B-027

要求分析モデルを用いたエンティティ間の関連と属性に対するデータライフサイクル検証手法の提案

Data Life-Cycle Verification Method for Entity Relationship and Attribute by Using Requirement Analysis Model

奥田 博隆† 小形 真平‡ 青木 善貫† 松浦 佐江子†
Hirotaka Okuda Shimpei Ogata Yoshitaka Aoki Saeko Matsuura

1. はじめに

業務システムではサービスに必要なデータを永続化し、全てのユーザに同等の誤りのないサービスを提供することが重要である。要求分析段階から、こうした必要なデータ（これをエンティティと呼ぶ）のライフサイクルに関わる基本的な操作である CRUD（Create、 Read、 Update、 Delete）を用いて、データが参照される前には必ず作られると言うようなシステムの普遍的なデータの性質（以降、データライフサイクル）を満たすことを保証し、後工程での手戻りを防ぐことが重要である。

我々は、業務系 Web システム開発を対象にユースケースに基づく利用者とシステムのインタラクションや入出力データを定義した数種の UML（Unified Modeling Language）モデルからなる要求分析モデルを用いて、要求の妥当性確認と要求の正確な仕様化支援を目的とした、プロトタイプ生成可能なモデル駆動要求分析手法を提案してきた[1]。

データライフサイクルを検証する為に、CRUD 記法を加えた要求分析モデルから CRUD テーブルを自動生成し、サービスの責務とエンティティの過不足、サービスに不必要なエンティティ、データライフサイクルという 3 観点からの検証方法を提案した[2]。しかし、[2]では、モデルに記述された処理フローに応じたオブジェクトのライフサイクルの妥当性を網羅的に保証できない。そこで、モデル検査ツールを用いた検査方法を提案した[3]。（以降、[2]、[3]手法を合わせて、オブジェクトのデータライフサイクル検証手法と呼ぶ）

[3]や[2]では、エンティティ間の関連と属性に基づくライフサイクルを検証する為の定義が不十分であり、エンティティ間の属性に対して、開発者の想定と検査結果が異なり、開発者の想定とは異なる解釈が発生していた。

そこで、本論では、開発者が定義する属性に対する機能を規定する概念である IRUN（Insert、 Refer、Update、 Nullset）[4]にオブジェクトの関連を明記できる概念を追加した IRUN テーブルを用いて属性間の関係を考慮したライフサイクルの検証を本学学習支援システムのサブシステムである「掲示板システム」（以下、BBS システム）及び、研究室で運営される「図書管理システム」の適用事例を通して、有効性を議論する。

2. モデル駆動要求分析手法

要求分析段階は、顧客要求を分析し、要求仕様としてまとめる段階である。 本研究では、基本的にはユースケース分析と同様に、 要求の最も核となる機能要求を中心に扱う。特に、以下の観点から、ユーザがシステムを直接操作する UI に機能要求が 1 顕在する部分(これを、ユーザとシステムのインタラクションと呼ぶ)を分析対象とする。具体的には、 業務プロセスをシステムのサービスとして定義する場合、以下の項目を分析する。

（1）業務規則に基づくサービス成立時の入力データとその条件
（2）サービス不成立時の条件
（3）各条件下の振る舞い
（4）振る舞いの結果としての出力

これらはすべて、ユーザが操作上で理解すべき点である。要求分析手法では、開発者が、上記（1）から（4）の観点から、業務遂行に必須な業務フローと業務データを UML モデルであるアクティビティ図とクラス図により定義する。ユースケースをユーザの利便性や柔軟性の観点から分割・統合した単位をサービスとして捉えてアクティビティ図に定義する。アクティビティ図では、フローを構成するアクションとオブジェクトノードの分類子となるエンティティに定義されるデータとの関連を定義する。特に上記の観点における、ユーザの入力、サービスの不成立の条件、出力に関わるフローとデータが UI に顕在する要求に当たることから、これらを識別できるように、アクティビティ図を「ユーザ」「インタラクション」「システム」パーティションに分けて定義する。

本稿の実験対象である「BBS システム」のサービスの一つである「FAQ を閲覧する」を例としたアクティビティ図を示す(図 1)。

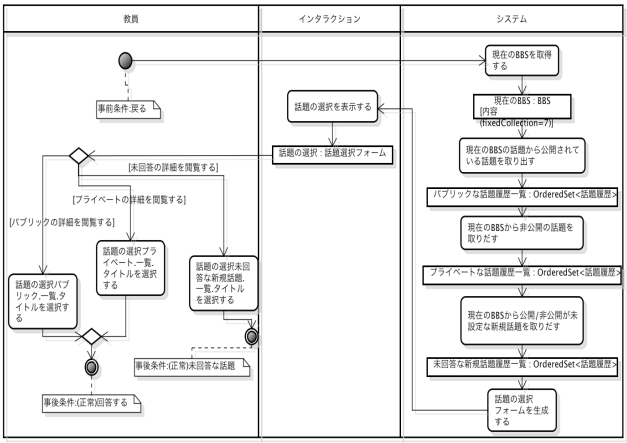

図 1 要求分析モデルによるアクティビティ図

システムパーティションにあるすべてのオブジェクトノードやアクションの目的語を表すデータが（1）の入力データから（4）の出力データを生成するために必要となるデータ群であり、これがエンティティの候補である。

さらに、業務データの具体例をオブジェクト図を用いて定義し、これらのモデルから HTML（Hyper Text Markup Language）の Web ページで構成される UI プロトタイプを自動生成する。このプロトタイプは、最終プロダクトの機

† 芝浦工業大学、 Shibaura Institute of Technology
‡ 信州大学、 Shinshu University

能におけるシステムの内部処理および UI における外観を除いたシステムの模型であり、顧客は、この最終プロダクトの模型を通して顧客が業務遂行に必須なフローとデータを確認する。UI プロトタイプは、顧客による要求の妥当性確認のみならず、開発者のモデル理解にも有効である。それは、複数のモデルの整合性を 1 つの模型を通して確認することができるからである。開発者が各モデルとプロトタイプの関係を十分に理解できるように、要求分析手法では、図 2 に示すような要求分析モデルの各種定義段階に応じて段階的にリッチ化する UI プロトタイプ自動生成を実現している。なお、UML モデリングツールは、ChangeVision 社の astah*[5]を利用する。

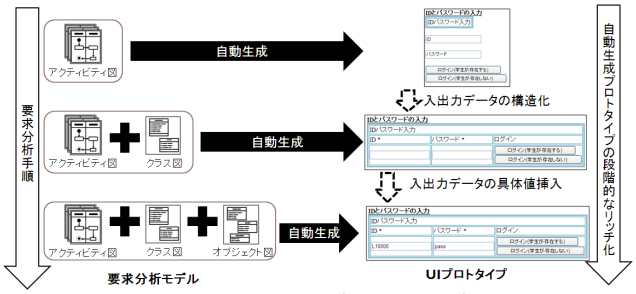

図 2　段階的な WebUI プロトタイプ自動生成

3. オブジェクトのデータライフサイクル検証方法

3.1 要求分析モデルに追加する CRUD 記法

エンティティのライフサイクルを検証するために、アクティビティ図のシステムパーティション内のアクションの記述（目的語と動詞）と目的語に対応するオブジェクトノードの位置関係を整理し、CRUD アクションを明確化する。CRUD アクションを表す動詞は、複数の適用事例で使用された動詞の調査に基づき「作成する、検索する、更新する、削除する」等の 9 種類に限定した。例えば，「作成する」アクションは、「○○な本を作成する」という形式で書かれ、目的語には作成対象のオブジェクトノード名又はクラス名をとり、定められた動詞を取る。さらに結果として得られる目的語に対応するオブジェクトノードをアクションの直後に置く。

3.2 CRUD テーブルを用いた検証手法

CRUD テーブルは、サービス名及びアクター名を縦に、エンティティ名を横に記述したテーブルを定義する。それらの交差するマス目に、アクティビティ図中の CRUD 記法に該当するアクションを識別し、図 3 に示す様に、関わる機能を記述する。

CRUD テーブルを用いた検査では、次の 3 つの観点から検査する[2]。

- サービスとエンティティの責務･･･開発者が意図するサービスの責務とエンティティの過不足を CRUD テーブルから判断する。
- サービス全体と関わりの薄いエンティティ･･･CRUD テーブルからシステムにとって不要なエンティティを洗い出し、必要性を判断できる。不要でない場合、システムにとって必要なエンティティの機能分析されていない可能性があり、エンティティの機能の過不足を判断できる。
- データライフサイクルの検証･･･各サービス内でエンティティの機能が CRUD のデータライフサイクルが満たされているかを検証する。

3.3 シナリオに沿った CRUD の網羅的な検査

アクティビティ図のアクションのフローによっては、CRUD テーブルと意図は異なる為に網羅的な検査が必要となる。そこで、モデル検査技術を用いて網羅的に検査する事で、データライフサイクルがどのフローでも完全に満たせるかどうかを検査できる[3]。

3.4 オブジェクトレベルの CRUD 検証手法の問題点

3.2 節で述べた手法では、オブジェクトの属性が他のオブジェクトと関連がある場合には、正確に検査する事ができない。何故なら、アクティビティ図のアクションの目的語と動詞だけでは、オブジェクトの関連に関する意図を正確に読み取る事はできない。

		BBS	コンテンツ	投稿内容	投稿者	日時	話題	話題履歴
学生	FAQに質問を新規投稿する		C		R	R	C	C
学生	FAQに質問を追加投稿する		C	C	R	R	C	U
教員	FAQを閲覧する	R						
学生	FAQ学生Viewを閲覧する	R			R			
学生	プライベートFAQに質問を追加投稿する		C	C	R	R	C	U
教員	新規質問に回答する		C		R	R	R	U
教員	質問に回答する		C		R	R	R	U
教員	質問に回答する_主担当		C		R	R	R	U

図 3　BBS システムの CRUD テーブル

次に BBS システムの例（図 3）を用いて説明する。

「FAQ を閲覧する」サービスを実行する場合の開発者の意図は、「BBS」が Read されると、関連を持つオブジェクトである「投稿内容」、「話題」、「話題履歴」等を Read する事である。しかし、図 3 では、「BBS」の Read 以外には何も Read されていない為、開発者の意図と異なる事が問題点である。

4. 提案手法

4.1 IRUN テーブルによる検証手法

3.4 節で述べた問題点を解決する為に、属性に対する機能を規定する概念である IRUN（Insert、Refer、 Update、Nullset）にオブジェクトの関連を明記できる概念を追加した IRUN テーブルの形式と検証方法を提案する。

IRUN テーブルは、サービス名とそのシステムパーティションのアクションに対してそれに関わる属性の機能の定義表である。それを用い、開発者が次の 3 点を検証する。

（1）既存の CRUD テーブルと差異を検討し、開発者の意図を確認する。
（2）各属性の実現可能性を検証する
（3）各属性の要求の妥当性を検証する

4.2 属性に対する機能（IRUN）の定義

CRUD で規定されたアクションの意味は、その属性に対する操作によって詳細化される。そこで、属性に対しては、値の挿入（Insert）、値の参照（Refer）、値の更新（Update）、Nullset（値の削除）を用いる。属性は値型と参照型に分類することができる。値型は、数値や文字列を表し、参照型はオブジェクトを表す。

IRUN の概念では、属性の型がオブジェクトである時に、他のオブジェクトを用いて Insert、Update、Nullset を行い

たい場合と関連するオブジェクトを連鎖的に Insert、Update、Nullset を行いたい場合が存在する。そこで、前者に対しては、Insert Pointer、Update Pointer、Nullset Pointer として明確に区別する。

従って、本研究では、属性に対する機能を Inset、Insert Pointer（IP）、Update、Update Pointer（UP）、Refer、Nullset、Nullset Pointer（NP）（以降、IRUN と呼ぶ）の合計 7 つ定義する。

4.3 IRUN テーブルの形式とその記述方法

IRUN テーブルの形式を図 4 に示す。横列に永続化対象とするエンティティとその属性をすべて列挙する。また、「Class」（灰色マス目）とは Class 全体に対する機能を記述する箇所である（オブジェクトレベルと同義）。縦列には、サービス名（灰色マス目）とサービスの中の CRUD 機能を持つアクションとそれらの実行主体であるアクターの３種類を記述する。各属性とアクションの交差した所に、属性の機能を定義する。

次にテーブルの記述形式について説明する。エンティティの「Class」とサービス名が交差するマスに対しては、オブジェクトに対する機能を定義する為、CRUD を記入できる。このマスは、CRUD テーブルと等価である為に、CRUD テーブル基づいて記入する。

エンティティの属性（白マス目）とアクション（白マス目）の交差するマス目に対しては、属性に対する機能を定義する為、4.2 の 7 種の機能を記入できる。ここで、属性の関連を意識して、機能を記入する必要がある。例えば、あるオブジェクトの「作成」が意図するのは、他のオブジェクトを参照に基づいて作成されるのか（IP）、そのオブジェクトもどの範囲まで連鎖的に作成するのか（I）によって使用する機能が変わる。また、R については、取得したい関連を含めた属性すべてにまたがり記入する。

		Entity1			Entity2			Entity3		
		Class	Attr1	Att2	Class	Attr1	Attr N	Class	Attr1	Attr2
Actor	Service 1									
Actor	Action 1									
Actor	Action N									
Actor	Service 2									
Actor	Action 1									
Actor	Action 2									
Actor	Action N									

図 4　IRUN テーブルのひな形

4.4 IRUN テーブルを用いた検証項目

（１）CRUD・IRUN テーブルによる比較検査

オブジェクトレベルで作成された CRUD テーブルと属性までの機能を定義した IRUN テーブルを比較する。CRUD テーブルで検証しきれなかった、オブジェクト間の関連が意図通りに反映されているかを検査する。

検証手順について説明する（図 5）。各属性とアクションに対して機能を記述し（図 5 の①）、次にサービス名とエンティティの「Class」部分（図 5 の②）に対して、①で使用された機能が何れか 1 つでもある場合に既存の機能との差異が明確になるように CRUD を追記する。

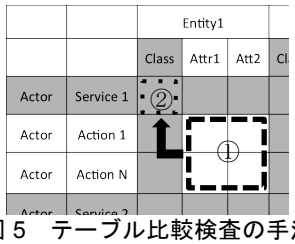

図 5　テーブル比較検査の手法

その結果を既存の CRUD テーブルと重ね合わせる事によって、比較を行い、検査することができる。

（２）実現可能性の検査

システムを実現する上で手戻りやバグを引き起こす致命的な誤りをデータライフサイクルの観点で検出する。特に以下の２点の違反項目がないか検査を行う。

- 属性毎に見て、Insert が存在しないが、Refer や Update や Nullset が存在する。
- 属性毎に見て、同一サービス内で Refer 又は Insert が存在しないが、Nullset や Update が存在する

（３）要求の妥当性の検査

システムを実現する上で、致命的な誤りではないが、要求の観点から妥当性が疑われる項目を検査する。

- 属性毎に見て、一度も使用されない属性があるか。（理由）使用されない属性は、システムにとって不必要の可能性が高く、混乱を与える。
- 属性毎に見て、Refer のみ行われている属性があるか。（理由）分析対象外のシステムで作成する意図であれば良いが、記入漏れの可能性がある。
- 属性毎に見て、Insert のみ行われている属性があるか。（理由）分析対象外のシステムで参照する意図であれば良いが、記入漏れの可能性がある。

また、分析対象外のシステムで意図がある場合は、ステレオタイプとして「Constant」をクラスに付加する

5. 適用事例

本手法を本学学習支援システムのサブシステムである「掲示板システム」に対して適用した結果を示す。

		BBS	コンテンツ	投稿内容	投稿者	日時	話題	話題履歴
		Class	Class	Class	Class	Class	Class	Class
学生	FAQに質問を新規投稿する	***U***	C		R	R	C	C
学生	FAQに質問を追加投稿する	***U***	C	C	R	R	C	U
教員	FAQを閲覧する	R	***R***	***R***	***R***	***R***	***R***	***R***
学生	FAQ学生Viewを閲覧する	R	***R***	***R***	R	***R***	***R***	***R***
学生	プライベートFAQに質問を追加投稿する	***U***	C	C	R	R	C	U
教員	新規質問に回答する	***U***	C***R***	***R***	R	R	R***U***	U
教員	質問に回答する	***U***	C	***R***	R	R	R***U***	U
教員	質問に回答する_主担当	***U***	C	***R***	R	R	R***U***	U

図 6　BBS の IRUN テーブル

最初に、4.4 節の（１）に対する検査について説明する。各アクションと属性を持つ機能を定義した結果、図 6 のテーブルの結果となった。

図 6 では、図 5 の機能は装飾がない文字で表し、追加された機能は、イタリック・太字・アンダーラインの組合せで装飾された文字で表している。その結果、3.4 節で問題点となったオブジェクトの関連を明確に定義した為、IRUN テーブルでは開発者の意図通りに反映されている。

「FAQ を閲覧する」というサービスでは、図 3 では、「BBS」というエンティティに対してのみ「Read」を行っていたが、図 6 では、関連するエンティティである「コンテンツ」、「投稿内容」、「話題履歴」などに対する「Read」がされ、より明確になった。

また、「話題」というエンティティでは、図 3 の CRUD テーブルの「新規質問に回答する」、「質問に回答する」サービスでは、「Read」の機能しか無いが、図 6 の IRUN テーブルでは、属性の機能をより明確にすることで「Update」の機能が意図として現れている。

次に、4.4 節の（2）に対する検査について説明する。実現可能性の観点から、IRUN テーブルを検査した。研究室内図書管理システムにおいて発見できた。図書管理システムでは、利用者が貸出記録を参照したいという要求があった。そこで、利用者情報クラスと貸出記録クラスに対して関連持たせ、利用者情報が複数の貸出記録を保持できるようにした。そのクラスを図 7 に示す。

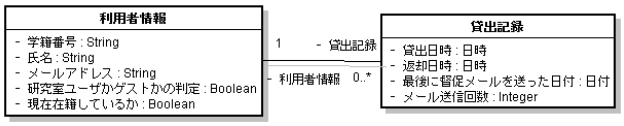

図 7　利用者情報と貸出記録のクラス図

しかし、実際に、IRUN テーブルを記述したところ、図 8 の点線で囲まれた列に Insert Pointer が存在しないことから、貸出が行われた場合に貸出記録を保持する利用者に対して貸出記録の追加が行われていない事がわかった。

A	B	L	O
		利用者情報	利用者情報
		Class	貸出記録.Set
研究室ユーザ	延滞を確認するact		
研究室ユーザ	本を登録する		
研究室ユーザ	キーワードで検索する		
研究室ユーザ	ログインする	C	
研究室ユーザ	利用者情報一覧を取得する		
研究室ユーザ	Guestユーザー覧を取得する		
研究室ユーザ	利用者情報を作成する		
研究室ユーザ	利用者情報一覧を登録する		
研究室ユーザ	本を登録する(ISBN未使用)		
研究室ユーザ	貸し出し記録を参照する	R	
研究室ユーザ	利用者情報一覧から利用者情報を取得する		R
研究室ユーザ	利用者情報から貸出記録一覧を取得する		R
研究室ユーザ	貸出記録一覧から貸出記録リストを取得する		
研究室ユーザ	タイトルから図書を取得する		
研究室ユーザ	図書から貸出記録一覧を取得する		
研究室ユーザ	利用者情報一覧から学籍番号に基づき利用者情報を取得する		
研究室ユーザ	本を返却する		
研究室ユーザ	本を借りる	R	
研究室ユーザ	図書情報を更新する		
研究室ユーザ	研究室蔵書一覧を取得する		
研究室ユーザ	貸出記録を作成する		
研究室ユーザ	現在日時を取得する		
研究室ユーザ	利用者情報を取得する		R
研究室ユーザ	ゲストユーザのmy貸出記録一覧を取得する		R
研究室ユーザ	貸出記録一覧を登録する		
研究室ユーザ	貸出記録一覧を取得する		

図 8　データライフサイクルが満たされない属性を示す IRUN テーブル

最後に 4.4 節の（3）に対する検査について説明する。

要求分析過程で、試行錯誤を繰り返した結果、変更に対するモデルへのトレーサビリティを完全に保つ事ができない属性が登場する。研究室内図書管理システムでは、一度も使用されない属性として「本」クラスの「画像」というエンティティがある事が発見された。しかし、本クラスの属性が Refer されている箇所でも、属性である「画像」は一度も Refer されることは無い。その為、属性である「画像」属性の必要性や想定を再度議論することができる。

提案する検証手法が有効であると共に、分析段階に応じた提案方法が有効である事を示した。エンティティの抽出段階では、開発者は、CRUD テーブルを用いて、サービスとエンティティの責務について議論することが可能である。次にエンティティを仕様化する段階では、開発者が、IRUN テーブルを用いて、サービスとそのアクション及びエンティティの属性の機能について議論し、検証する事ができる。

6. 関連研究

我々は、要求分析過程の成果物に対して、容易に網羅的かつ段階的に普遍的な事象（本論では、CRUD）に注目して検証できる事を強みと考えている。

谷崎らは、システムはアプリケーションとその実行環境から成ると捉え、そのフィーチャーモデルを構築し、アプリケーションや環境の移植によってシステムに対して変化が起きる時に競合などの問題点に対処しようとしている[6]。既に構築されているシステムや仕様化された段階にあるシステムに対して、網羅的に変更範囲を特定することは有効である。しかし、要求の変化が激しい分析過程に用いる事は、フィーチャーモデルの記述にコストがかかり難しい。

7. まとめ

我々は、要求の妥当性確認と要求の正確な仕様化支援を目的とした、プロトタイプ生成可能なモデル駆動要求分析手法を提案してきた。要求分析モデルの実現可能性向上の為に 1 番に必要な事は、業務システムの特徴はサービスを継続的に提供し続ける事にあると考え、データのライフサイクルに注目し、CRUD テーブルによる検証と CRUD の網羅的な検査手法を提案してきた。

しかし、アクションの目的語とオブジェクトノードの関連から CRUD 機能を抽出するだけでは、オブジェクト間の関連を正しく捉えるのが難しい。そこで、本研究では、オブジェクト間の関連を解決する為に、オブジェクトの属性の基本機能である IRUN に注目した。本研究は、IRUN テーブルの形式とその検証手法を、本学授業支援システムのサブ機能である BBS システムを例にその有効性を示した。

現在の手法では、要求分析モデルに記述された処理フローに応じたエンティティ間の関連と属性に基づくライフサイクルの妥当性を網羅的に保証できない。そこで、網羅的な検査手法を今後、提案したい。

謝辞

実験の題材の提供と提案手法の実験に協力して頂いた、竹本君を始めとする松浦研究室所属の 4 年生に感謝する。

参考文献

[1] 小形 真平, 松浦 佐江子, “UML 要求分析モデルからの段階的な Web UI プロトタイプ自動生成手法”, コンピュータソフトウェア, vol. 27, No.2 (2010).
[2] 奥田 博隆, 小形 真平, 松浦 佐江子, “データライフサイクル保証の為の要求分析モデル検証, ソフトウェア科学会第 28 回全国大会, 2E-4 (2011).
[3] 青木 善貴, 小形 真平, 奥田 博隆, 松浦 佐江子, “UML 要求分析モデルにおける CRUD 観点のデータライフサイクルの妥当性検査手法”, 情報処理学会第 74 回全国大会, Vol.74, No.1(2012)
[4] 真野 正, “独習データベース設計”, 翔泳社 2009
[5] astah, http://astah.change-vision.com/
[6] 谷崎 裕明, 片山卓也, “ソフトウェアシステムの構成変更における整合性判定フレームワークの提案”, コンピュータソフトウェア, vol. 28, No.2 (2011)