

モデル・ハーネス

モデルを開発と実行の基盤に

ドメイン知識に近い表現を
AI開発に活かす

ソフトウェアをモデルで記述

構造・規則・状態機械



処理系で検査・実行

AIが生成する変更にも制約

シリーズにおける位置付け

第2～4回は、モデル・ハーネスを共通原理・静的構造・動的振る舞いに分けて扱う



第5回アーキテクチャ、第6回プログラミング、第7回ランタイム。第8～10回で知識・観測・検証へ。

用語集で結ぶモデルの基盤

前回シリーズのモデル駆動開発を、AI開発の基盤として活用する

文芸モデル

要求・文脈・判断

オブジェクト・モデル

形式的な構造と振る舞い

知識モデル

意味関係・根拠・出典

用語集が名称・意味・識別子を管理し、モデル間の対応を支える。

ドメイン知識に近いソフトウェアの記述

モデルによって、業務の概念とプログラムの表現の隔たりを小さくする

CMLで記述

概念・関係・規則

Textusで直接実行

実行可能モデルを動かす

AIによる実装補完

モデルだけでは不足する部分

Cozyがモデルを変換する。補完実装にも、モデルの構造・規則・振る舞いを反映する。

状態機械による振る舞いの記述

状態と許される遷移を明示し、人間・AI・処理系が同じ構造を扱う



失敗時の遷移や在庫予約の扱いも、業務規則として記述する。

モデル・ハーネスを支える仕組み

レビュー・静的検査・直接実行を組み合わせ、生成・変更に制約を働かせる

意味と対応のレビュー

用語集・textus-cbd-support

ビルド時の静的検査

文法・参照・制約を確認

Textusによる直接実行

モデルの規則で動作を規定

実行可能モデルにもユースケースにもレビューが作用する。意味の選択と採否は人間が判断する。

モデル・アップ・ダウン

モデルを実現へ具体化し、レビューでは目的に応じて抽象化する



モデルアップの例：状態機械をフロー図へ。ユースケース・モデルをアクター・ゴール一覧へ。

要求から実現・検証への対応

要求とモデルの対応を残し、形式的な整合性と業務の目的への適合を確かめる



ドメインの共通規則とアプリケーション固有の流れを分け、変更先と影響範囲を判断する。

モデルと実行結果の照合

明確性・一貫性・妥当性・正確性を、構造検査と業務上の判断で確認する



性能・信頼性・セキュリティは、要求から実行までを横断して確認する。

モデルでソフトウェアを記述する

ドメイン知識に近い表現を、開発と実行の基盤にする

構造と振る舞いをモデルにする

静的モデル：概念・関係・規則

動的モデル：状態・イベント・遷移

処理系がAIの変更を制約する

モデルのレビュー・静的検査

実行可能モデルをTextusで直接実行

要求との対応で目的を確かめる

要求・ユースケース・実現先を結ぶ

規則と検証例を実行結果に照合する

モデル・アップ・ダウン

ダウン：モデルから実現へ具体化

アップ：レビュー用にモデルを抽象化

例：状態機械をフロー図で表示

次回：概念・関係・規則による静的モデル

モデル・アップ・ダウンはSimpleModeling独自の概念