作り込みを支援

point 2

ジェネレート機構により、実装の自動化、設計情報と実装情報の乖離を防止

point 3

高度な技術をラッピングするアーキテクチャおよびフレームワークにより、属人性を排除した、高品質なシステムを支援



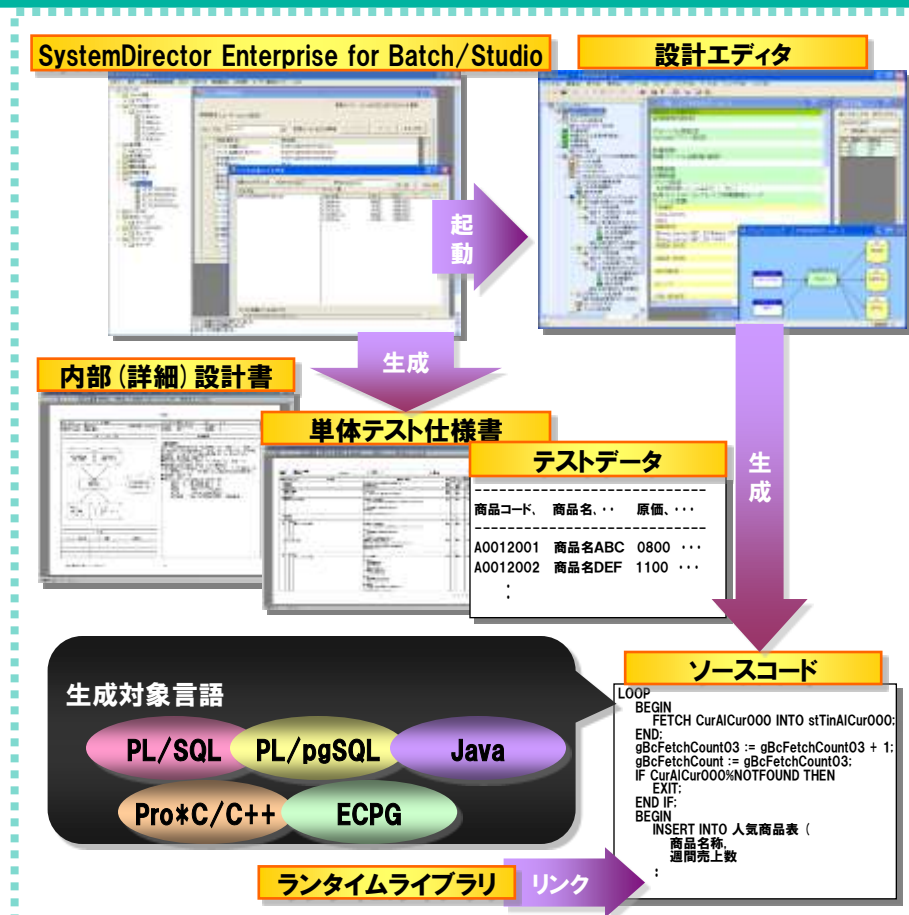
# 大規模Java／Cバッチ開発環境

バッチ系システム構築における開発方法論、開発基盤、サポートサービスを体系化し、各開発工程間の作業をシームレスに連携したバッチ系SI統合開発環境です。

過去10年以上に渡り、公共・製造・流通系のお客様のSI案件で適用されており、大規模案件の事例が多数あります。

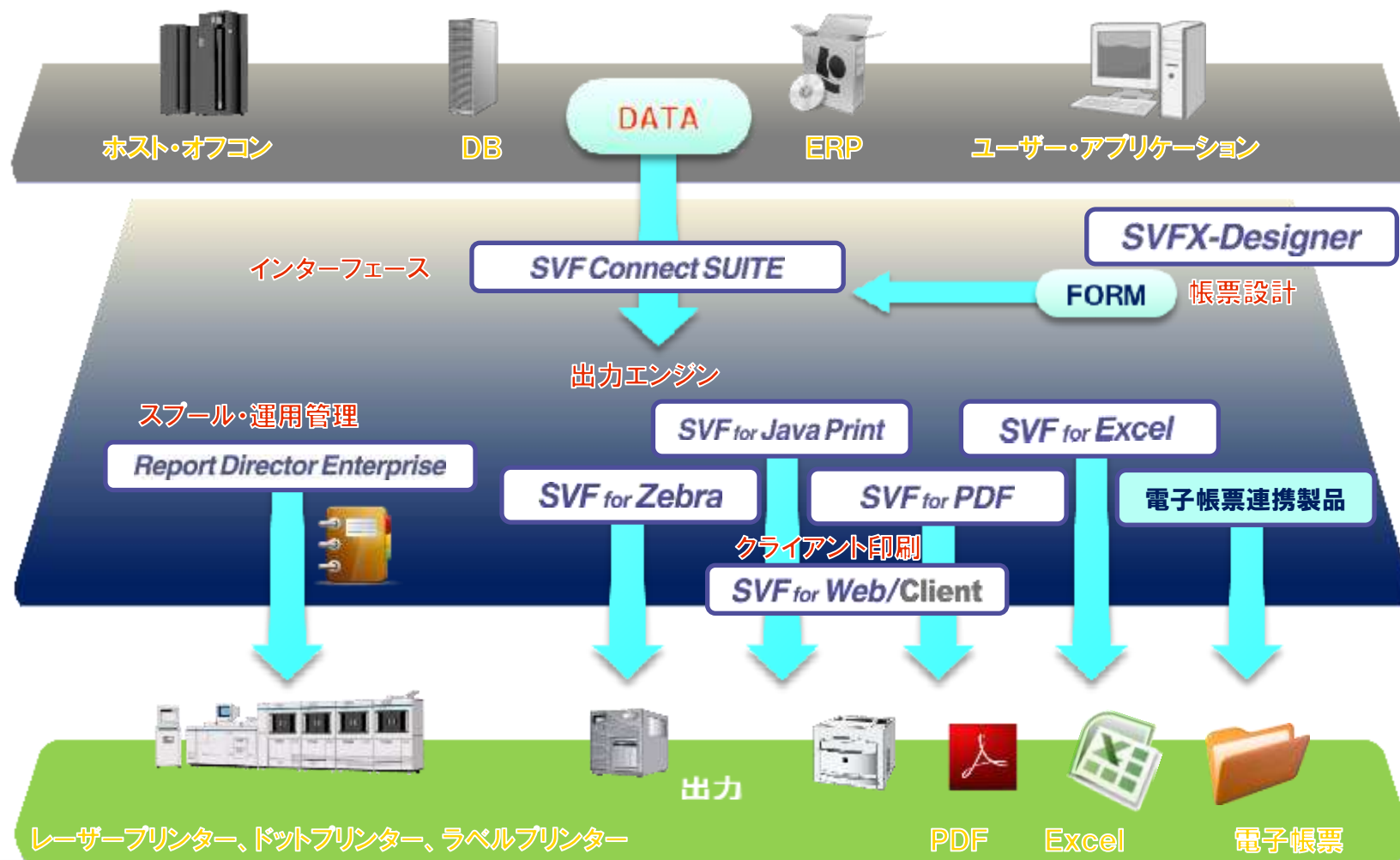
## 主な機能と導入効果

1. バッチ系システム開発 (SI) の生産性向上
  - ビジュアルなアプリケーション設計を可能とし、簡単な操作で**ソースコードを自動生成**
  - 生成対象言語は、**Java**、Pro\*C/C++、PL/SQLをサポート
2. バッチAPの品質・実行性能の向上
  - **フレームワークに依存しない柔軟な設計**ができ、PJ毎のフレームワークカスタマイズも可能
  - Javaバッチに特化した**高速DB/ファイル処理**ライブラリを提供、**JavaとCで同等の実行性能**
3. バッチAPの保守性の向上
  - **Javaバッチ詳細/テスト仕様書**を生成し、ソースと仕様書の一貫性を維持
  - プロセス指向の処理方式を採用し**現行資産移行も容易**



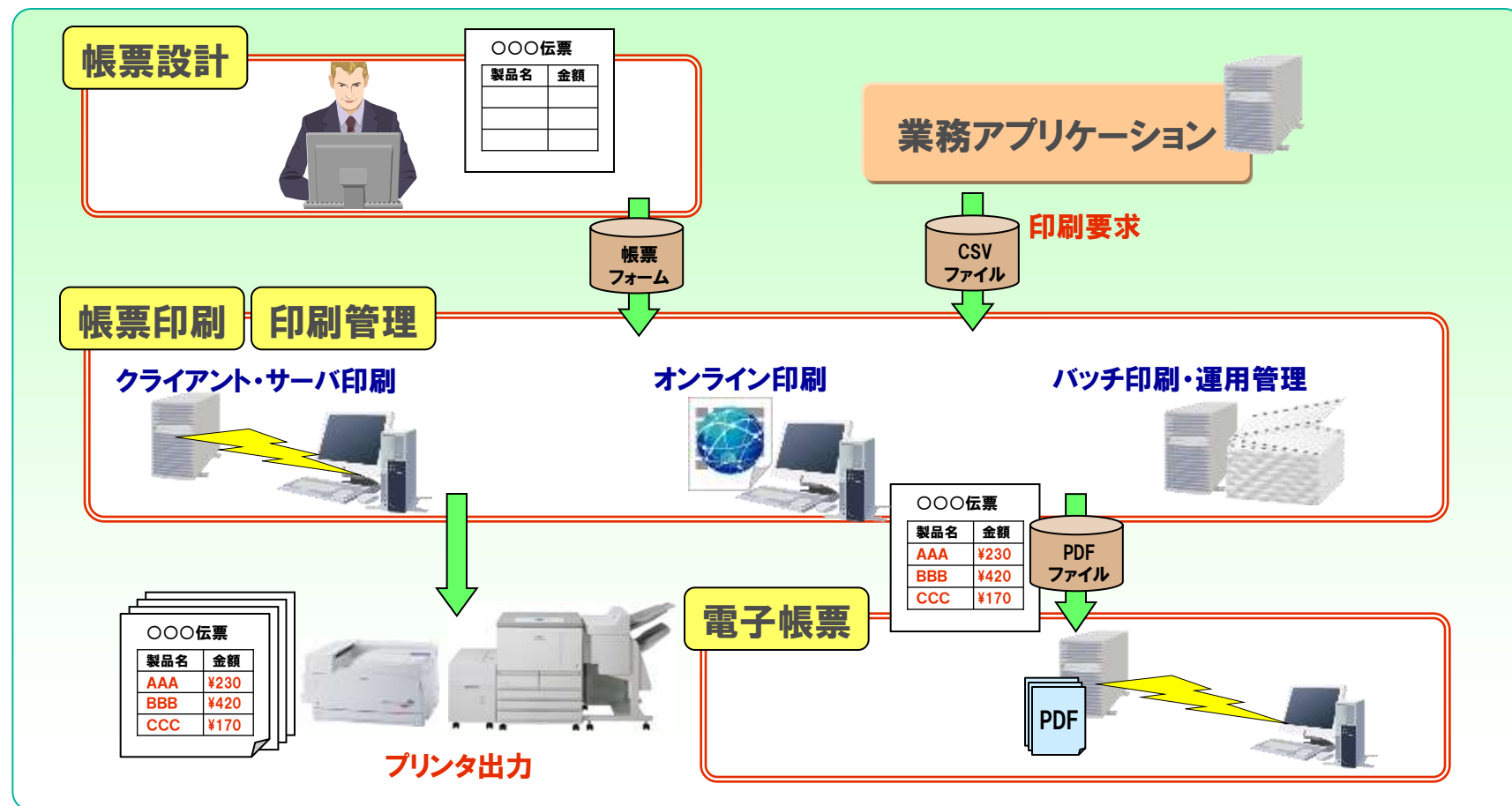
# 総合帳票基盤 WebSAM SVF

- 「SVF」1つで業種・業務を問わず、社内帳票から顧客向け帳票まで設計可能
- 要件に応じて複数の製品を使い分ける必要がなく、開発コスト削減を実現



# 統合帳票運用支援ソフトウェア WebSAM Rakuform

「WebSAM Rakuform」は、企業における帳票印刷システムの構築から運用までをカバーする、「統合帳票運用支援ソフトウェア」です。



# 大量データ高速ソート・マージユーティリティ SORTKIT

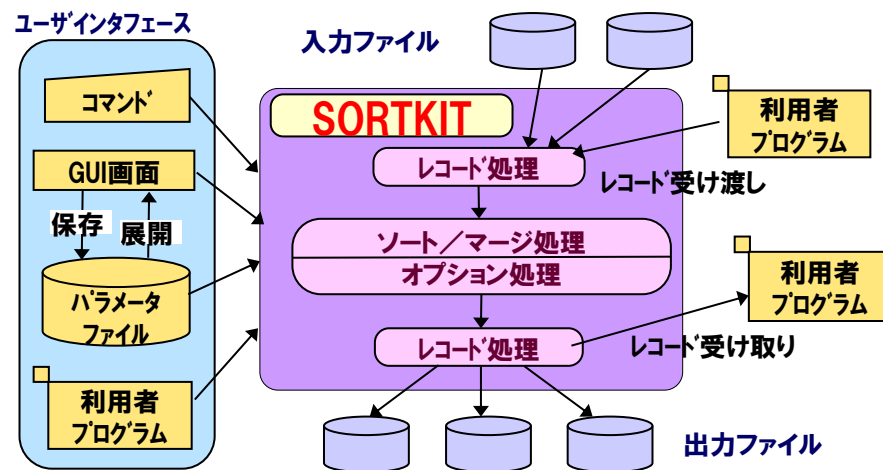
## <製品概要>

大量データを高速処理するためのソート／マージ専用ユーティリティ。多様なデータフォーマットやレコードフォーマットに対応し、大量データを処理するバッチ処理や業務アプリケーションのソート処理を高速化。

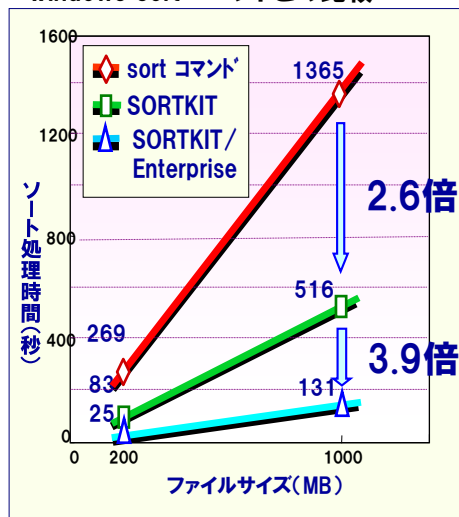
## <導入効果と機能>

- 大量データの処理に適した高速ソートアルゴリズムの採用によるソート処理時間を短縮。
- 既存のバッチプログラムや業務アプリケーションと連携し、大量データ処理の高速化を実現
  - ・充実したインタフェースで各種の実行形態に対応。
    - バッチコマンドによるバッチ処理
    - GUIによる対話的なオプション設定や実行
    - NEC COBOLの整列併合機能との連携
    - C言語で作成されたアプリケーションからAPIとして利用
    - MS Visual BasicなどのアプリケーションからActiveXコントロールを利用
  - ・ビジネス用途に必要な各ファイルタイプをサポート。
    - テキストファイル(可変長レコード・可変長フィールド形式)
    - NEC COBOL形式、MicroFocus COBOL形式
    - NEC版C-ISAM形式、IFASPRO 形式、RAF 形式
  - ・ビジネス業務に適した各データタイプをサポート。
    - ASCII／1バイトJIS
    - EBCDIC／EBCDICカタカナ
    - 漢字コード(シフトJIS／JIPS／EUC／UCS-2)

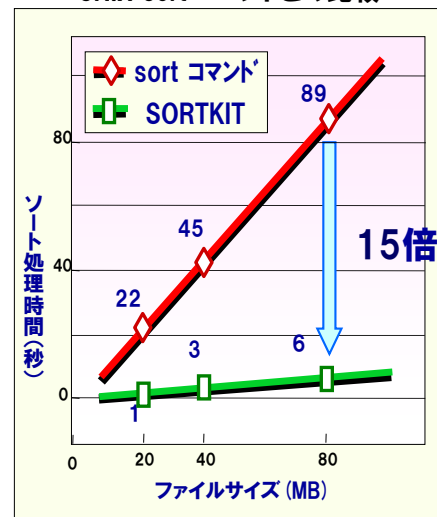
## イメージ図



Windows sort コマンドとの比較



UNIX sort コマンドとの比較



## 汎用マクロプロセッサ GMP

言語に依存しない強力なマクロ定義・参照機能によって、APの開発品質・生産性、保守性を高め、コスト削減に効果を発揮します

## 【GMPの特徴】

① APとマクロ定義を分離→APのマクロ参照でマクロ定義を引用しマクロ展開

マクロ定義	雛形ソースとカスタマイズ規則、生成規則をプログラミング
マクロ参照	雛形ソースに対するカスタマイズ/生成パラメータを伴うマクロ呼び出しをAPソースプログラムに記述
マクロ展開	マクロ定義にカスタマイズ/生成パラメータを適用してカスタマイズされた雛形ソース等を生成

② プログラミング言語非依存→様々なプログラミング言語で活用可能

③ 強力なマクロプログラミング機能で複雑なルールによるソース生成にも対応

## 【GMP利用により期待される効果】

メインフレームの生産性、品質、保守性を支えたマクロ機能をオープン環境でも提供

## 開発生産性向上

雛形ソースの自動カスタマイズ、ルールに基づくソース生成等によってコーディング量を大幅に削減し、生産性を向上

## 開発品質向上

コーディング量削減による品質低下機会の低減に加えて、具体的な処理内容のブラックボックス化でスキル不足による品質低下等を抑止

## 保守性向上

OS等のI/F、他プログラムI/Fの隠ぺいにより、APソースの他への依存性を低減し、I/F変更をマクロ定義に限定することで保守性を向上

## ② マクロを利用したプログラミング



開発者  
保守担当者

利用

開発・保守  
対象ソース  
(マクロ参照利用)

```
...
¥M1 Y1,Y2;←マクロ参照
...
```

## ③ マクロ参照のソース生成(マクロ展開)

汎用マクロプロセッサ  
GMP

コンパイル  
対象ソース  
(マクロ展開済)

```
...
/*¥M1 Y1,Y2*/;
X=Lookup(Y1);
Y=Read(X,Y2);
...
```

マクロ展開  
ソース

コンパイラ  
(COBOL,C,Java,...)

オブジェクト  
プログラム

マクロ仕様ドキュメント



設計者

## ① マクロを設計し、マクロ定義を登録

マクロ定義

```
¥DEF M1(...);
...
¥END;
```

参照

マクロ定義

登録

C言語のマクロ機能を凌ぐ強力なマクロ機能によりAPの生産性、品質、保守性の改善に効果を発揮



# OpenMeisterEnterprise

**業務アプリケーションを開発するためのテンプレートベースの開発フレームワークです。「コスト削減」「期間短縮」「技術的リスク低減」を実現します**

JavaEEを利用したEF版と、.NET Frameworkを利用した.NET版を提供

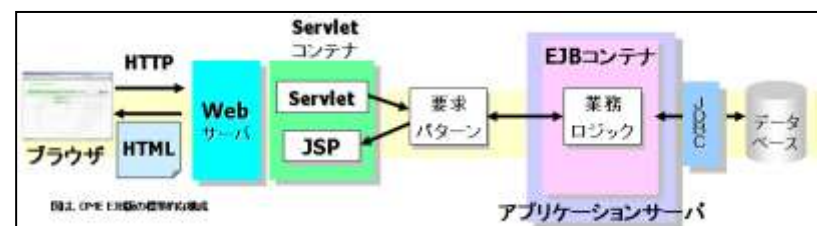
品質の良い業務アプリケーション開発に必要な「システム基盤（フレームワーク）」と「開発ガイド（開発標準・ドキュメント群）」を提供

評価版貸出サービスにより、ご購入前の事前評価が可能

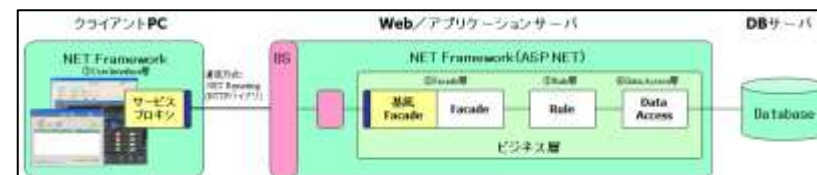
業種・業務を問わず、さまざまなユーザやプロジェクトでの豊富な利用実績

お客様の主な業種：医療機器メーカー、飲料メーカー、官庁・公共団体、金属メーカー、金融機関、娯楽サービス会社、自動車メーカー、自動車部品メーカー、商社、食品メーカー、スポーツ用品メーカー、製薬会社、ディスプレイメーカー、鉄道会社、電力会社、物流会社など（1000プロジェクト以上の利用実績）

OpenMeisterEnterprise/EF システム基盤の構成例



OpenMeisterEnterprise.NET 2005 システム基盤の構成例



## InfoFrame DataBooster

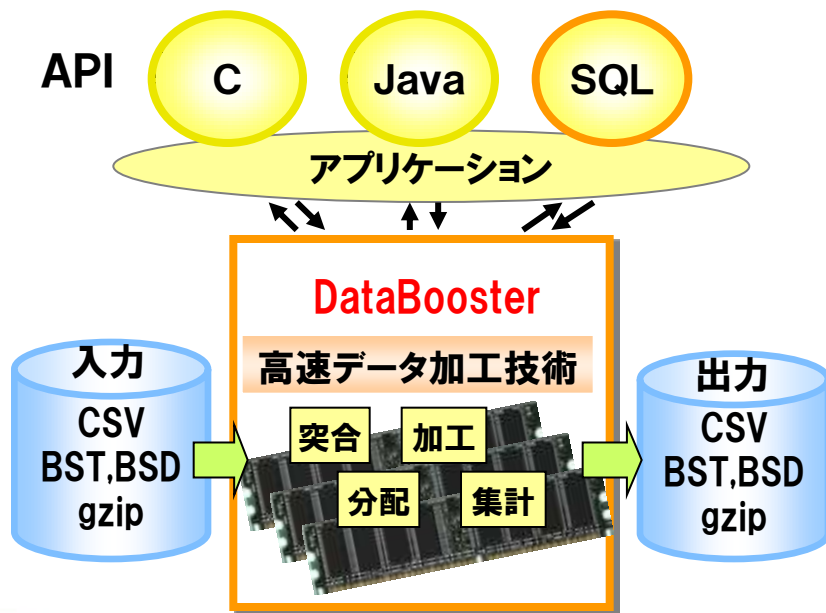
インメモリ技術

カラムストア技術

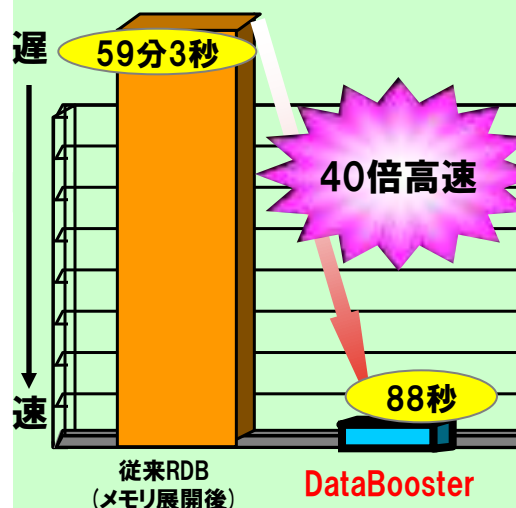
高速データ加工技術を核としたバッチ処理高速化製品  
データベースチューニングでは到達不可能な高速性を実現

DataBoosterはリレーショナルデータベースが苦手とする処理が得意です

大量データの { グループ処理 (集計)  
ジョイン処理 (結合)  
ソート処理 (並び替え)  
一括更新 (コード洗替処理) } などで高速処理を実現します



## インメモリ技術による高速性の一例



商品マスタ: 1万件、取引明細表: 2億件をジョインし、商品で集計

※SQL

```
SELECT 商品, SUM(売上)
FROM 売上マスタ a, 取引明細表 b
WHERE  a.商品ID = b.商品ID
      AND 日付 >= '20070101'
GROUP BY 商品
ORDER BY 2 DESC
```

【測定条件】

CPU: Intel Xeon MP 3.66GHz\*4  
メモリ: 24GB



# ソフトウェア・ファクトリ

集約

ツール・資産・開発管理技術標準類・  
テンプレート類をセンターにて集中管理

マネジメント力向上

リアルタイム性と  
データ精度向上

標準化

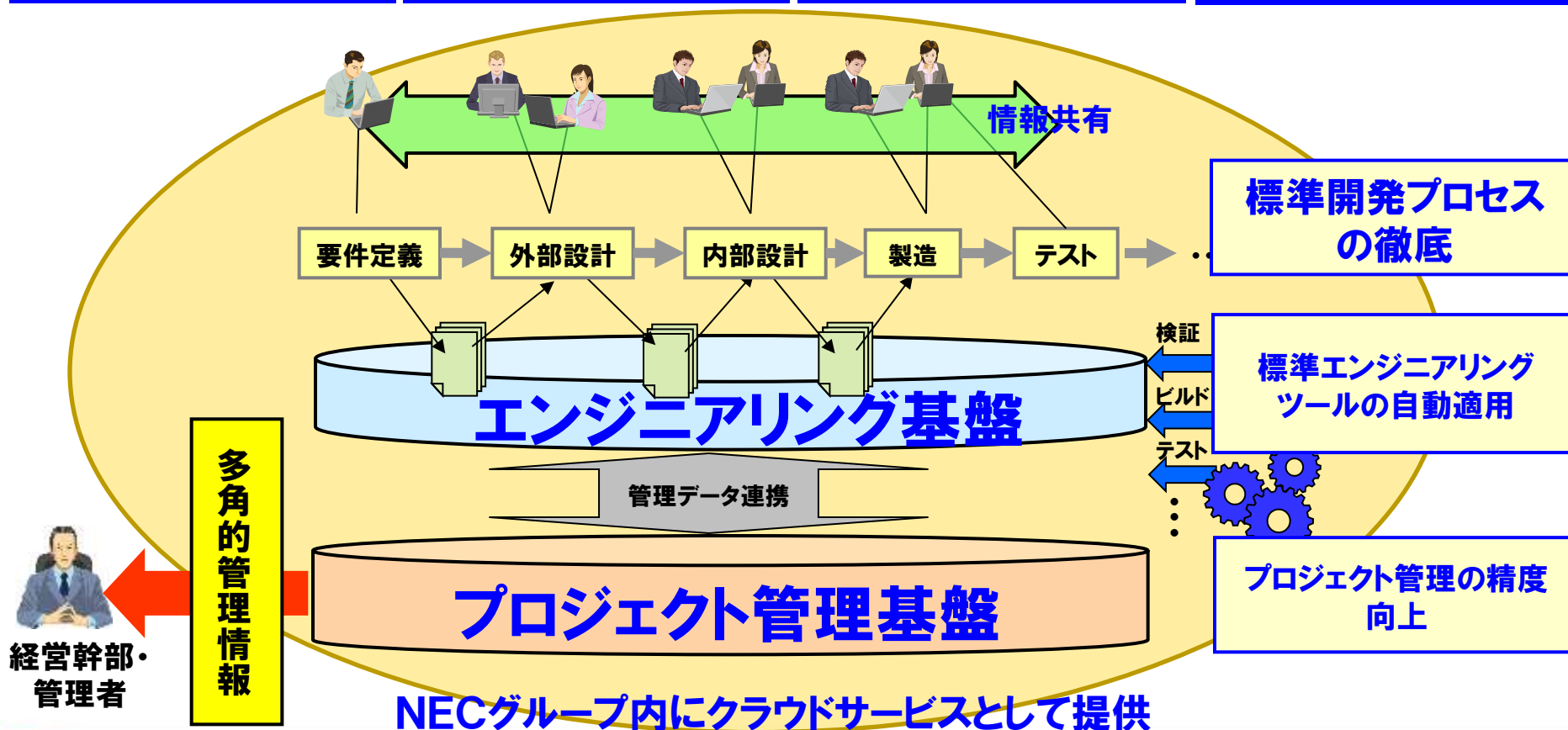
開発プロセス、開発方法論、  
ツールの一元化

自動化

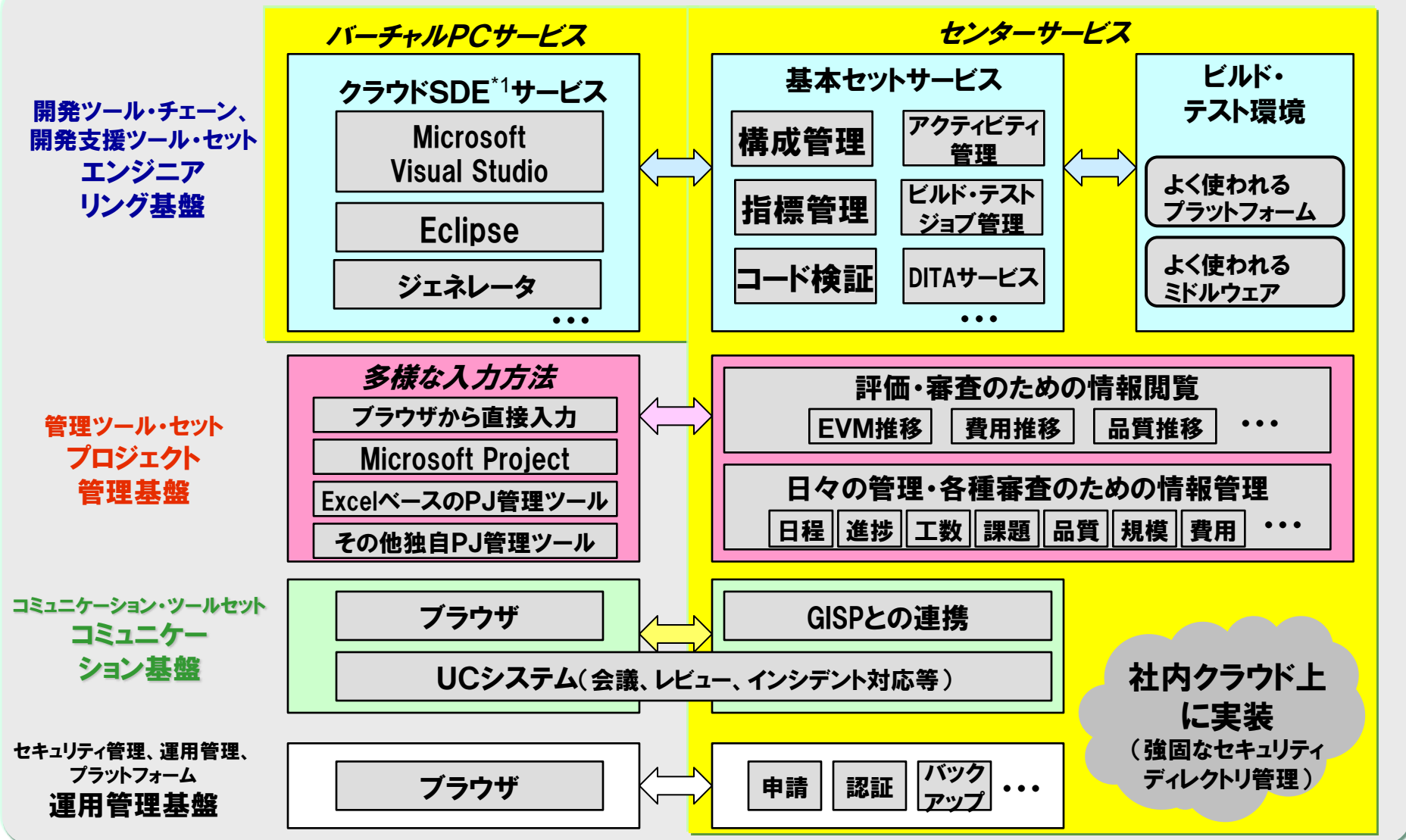
エンジニアリング技術・  
ツールを全てに適用

再利用

設計情報、設計ノウハウ等  
の展開、検索性向上



# ソフトウェア・ファクトリの機能構成



\*1. SDE: SystemDirector Enterprise

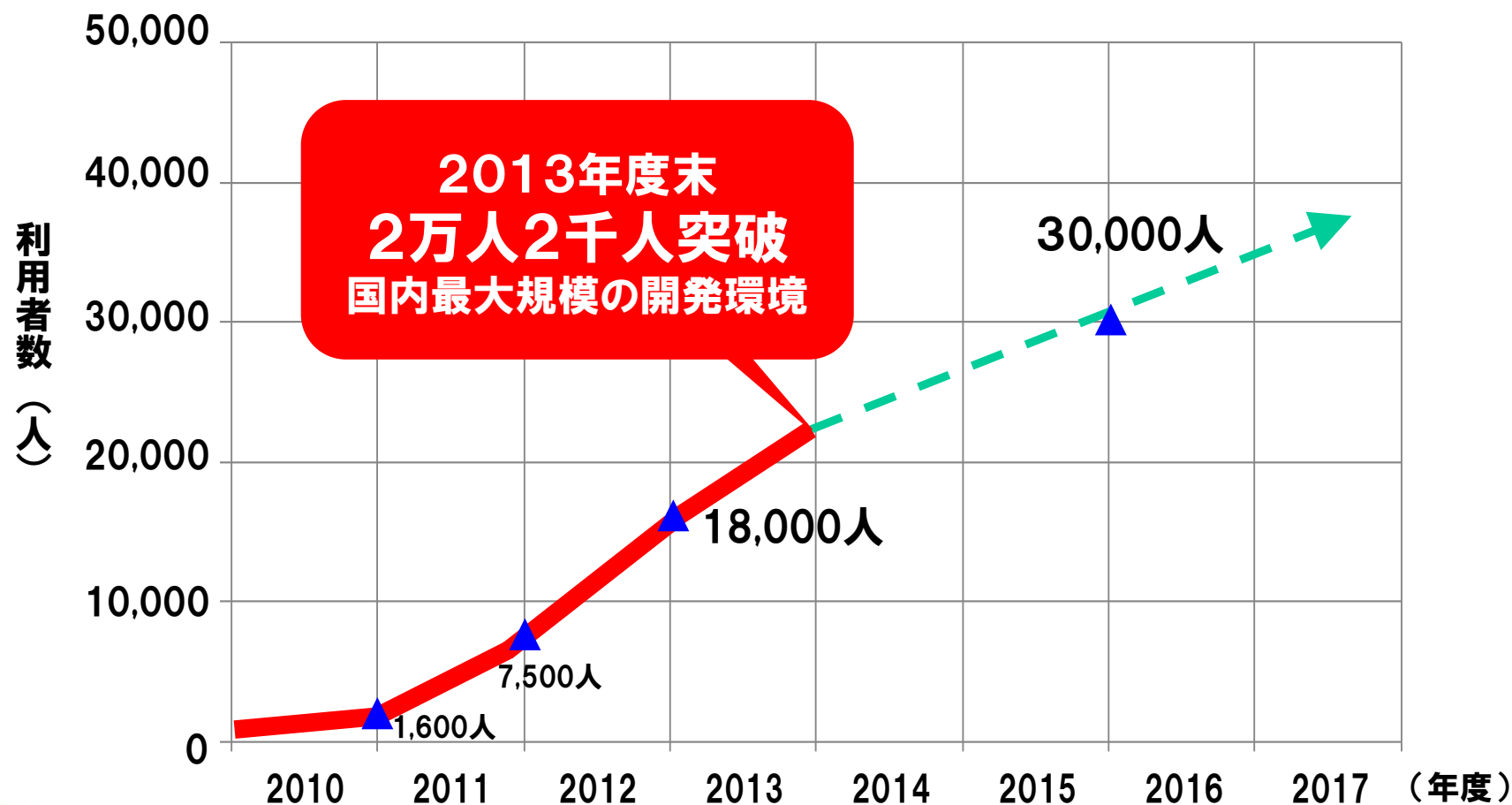
# ソフトウェア・ファクトリのメリット・効果

立場	区分	メリット・効果
経営視点	コスト	開発投資全体(設備費、教育費など)の最適化実現
	統制	指示・施策の展開、徹底が可能
	リスク管理	社会的要請へ対応の容易化(電力節約、CO2排出量規制など)
		BC/DR対策が可能
	HR関連	属人性からの脱却
		人材流動性向上
実務者視点	コスト	商用開発ツールの安価な活用を含む開発環境が入手可能
		開発標準の整備が不要(方法論、アプリケーションフレームワーク)
		開発付帯業務(購入、導入、教育、バックアップなど)
		一時的な設備に関わるコストが不要
	スピード	選定(HW、SW)、納品までのリードタイム、設置・インストールが不要
	見える化	現物管理による正確な状況把握が可能
	リスク管理	セキュリティ問題、データ紛失などを委託可能

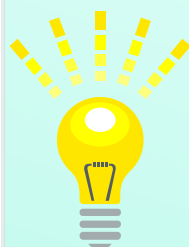
# NECグループ内への展開状況

■ 既に社内のSI/SW開発の主要な開発環境として定着(2009年度より適用を開始)

■ 参加可能な開発者全てがソフトウェアファクトリを利用することが目標  
⇒2015年度末に、3万人の利用を目指す



# お問い合わせ先



SystemDirectorでは業務システム開発環境に関する  
様々なご質問やご相談にお応えします  
例えば・・・

再構築を検討中のお客様に何を提案したら良いか  
業務アプリケーションの開発や保守を効率化するにはどうしたら良いか  
既存資産をどう活用したら良いか

などお気軽にご相談ください



## ●ご購入前のお問い合わせ

NEC システムソフトウェア事業部 SystemDirector ご相談窓口

● Web <http://jpn.nec.com/SystemDirector/contact.html>

本書に記載されている会社名、製品名は、各社の登録商標または商標です。

# Orchestrating a brighter world

世界の想いを、未来へつなげる。

**未来に向かい、人が生きる、豊かに生きるために欠かせないもの。  
それは「安全」「安心」「効率」「公平」という価値が実現された社会です。**

**NECは、ネットワーク技術とコンピューティング技術をあわせ持つ類のないインテグレーターとして  
リーダーシップを発揮し、卓越した技術とさまざまな知見やアイデアを融合することで、  
世界の国々や地域の人々と協奏しながら、  
明るく希望に満ちた暮らしと社会を実現し、未来につなげていきます。**

Empowered by Innovation

**NEC**