B-001

要求分析モデルの妥当性検証のための インタラクティブなテストデータ定義支援ツール

Interactive Test Data Development Tool for Validating Requirements Analysis Model

式見 遼† 小形 真平‡ 松浦 佐江子†
Ryo Shikimi Shinpei Ogata Saeko Matsuura

1．はじめに

我々は Web アプリケーションを対象に，業務のワークフロー分析からシステムが提供するユースケースを抽出し，UML モデルからツールを用いてプロトタイプ(HTML で定義された Web ページ)を自動生成しながら，要求分析モデルを定義する手法を提案している[1]．この手法の特徴は，システムが提供するサービスへの要求を，振舞い・データの構造と具体値といった観点から，作成したモデルを最終的なプロダクトを模した具体例を含むプロトタイプで顧客の要求を可視化することにより，顧客および開発者が最終プロダクトに近い形で要求の満足度を確認できることにある．図 1に示すように，要求分析モデルでは，システムの振る舞いを記述したシステムパーティション，アクターのシステム利用手順を記述したアクターパーティション，システム・アクター間のデータの受け渡しと画面表示を記述したインタラクションパーティションの 3 種類のパーティションからなるアクティビティ図を定義する．

しかし，既存のプロトタイプ生成手法では，システムパーティションの記述が不十分で，システムの実現可能性が不明なままであっても画面イメージが生成することができた．つまり，アクター視点でシステムを利用する流れを表す画面イメージが生成されるが，プロトタイプからはシステムが行う処理を確認することが難しいため，システムへの入力データから期待する出力データをつくれるように要求分析モデルに記述したシステムの振舞いが記述されているという意味で，要求分析モデルの妥当性が検証されていることの保証はできない．さらに，顧客自体がプロトタイプ上で確認できるはずの要求分析モデルに記述されたシステム要件の誤りや過不足を見逃す可能性も考えられる．

そこで，この問題を解決するために我々は，テストを設計する観点から要求分析モデルで記述されたシステムの具体データを定義し，定義したデータを要求分析モデルに当てはめることで要求分析モデルの実現可能性を検証するとともに，顧客のシステムの動作イメージの理解を向上することを目的とした，テストデータ定義・テストケース生成を支援するツールを開発した[2]．これは，W 字モデル開発でも提唱されている，要求分析からテスト設計を行うことにより要求分析の妥当性を早期に検証するという考えに則ったものである．しかし，要求分析段階でシステムの実現可能性が十分に検証出来ていないと，実装段階で要求分析の通りに実装することができなくなる．こうして実装したシステムには要求分析との間にずれが存在するため，要求分析段階で作成したテストケースが使用できなくなるという問題がある．本研究では，テストを設計する観点から要求分析モデルにおけるシステムパーティションの記述を十分に精査し，品質を向上させることで，要求分析モデルで表されたシステムの実現可能性を検証することを目的とする．

† 芝浦工業大学 大学院理工学研究科 電気電子情報工学専攻

‡ 芝浦工業大学 大学院工学研究科 機能制御システム専攻

2．テストケース生成ツールの課題と解決方法

[2]では，ユースケース開始時にシステムが保持するデータと外部から入力されるデータと，そのデータを変化させるシステムの処理を要求分析モデルのパスに沿って定義することで，システムで行われる処理をシミュレートし，テストケースを生成していた．この手法は，一度システムの処理を定義すれば，ユースケース開始時にシステムが保持するデータと外部から入力されるデータを変えるだけで複数のテストケースが生成できるという点で優れているが，テストケースを生成の核であるシステムの処理の定義自体に誤りが無い事を保証できていなかった．

我々は，システムの処理の定義に誤りが無いことを保証するには，システム全体のデータの変化の定義と契約による，システム全体のデータと処理の定義が必要であると考えている．これは，データでシステムの処理を具体的に定義し，契約でシステムの処理を仕様の観点から定義することで，システムの処理の定義の誤りを防ぐ仕組みである．そして，本稿ではシステム全体のデータ変化の定義を行う手法に取り組んだ．システム全体のデータの変化を定義することは，システムの処理中のシステム全体のデータ状態の変化を理解することにつながるため，要求分析モデルで記述されたシステムの実現可能性の検証としての効果も期待できる．

本研究におけるシステム全体のデータ変化の定義手法は，要求分析モデルで定義されたシステムのデータ変化を要求分析モデルのパスと対比しながらインタラクティブに定義する手法である．

なお今回の手法で定義するデータの型は，将来的にデータの定義に契約による制約条件を付加することを考慮して，オブジェクトの制約条件の定義に仕様として対応している OCL(Object Constraint Language)の型に統一した．

3．提案手法

3.1 データ定義プロセス

課題を解決するために，次のプロセスでテストデータの定義を行う

1. テストデータを定義するパスを特定する
　アクティビティ図の中からテストデータを設計したいフローを選択する．

2. 1.で選んだパスの各ノードにおけるシステム全体のオブジェクトの状態を入力する
　選択したパス内のすべてのノードに対してそのノード時点におけるシステム全体のデータの状態を入力して，システム全体のデータの変化を定義する．これにより，作成した要求分析モデルで使用できるオブジェクトとその構造が具体値で定義され，システムがデータに対して行う処理や分岐条件が定義された具体値で考えられるため理解しやすくなる．結果として，アクティビティ図内のノードの不足やクラスのフィールド項目の不足などが発見でき，要求分析モデルの妥当性確認ができる．また，オブジェクトの状態の入力はパスに対して任意の順番で行って良い．すなわち，必ずしも開始ノードから終了ノ

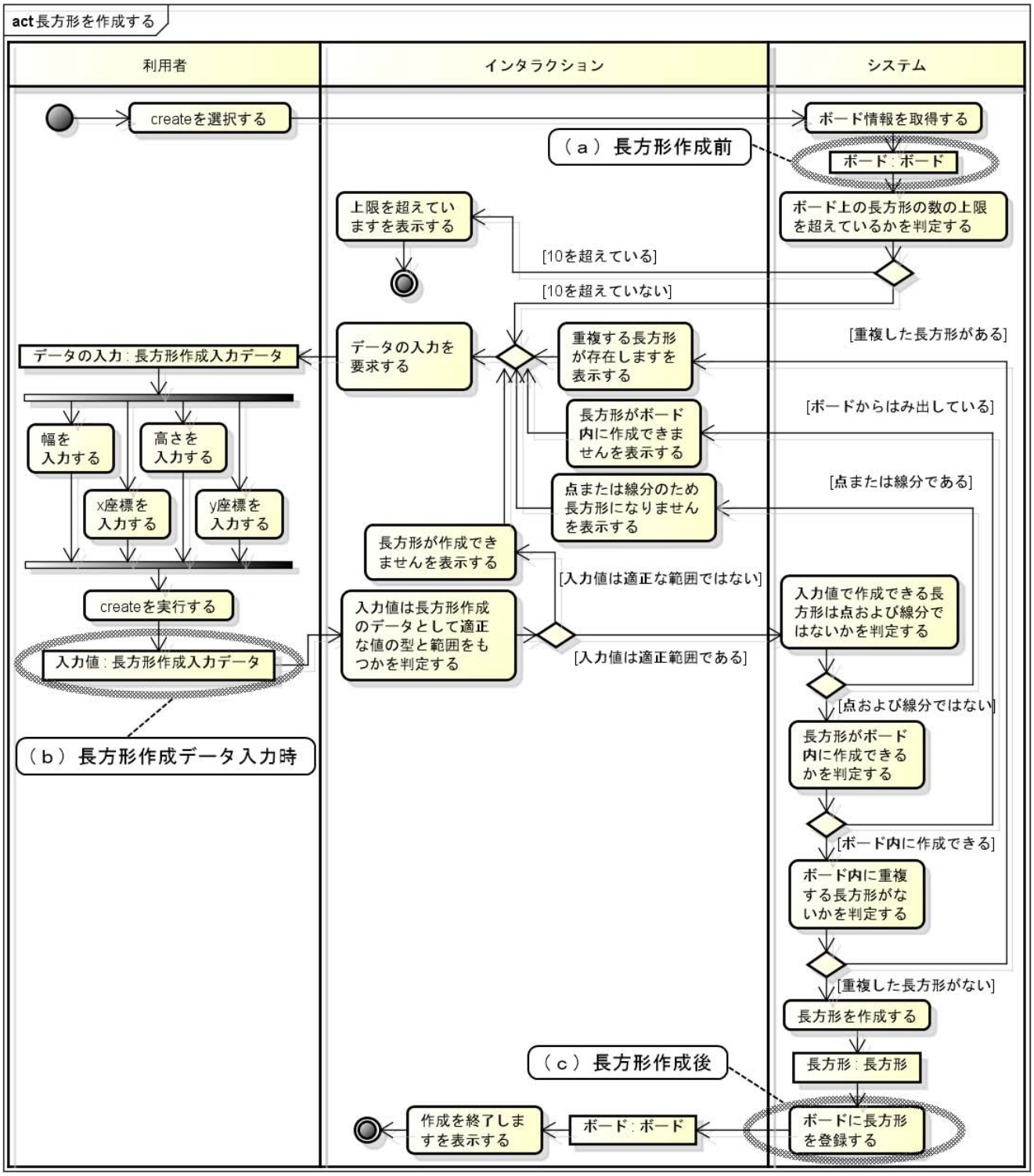

図 1「長方形を作成する」アクティビティ図

図 2 「長方形を作成する」クラス図

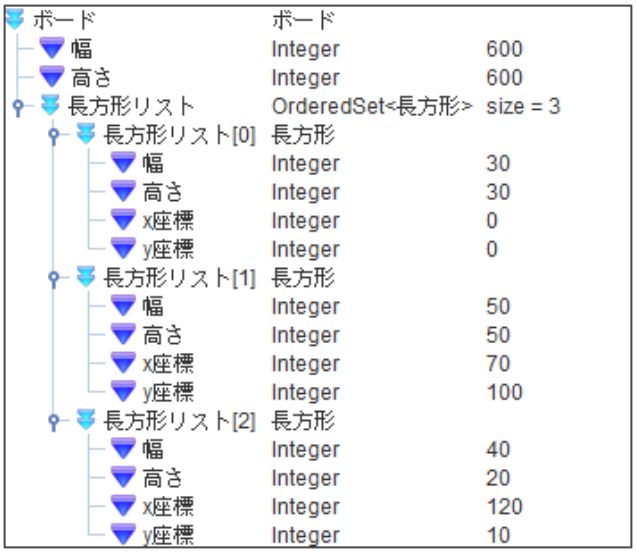

図 3 長方形作成後のシステム全体のデータ

図 4 長方形作成前のシステム全体のデータ

ードまで順番に入力する必要はない．例えば，長方形を 10 個まで同時に作成して表示するソフトウェア，長方形エディタで「長方形を作成する」というユースケースが図 1のアクティビティ図と図 2のクラス図で定義されている場合，システム全体のオブジェクトの状態を長方形が作成された（図 1-(c)）状態（図 3）から定義し，続いて，長方形作成前（図 1-(a)）の状態（図 4）を定義し，最後に作成する長方形の入力値（図 1-(b)）を定義するという順番で，システム全体のオブジェクトの状態変化を定義する事ができる．入力値に対するユースケースを正常に完了するための条件が多数あるときは，開始ノードから順番にオブジェクトの状態変化を考えるより

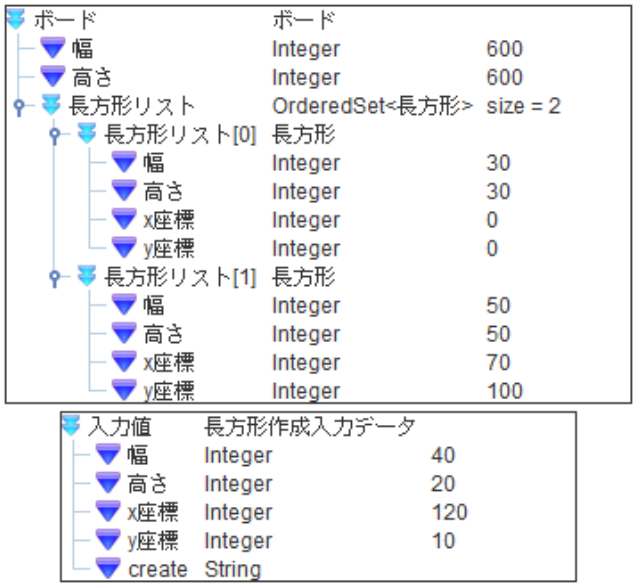

図5 長方形作成データ入力時のシステム全体のデータ

も，終了状態と合わせて考えたほうがオブジェクトの状態の定義が容易になる場合が多いため，このようにオブジェクト状態の入力をパスに対して任意の順番で行うことを可能にしている．

3. 作成したオブジェクトのデータのうち，テストに使用するオブジェクトのみを選んで，テストデータとして登録する

本稿では[2]の手法の課題点であったデータ定義方法を対象としており，テストケースの生成に関しては[2]の手法を用いる．そのため，テストケースを生成するためには本手法で定義したデータのうち，[2]の手法でテストケース生成のために必要となるデータを選択する必要がある．その後，生成したテストケースのオブジェクトの状態と，本稿の手法で定義したオブジェクトの状態を比較し，両者に差異がないことを確認することでデータ定義とシステムの処理の定義に誤りがないことを検証することができる．

以上が，システム全体のデータ変化の一貫性を保ちながらテストデータを定義する手法である．

3.2 利点

本稿で提案したデータ定義手法には，システムの動作理解とデータ定義作業負荷削減の観点から，次の利点がある．

- システムのパスのオブジェクトの状態の変化をテスト設計者の望む順番で定義できる

- データの遷移を定義することにより，オブジェクト間で行われる値の代入によるオブジェクトの状態の変化の入力を削減できる

  システム全体のデータ変化を追うと，あるオブジェクトのフィールドの値が，そのまま別のオブジェクトのフィールドに代入されることがある．これは，アクターパーティションでアクターが入力した値がシステムのエンティティに渡されたり，エンティティの値からインタラクションパーティションの表示データが生成されたりする際によく見られる．このような場合，テスト対象としているアクティビティ図のパスが変わるたびに，何度も同じ類のデータの変化を入力しなくてはならない．そこで，本手法では，オブジェクト間で行われる値の代入処理自体を定義することで，ツールがオブジェクト間の値の遷移によって定義されるデータの入力を自動で行い，この種の手間を省くことができる．

- コレクションなどのデータの集合をランダムに複数生成できる

  本手法では，コレクションで保持されているデータを一括で生成できる．生成されるオブジェクトの値は，擬似乱数で作成した値になる．例えば，先に挙げた長方形エディタの例では，複数個の長方形オブジェクトの一括生成などに用いることができる．生成されるオブジェクトは擬似乱数で値が定義されたものになるが，複数のモックデータが必要になる際に有力である．表 1 に，擬似乱数で生成できる値の指定の種類を示す．

表 1 擬似乱数によるオブジェクトの生成条件

型（OCL）	生成条件
Boolean	候補(true/false)からの選択
Integer	指定範囲から生成 or 候補からの選択
Real	指定範囲と指定有効桁数範囲から生成 or 候補からの選択
String	指定文字数範囲と指定使用可能文字候補から生成 or 候補からの選択

- アクティビティ図内のノードの不足やクラスのフィールド項目の不足などが発見できる

  アクティビティ図のノードと対比しながらシステム全体のデータをインタラクティブに定義する事によって，データ変化の観点からアクティビティ図におけるシステムのデータの変化を規定するようなアクションノードが不足していることが発見できる．また，同様に具体データをもってデータの変化を見ることで，システムの動作を実現するために必要なデータ項目がオブジェクトに不足していることがある場合に，これを発見できる．このように，パスの各時点におけるシステム全体のオブジェクトの状態をグラフィカルに定義することで，システムへの入力データから期待する出力データをつくれるように要求分析モデルに記述したシステムの振舞いが記述されているという点での要求分析モデルの妥当性を検証できる．

4. 課題と解決方法

今回の研究では，前回開発したツールの課題点の解消にデータ定義の観点から取り組んだが，それに伴って今後解決すべき新しい課題が発見できた．この節では，新たに見つかった課題とそれに対する今後の展望を述べる．

1. 擬似乱数で生成したオブジェクトが，満たすべき不変条件を満たさない場合がある

本手法では，コレクションのモックデータを擬似乱数によって複数個生成できる．例えば，(年 : Integer, 月: Integer, 日 : Integer)の情報を持つ日付クラスを複数個ランダムに生成する場合，(年 : [2011～2011], 月: [1～12], 日 : [1～31])と条件を指定することで，2011 年の日付を表すオブジェクトデータを作成することができるが，(年 = 2011,月 = 2, 日 = 31)などの本来存在しないはずの日付を生成する可能性がある．この問題に対して，今後は各クラスに OCL で不変条件を定義して，その条件を満たさないデータを生成しないようにすることが考えられる．また，テストの観点からは境界値分析を行うために，不変条件を満たさないデータが欲しい場合も考えられる．そのため，不変条件を満たすものと満たさないものの作り分けをできるようにする必要がある．

2. 生成したテストケースの妥当性確認が難しい

現在，ツールは Excel で閲覧できる形式でテストケースを生成するがこの形式では，テスト設計者や顧客による生成したテストケースの妥当性確認が困難である．そこで，生成したテストケースの妥当性確認を目的として，[1]の研究の web プロトタイプ上にテストケース内で記述された入出力データを表示して生成することが考えられる．これにより，プロトタイプ上でテストケースを確認でき，より直感的にテストケースの妥当性確認が行えるようになる．

5. まとめ

要求分析モデルの妥当性確認をするためにテストを設計することは重要であるが，要求分析モデルに対してテストデータを考えることは難しい．今回作成したツールの手法ではアクティビティ図に対してインタラクティブにオブジェクトの状態を定義することでアクティビティ図のフロー全体を通して一貫性のあるテストデータの定義を誤りなく容易に行えるようにした．その一方で，生成したデータの不整合やテストケースの妥当性確認の点で課題が残されており，今後は不変条件の契約の使用や web プロトタイプの使用によりこれらの問題を解決していく．

6. 参考文献

[1] 小形真平, 松浦佐江子：妥当性確認可能なモデルベース要求仕様からの統合テスト仕様自動生成，情報処理学会ソフトウェアエンジニアリング最前線 2009, pp.127-132,2009.

[2] 式見遼，小形真平, 松浦佐江子：UML 要求仕様からのカバレッジに基づく機能テストのテストケース生成，信学技法 vol.110，no 468，KBSE2010-603，pp.7-12，2011.