

モデル駆動開発入門

8

masaki@metabolics.co.jp

2006.09.04

まとめ

モデルから知識へ

- モデル駆動開発のさまざまな実例を見てきた
 - a) Django, Ruby on Rails, TurboGears, ...
 - b) EMF
 - c) MDA
 - d) xUML
 - e) Wf/BPM
 - f) オントロジ

モデル駆動開発の比較属性

- 変換ステップ数
- 変換ステップ
- 入力
- 出力
- 同期性
- 他の方法との混合
- 類似製品
- 適合分野
- 難易度
- ツール規模
- 対象モデル
- 安定度
- ライセンス
- 費用
- 教育コスト
- 周知度
- 将来性
- 実績
- プログラミング言語
- 標準への準拠度

変換ステップ数

django	1
EMF	1 (実装), 3 (Editor), 12 (GMF)
MDA	任意
xUML	1
Wf/BPM	?
Protégé	1?

変換ステップ

django	実行時
EMF	JET (テンプレート)
MDA	MOF/QVT
xUML	...
Wf/BPM	?
Protégé	...

入力

django	ER
EMF	UML, XMLSchema, アノテーション付きJava
MDA	PIM
xUML	UML
Wf/BPM	ワークフロー
Protégé	オントロジ

出力

django	マスタ画面
EMF	Ecore (MOF), Javaコード, エディタ
MDA	任意
xUML	実行可能コード
Wf/BPM	?
Protégé	ファクト入力ツール

同期性

django	同期 (単方向)
EMF	同期 (Ecoreと双方向)
MDA	同期 (基本的に双方向)
xUML	同期 (単方向)
Wf/BPM	?
Protégé	同期 (単方向)

他の方法との混合

django	難しい
EMF	可能
MDA	? (使い方次第)
xUML	難しい
Wf/BPM	可能 (必要)
Protégé	可能 (必要)

類似製品

django	R on Rなど多数
EMF	MDAの実装に近い
MDA	Optimal/Jなどいくつか
xUML	BridgePointなどいくつか
Wf/BPM	多数
Protégé	あまりない

適合分野

django	Webアプリケーション
EMF	Eclipseプラグイン, 任意
MDA	任意 (今は業務系中心)
xUML	任意 (今はリアルタイム系中心)
Wf/BPM	業務系中心
Protégé	任意

難易度

django	◎
EMF	△
MDA	×
xUML	△
Wf/BPM	○
Protégé	△

ツール規模

django	小
EMF	大
MDA	大
xUML	大
Wf/BPM	小 ~ 中
Protégé	中 ~ 大

対象モデル

django	ER
EMF	静的構造
MDA	静的構造
xUML	ステートチャート
Wf/BPM	ワークフロー/BPM
Protégé	OWL (知識表現), ルール

安定度

django	大
EMF	中
MDA	まだ開発中
xUML	大
Wf/BPM	中 ~ 大
Protégé	中

ライセンス

django	BSD
EMF	CPL (Eclipse)
MDA	標準としてはオープン
xUML	商用
Wf/BPM	さまざま
Protégé	Mozilla

費用

django	¥0
EMF	¥0
MDA	ツールはおおむね高価
xUML	大変高価
Wf/BPM	さまざま
Protégé	¥0

教育コスト

django	低
EMF	中～高
MDA	高
xUML	高
Wf/BPM	中
Protégé	中～高

周知度

django	(ファミリーとして) 高
EMF	中 ~ 高
MDA	中
xUML	低 ~ 中
Wf/BPM	中
Protégé	低 ~ 中

将来性

django	中
EMF	高
MDA	高
xUML	中
Wf/BPM	中 ~ 高
Protégé	中 ~ 高

実績

django	多い
EMF	ある程度 (Eclipse plug-inが中心)
MDA	まだこれから
xUML	狭いが多い
Wf/BPM	今増えつつある
Protégé	まだこれから

プログラミング言語

django	Python (Ruby, Groovyなど)
EMF	Java, OCL, JET
MDA	OCL, Action Language?, MOF/QVT
xUML	C++がほとんど
Wf/BPM	Javaが多い
Protégé	SPARQL, SWRL

Model

django	Python DSL
EMF	UML, annotated Java, XSD
MDA	UML
xUML	UML
Wf/BPM	特になし
Protégé	OWL

View

django	Python DSL
EMF	Eclipse SWT/GEF
MDA	特になし
xUML	特になし
Wf/BPM	特になし
Protégé	フォーム

Control

django	django埋め込み
EMF	生成コードに埋め込み
MDA	カートリッジに埋め込み
xUML	状態機械
Wf/BPM	ペトリネット
Protégé	なし

Persistence

django	RDBMS
EMF	XMI
MDA	カートリッジ次第
xUML	アーキタイプ次第
Wf/BPM	特になし
Protégé	.owl, RDBMS

標準への準拠度

django	あまりない
EMF	ある程度有り
MDA	標準そのもの
xUML	今のところあまりない
Wf/BPM	標準自身がさまざま
Protégé	ある程度有り

モデル駆動開発のポイント

- モデル
 - プログラムよりも抽象度が高い
 - プラットホームへの依存度が低い
 - 生産性が高い
 - ユーザの世界に近い
 - 初期に検証, 妥当性確認が可能
 - 文書に較べて追跡可能
- ➡ 少数の優秀な人間で, 品質が高く, ユーザの満足度の高い製品を低いコストで作ることができるはず

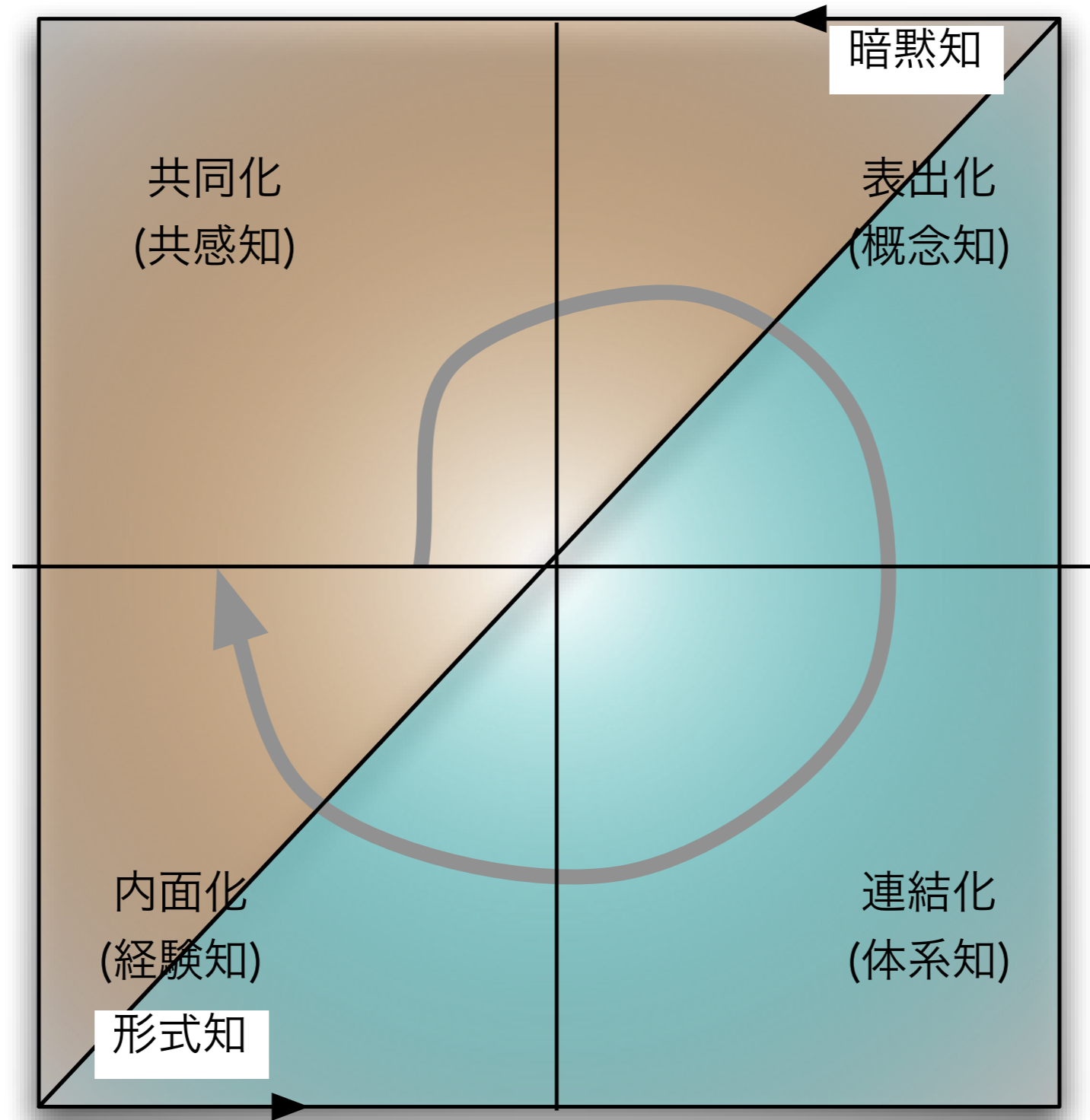
モデル駆動開発のポイント

- 変換ルール
 - 属人的/暗黙のノウハウの形式化, 資産化が可能
 - 知識を変換ルール (同時にモデル) としてパッケージ化できる
 - 正しい変換ルールの記述は難しい
 - 商品としての変換ルールもあり得る
- ➡ 優秀なエンジニアがいれば, 長期的な知識資産, プロセス改善の蓄積が可能
- ➡ 初期のコンパイラと同じように, 変換ルール (あるいはその作成手法) 自身が高い価値を持つ

モデル駆動開発の落とし穴

- 技術としてはまだ成熟していない
- 社会的な影響が大きい
- 重量級のモデル駆動開発は, 利点を打ち消してしまう可能性がある
- モデルにだけ依存してしまうと「痩せ」てしまう可能性がある
- 現時点では複雑すぎる約束事を決めておいて, それを少しでも簡単にする手段 (それを使うにもまた別の約束事が必要!) となっているような側面もある

- モデルとは実は知識の一表現形式
 - 現実世界についての
 - 顧客についての
 - 技術についての
 - 開発手法についての
- 文書やプログラムも知識の一表現形式だが
 - 文書やプログラムよりも知識としての質が高い
 - 暗黙知を形式知に変換する仕掛け
 - モデル駆動開発を続けることによって形式知がまた暗黙知に戻る (モデル駆動開発のやり方に関する暗黙知が蓄積される)
 - 「実際に動く」ことによって,従来の純粋な形式的手法よりも理解しやすく,ユーザ/開発者にとって敷居が低い



- ソフトウェア開発が本当の意味で「知識」を取り扱う, 「知識」に価値を与えることのできる産業になることができるかどうかの試金石になるかもしれない