



Javaオブジェクト・モデリング

2008年4月30日

JJUG

浅海智晴



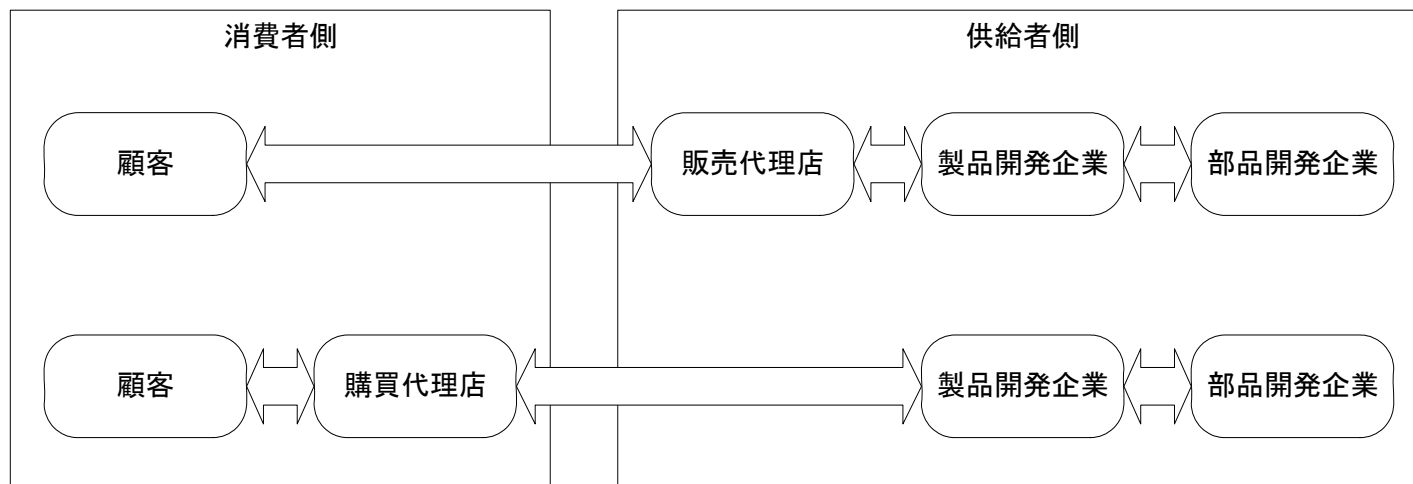
テーマ

- どこまでをモデルで作るのか？
- どこからJavaで作るのか？
- モデリングとJavaの接点

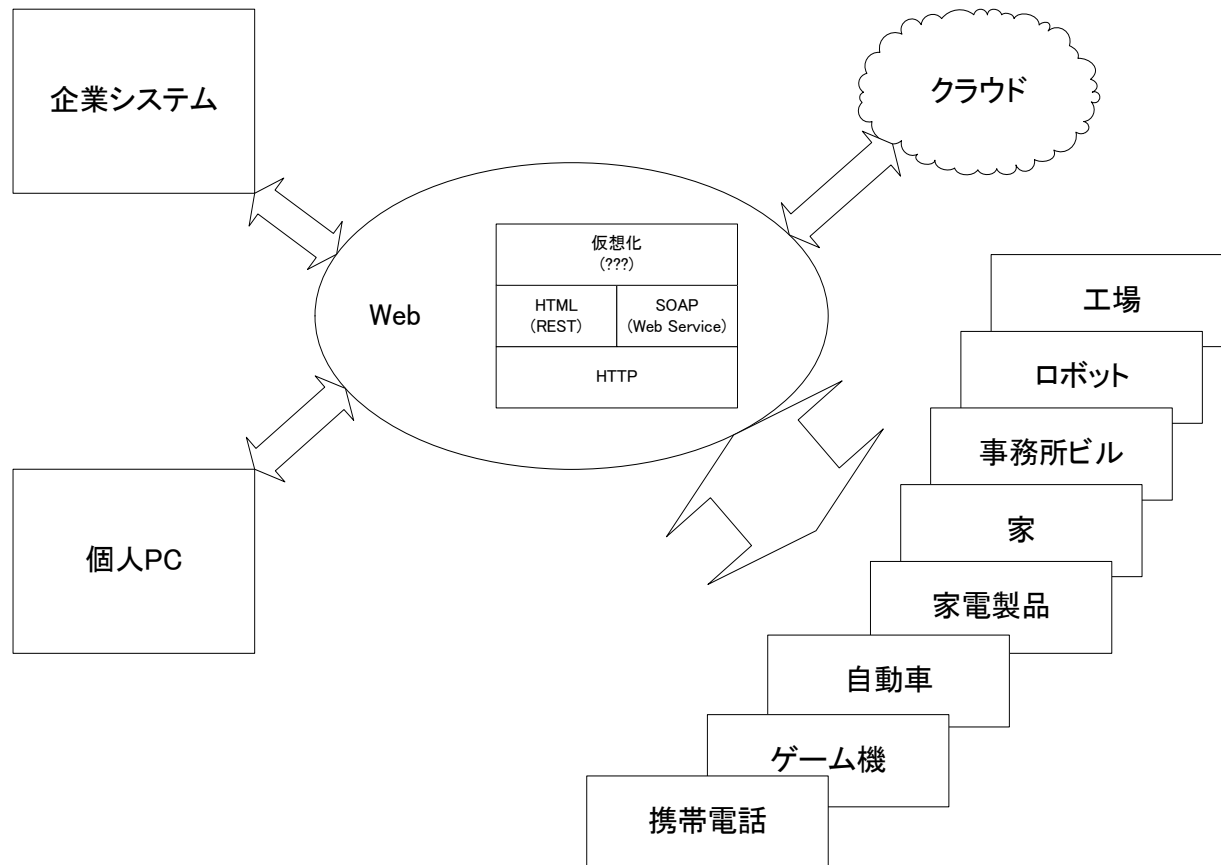


現状認識

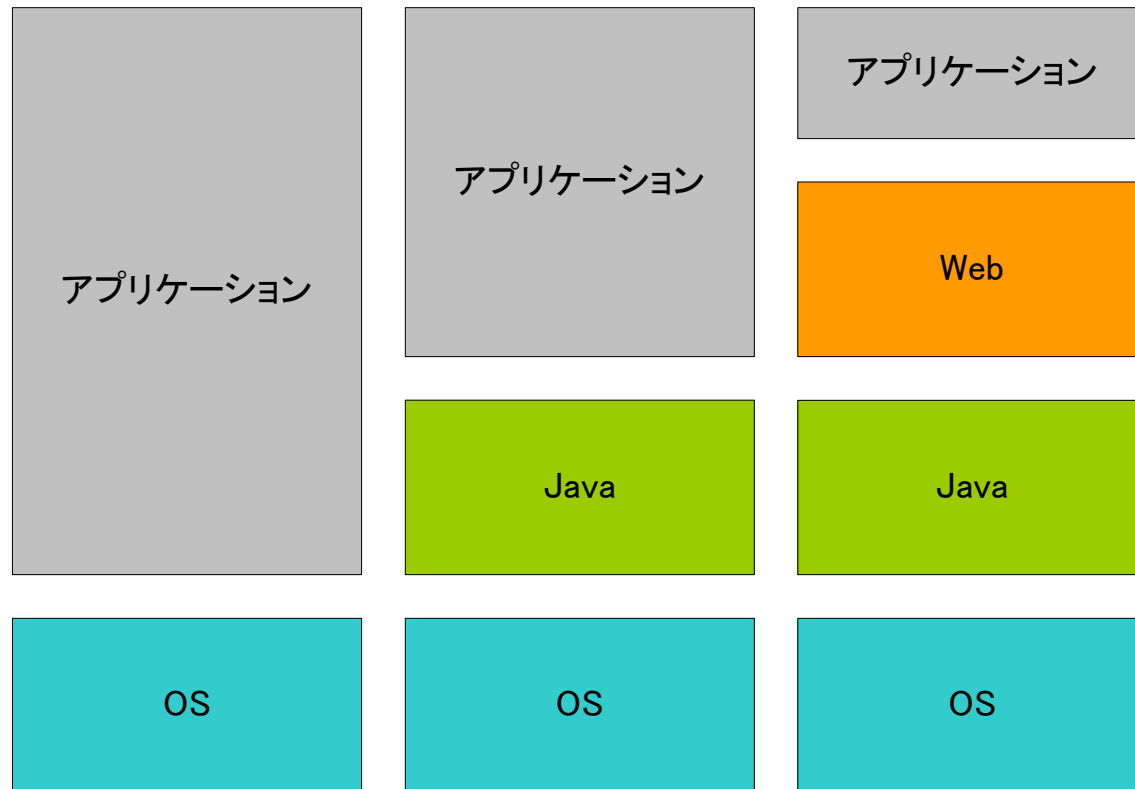
販売代理店から購買代理店へ



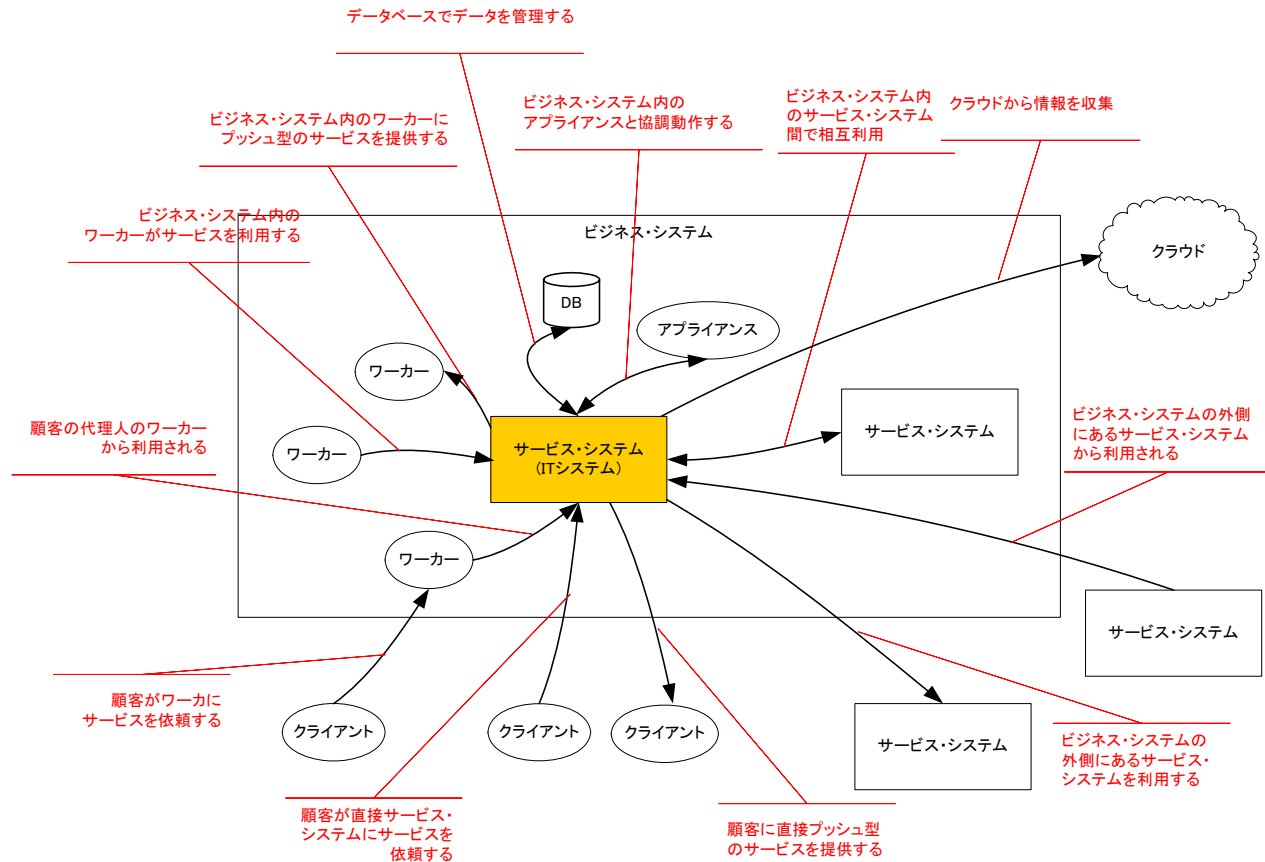
クラウド&アプライアンス・ コンピューティング



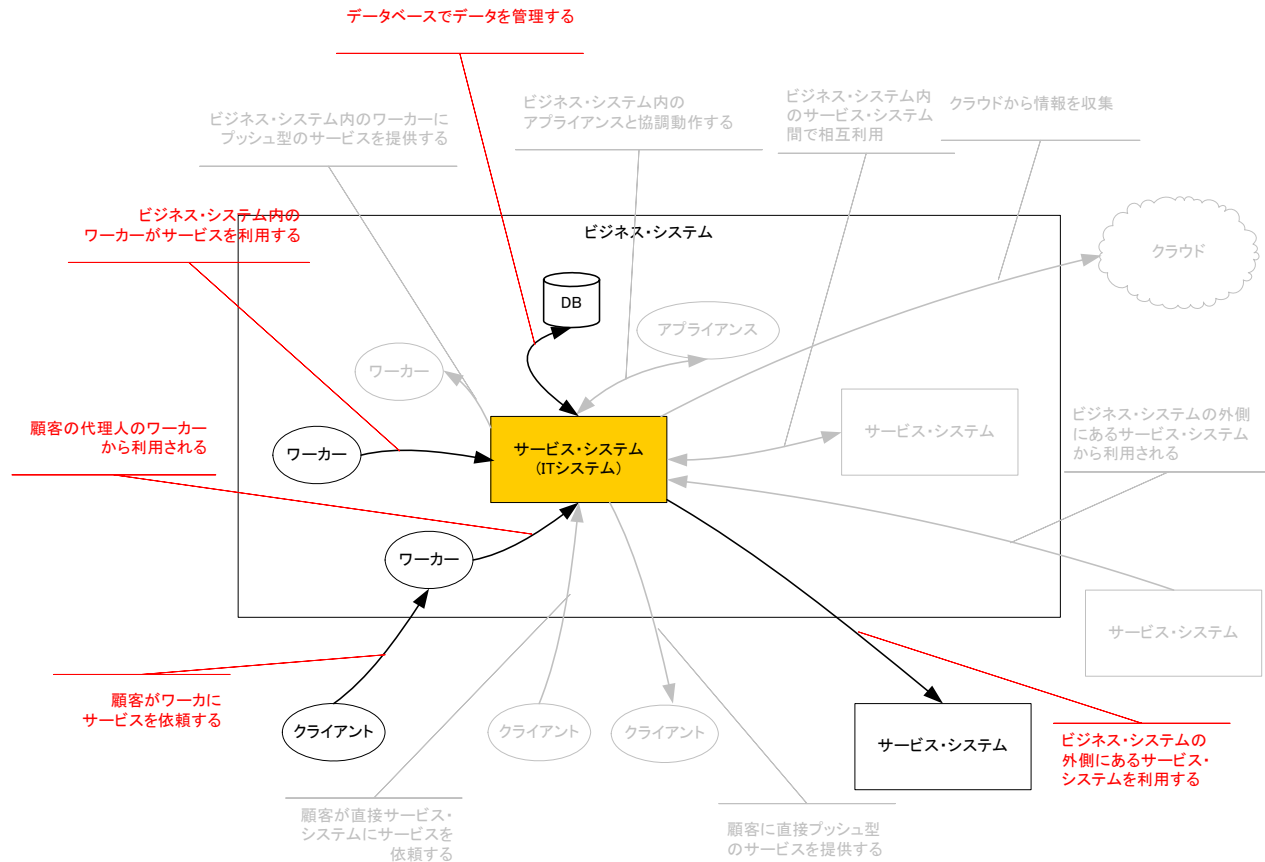
Webがプラットフォームになる



新世代システムの相互作用

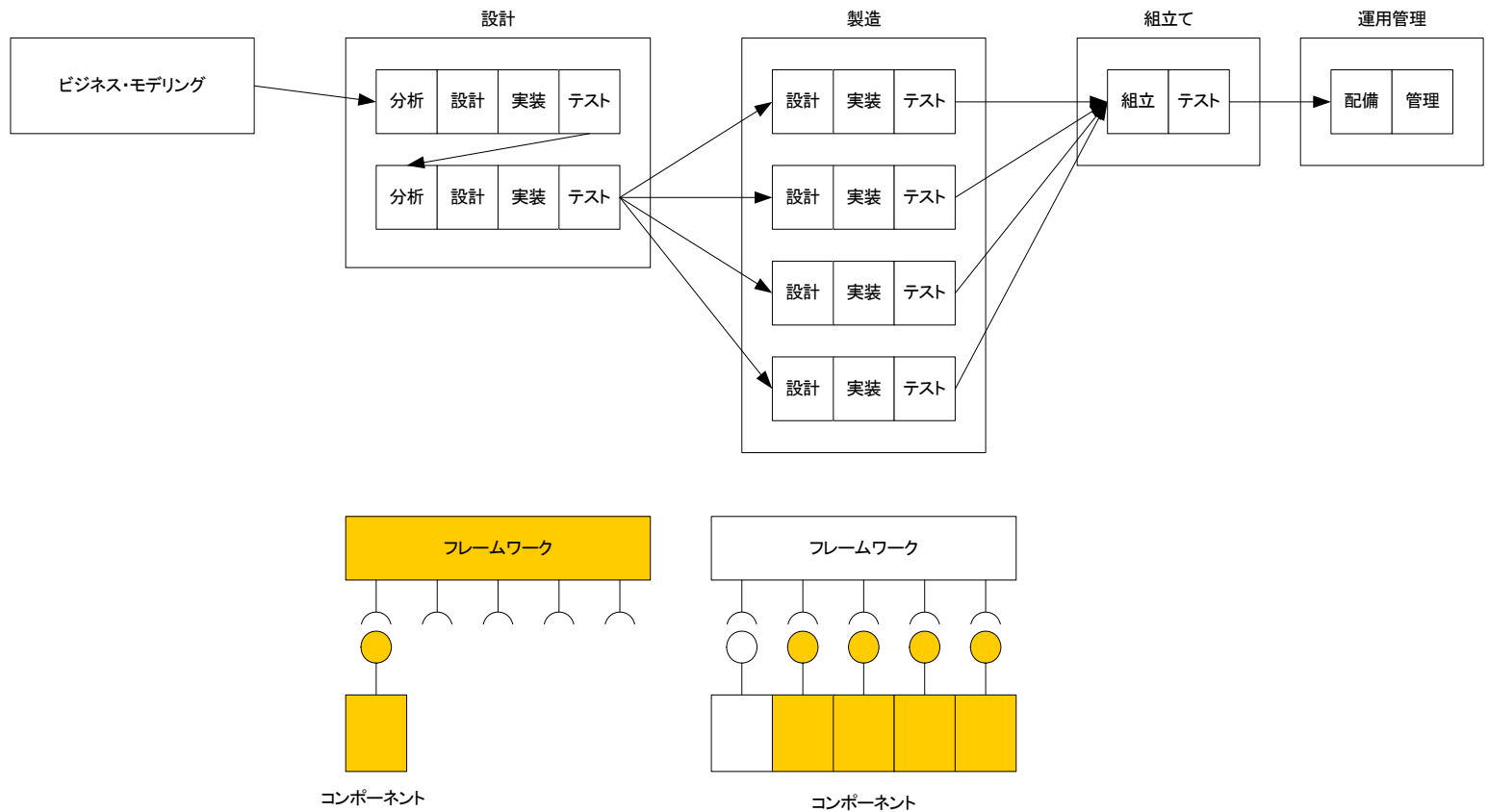


旧世代システムの相互作用(比較)



開発の流れ

Component Based Development



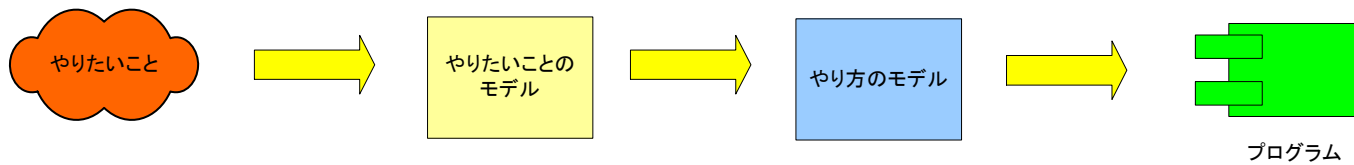
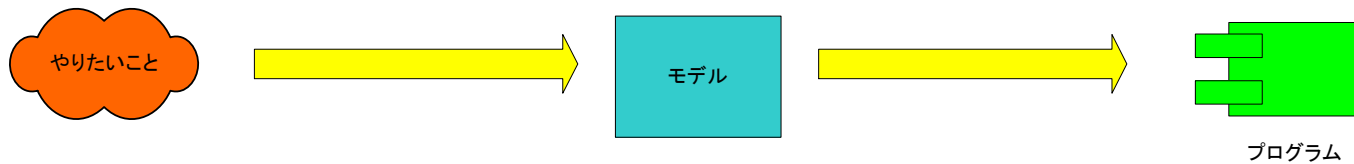
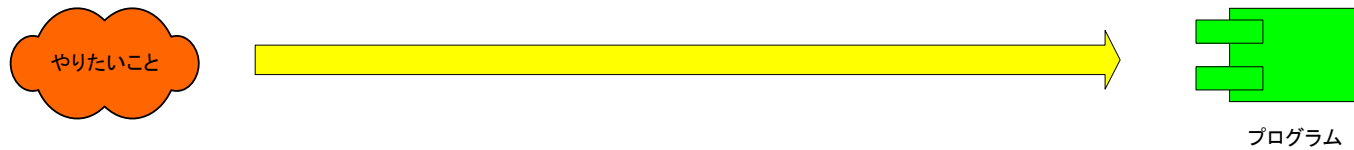
現状認識

- Webプラットフォーム上で、
- さまざまな形のエージェント（企業システム、デスクトップ、クラウド、アプライアンス、...）が参加者となる中で、
- サービスやコンポーネントを部品とし、
- 複雑化、高度化したコラボレーションを駆使して、
- 利用者視点のサービスを提供する。



SimpleModeling

モデリングの意味



"中流"モデリング

- 上流・"中流"・下流
- 上流のテーマ
 - ビジネス・モデリング
 - EA (Enterprise Architecture)
 - SOA (Service Oriented Architecture)
 - アーキテクト視点
- 下流のテーマ
 - オブジェクト指向プログラミング
 - Java, C#
 - JavaEE, .NET
 - Webサービス
 - エンジニア視点
- 中流の現状
 - ビジネス・モデリングを実装に結びつける具体的な手法
 - この分野の技術が開拓されていない
 - 2000年頃から停滞状態



SimpleModeling

- SimpleModeling HP
 - <http://simplemodeling.jp>
 - MindmapModelingの基盤となるメタ・モデル
- MindmapModeling HP
 - <http://mindmapmodeling.jp>
 - MindmapModelingの文法、サンプルなど
 - 開発中の情報
 - <http://mindmapmodeling.jp/model/index2.html>
 - <http://mindmapmodeling.jp/sample/yorozu/domain2.html>
- JavaDSL HP
 - <http://javads1.jp>
 - SimpleModelingのもう一つのDSL

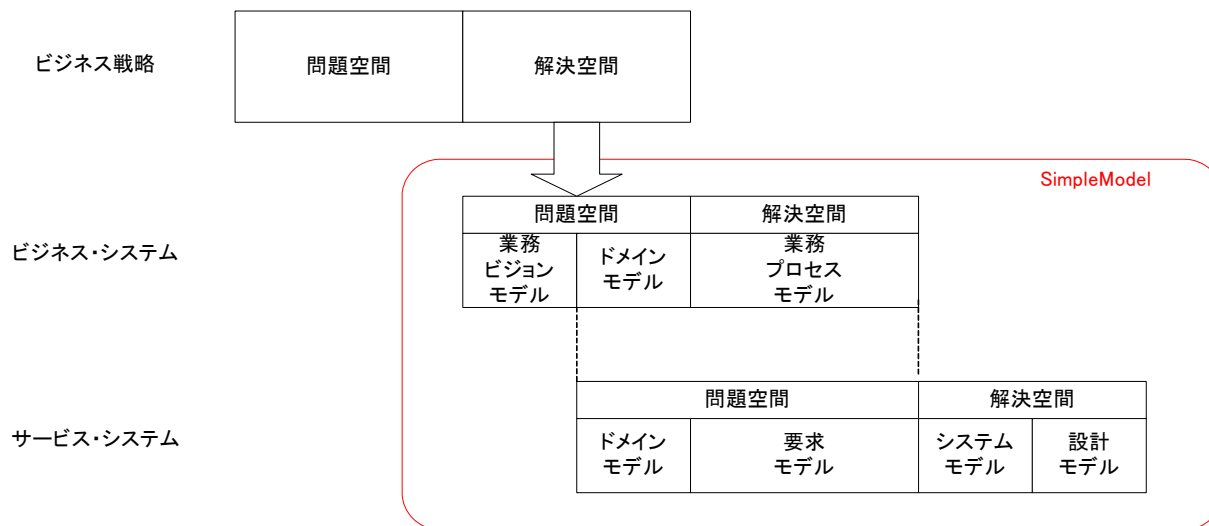
SimpleModeling

- モデル化対象の絞込み
 - 教育に適した範囲
 - ただし、実務に活用できないような単純化はしない
- 業務モデルとシステム・モデルの連携
 - 業務モデル、ドメイン・モデル、要求モデル
- ユースケース
 - 利用者視点
 - 業務フローで表現できないこと
 - 業務ユースケースとシステム・ユースケース
 - コラボレーションのモデル化
- 準備モデル
 - 補助線として利用できるモデル
 - マインドマップ
- Excel
 - 帳票ベースでモデリングすることで理解が深まる
 - UMLの習得だけでは、肝心なことは分からない

企業システム・モデリングの3階層構造

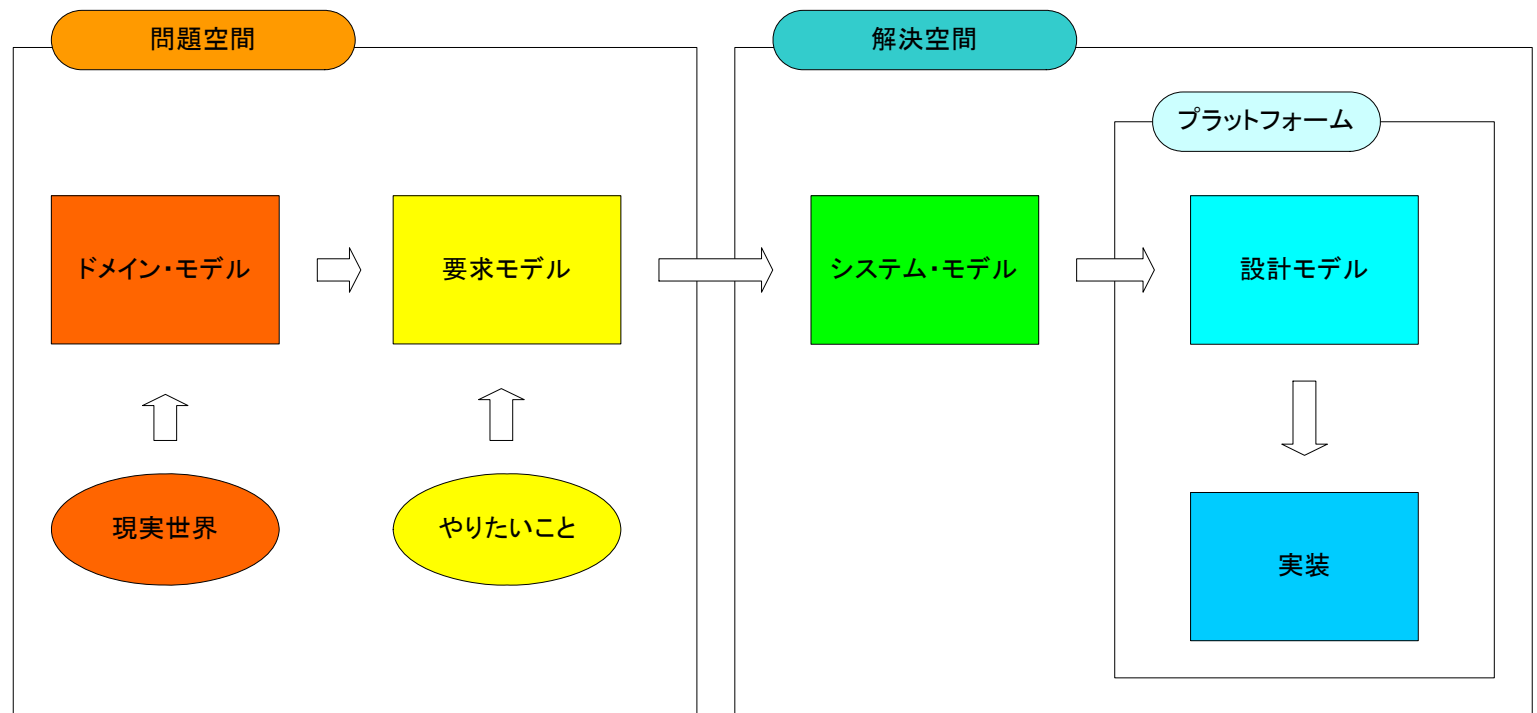


SimpleModelの対象範囲



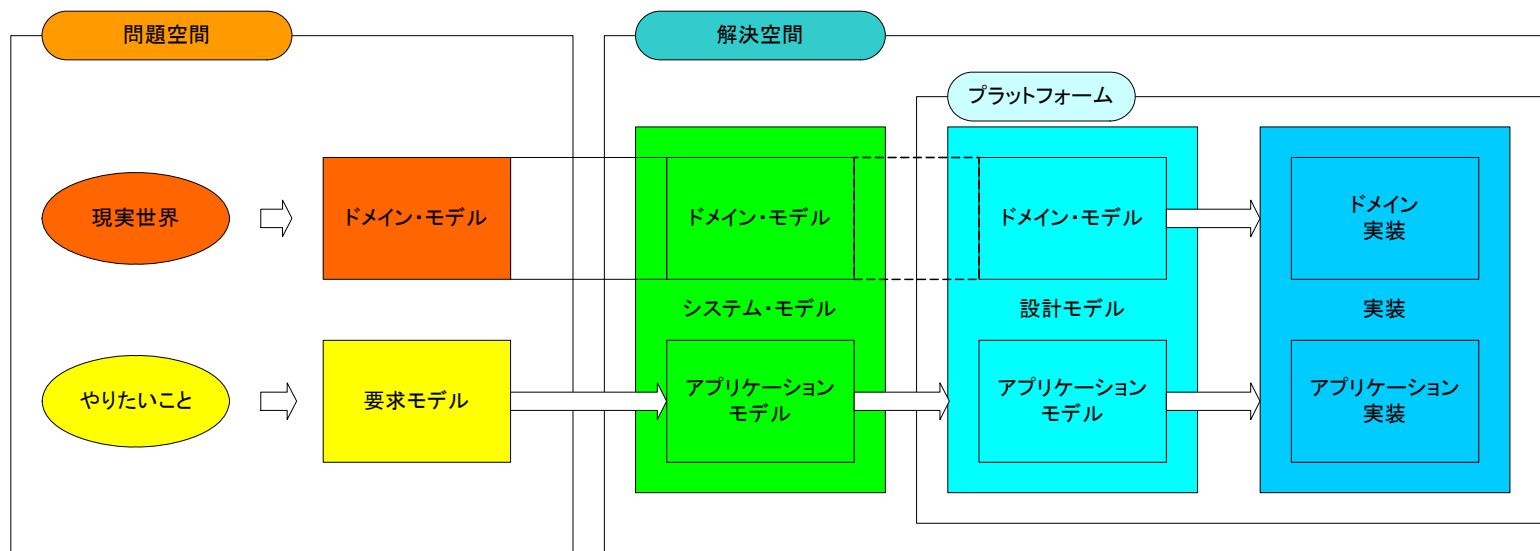
SimpleModeling

モデル変換/作業の流れ



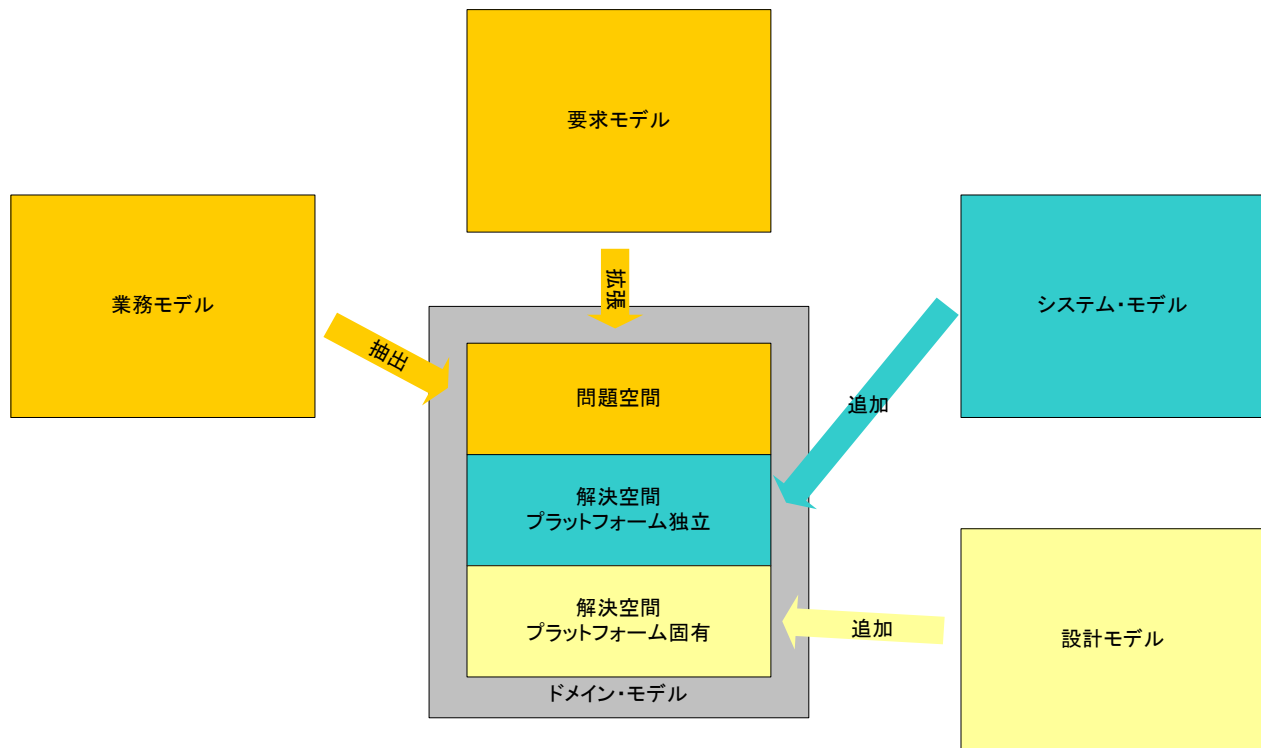
SimpleModeling

モデル変換/モデルの観点から



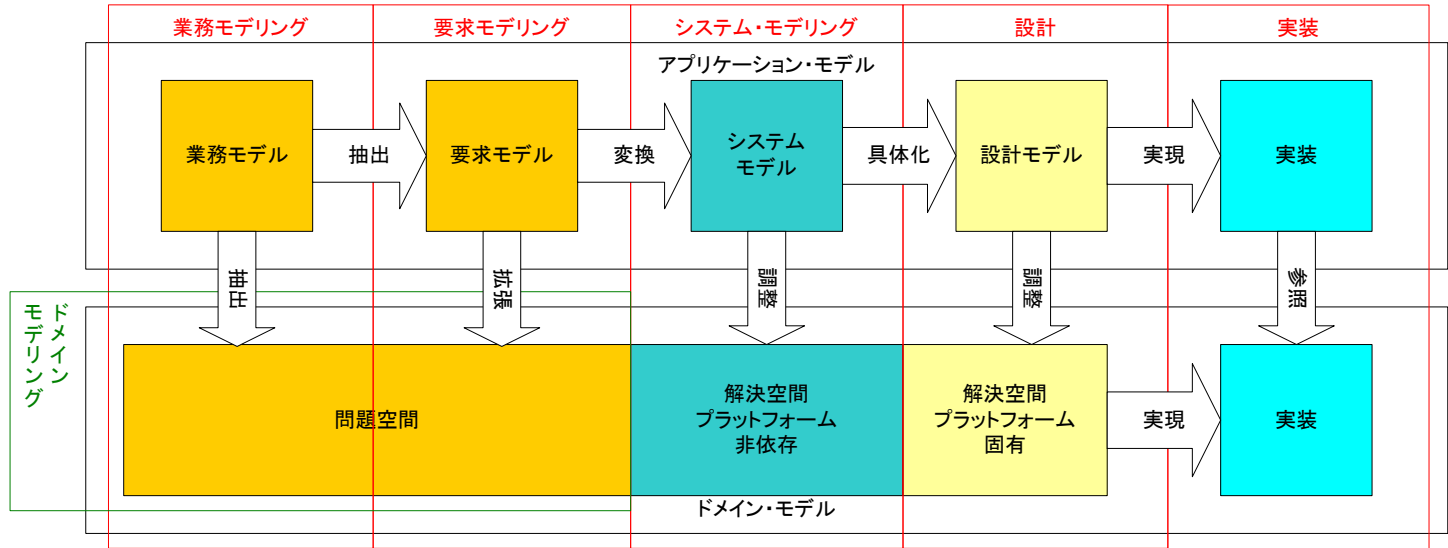
SimpleModeling

ドメイン・モデルをハブとした連携

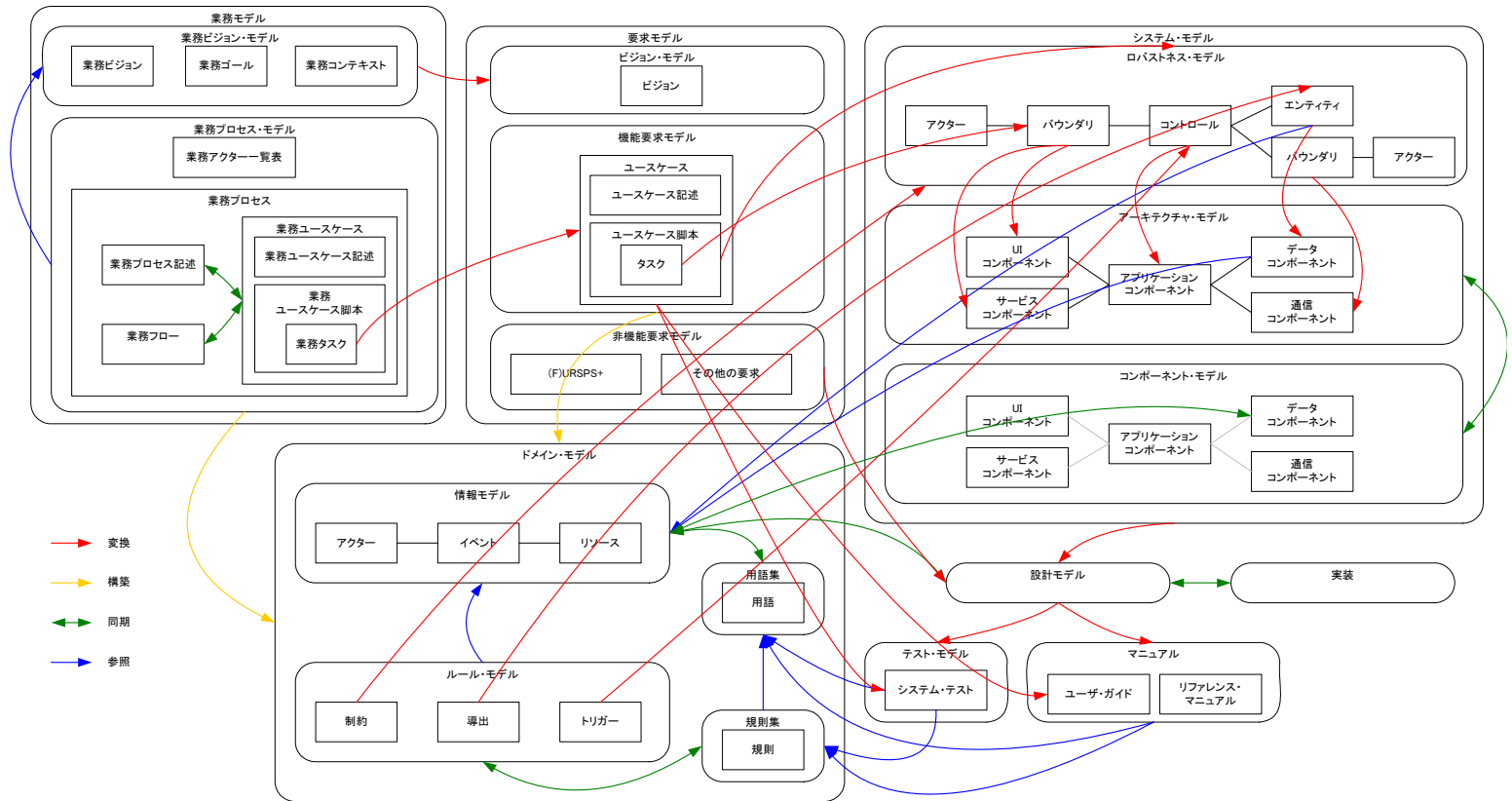


SimpleModeling

モデル変換の流れ

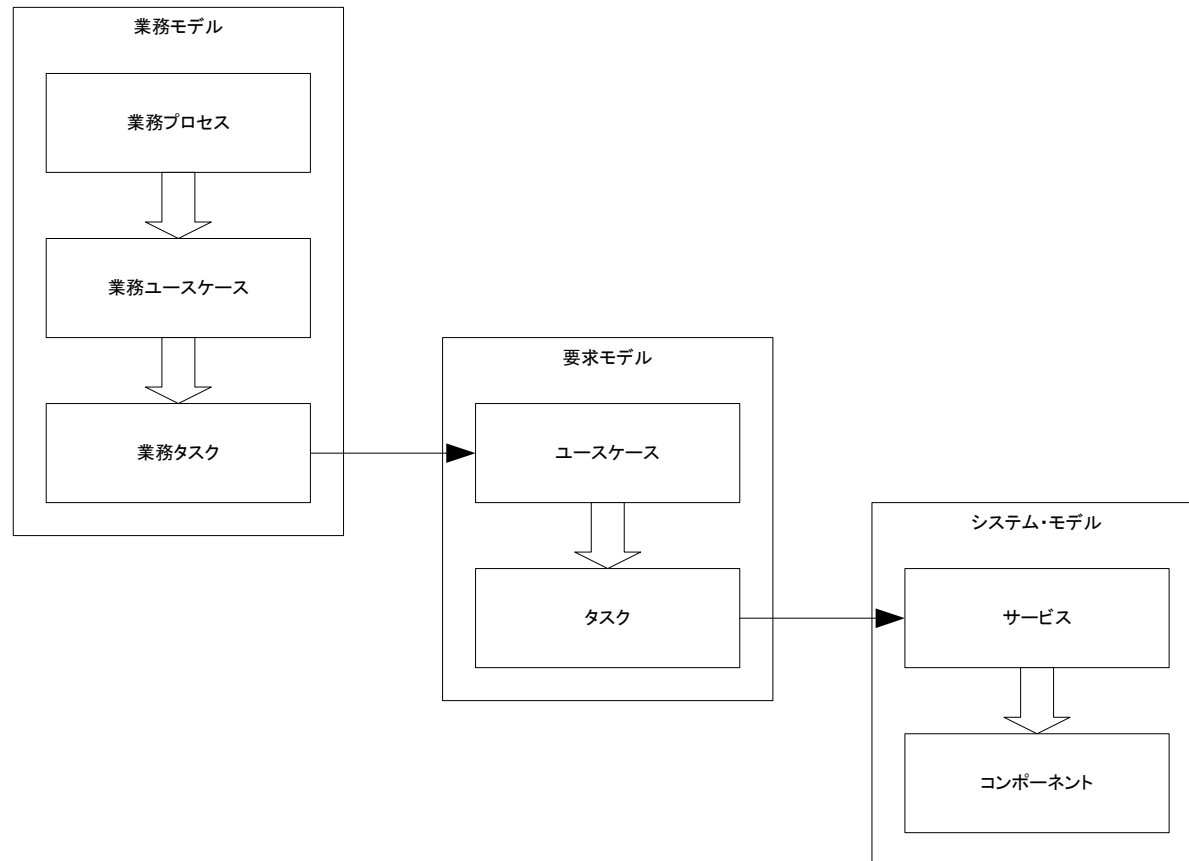


モデル体系

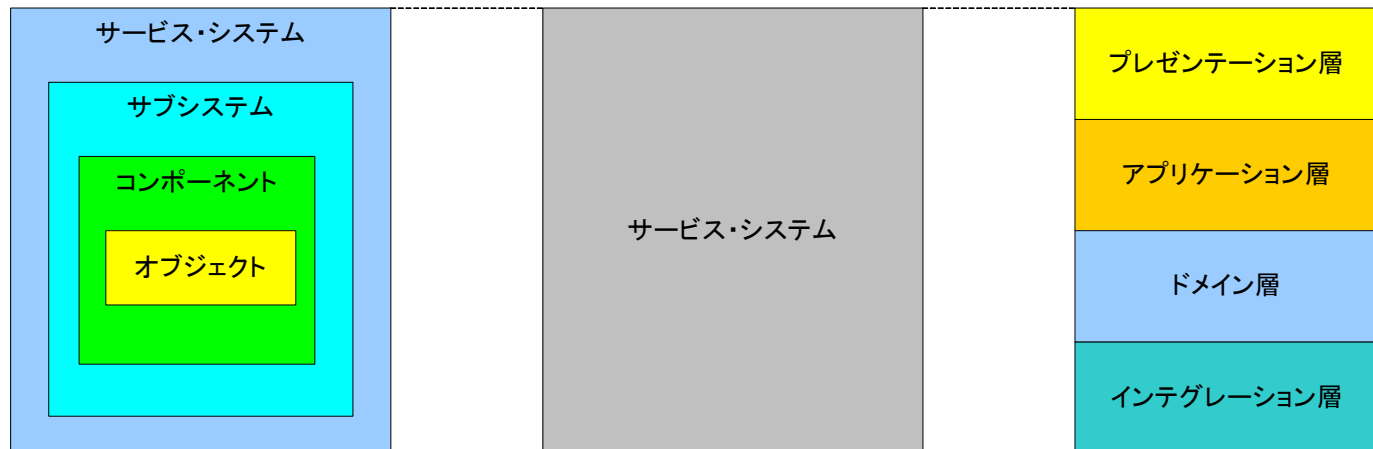


SimpleModeling

軸となるモデル連携



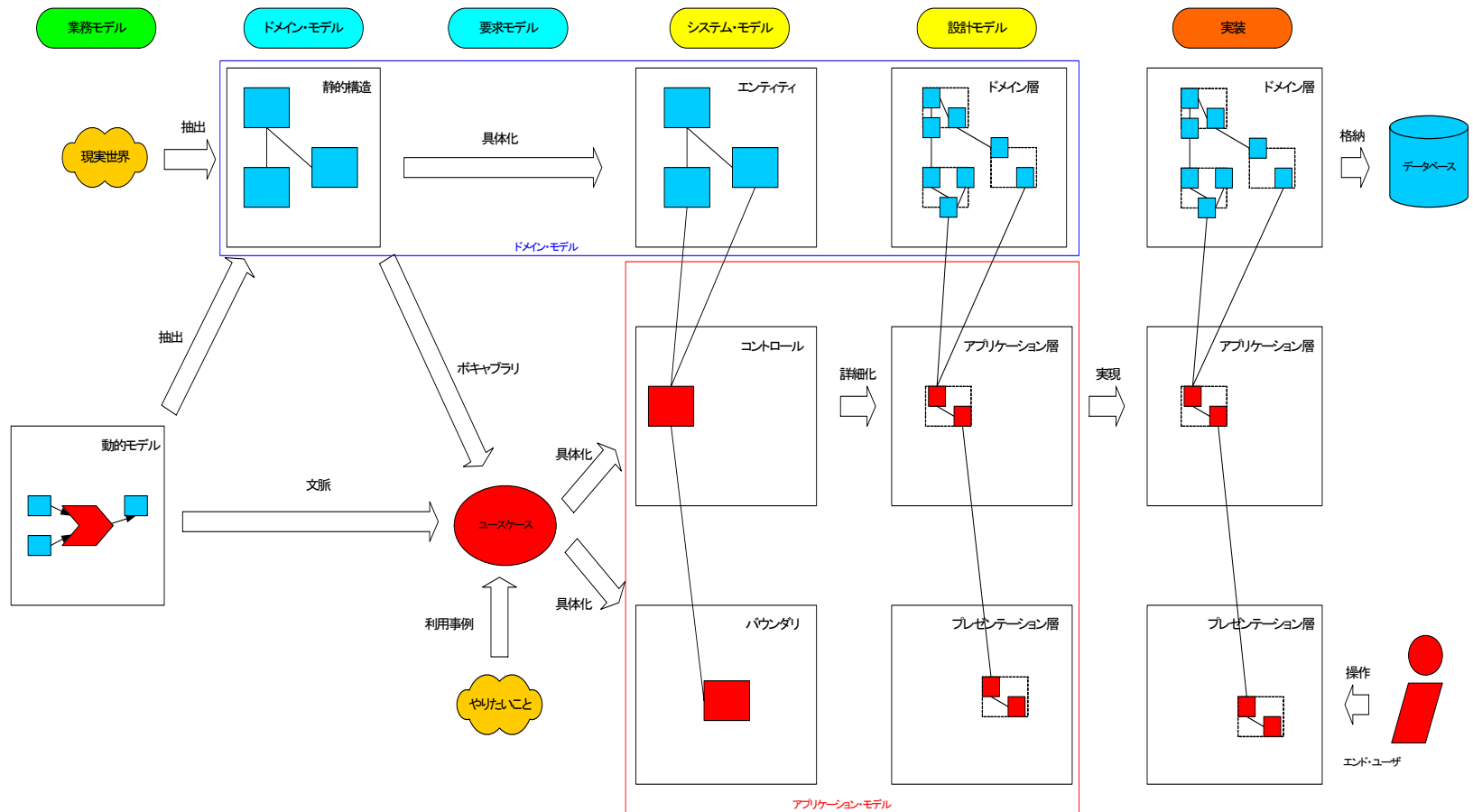
サービス・システムの構成



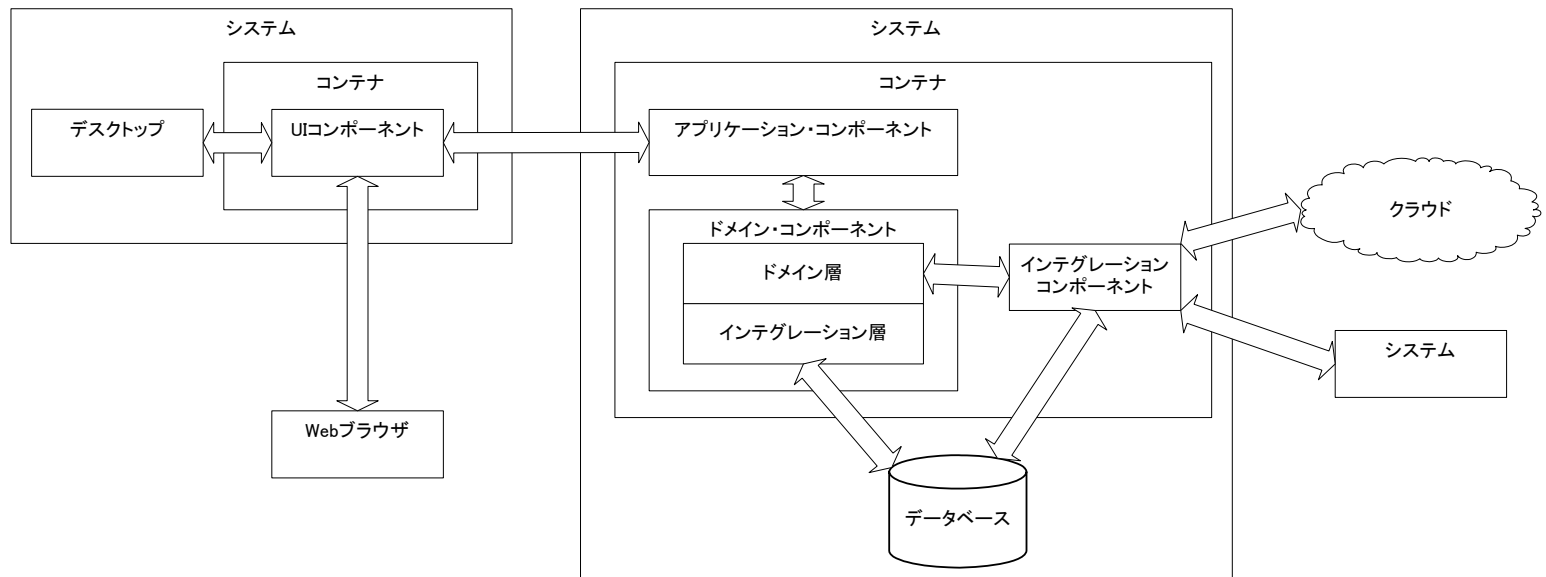
論理モデル視点

アーキテクチャ層ビュー

モデルとアーキテクチャ



システム・アーキテクチャ





解決空間モデル

○ システム・モデル

- PIM(Platform Independent Model)
- プラットフォーム独立の抽象度の高いモデル

○ 設計モデル

- PSM(Platform Specific Model)
- Java/JEEによる実装のためのモデル

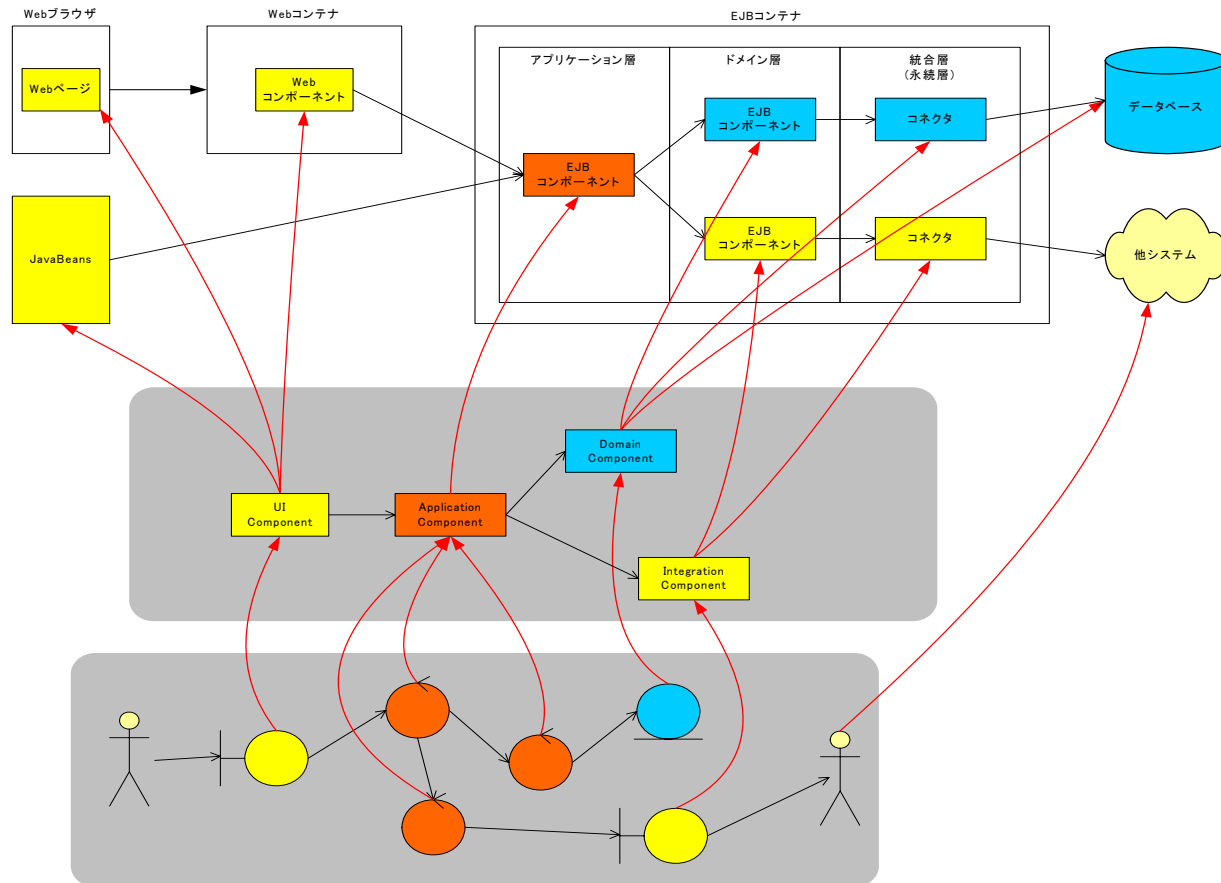


SimpleModelingのPIMモデル

- PIM(Platform Independent Model)
- ドメイン・モデル
 - 情報モデル
 - ルール・モデル
- システム・モデル
 - システム・アーキテクチャ・モデル
 - コンポーネント・モデル
 - モジュール・モデル

コンポーネントとJavaEE

クライアント	Web	EJB		EIS
クライアント	プレゼンテーション	ビジネス	インテグレーション	リソース



ドメイン・モデル

- ドメイン・コンポーネントとして実現
 - 外部仕様は通常のコンポーネント
- エンティティやルールはドメイン・コンポーネント内のオブジェクトとして実現
- 公開方法
 - オペレーション経由で間接的に公開
 - オブジェクトを直接公開
- 永続化の実現場所
 - ドメイン・コンポーネント内で実現
 - インテグレーション・コンポーネントを使用
- 永続化の実現方法
 - プログラミング
 - O/Rマッピング

システム・モデル

- システム・アーキテクチャ・モデル
 - コンポーネントを組み立ててシステムを構築する。
- コンポーネント・モデル
 - UIコンポーネント
 - User Interfaceを実現
 - サービス・コンポーネント
 - 他システムに公開するサービスを実現
 - アプリケーション・コンポーネント
 - アプリケーション・ロジックを実現
 - ドメイン・コンポーネント
 - ドメイン・モデルを実現(情報モデル、ルール・モデル)
 - インテグレーション・コンポーネント
 - システム・リソース(データベースなど)へのアクセス
 - 他システム、サービスとの通信を実現
- モジュール・モデル
 - 配備の単位、流通の単位、開発の単位でコンポーネントを束ねる

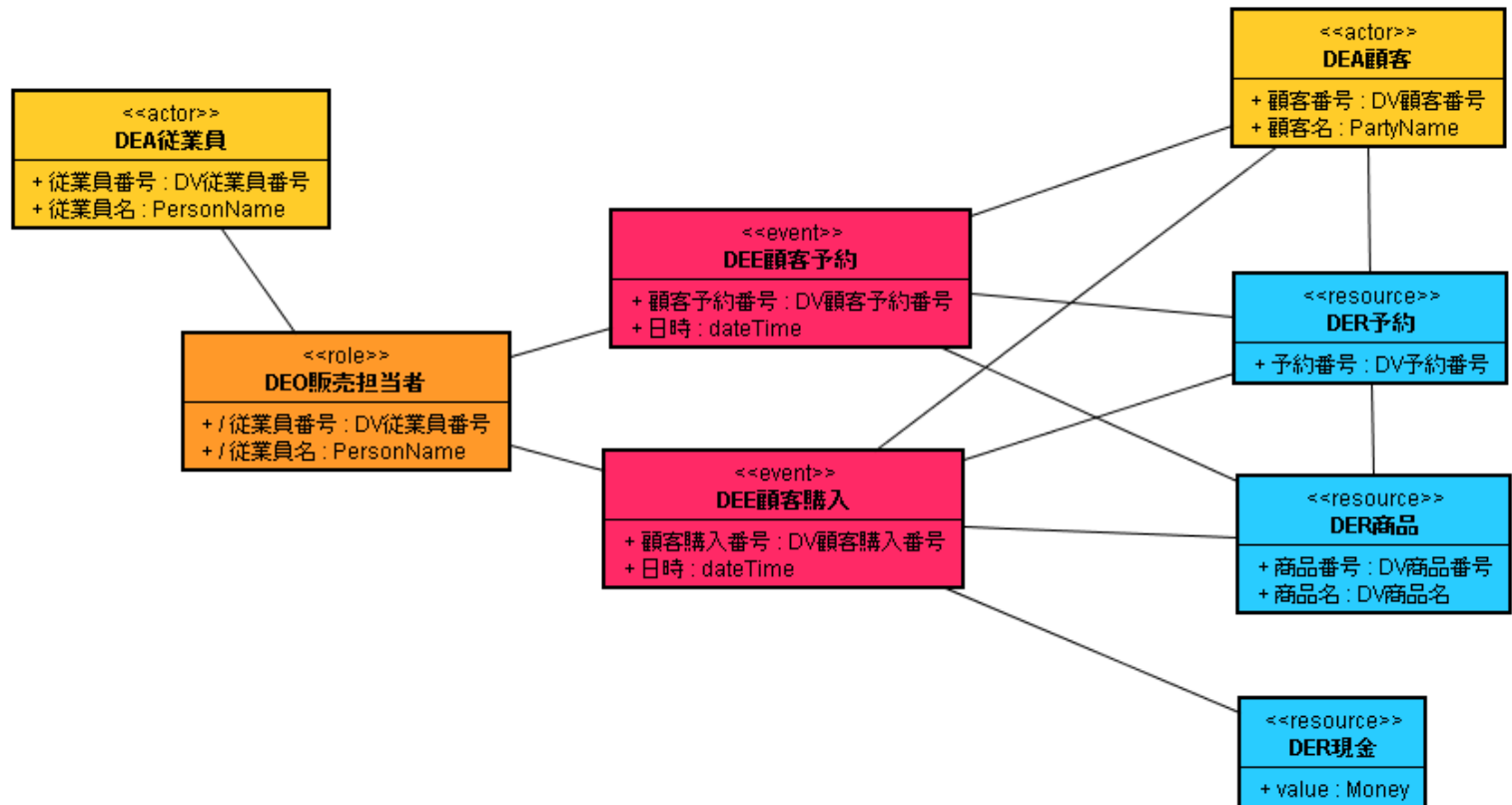
サービスとコンポーネント

- サービス
 - 再利用可能な部品
 - 利用者の目標を解決するための機能とインタフェース
 - プログラム呼出し以外の方法での利用が主
 - UI、RPC(SOAP、IIOPなど)、REST
 - 遠隔呼出しに適した粒度と信頼性
 - 大きな粒度
 - 低信頼性を想定
 - RESTが重要な理由の一つは、UIとAPIの両方を同時に提供しているから
- コンポーネント
 - 再利用可能な部品
 - 提供者が提供できる機能とインタフェース
 - プログラム呼出しでの利用が主
 - 静的な結合(少なくともプログラム起動時)
 - 遠隔呼出しには必ずしも適さない粒度と信頼性
 - 小さな粒度
 - 高信頼性を想定

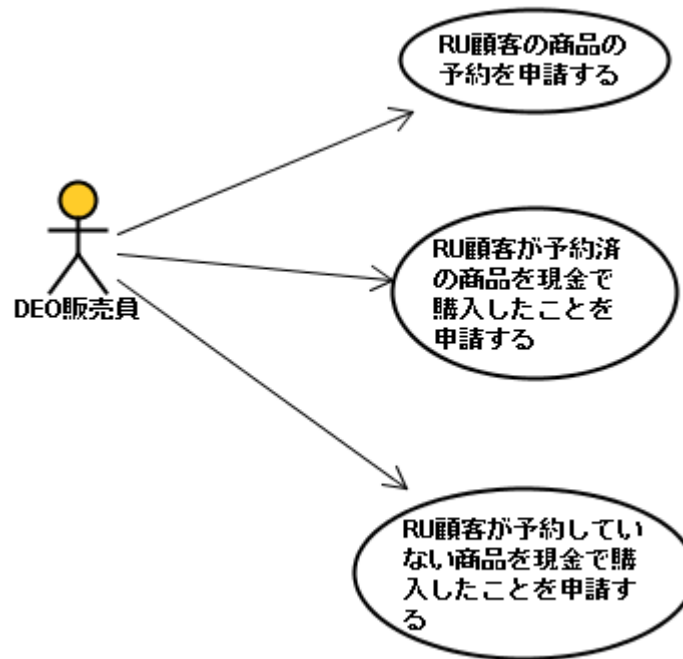


モデル・サンプル

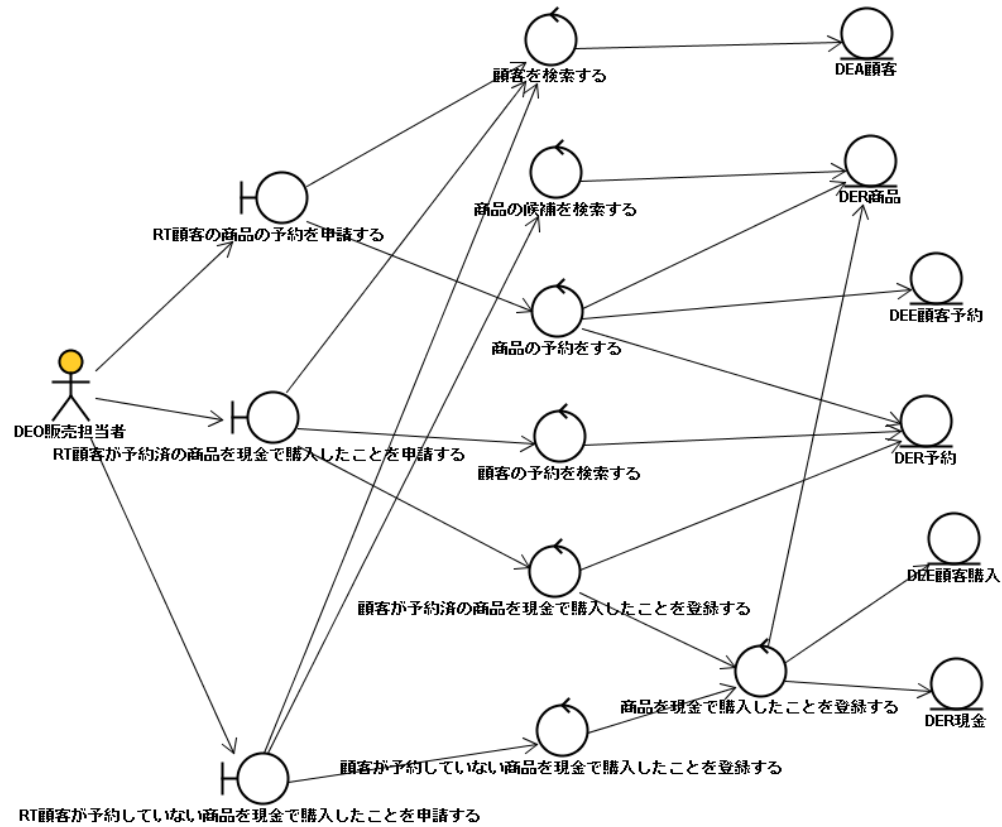
ドメイン情報図



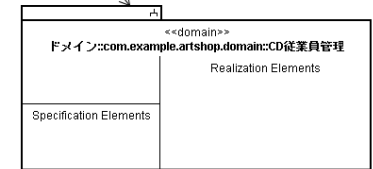
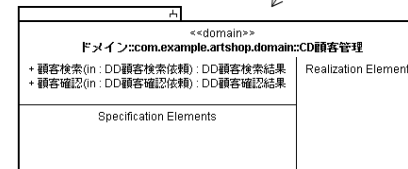
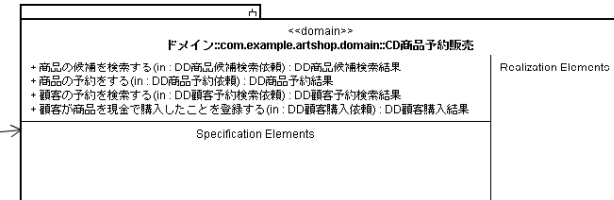
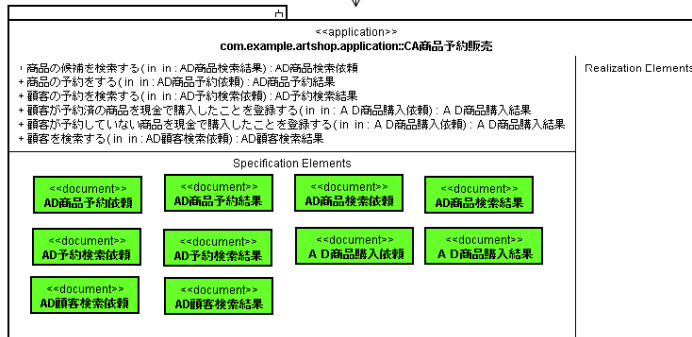
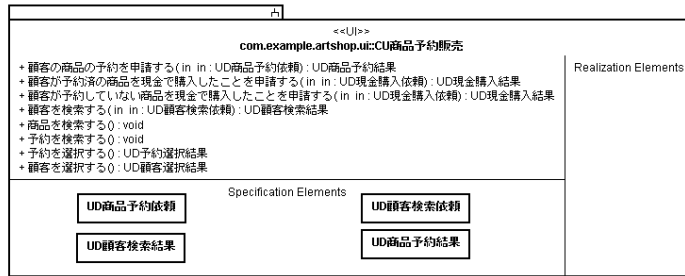
ユースケース図



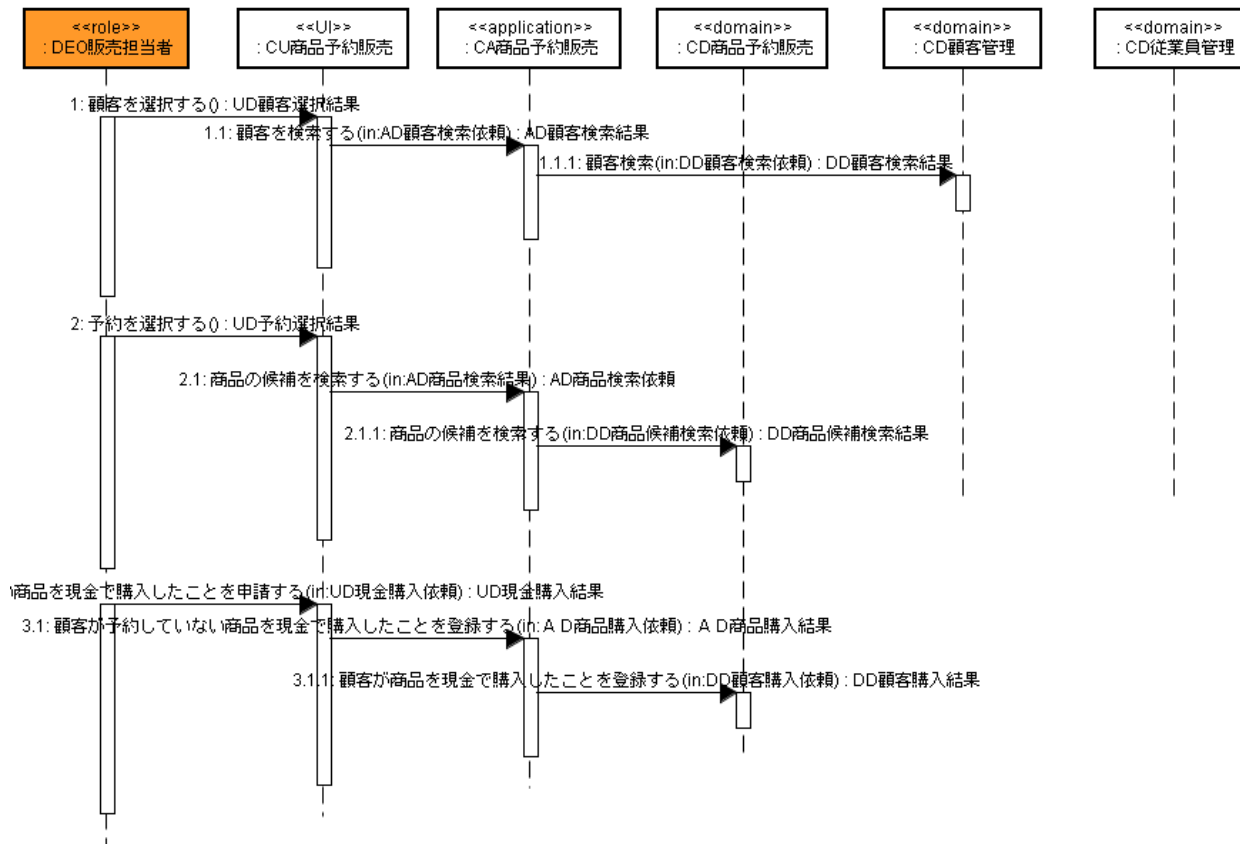
ロバストネス図



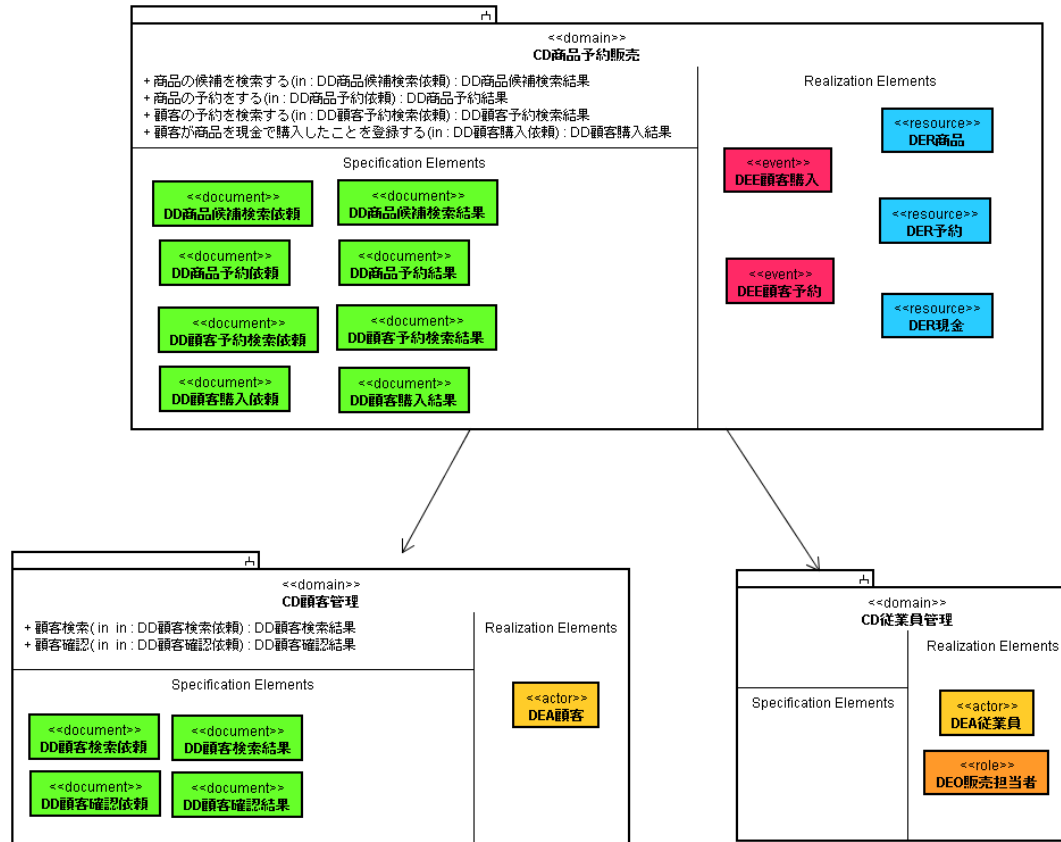
システム・アーキテクチャ図



ユースケース・シーケンス図



ドメイン・コンポーネント図





Javaプログラミング



Javaプログラミングでの考え方

- PIMモデルをそのまま入力とする
 - コンポーネントの仕様が定まっていれば十分
 - Javaはオブジェクト指向言語として十分な機能を持っているので、いわゆる”詳細設計”は不要
- 設計が必要な場合
 - GUIの画面構成やデータベースの物理モデルなど、Javaプログラミング以外のもの



Javaプログラミング

- ドメイン・モデル
 - ドメイン・コンポーネントとして実現
- コンポーネント・モデル
 - コンポーネント
 - モジュール
- システム・アーキテクチャ・モデル
 - コンポーネントの組立て



Javaプログラミング

システム・アーキテクチャ・モデル

- コンポーネントの組立てとして実現
- JEE
 - 配備ディスクリプタ
- DIコンテナ
 - 定義ファイル、命名規約

Javaプログラミング

コンポーネント？モジュール？

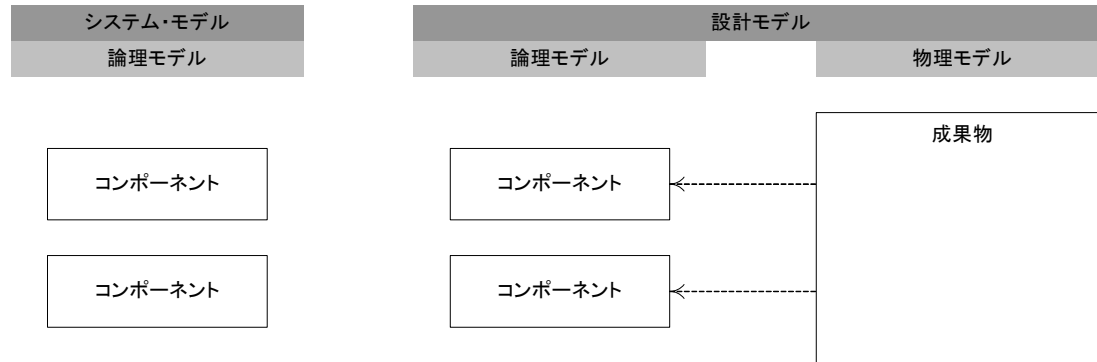
- JavaBeans
 - JAR (Java Archive)
 - Java classファイル
 - META-INF/MANIFEST.MF
- Enterprise JavaBeans
 - EJB-JAR
 - Java classファイル
 - META-INF/ejb-jar.xml
 - WAR (Web Archive)
 - Java classファイル, HTML, JSP
 - WEB-INF/web.xml
 - RAR (Resource Archive)
 - Java classファイル
 - META-INF/ra.xml
 - Client-JAR
 - Java classファイル
 - META-INF/client-jar.xml

Javaプログラミング

コンポーネントとモジュール

- コンポーネントとモジュールの考え方を整理しなければならない
 - コンポーネントの一般的な定義: 再利用可能なソフトウェア部品
- UMLでは...
 - UML 1.x: コンポーネントは配備の単位 (物理的なモデル要素)
 - UML 2.x: コンポーネントは配備の単位 (物理的なモデル要素) かつ再利用可能なソフトウェア部品 (論理的なモデル要素)
 - モジュールというモデル要素はない
- 今後は以下のように整理されてくると思う。
 - コンポーネント: 再利用可能なソフトウェア部品
 - モジュール: 配備の単位

コンポーネントとモジュール





Javaプログラミング モジュール

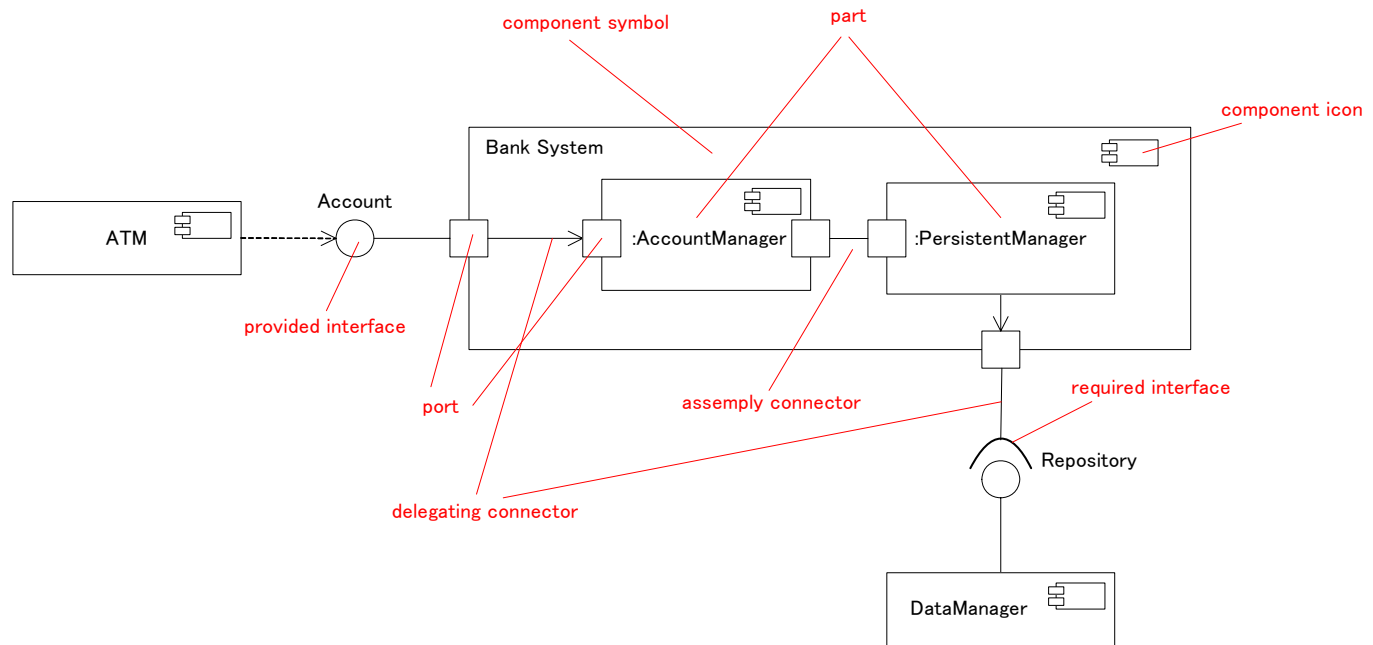
- JavaBeans
 - JAR (Java Archive)
- Enterprise JavaBeans
 - EJB-JAR
 - WAR
 - RAR
 - Client-JAR
- Maven2
 - POM(Project Object Model)
- Java7
 - JAM(Java Application Modules)
 - JSR 277:Java Module System
 - JSR 294: SuperPackage



Javaプログラミング コンポーネント

- 「論理的なソフトウェア部品」
 - 物理的な側面の表現はモジュールに移動
- コンポーネントを集めて配備の単位であるモジュールを作成する

UMLコンポーネント図



Javaプログラミング コンポーネントの実現

- パート(part)
 - Javaクラス
- 提供インタフェース(provided interface)
 - Javaインタフェース
 - UI(UIコンポーネントの場合)
 - Web(REST, SOAP)
- 必要インタフェース(required interface)
 - Javaインタフェース
 - Web(REST, SOAP)
- ポート(port)
 - メソッドの集まり
 - UI部品(画面など)の集まり(UIコンポーネントの場合)
 - WSDL/Port
- 委譲コネクタ(delegating connector)
 - Javaインタフェース&メソッド
 - ツールによる自動生成
- 組立てコネクタ(assembly connector)
 - インスタンス変数
 - プログラム(糊コード)またはDIコンテナで設定



Javaプログラミング コンポーネントの種類(1)

- UIコンポーネント
 - User Interfaceを実現
 - HTML, JSP, JSF
 - Swing
- サービス・コンポーネント
 - 他システムに公開するサービスを実現
 - JAX-WS
 - RMI
- アプリケーション・コンポーネント
 - アプリケーション・ロジックを実現
 - Javaオブジェクト

Javaプログラミング コンポーネントの種類(2)

- ドメイン・コンポーネント
 - ドメイン・モデルを実現(情報モデル、ルール・モデル)
 - JDBC
 - JPA, Hibernate
- インテグレーション・コンポーネント
 - システム・リソース(データベースなど)へのアクセス
 - 他システム、サービスとの通信を実現
 - JDBC
 - JPA, Hibernate
 - JMS
 - JCA
 - JAX-WS
 - RMI

Javaプログラミング ドメイン・モデル

- ドメイン・コンポーネントとして実現
 - 外部仕様は通常のコンポーネント
- エンティティやルールはドメイン・コンポーネント内のJavaオブジェクトとして実現
- 公開方法
 - メソッド経由で間接的に公開
 - Javaオブジェクトを直接公開
- 永続化の実現場所
 - ドメイン・コンポーネント内で実現
 - インテグレーション・コンポーネントを使用
- 永続化の実現方法
 - プログラミング
 - JDBC
 - O/Rマッピング
 - JPA, Hibernate

まとめ

- どこまでをモデルで作るのか?
 - ドメイン・モデル
 - システム・アーキテクチャ・モデル
 - コンポーネント外部仕様
- どこからJavaで作るのか?
 - ドメイン・モデルの実現
 - コンポーネントの実現
 - システムの構築(コンポーネントの組立て)
- モデリングとJavaの接点
 - ドメイン・モデル
 - コンポーネント
 - システム・アーキテクチャ・モデル