社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－SE－147 (2)
2005／3／17

日本語ベースのオブジェクト指向記述言語 OCDJにおけるモデリング支援ツールの開発

木本哲史　北瀬裕丈　上田賀一
茨城大学 工学部 情報工学科

本研究では，ドメインユーザがシステム化対象モデルをモデリングするためのOCDJの記述仕様の具体化を行った．以前より自然日本語記述で対象モデルをモデリングするOONJが提案されている．ドメインユーザの利用を想定しているので，OONJはUMLなどの専門知識を必要としないという条件に適合する．しかし，OONJは真性実世界モデリングのための言語なので，ソフトウェア開発の擬似実世界モデリングには適さない．この問題を解決すべく，本研究でOCDJを提案した所以である．更に，ドメインユーザ向けのOCDJ記述をソフトウェア技術者向けの記述であるUML図へ変換するツールも開発した．

A Modeling Support Tool in Object-oriented Computation Description Japanese

Satoshi Kimoto, Hirotake Kitase, Yoshikazu Ueda
Ibaraki University

In this study, the description specification of OCDJ for domain users to model a target system is realized. OONJ to model a target system by natural Japanese description has been already proposed. And, OONJ conforms to the condition that domain users need no expert knowledge like UML. However, OONJ isn't suitable for pseudo real world modeling of software development because it is a language for truth real world modeling. We propose OCDJ to solve this problem. Furthermore, we developed the tool which translates OCDJ description for the domain user into the UML diagrams which were descriptions for the software engineer.

1 はじめに

現在，ソフトウェア開発の抽象設計段階にてUMLがよく使われている．通常，UML記述を行うにあたり，ソフトウェア技術者がシステム開発依頼者より要求を獲得する．更にそこから要求定義を行い，そしてUML記述を行う．本研究では，依頼者がシステム化対象を熟知している場合を考えている．その様な状況において，依頼者が要求定義を行える能力を備えていると仮定することは自然であると考えられる．しかし，そのような依頼者が実装能力を備えていると仮定するのは難しい．本研究では，要求定義を行えるが実装能力は備えていない依頼者をドメインユーザとする．

ドメインユーザはソフトウェア工学の専門家ではないのでUML記述を熟知していると仮定することはできない．そこでUMLと言った専門知識を用いずに抽象設計を行える記法を，本研究では考えていきたい．

以前より，自然日本語にてモデリングを行うOONJという記法[2],[3]が提案されている．自然日本語による記述という点により，OONJは専門知識を必要としないという条件を満たしている．しかし，その誕生の背景が，科学技術分野におけるコンピュータシミュレーション開発にある．本研究が考えているソフトウェア開発とは，図書館業務のコンピュータシステム化と言った分野であり，OONJが対象としている分野とは異なる．シミューレーションは対象モデルを全てコンピュータ上に再現し観察するのに対して，ソフトウェア開発は対象モデルを全てコンピュータ上に再現することはない．そのため，OONJをそのままソフトウェア開発のモデリング言語として利用することはできない．

そこで，OONJをソフトウェア開発用モデリング言語に適させたOCDJを考える．以前よりOCDJの基本概念は提案[1]されていた．本研究はその提案の具体化の第一段階である．

OCDJはドメインユーザ向けの記法でありソフトウェア技術者向けではない．ソフトウェア技術者には，UMLのような図式表現のほうが一目見てどのような設計であるかの情報を読み取りやすい．本研究ではOCDJの記述からUML図を作成することも考える．それによりドメインユーザはUMLの記法を知らなくても間接的にUML図を作成することができるようになり，ソフトウェア技術者との情報交換にも使える．さらに，生成されたUML図をUMLツール[5]-[8]に読み込ませることで，スケルトンプログラムを作成することができる．本研究室で同時期に進められている「UMLとUI設計を利用した開発支援ツールの構築」[5]と連動させることによって，同様の効果を得ることができる．

2 OONJとOCDJ

オブジェクト指向自然日本語記述言語OONJの基本事項を説明し，拡張されたオブジェクト指向計算記述日本語OCDJに特有の言語仕様を説明する．

2.1 OONJの基本特性

OONJとは，フレームシステムを用いオブジェクト指向構造化の記法である．フレームは外枠であり，スロットと呼ばれる行を複数個囲い込んでいる．記述対象モデルの一つのモノを一つのフレームとして囲い込む．モノを鉛筆とした場合，鉛筆は，芯の色・太さと言った特徴を持っている．また，線を描くという機能を持っている．鉛筆を一つのフレームに対応づけたら，そのフレームには芯の色・太さと言った特徴を表現するスロットと線を描くという機能のスロットを記述できる．特徴と機能を記述したスロットには，それぞれファセットナンバfn2*とfn3*も追記する．人間がその記述を読めば，ファセットナンバが無くともどのスロットが特徴(属性)か機能(振舞い)かを理解できる．しかし，コンピュータには，日本語記述から属性か振舞いかの判別は難しい．ファセットナンバを割り当てることで，フレームによる記述をある程度コンピュータが理解できる．図1にその様子を示す．

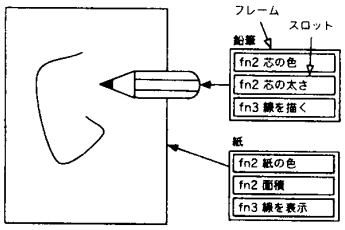

図1: モデリング対象モデルとフレーム記述

更に，記述対象モデルには鉛筆だけでなく他のモノも存在することが考えられる．それらのモノ同士が互いにやり取りすることもある．例えば，紙というフレームを追加してみる．鉛筆は自分の持つ芯の炭素を削り，紙にその炭素を付着させ線を描くといことがある．このモノ同士のやり取りも記述可能である．その様子を，フレームを用いて記述すると図2のようになる．

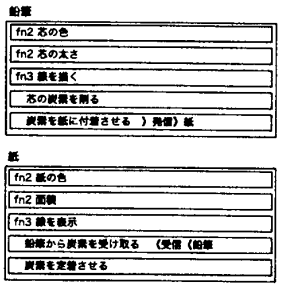

図2: モノ同士の相互作用記述

OONJは，以上のようなフレーム・スロットを自然日本語で記述していくモデリング記法である．OONJを使えば，コンピュータ化したい対象を記述し，その記述をコンピュータ処理し他の表現に変換することができる．

2.2 モデリング範囲

UMLのような専門的な知識を必要とせずにモデリングを行えるので，ソフトウェア開発者でなくとも扱いが可能である．しかし，OONJをそのままソフトウェア開発におけるモデリング言語として利用はできない．OONJは，対象モデル全てを一つの他の世界へ記述することを前提としている．例えば，気象のシミュレーションをコンピュータ上で行うにあたり，気象の世界の様々なモノをフレームで記述し，それらのフレームをコンピュータ上で再現するということをOONJは目標としている．そのモデリングを真性実世界モデリング[4]

と呼ぶ.

一方，コンピュータシミュレーションを除くソフトウェア開発の目標は，対象モデルをコンピュータ上で現実を再現するという点では，OONJ の目標と同じだが，対象モデルの全てをコンピュータ上で再現することはない．コンピュータ化したい現実を OONJ で記述することは可能だが，その記述はすべてをコンピュータ上に再現することになる．コンピュータ化したいことは自動化したい作業であり，その作業を指示する人間は再現される必要はない．こちらは，真性実世界モデリングに対し擬似実世界モデリング [4] と呼ぶ.

図書館システムを開発するにあたり，その世界を OONJ で記述することを例にとってみる．図書館には，本・利用者・司書・貸し出しノート・利用者名簿と言ったモノが存在する．図 3 のように，OONJ 記述はこれらのモノ全てを一つの世界として捉える.

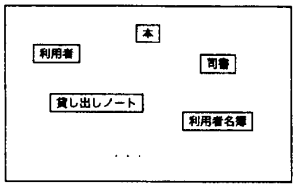

図 3: OONJ による図書館

ソフトウェア開発において，図 4 のように，コンピュータ上に再現するかしないかの区別を付けなくてはならない．システムとの関わりを持っていながら，システム上に再現されないものをアクタとして取り扱う.

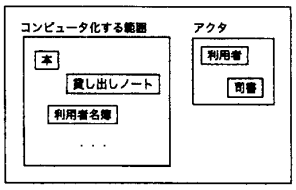

図 4: ソフトウェア開発を考慮した図書館システム

本研究では，OONJ にアクタという概念を導入した OCDJ を考える.

2.3 OCDJ の仕様

表 1〜表 3 に OCDJ の記述要素を列挙した．その中で下線が引いてある要素が，OCDJ にて加えられた要素である．残りは OONJ の記述要素と同じである.

OCDJ と OONJ の違いとして，アクタとそうでないものの区別を付けるか付けないかにある．その違いを明確にするために fn1.1 を更に細分化した.

- fn1.1.1　内部存在 システムの内部で働くモノ/概念
- fn1.1.2　外部存在 システムの外部で働くアクタとなるモノ/概念

システムの中と外とを結ぶ要素として fn1.7 インタフェースを定義した．インタフェースは以下のように分かれている.

- fn1.7.1 CUI ユーザ (アクタが人の場合) との対話をテキストにより行う
- fn1.7.2 GUI ユーザとの対話をウインドウ環境により行う
- fn1.7.3 プロトコル 他のシステム (アクタが人でない場合) との対話を TCP/IP と言ったプロトコルで行う

インタフェースと内部のモデルを接続するコントロールを fn1.4 として記述することにした．コントローラのフレームは以下のスロットから形成される.

- モデルが必要とする値を属性として保持
- モデルによる値の入力要求に応答するメソッド
- アクタの入力を管理するメソッド (イベントハンドラ)
- モデルへ値を渡すメソッド
- このコントローラを利用するモデルを集約として記述
- このコントローラが利用するビューを集約として記述

インタフェースが GUI の場合，fn1.7.2 以下のスロットからなる

- ボタン，テキストフィールドと言った要素
- 各要素の属性 (サブスロットに記述)
  - 表示座標
  - 初期値
  - 値
- それら要素を表示するメソッド
- ビューを利用するコントローラを被集約として記述
- ビューを利用するアクタを被集約として記述

表 3 に，ファセットナンバが 6 で始まる要素を列挙した．これらの要素を集約しインタフェースを記述する.

- fn6.1.* CUI 要素： テキストベースのインタフェースを構成する要素
- fn6.2.* GUI 要素: ウィンドウ環境におけるボタン，テキスト入力フィールドと言った要素
- fn6.3 プロトコル： アクタが他のコンピュータシステムである場合の更新手段としてのプロトコル

  OCDJ の記述段階ではプロトコルの詳細は記述せず，プロトコルが存在するとすればよい．

表 1: OCDJ 記述の要素リスト 1

fn04 個別の離散モデリング単位 (DMU)
fn1.0 要素種類間交差 DMU
fn1.1 モノ/概念 DMU 名
  fn1.1.1 内部存在
  fn1.1.2 外部存在
fn1.2 属性 DMU 名
fn1.3 振舞い DMU 名 / fn1.4 相互関係 DMU 名
fn1.5 対象世界共通要素 DMU 名
fn1.7 インタフェース
  fn1.7.1 CUI
  fn1.7.2 GUI
  fn1.7.3 プロトコル

fn2 属性
fn2.0 要素種類間交差属性
fn2.1 振舞い参照属性 / fn2.2 汎化/特化属性
fn2.3 集約/被集約属性 / fn2.4 一般関連属性
fn2.5 相互作用 (伝達付置) 属性
fn2.6 制約属性
fn2.7 アクセス属性 (情報隠蔽を含む)
  fn2.7.1 DMU アクセス属性
  fn2.7.2 属性アクセス属性
  fn2.7.3 振舞いアクセス属性 (振舞い発起条件)
  fn2.7.4 相互関連アクセス属性
fn2.8 所有 (所属)/所用属性
  fn2.8.1 共通属性 / fn2.8.2 共用属性
  fn2.8.3 共有/共属属性
fn2.9 複合 (抽象) 構造化属性 (CSA と略)
  fn2.9.1 状態 CSA / fn2.9.2 遷移 CSA
  fn2.9.3 初期化 CSA / fn2.9.4 変身後 CSA
  fn2.9.5 相互作用伝達付置 CSA

fn3 振舞い
fn3.1 内部振舞い
  fn3.1.1 一般内部振舞い / fn3.1.2 内部変身
  fn3.1.3 振舞い開始 / fn3.1.4 振舞い停止
  fn3.1.5 制約属性有効化 (活性化) 実行振舞い
  fn3.1.6 相互関連有効化 (活性化) 実行振舞い
fn3.2 相互作用伝達 (message passing, mp と略)
  fn3.2.1 mp 発信 (直接目的で fn1.1 モノを伝達付置)
  fn3.2.2 mp 発信 (情報のみを伝達付置)
  fn3.2.3 mp 受信 (直接目的で fn1.1 モノを伝達付置)
  fn3.2.4 mp 受信 (情報のみを伝達付置)
fn3.3 相互作用振舞い (=fn3.1+fn3.2)
  fn3.3.1 内部振舞い+mp 発信
  fn3.3.2 mp 受信+内部振舞い

表 2: OCDJ 記述の要素リスト 2

fn3.4 変身振舞い
  fn3.4.1 発生/消滅 / fn3.4.2 変形/復元
  fn3.4.3 コピー / fn3.4.4 破壊/復元
  fn3.4.5 分裂/融合 fn3.4.6 名称や数の変化
fn3.5 複合 (抽象) 構造化振舞い (CSB と略)
  fn3.5.1 状態表現 CSB / fn3.5.2 遷移表現 CSB
  fn3.5.3 初期化 CSB / fn3.5.4 変身 CSB
  fn3.5.5 相互関連属性活性化 CSB

fn4 相互関係
fn4.0 要素種類間交差相互関係
fn4.1 相互作用伝達 (mp)
fn4.2 汎化/特化 / fn4.3 集約/被集約
fn4.4 グループ化 (カプセル化を含む)
fn4.5 再帰展開/縮約 / fn4.6 離散化/修復再構成
fn4.7 詳細化/簡略化
fn4.8 時間的/空間的相互関係
  fn4.8.1 連接 / fn4.8.2 反復 / fn4.8.3 分岐
  fn4.8.4 逐次 / fn4.8.5 並列/並行
  fn4.8.6 分散 / fn4.8.7 接着/固着
  fn4.8.8 その他の時間的/空間的相互関係
fn4.9 所有 (所属)/所用
  fn4.9.1 共通/特有 / fn4.9.2 共用/専用
  fn4.9.3 共有・共属/専有・専属
fn4.10 変身前後関係
  fn4.10.3 破壊/復元 fn4.10.4 分裂/融合
  fn4.10.5 名称や数の変化 / fn4.10.6 変異・遷移
fn4.11 その他諸々の相互関係

fn05 対象世界共通要素
fn5.1 離散時間定義
  fn5.1.1 離散時間単位 / fn5.1.2 離散時刻
fn5.2 離散空間定義
  fn5.2.1 離散空間単位 / fn5.2.2 離散空間
fn5.3 初期状況記述
  fn5.3.1 時間/空間初期化
  fn5.3.2 属性初期化記述
fn5.4 対象世界駆動シナリオ
fn5.5 対象世界共有要素 / fn5.6 注釈

表 3: OCDJ 記述の要素リスト 3

fn6 インタフェース記述要素
fn6.1 CUI 要素
  fn6.1.1 文字列入力
  fn6.1.2 数値入力
  fn6.1.3 指定文字列表示
fn6.2 GUI 要素
  fn6.2.1 プッシュボタン
  fn6.2.2 ラジオボタン
  fn6.2.3 チェックボタン
  fn6.2.4 ポップアップボタン
  fn6.2.5 コンボボックス
  fn6.2.6 テキストフィールド
  fn6.2.7 固定テキスト
  fn6.2.8 クローズボタン
  fn6.2.9 パネル
fn6.3 プロトコル

3 OCDJ記述例

ここでは，図書館システムにおいて「司書」が「書籍検索」を行うOCDJ記述を紹介する．

1 fn1.1.2 司書

1	fn2.1	役職
2	fn3.3	本を探す ≫詳細化≫ 2:本を探す
3	fn3.3	予約をうける
4	fn3.3	リクエストを受ける
5	fn3.3	貸す
6	fn3.3	返却される
7	fn3.3	本を追加する
8	fn3.3	本を破棄する
9	fn3.3	催促する
10	fn3.3	利用者を追加する
11	fn2.1	職員番号を示すカード
12	fn4.3	≫集約≫ 6:書籍情報入力画面

2 fn1.3 本を探す ≪詳細化≪ 1:司書 [2]

1	fn3.3	本を探してもらうというメッセージを受ける ≪受信≪ 1:利用者 [15]
	fn2.5	| 書籍情報
	fn3.3	探したい本の書籍情報を元に検索用 PC で探す ≫発信≫ 3:司書業務 [1]
	fn2.5	| 書籍情報
	fn3.3	検索結果を利用者に伝える

3 fn1.1.1 司書業務

1	fn3.3	書籍検索 ≫詳細化≫ 4:書籍検索
2	fn3.3	予約受付
3	fn3.3	リクエスト受付
4	fn3.3	貸し出し
5	fn3.3	返却
6	fn3.3	本を追加
7	fn3.3	本を破棄
8	fn3.3	催促
9	fn3.3	利用者を追加
10	fn4.3	≪被集約≪ 5:書籍情報取得

4 fn1.3 書籍検索 ≪詳細化≪ 3:司書業務 [1]

1	fn3.3	書籍検索メッセージを受ける ≪受信≪ 1:司書 [2]
	fn2.5	| 書籍情報
	fn3.3	探したい本の書籍情報を問う ≫発信≫ 5:書籍情報取得 [6]
	fn2.5	| タイトル，著者，訳者，出版社，初版出版日
	fn3.3	検索結果を利用者に伝える

5 fn1.4 書籍情報取得

1	fn2.8	タイトル
2	fn2.8	著者
3	fn2.8	訳者
4	fn2.8	出版社
5	fn2.8	初版出版日
6	fn3.3	書籍情報取得 ≪受信≪ 3:司書業務 [1]
	fn2.5	| タイトル，著者，訳者，出版社，初版出版日
	fn3.3	入力画面を用意する ≫発信≫ 6:書籍情報入力画面 [8]
	fn3.1	入力情報制御 ≫発信≫ 5:書籍情報取得 [7]
	fn3.1	書籍情報伝達 ≫発信≫ 5:書籍情報取得 [8]
7	fn3.1	入力情報制御
8	fn3.2	入力情報伝達
	fn3.3	モデルに情報を返す
9	fn4.3	≫集約≫ 6:書籍情報入力画面
10	fn4.3	≫集約≫ 3:司書業務

6 fn1.7.2 書籍情報入力画面

1	fn6.2.6	タイトル入力フィールド
	fn2.8	リンク情報 5:書籍情報取得 [1]
	fn2.1	座標＝{10, 10, 50, 30}
	fn2.1	初期値＝""
2	fn6.2.6	著者入力フィールド
	fn2.8	リンク情報 5:書籍情報取得 [2]
	fn2.1	初期値＝""
	fn2.1	座標＝{ 10, 10, 50, 30}
3	fn6.2.6	訳者入力フィールド
	fn2.8	リンク情報 5:書籍情報取得 [3]
	fn2.1	初期値＝""
	fn2.1	座標＝{10,10,50,30}
4	fn6.2.6	出版社入力フィールド
	fn2.8	リンク情報 5:書籍情報取得 [4]
	fn2.1	初期値＝""
	fn2.1	座標＝{10,10,50,30}
5	fn6.2.6	初版出版日入力フィールド
	fn2.8	リンク情報 5:書籍情報取得 [5]
	fn2.1	初期値＝""
	fn2.1	座標＝{10,10,50,30}
6	fn6.2.1	OK ボタン
	fn2.1	表示＝"OK"
	fn2.1	座標＝{10,10,50,30}
7	fn6.2.9	入力パネル枠
	fn2.1	座標＝{20, 30, 420, 430}
8	fn3.2	パネル表示 《受信《 37:書籍情報取得 [6]
	fn3.1	fn6.2.*の要素を用意する
9	fn4.3	≪被集約≪ 5:書籍情報取得
10	fn4.3	≪被集約≪ 1:司書

4 OCDJ記述支援ツール

4.1 記述支援手順

OCDJ記述を作成する段階において，記述者にアクタとビュー，コントローラと言った概念を理解してもらって，OCDJ記述を直接行ってもらうようなことは考えていない．まずは，システム化対象モデルをOONJで記述してもらい，その記述を支援ツールを用いOCDJに変換する手順を考えた．

OCDJで直接記述しないことを想定した理由として，ソフトウェア開発に精通していないドメインユーザである記述者にビュー，コントローラの概念を理解してもらうことは難しいと考えたからである．

一方，OONJは対象モデルの存在するモノをフレームと捉え，そのフレームに捉えたモノの属性や振舞いを記述していけばよい．人間の認識した結果をそのまま自然日本語で記述していけばよいので，ドメインユーザにとって敷居が低いと考えられる．

4.2 ツールの要求事項

OONJ記述をOCDJ記述に変換するにあたり，OONJ記述中からアクタとなりうるものを指定してもらえば，アクタとそうでないものとの間にインタフェースが存在することがわかる．インタフェースがあれば，ビューとコントローラが存在する．ツールとしては，そのインタフェースにてどのような情報のやり取りを行うかを理解するのは難しいが，インタフェースが存在するということは理解できる．後は，ツールは「ここにインタフェースが存在し，そのやり取りを記述して下さい」ということを伝えればよい．

他に支援すべきことは，アクタは二種類に分類でき，それぞれが異なった記述を必要とするので，それを支援することが必要となる．

4.3 アクタの分類

記述対象として図書館システムを用い，アクタの分類について説明する．記述者は，図書館システムにて次のモノを捉えたとする．

- 利用者
- 本
- 司書
- 貸し出し記録ノート

- 書籍台帳

ここで，アクタとなりうるモノは「利用者」と「司書」である.「利用者」は「司書」に本の貸出し・本の返却そして本の検索を依頼するものとする.「司書」は「利用者」より依頼された内容ごとに,「書籍台帳」や「貸し出し記録ノート」を参照したり書き込んだりする．この場合，システム化したい部分は司書の業務となる.「司書」が行う作業をコンピュータ上で再現することが開発の目的となる.

では,「利用者」はシステム化する必要がないからOCDJで記述する場合は考慮する必要はない.また「利用者」の作業は「司書」を介して行うので，この点でも考慮する必要はない．しかし,「利用者」は「司書」が貸し出しを行う際に,「利用者」が図書館に登録されているかどうかを知っておく必要がある.「利用者」の振舞いをシステム化する必要はないが，システムは「利用者」の存在を知っておく必要がある.

以上の「司書」と「利用者」の性質より，アクタとなるものを二つに分類できた．この論文では，それぞれの性質のアクタをバウンダリ型と分身型と呼ぶこととする．バウンダリ型のアクタはその振舞いをシステム化する必要があり，分身型のアクタはその存在をシステムに留めておく必要がある．図5にその様子を示す.

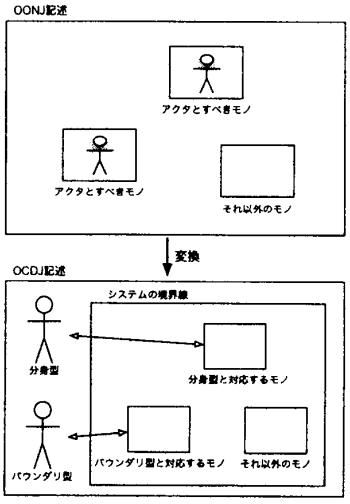

図5: OONJからOCDJの変換

4.4 ツールの機能

前節より，ツールはOONJ中の記述よりモノを列挙し，その一覧からアクタとなるものを選択してもらう，さらに

- アクタ
  - バウンダリ型のアクタ
  - 分身型のアクタ

と言った分類を要求する.

バウンダリ型のアクタはシステム内部と接するのでその接点にインタフェースが存在する．支援ツールはビューとコントローラの外枠を追加し，記述者にその中身を記述してもらう．一方，分身型のアクタはその存在を示すコピーをシステム内部に残し外部に出す．図6にその様子を示す.

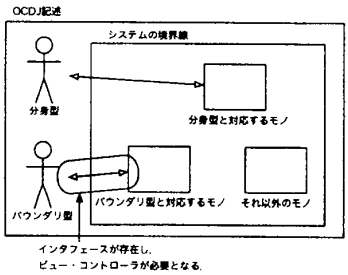

図6: インタフェースの存在

OONJ記述に新たなモノを追加することにより，オブジェクト間の振舞いを修正する必要がある．例えば，図7を変更の前のOONJとし「A」をバウンダリ型アクタ「C」をアクタ(バウンダリか分身のどちらでも可)そして「B」をそのまま残すモノとする．矢印は相互振舞いを示している.

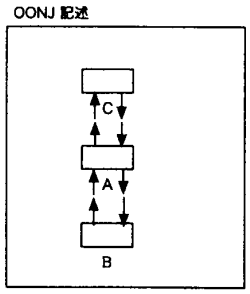

図7: OONJ記述の振舞い

図7をOCDJに変換する際,「A」はバウンダリ型アクタなので「A」をシステム内部に位置させ「A」のコピーを「A'」としシステム外部に配置する．OCDJ記述では，図8の様になる．このような相互振舞い関係を支援ツールは修復する.

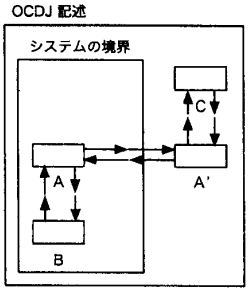

図 8: OCDJ 記述の振舞い

5 UML 図作成支援ツール

この章では，OCDJ から UML 図に変換するツールの機能について述べる．

5.1 機能

今回作成したツールは OCDJ 記述を入力として与えることによって，UML 図を自動で作成する機能を有している．入力される OCDJ 記述は UML 図を作成するのに必要な情報が書かれていることを仮定しており，必要な情報がなかった場合，その部分を無視することによりなるべく多くの UML 図を作成するようになっている．

作成された UML 図の情報は XMI 形式で保存される．しかし，XMI では座標情報が保持できないため，UML のそれぞれの図や文字のレイアウトが一意に定まらない．よって，OCDJ 記述から作成された UML 図の情報に加えて，座標データも加えた独自の XMI 形式のファイルに保存する．この規格は，XMI 単体としても読み込みが可能なものである．座標情報を含めた UML 図を作成する際には，図や文字が重ならないようになっており，見やすい図を出力できるようになっている．

この独自規格の XMI ファイルを，オブジェクト指向 CASE ツールである IIOSS を読み込み対応にしたものを本研究にて作成したため，そのソフトで読み込み，使用することができる．また，ブラウザでも読み込めるように，この独自規格の XMI ファイルを HTML と図にコンバートするツールも作成した．このツールに通すことによってブラウザでも UML 図が表示できるようになる．さらに，標準の XMI 形式としても読み込めるようにすることによって，XMI の読み込みに対応している UML ツールで，UML 図として読み込むことができる．

処理の流れを図 9 に表す．

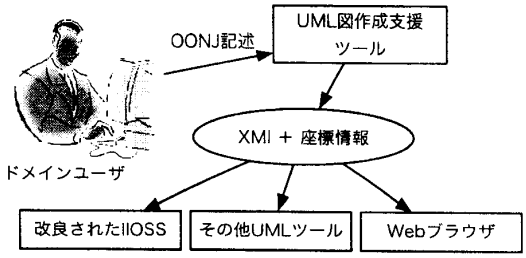

図 9: OCDJ 記述利用の流れ

本ツールは，Java にて実装した．

5.2 変換例

図 11 に示す図書館システムの OCDJ 記述を入力として，本ツールにより変換し，出力された XMI ファイルを IIOSS で読み込み，表示したものを，図 11 に示す．

```
1 fn1.1 利用者
 1 fn4.2          》集約》  2 個人識別情報
 2 fn3.3 登録する      》詳細化》8 登録する
 3 fn3.3 探す          》詳細化》9 本を探す
 4 fn3.3 予約する      》詳細化》10 本を予約する
 5 fn3.3 リクエストする   》詳細化》11 本をリクエストする
 6 fn3.3 借りる        》詳細化》12 本を借りる
 7 fn3.3 返す          》詳細化》13 本を返す
 8 fn3.3 催促される      》詳細化》14 返却を催促される
 9 fn4.2          甘集約《6 貸し出し記録
10 fn4.2       》集約》  7 予約情報
11 fn2.1 利用者カード
12 fn3.3 見つける      》詳細化》15 本を見つける
2 fn1.1 個人識別情報
 1 fn2 名前
 2 fn2 性別
 3 fn2 生年月日
 4 fn2 住所
 5 fn2 電話番号
 6 fn4.2       《集約《1 利用者
 7 fn4.2       《集約《2 司書
```

図 10: 図書館システムの OCDJ 記述

XMI としての出力結果は，XMI の文法通りに正しく出力されており，OCDJ で記述した意味もファセットナンバに基づく部分は適切に UML の情報に対応づけることができた．

UML 図としての出力結果は，IIOSS や eclipse などの UML ツールで XMI として読み込んだものに比べ，本研究で改良した IIOSS で独自規格の XMI を読み込んだ場合，図要素や文字の重なりを回避しており，初期配置としては見やすいものにすることができた．

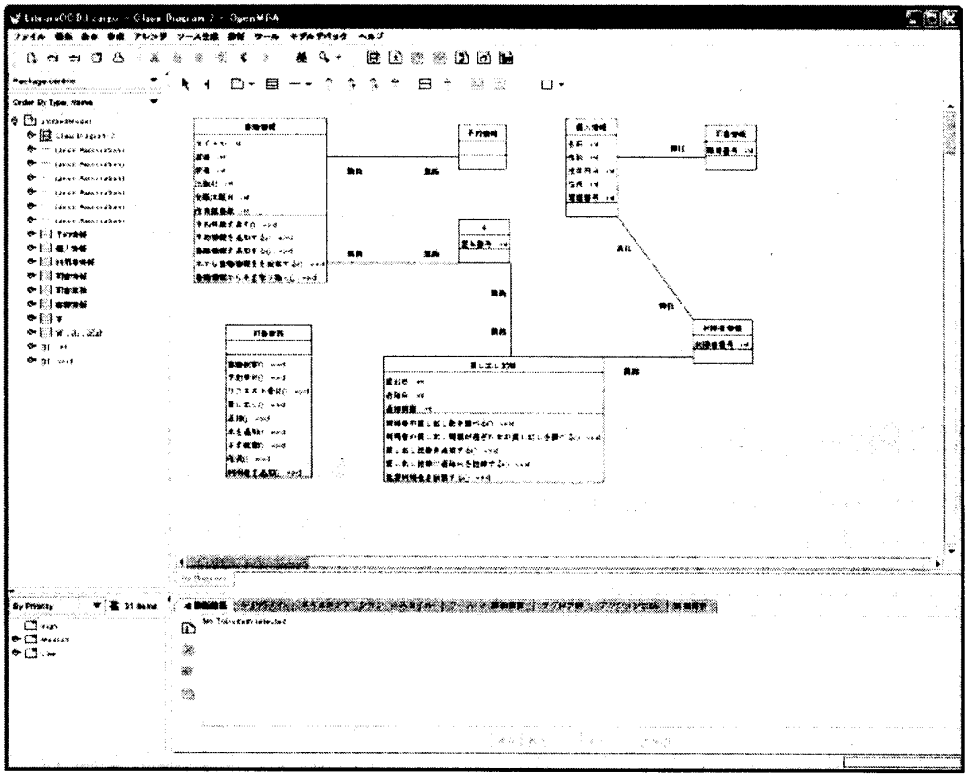

図 11: OCDJ 記述より生成した UML 図

6 まとめ

本研究では，以前より提案されていた OCDJ の仕様を具体化し，ドメインユーザが OONJ 記述より OCDJ 記述を生成する支援ツールを開発した．更に，UML 図生成のためのモデル情報として OCDJ が利用できることを示した．

今後の課題として，本研究で示したアクタの概念をドメインユーザが理解し OCDJ 記述を行えるかの検証を上げることができる．また，OCDJ 記述からの UML 図の生成に関しては，図要素や文字の重なりは起こらないようにすることができたが，さらに見やすい配置とする工夫を検討する必要がある．

参考文献

[1] 上田賀一，畠山正行，加藤木和夫: オブジェクト指向計算記述日本語 OCDJ の設計，電子情報通信学会，技術報告 (SS)，Vol.101，No.629 (2002.01).

[2] 畠山正行，加藤木和夫，上田賀一: オブジェクト指向自然日本語 OONJ の設計と評価，情報処理学会，研究報告 (SE)，Vol.2001，No.31 (2001.03).

[3] 畠山正行: オブジェクト指向自然日本語記述言語 OONJ の設計とその記述例，情報処理学会，研究報告 (SE)，Vol.2004，No.87 (2004.08).

[4] 磯田定宏: 実世界モデリング有害論-オブジェクト指向モデリング技法の解明，電子情報通信学会論文誌，Vol.J83-D1 No.9 pp.946-959 (2000.09).

[5] 銅島康，上田賀一: UML 記述に UI 設計を連携させた開発支援ツールの提案，情報処理学会，研究報告 (SE) (2005.03).

[6] http://www.iioss.org/, The IIOSS Project

[7] http://www.eclipse.org/, eclipse.org

[8] http://www.openmda.org/, OpenMDA.org