B-014

UML で記述された要求分析モデルからのプロトタイプ自動生成

An Automatic Generation of Prototypes from Requirements Analysis Models in UML

小形 真平† 松浦 佐江子‡
Shinpei Ogata Saeko Matsuura

1 はじめに

システム開発では，顧客がプロトタイプを通して要求分析モデルの妥当性を確認する方法がある[1]．このとき，顧客はシステムの提供する業務手順や入出力項目が要求を反映しているか確認し，満足しない箇所に対して開発者はモデルを修正する．しかし，モデルとプロトタイプの整合性の保証がなければ，モデル修正の妥当性が保証できない．そこで，要求分析モデルからのプロトタイプ自動生成により整合性を保証する．本研究の特徴として，

- UML2.0(Unified Modeling Language)を用いて要求分析モデルを定義する上で，顧客の要求レベルに応じたモデルの作成・修正を実現するため，顧客がプロトタイプで確認する事項を基準に数種のモデルを使い分けて定義する．
- 要求分析モデルに対し，プロトタイプ自動生成のために与える制約を少なくすることで，洗練したモデルをオブジェクト指向に則った後開発工程へ適用しやすくする．
- UML モデルの情報からプロトタイプを自動生成するツールを開発し，プロトタイプ開発の負担を減らす．

本稿ではプロトタイプ自動生成ツールを用いて，実際に稼動している本学の Web システムであるグループワーク支援システム(以下，GWSS)に適用し，有用性を議論する．

2 プロトタイプと要求分析モデルの定義

2.1 プロトタイプ

プロトタイプは HTML(Hyper Text Markup Language)で構成されるユーザインタフェースプロトタイプとし，顧客が表 1 の事項を確認するものとする．

表 1 プロトタイプにおける顧客の確認事項

確認事項	確認事項の詳細内容
業務手順	・サービス実行手順としての画面遷移 ・サービスに対する入出力項目の過不足
入出力項目	・入力項目の入力形式 ・グループ化すべき入出力項目

要求分析モデルをユースケース図，アクティビティ図，クラス図，オブジェクト図とし，本提案用に拡張する．各モデルと確認事項の関連は表 2 に示す．特に入出力項目では，表示のタイミング，入出力項目のグループ化，入出力項目の具体例から直感的理解の容易さを考慮する．モデルとの対応は，表示のタイミングをアクティビティ図に定義し，入出力項目のグループ化をクラス図へ定義し，入出力項目の具体例をオブジェクト図へ定義する．

†芝浦工業大学大学院工学研究科電気電子情報工学専攻,
Graduate School of Engineering,
Shibaura institute of technology
Department of electronic engineering and computer science

‡芝浦工業大学システム工学部電子情報システム学科,
Shibaura institute of technology
Department of electronic information system

表 2 モデルの定義内容と確認事項の対応

モデル名	確認事項
アクティビティ図	・サービス実行手順としての画面遷移 ・サービスに対する入出力項目の過不足 ・入力項目の入力形式
クラス図	・グループ化すべき入出力項目
オブジェクト図	・入出力項目の具体例

2.2 アクティビティ図の定義

アクティビティ図はユースケース単位で定義し，入出力操作レベルのユーザとシステムのやりとりを表現する．

- デシジョンノードとマージノードを別々のノードで表現するか 1 つのノードで同時に表現するかのように，1 つの意味を複数の記述で表現できるが，要素の制御フローの入出力数に制約を設けることで表現方法を統一する．
- ユーザとシステムの境界として，パーティション名をアクター名としたパーティションを定義する．(図 1-(1))
- デシジョンノード直後の制御フローに，必ずガードを定義する．(図 1-(2))
- アクション名の形式を「～を…する」に統一する．「～」の名詞部分に入出力項目名を定義し，「…」に動詞部分に処理動作を定義する．(図 1-(3))
- 入力形式表現用の動詞があり，その動詞を必要に応じてアクション名に定義する．(図 1-(4)及び 4.2 節で後述)
- グループ化された入出力項目を表示する場合は，オブジェクトノードをユーザ側の処理に定義する．(図 1-(5))

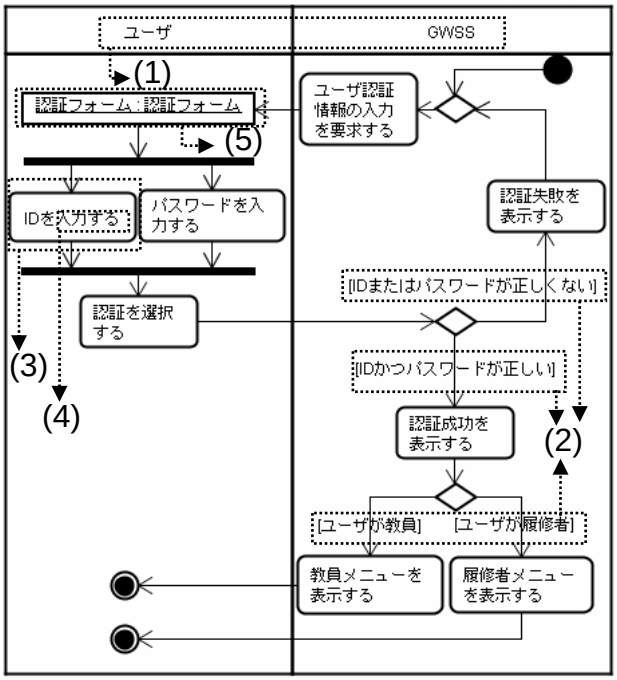

図 1 「ユーザ認証する」のアクティビティ図と制約

2.3　クラス図の定義

クラス図では，プロトタイプ上で表現すべき入出力項目のグループ化を目的としてクラスを定義する．これにより，ユーザとシステムのインタラクションを決定していくとともに，クラス候補やクラス内の属性候補を抽出することができる．記述内容として，関連や操作は定義せず，1 つのクラスを 1 つのグループとみなし，クラスの 1 つの属性を 1 つの入出力項目として定義する．

2.4　オブジェクト図の定義

オブジェクト図では，2.3 のクラスに対する具体例としてインスタンス仕様を定義する．なお，1 つのスロットに複数の値を定義する場合は，カンマ区切りで定義する．

3　要求分析プロセスとプロトタイプの確認

本研究の要求分析プロセスを以下に定義し，図 2 に示す．

1. 既存業務や顧客要求から必要サービスをユースケースとして導出し，ユースケース図に定義する．
2. 導出したユースケース単位でシステム化した業務手順をアクティビティ図に定義する．
3. アクティビティで必要とされる入出力項目をグループ化し，クラスとしてクラス図に定義する．
4. 3.で定義したクラスに対して，インスタンス仕様をオブジェクト図に定義する．
5. システム化後のアクティビティ図，クラス図，オブジェクト図の情報からプロトタイプ自動生成ツールにより，プロトタイプを自動生成する．
6. 顧客がプロトタイプにより要求分析モデルの妥当性を確認し，満足しない箇所を指摘する．
7. 指摘の要求レベルを表 2 の確認事項にて判断し，対応するモデルを修正する．(以降，顧客の了承が得られるまで 5.-7.を反復的に繰り返す．)

本研究では，現状として 1.のユースケースの導出方法や 6.のプロトタイプの使い方は考慮していない．また，2.-4.のモデル定義では，順序的に行う必要は必ずしもない．

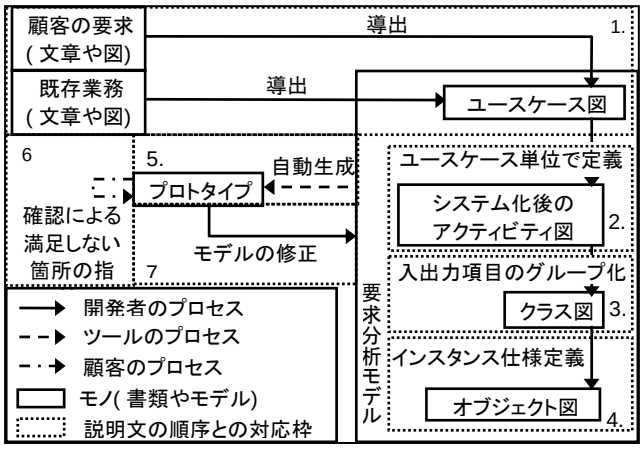

図 2　プロトタイプ生成および妥当性確認までの手順

4　プロトタイプ自動生成ツールの設計と実装

4.1　アクティビティ図の解釈

プロトタイプ自動生成ツールでは，以下に示すような概念をアクティビティ図に設けており，この概念を用いて画面分割や遷移を表現している．

(1) ユーザ側へ処理が移行する直前のシステム側のアクションを「システム外部遷移トリガー」と呼ぶ．
(2) システム側へ処理が移行する直前のユーザ側のアクションを「画面遷移トリガー」と呼ぶ．
(3) 1 画面の区切りを画面遷移トリガー直後とする．
(4) システム側の処理で定義されるガードを，複数の遷移先を表現するための遷移条件とする．

図 3 に各番号とアクティビティ図との対応と分割された 1 画面の例を示す．

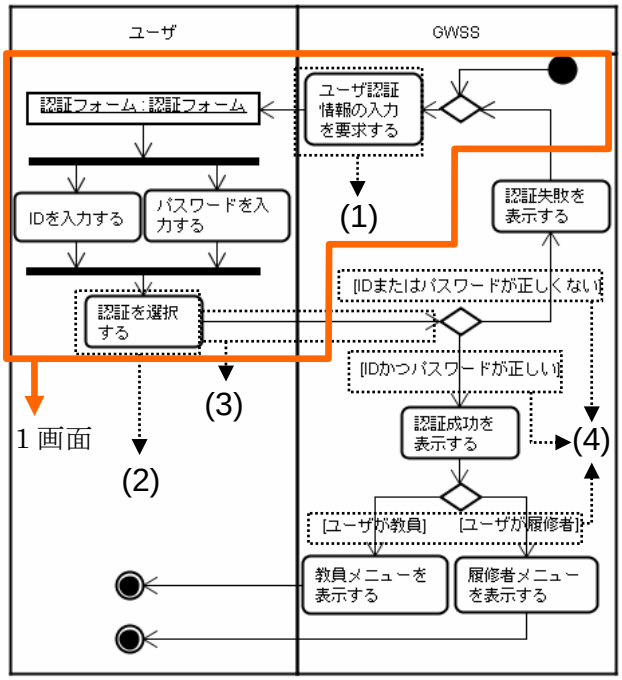

図 3　「ユーザ認証する」のアクティビティ図と概念

1 画面中の入出力項目は，ユーザ側の処理のオブジェクトノードとアクションによって決定し，図 3 の 1 画面の枠の例では，オブジェクトノードの認証フォームクラス，アクションの「ID」，「パスワード」，「認証」となる．

4.2　アクションの動詞と入力形式

表 2 の(5)で述べた入力形式の表現用に用意した動詞とその動詞に対応する入力形式の種類を表 3 に示す．

表 3　動詞と入力形式の対応

動詞	入力形式の種類
入力する	テキスト入力形式
単数選択する	ラジオボタン選択形式
複数選択する	チェックボックス選択形式
選択する	リンク選択形式(画面遷移トリガーに必要)
確認する	項目名表示(入力ではない)

例えば，図 3 の例では，「ID」と「パスワード」はテキスト入力形式の入力項目であり，「認証」はリンク選択形式の入力項目となる．そして，4.1 の(3)より「認証」のリンクは，「ID またはパスワードが正しくない」，「ID かつパスワードが正しい&ユーザが教員」，「ID かつパスワードが正しい&ユーザが履修者」の 3 パターンの遷移条件に応じた遷移先が存在する．

4.3　クラス定義の入出力項目とテーブル表示

入出力項目のグループ化をクラスで定義した場合，プロトタイプ確認の際にその構造が理解しやすいように，グループ化された入出力項目はテーブル形式で表示する．テー

ブルの構成要素と各要素の名前を図 4 に示し，その構成要素と要求分析モデルの要素の対応を表 4 に示す．

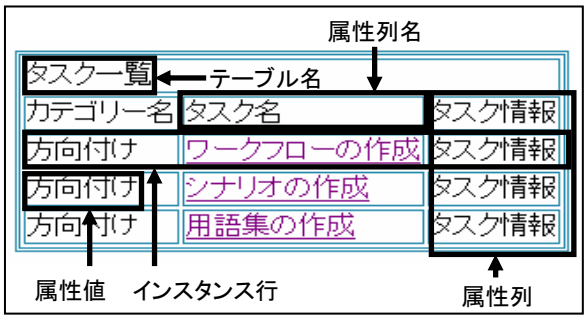

図 4　プロトタイプのテーブル構成要素

表 4　テーブル構成要素と要求分析モデル要素の対応

図 4 の構成要素	要求分析モデル要素
テーブル名	オブジェクトノードのオブジェクト名
属性列名	クラスの属性
属性列	クラスの属性およびインスタンス仕様のスロット
属性値	インスタンス仕様のスロット
インスタンス行	インスタンス仕様

なお，属性値の表現形式について，その属性列名を含んだアクションが定義された場合は表 3 に示した形式で表現される．逆にそのようなアクションが定義されていない場合は，インスタンス仕様のデータをそのまま表示する．また，オブジェクトノードで指定されたクラスから対応する全てのインスタンス仕様を取得し，インスタンス行を生成する．その際の属性列の数はクラスの全属性数と同値となる．そこで，問題となるのが，テーブルの情報を操作したい場合である．例えば，

(1)特定の属性列自体を非表示にする．
(2)特定の属性列の属性値を非表示にする．
(3)インスタンス仕様の取得数に制限をかける．
(4)インスタンス行に新規欄として空欄を設けたい．

という場合である．(1)の例として，「ログイン ID」，「パスワード」，「名前」を属性として持つ「ユーザアカウント」クラスのインスタンス仕様を定義したとする．このとき，プロトタイプ上で「ユーザアカウント」クラスに対するテーブル表示を行う場合に「パスワード」の属性列を非表示にしたいといった事がある．(3)の例としては，プロトタイプで検索機能を見せる際に，ある属性値が同値のインスタンス仕様のみを見せたいといった事がある．

このような要求をオブジェクトノードの状態に対し，表 5 に示す本研究独自の命令記述を設けることで解決する．この命令は原則として属性列名を指定して行う．

表 5　オブジェクトノードの状態に対する命令

命令の種類	命令値
属性列の表示	表示　または　非表示
属性値の表示	表示　または　非表示
インスタンス行の絞込み	指定属性列の属性値が同値なインスタンス仕様数
新規欄	有　または　無

特にインスタンス行の絞込みの意図として，命令値で指定された要素数を上限としたとき，上限を超えない最大要素数のインスタンス仕様の集合を取得する．

また，新規欄の命令はインスタンス行の絞込みの命令と併用することが原則である．そして，この命令値を記述された属性では，インスタンス行の絞込みにより属性値が 1 つに決定しており，新規欄として設けられた空白のインスタンス行に，その属性列の決定した属性値を表現することができる．もし，オブジェクトノードの状態に命令がない場合は，属性列は表示，属性値も表示，インスタンス行の絞込みは行わず，新規欄は無，となる．

4.4　実装

プロトタイプ自動生成ツールは Java により実装した．要求分析モデルが定義された JUDE5.0 の JUDE ファイルとシステムのパーティション名をツールへの入力とし，HTML で記述された Web ページをツールからの出力とする．内部処理としては，アクティビティ図毎に制約に基づくモデル記述形式の検査を行い，検査を通った図に対しプロトタイプを生成する．

5　要求分析モデル間の連携

アクティビティ図，クラス図，オブジェクト図の 3 種のモデルを連携させた情報からプロトタイプを自動生成するので，開発者は 3 種のモデルをフローとデータを区別して定義できるため，分析内容の整理が可能となる．一方で，モデルの段階的な定義に応じたプロトタイプ生成を可能にするため，プロトタイプ自動生成ツールはアクティビティ図のみからでもプロトタイプが生成できる．

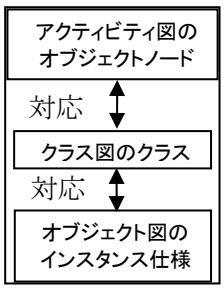

図 5 モデルの連携

図 5 に示すモデルの要素に関して，

・オブジェクトノードのクラス
・インスタンス仕様のクラス

のクラスを共通にすることで，ツールはモデルを連携させて情報取得することが可能となる．従って，オブジェクトノードの定義箇所において，グループ化された入出力項目とその具体例を表現したプロトタイプが生成可能となる．

図 6 に「ユーザ認証する」の認証フォームのアクティビティ(一部)，クラス，インスタンス仕様を示し，図 7 に連携の有無によるプロトタイプの表示の変化を示す．

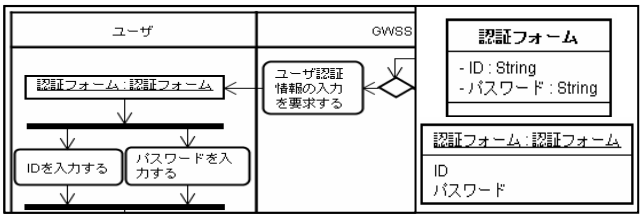

図 6　認証のアクティビティ，クラス，インスタンス仕様

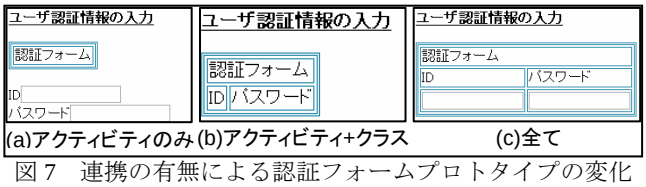

図 7　連携の有無による認証フォームプロトタイプの変化

6　後開発工程におけるモデルの利用可能性

アクティビティ図では，入出力操作レベルのユーザとシステムのインタラクションを定義し，プロトタイプを通し

てモデルの洗練するため，シーケンス図のアクターが直接メッセージを送るライフラインとのインタラクションに適用しやすい．また，オブジェクトノードの命令記述は，シーケンス図で詳細なロジックを定義する上で，データに対しどのような加工をする必要があるかの指標となる．

本研究におけるクラス図では，サービスに対する入出力項目のグループ化を念頭おいたクラスの作成を行う．これを元にモノの性質や関連を考慮したクラス図へ洗練していくことで，業務上必要な用語や項目を漏らさず要求分析モデルに定義できる．

本研究におけるオブジェクト図では，システムにおけるサービス実行手順に即した必要データを定義したものであるため，テストデータとして再利用できる．

7　適用事例と考察

7.1　GWSSへの適用

本学の GWSS はシステム開発時におけるグループワーク支援目的ために開発された．システムの提供するサービスの内，ファイル共有機能に対し本提案を適用した．GWSS では，フェーズ→カテゴリー→タスク→作業項目の順でグループの行う作業を分類しており，ファイル共有機能では，作業項目単位に共有場所を提供することで，グループ内のファイルの共有を実現している．本稿ではプロトタイプの表現力に着目し，この機能の実行手順や入出力項目について，現行システムでの表現と自動生成されたプロトタイプの表現の差異を検証した．

7.2　適用結果

適用結果として，確認事項が確認できたかや確認事項に応じたモデルの修正が可能だったかを表 6 に示す．

画面遷移は，どのような遷移条件の元で遷移先が決定するのかを確認でき，また，修正に関しては，アクティビティ図のフローを変更することで行える．

入出力項目の過不足は，オブジェクトノードに対応するクラスの属性の変更やアクションの追加・削除によってプロトタイプ上で表現する入出力項目の変更が行える．

入力項目の形式は，本稿で用意されてない動詞に関しては，ツールの処理にその動詞と対応する入力形式を追加することで可能となるため，確認および修正が行える．

グループ化すべき入出力項目は確認ができ，グループ化される入出力項目を追加・変更する修正は行えるが，テーブルのレイアウトに関する修正が行えない．

入出力項目の具体例は，確認および追加・変更による修正は行えるが，ツールのインスタンス仕様の取得順序に対して明確に順序を定義できないため，インスタンス行の順番を変更することや，特定のインスタンス行に対して命令記述を有効にすることが行えない．

表 6　プロトタイプ確認とモデル修正の適用結果

確認事項	確認	修正
・サービス実行手順としての画面遷移	○	○
・サービスに対する入出力項目の過不足	○	○
・入力項目の入力形式	○	○
・グループ化すべき入出力項目	○	△
・入出力項目の具体例	○	△

○：可能，△：一部可能，×：不可能

7.3　考察

7.2 の結果から，プロトタイプのテーブル構成に問題点があった．ファイルを共有するためのタスク選択画面を現行システムとプロトタイプの両方を図 8 に示す．

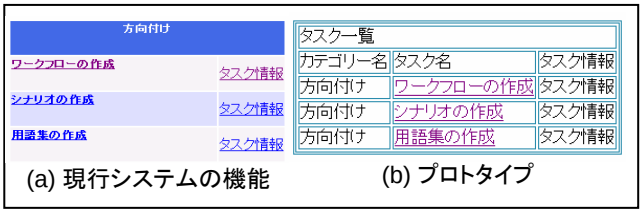

図 8　現行システムの機能とプロトタイプの比較

図 8 より，プロトタイプでは同値の属性値のセルを 1 つにまとめられていないことである．「同じ値ならまとめて 1 つで表示した方が見やすい」という顧客の要求は容易に想定できる．しかし，ツールではインスタンス行間の関連性は考慮していないため，まとめることは現段階では不可能となっている．他の問題として，例えば図 8 で現行システムでは表現されていない属性列名がプロトタイプで表現されているように，プロトタイプでは現行システムよりも情報量が多い場合がある．これは，プロトタイプを確認してもらう上で「説明のための表示」が必要だからである．しかし，顧客は実際のシステムの情報量を確認したい場合があるため，「実際を想定した情報量による表示」を別途実現する必要がある．これを解決するために，要求分析モデル，または，ツールによるプロトタイプ自動生成過程において，表示方法の種類を指定することによって，プロトタイプの表現を変化させる必要がある．

8　まとめ

本稿では，アクティビティ図，クラス図，オブジェクト図を連携させることにより，入出力項目の表示に具体的なデータを表示できるようになり，直感的な理解の容易さを向上する意味では，プロトタイプの表現力の向上が実現できた．しかし，適用事例により新たな問題も出てきた．それは，使用性に関わるデザインの問題や，プロトタイプの実際的な利用方法に関わる問題であった．

今後の展望は上記の問題の解決も考慮しつつ，顧客が
(1)サービス実行手順としてのサービス間の連携
(2)システムの使用権限と権限による実行可能なサービス
の 2 つを確認できるプロトタイプを生成することを目標とする．具体的には，(1)に対しては，ユースケース単位で表現していたアクティビティ図に対し，ユースケース自体の実行順序を表現するアクティビティ図を定義し，ツールにその関係を解釈するような処理を加えることで解決を目指す．また，(2)に対しては，権限をアクターに置き換えて考え，プロトタイプを生成する際にアクターを指定する．そして，ツールに，指定されたアクターがパーティション名として関わるアクティビティ図のみを解釈するような処理を加えることで解決を目指す．

9　参考文献

[1] 大西淳，郷健太郎，要求工学，共立出版，2002
[2] 小形真平，松浦佐江子，UML・プロトタイプを組み合わせた要求仕様の妥当性確認，第 69 回情報処理学会全国大会，6L-7，2007