オントロジーを用いたソフトウェア資産再利用法の検討

A study of how to reuse the deliverables of software development with ontology

堀内信吾　井上正輝　井上貴司　松雪康巳　山村哲哉
HORIUCHI Shingo, INOUE Masateru, INOUE Takashi, MATSUYUKI Yasumi, YAMAMURA Tetsuya

日本電信電話株式会社 アクセスサービスシステム研究所
NTT Access Network Service Systems Laboratories

〒 261-0023 千葉県千葉市美浜区中瀬１－６
1-6 Nakase Mihama-ku Chiba-shi, Chiba 261-0023 Japan

Abstract: Recently, network services have been increasing and we think it is very important to develop many OSSs (Operation Support Systems) to support these services. We think it will be possible to develop OSS efficiently by the reuse of the deliverables in the past developments. Using ontologies in the new OSS development, we could find models similar to new ones from the deliverables in the past developments. We propose how to calculate the similarity between ontologies.

1　はじめに

通信業界では，通信サービスや通信設備を管理・制御する一連のソフトウェアシステムとしてオペレーションサポートシステム (OSS：Operation Support System) の研究開発が進められている [1].
近年では，サービス競争の激化に伴い，多様なサービスを短い期間で提供する必要があり，この OSS 自体も低コスト且つ短期間で開発することが求められている．

OSS のようなソフトウェアの開発は大規模になることが多く，一般的なソフトウェア開発手法に準じて，要求/分析/設計/実装/テストのフェーズからなる．しかしながら，ソフトウェア開発を効率的に進め且つソフトウェアの品質を向上させる上でいくつかの問題点がある．

多くのソフトウェア開発においては，設計/実装フェーズにおいても顧客からの要求条件が変更されることがあり，分析作業のやり直しなどの手戻りが発生することがある．また，実際にソフトウェアを利用する顧客の要望が要求フェーズ完了時点であっても設計者に対して把握されていないこともある．このような問題がありソフトウェアを効率的に開発することは難しい．一方，既存のソフトウェア開発の資産を新規開発に再利用することで開発の稼動を減らし効率的な開発が実現できると考えている．しかしながら，既存の資産を再利用する方法は確立されていない．

近年，ソフトウェア開発の上流工程において UML (Unified Modeling Language)[2] が標準的に利用されている．UML は顧客の要求や外部システムとの関係などを視覚的に表現するので，複雑なシステムにおいて要求者/設計者/実装者の相互理解が容易である．また，この UML を用いた開発プロセスの一つとして UP (Unified Process)[3] がある．この開発プロセスにおいては主に表 1 に示すようなソフトウェア資産が作成される．この開発プロセスを実践しても，ソフトウェア資産である UML で記述されたダイアグラムであるモデルの管理が難しく，効率的な開発が実現されているとはいえない．

表 1 ソフトウェア資産

開発フェーズ	ソフトウェア資産
要求	要求仕様書
分析	概念モデル図，ユースケース
設計	クラス図，シーケンス図
実装/テスト	コンポーネント

そこで，このようなソフトウェア開発における問題点を克服するために MDA (Model Driven Architecture)[4] に基づいた統合開発環境の検討が進められている．MDA により，実装非依存のモデルと実装依存のモデルを分離し，モデル間のトレーサビリティを確保することが可能となる．しかしながら，設計フェーズの後半から実装フェーズにかけての下流工程に対する取組みが多く，要求フェーズから設計フェーズにかけての上流工程から，下流工程まで一貫してサポートするものはほとんどない．

本稿では，UML/UP を前提とし，OSS をはじめとするソフトウェア開発において，上流工程から下流工程における既存のソフトウェア資産を再利用する方法を考え，開発を効率化し OSS の品質を向上させることを目指す．

一般的なソフトウェア資産はシステムの機能や管理す

べき情報等を表現している．しかしながら，何故そのような資産を作成したかという DR (Design Rationale)，資産の前提条件などの暗黙知は表現されていない．その結果，ソフトウェア資産に含まれる各モデルを新規の OSS 開発時に参考モデルとして再利用する場合に再利用に適しているかどうかの判断が困難になる．

ソフトウェアの資産を再利用する為には暗黙知を理解することが重要となる．暗黙知となっている概念を開発者が相互に理解するためには，概念に対して同一性の判定を行なうことが必要となる．そこで，暗黙知を体系化し，概念特定に役立つオントロジー技術の利用を考え，UML のシーケンス図を表現するタスクオントロジー間の類似性を定量的に計算する方法を提案する．

2 オントロジー

オントロジー技術は，工学のドメインにおいて無形の知識を有形化するのに役立つ．オントロジー技術には様々なものがある [5]．

OSS では，光アクセスサービスを提供する場合，図 1 のような通信装置や所内外の光ファイバやスプリッタなどの通信媒体の管理や，通信装置や通信媒体で構成されるネットワークの回線開通や帯域制御など行う．その為，通信装置や通信媒体を物理構成や論理構成に基づいて体系化することと，回線開通や帯域制御を実現する為のシステムの振る舞いを体系化し明確化することが OSS 開発におけるモデルを正確に理解する上で必要となる．以上から，前者の体系化の為にドメインオントロジー，後者の体系化の為にタスクオントロジーが有効であると考えられる [6]．

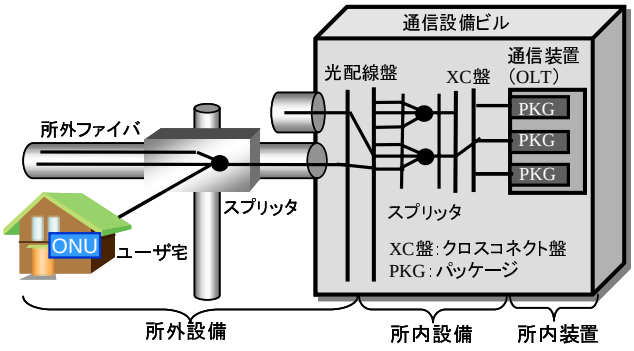


図 1 光アクセスサービスにおける設備/装置

2.1 タスクオントロジー

オントロジーを構築する為には，具体的なオントロジー構築方法を明確にする必要がある．その為，振る舞いを表現するタスクオントロジーを構築する方法として，AFM (Activity-First Method)[7] を参考にする．AFM では，先ずマニュアルなどの文章から動詞であるタスクアクティビティを抽出し，タスクアクティビティの概念化/一般化を行い，タスクアクティビティを時系列的に並べたタスクフローの構造化を行うことで is-achieved-by 構造を持つタスクオントロジーを構築する．AFM は漏れのないフローを記述できるという点で振る舞いを定義することに適した手法である．

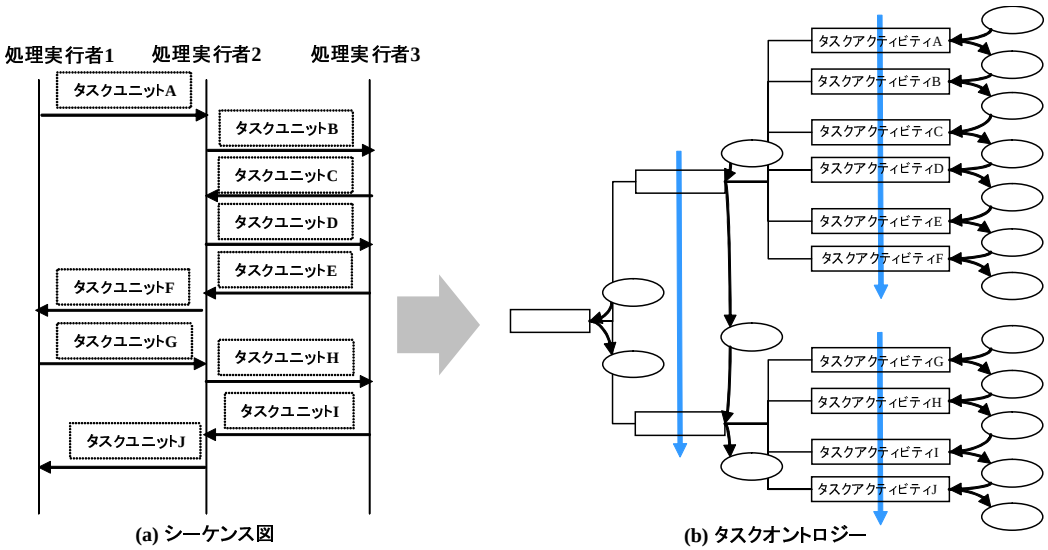
OSS をはじめとするソフトウェア開発において，動的な振る舞いを表現するモデルとしてユースケース，シーケンス図などがあり，シーケンス図は特に多くの開発において，システム全体の振る舞いを記述するモデルとして利用されている．従って，シーケンス図を基にしてタスクオントロジーを構築する必要がある．また，ソフトウェア開発者は AFM をはじめとするオントロジー技術に関する知識を持っていない為，機械的にオントロジーを構築することを考える．これらのことから，公開された AFM に対して，下記の点について手法を一部変更した．

- シーケンス図からタスクオントロジーを構築する為に，シーケンス図の構造を利用しタスクフローの構造化を行なう．
- 機械的にラベル統一を進める為に，モデルで利用するタスクアクティビティを事前にリスト化しておく．

これらの変更点について詳しく述べる．

(1) シーケンス図への対応
シーケンス図の構造をタスクの構造化に利用する．シーケンス図において，ある処理実行者からの働きかけによって一連の処理が実行されると考え，一連の処理をグループ化し図 2 のようにタスク構造を決定する．例えば，図 2 の (a) のように処理実行者 1 のタスクユニット A をトリガーに，タスクユニット F までの処理が実行されるシーケンスにおいて，タスクユニット A から F の各々に対応するタスクアクティビティ A から F があるとする．このシーケンス図から，タスクアクティビティ A から F を図 2 の (b) のようにひとつの階層でグループ化する．このようにしてタスクフローの構造化を実現する．

(2) タスクアクティビティのリスト化
OSS 開発におけるソフトウェア資産は，システムの振る舞いに特有のタスクアクティビティを用いて表現されることが多い．このことより，ソフトウェア



(a) シーケンス図　　(b) タスクオントロジー

図 2 シーケンス図を用いたタスク構造の決定

資産が実際に記載されている設計書に現れるタスクアクティビティを整理し，新規の設計書を作成する際に限定されたタスクアクティビティを用いることで表現の揺らぎを抑制することができると考える．例えば,「表す」「表現する」「表示する」といった異なるラベルを「表す」に統一することである．
そこで，既存の OSS の設計書を基にタスクアクティビティを抽出/整理し，オントロジー構築で利用するタスクアクティビティをリスト化する．これにより，ロール定義を事前に行なうことができる．このようなリストはオントロジー構築において，同じ意味で用いられている複数のラベルを持つタスクアクティビティをひとつのタスクアクティビティに置き換える処理を機械的に進める為に必要となる．タスクアクティビティのリストの一部を表 2 に示す．

表 2 タスクアクティビティリスト (一部)

上位タスクアクティビティ	タスクアクティビティ
アクセスする	アクセスする
	接続する
扱う	扱う
圧縮する	圧縮する
誤る	誤る
	間違える
	ミスする
表す	表す
	表現する
	表示する

2.2 ドメインオントロジー

ドメインオントロジーは過去の設計書から概念抽出を行い，UML 設計の手法を参考に次のトップレベルのカテゴリーに基づいて概念を，オブジェクト候補，状態の候補，操作の候補，属性候補，属性値候補，その他の 6 つのカテゴリに分類する [8]．

概念間の関係は，ドメインエキスパートにより，TMF (TeleManagement Forum) の MTNM (Multi Technology Network Management)[9] などの標準勧告を参考に整理を行なう．標準化されているデータモデルとして STEP[10] におけるオントロジーがある．この STEP のものと比較して，MTNM などの勧告はオントロジーとして構築されたわけではない為，ドメインの概念がどのようにシステムの振る舞いと関連性が明記されていないという違いがある．

3 モデルの再利用の為のオントロジーの類似性

新規のソフトウェア開発において，ソフトウェア資産に含まれるモデルを再利用する為に，次のようなシナリオをサポートする設計支援システムにより再利用が促進されると考えている．

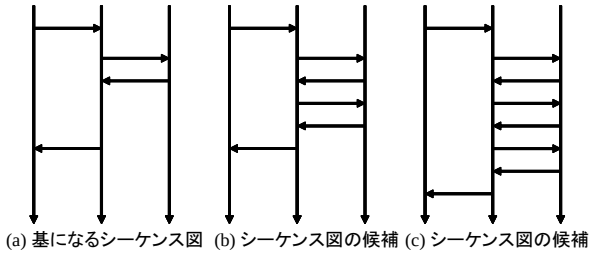


図 3 シーケンス図の類似性

表 3 類似度計算方法の比較

方法	計算量	拡張性	実現容易性	フローとしての類似性	備考
(1)	○	○	○	△	単語間の類似度が必要
(2)	○	△	△	△	編集操作の定量的な評価が必要
(3)	×	×	×	○	単語間の遷移確率やタスクフローのクラスタリングが必要

(1) 開発者が実現しようと考えている要求仕様に基づいたシーケンス図などのモデルの原案をシステムに投入する．

(2) システムでは入力されたシーケンス図に類似するシーケンス図などのモデルを開発者に提示する．

(3) 開発者はシステムから提示されたシーケンス図などのモデルを基に完全なシーケンス図などのモデルを作成する．

このシナリオによりシーケンス図などのモデルの再利用が可能となり，シーケンス図などのモデルを完成させるまでにかかるコストを削減できると考えている．この為には，新規開発中のシーケンス図などのモデルが既存のどのモデルと類似しているかを知る必要がある．本稿では，シーケンス図などの動的なモデルの類似性を知るために，タスクオントロジーの類似性に着目し，オントロジーの類似度の計算方法を提案する．

3.1 モデルの類似性

新規に開発するシステムのモデルが既存のシステムのモデルに如何に修正を加えて作成されるかに応じてモデルの類似性が決定されると考えている．例えば，図 3 のように (a) のシーケンス図に対して，(b) のシーケンス図と (c) のシーケンス図を考える．(b) のシーケンス図は (a) のシーケンス図に対して 2 つの処理が付け加えられている．一方，(c) のシーケンス図は (a) のシーケンス図に対して 4 つの処理が付け加えられている．このことから，(b) のシーケンス図のほうが (c) のシーケンス図より基になる (a) のシーケンス図に類似していると考える．但し，ここでのシーケンスの処理は，1 つの処理を複数の処理に分けて表現してしまうような表現上の揺らぎを取り除いた形になっているものとする．

3.2 オントロジーの類似度

モデルでの類似性をモデルの共通点であるとすると，その類似性を判断することはモデルがどのように作成/変更されるかに依存する．あるモデルから別のモデルに変更する際のコストを客観的に求めることは難しい．

そこで，モデルに対応するオントロジーで類似性を評価することを考える．本稿ではシーケンス図というモデルに対して，タスクオントロジーの類似度を計算する方法を提案する．本稿では，タスクオントロジーの類似度を，タスクオントロジーを構成するタスクフローの類似度として捉える．タスクフローの類似度を計算する方法として次の 3 つの方法を比較する．

(1) オントロジーに現れるタスクアクティビティに単語間の類似度を利用して DP (Dynamic Programming) によるマッチングを行いタスクフロー間の類似度を計る．

(2) 基にするタスクフローから比較するタスクフローに，動詞を入れ替えるなどの編集操作により変更する．その際の操作量によりタスクフロー間の類似度を計る．

(3) 確率モデルをタスクフローに当てはめ，基にするタスクフローから比較するタスクフローが最も起こり

やすいパラメータを決定することで，タスクフロー間の類似度を計る．

これらの方法を比較すると表 3 のようになる． この比較から，OSS の機能追加に対して柔軟に対応できるという拡張性の観点から本稿では 1 を選択する．

3.3 単語の類似度

タスクフロー間の類似度を求めるためには，タスクフローを構成する動詞間の類似度を知る必要がある．単語間の類似度を計算する方法として，EDR を用いて共通する意味概念の数により判断する方法 [11] や，既存の文章の中での共起する度合いを参考に計算する方法がある．前者の方法では，一般的な分野に関する語彙体系をベースとするのに対して，後者では OSS の分野の文章をベースとすることが可能である．ひとつの動詞に対する複数の意味概念の中から OSS の分野で用いられる意味概念だけに注目する必要がある．その為，本稿では後者の方法を用いる．実際に計算する為に，ツール [12] を用いた．

3.4 オントロジーの類似度計算方法

比較する 2 つのタスクオントロジーから、タスクオントロジーを構成する部分タスクフローを上位から順に選択する．部分タスクフローごとにオントロジーを次のように比較する．

まず，k 番目の部分タスクフローに含まれるタスクアクティビティである動詞間の類似度を求める．

次に，部分タスクフローの各々の最初の動詞の組からスタートして、類似度が最大になるように部分タスクフロー間のマッピングを行なう．部分タスクフローを $A = a_1, \cdots, a_M, B = b_1, \cdots, b_N$ とする．但し，a_i, b_j はそれぞれ部分タスクフロー A, B の i 番目,j 番目の動詞とする．これらのタスクフローを図 4 のように配置する．

動詞 $A-i$ と動詞 $B-j$ の組をノード (i, j) として，各ノードに対して動詞 $A-i$ と動詞 $B-j$ の類似度 w_{ij} を重みとして定義する．更に，ノード $(i-1, j)$ からノード (i, j) へとノード $(i, j-1)$ からノード (i, j) へのノード間の辺に対して重み $\tilde{w}_{ij} = 1 - w_{ij}$ を与え，2 つのタスクフローから有向グラフを考える．

このグラフに対して，ノード $(1, 1)$ からノード (M, N) への最短経路をダイクストラ法により求める．最短経路長を d_k とし，タスクフロー A とタスクフロー B の間の類似度 S_k を次式により求める．

$$S_k = \frac{M + N - 2 - d_k}{M + N - 2}. \quad (1)$$

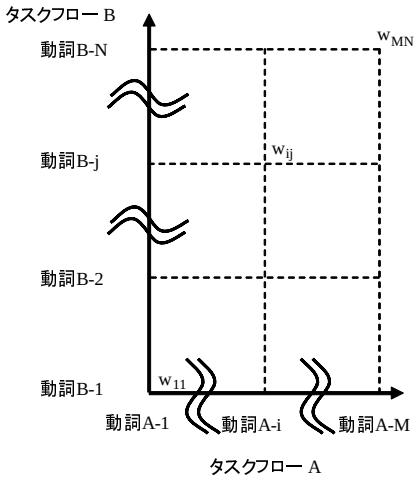


図 4 タスクフローの配置

このように求めることで，類似するタスクアクティビティによって構成されている 2 つのタスクフローに対する類似度はタスクアクティビティの構成を反映した値になると考えている．

4 適用例

光アクセスサービスを支えるシステムとして、GE-PON (Gigabit Ethernet Passive Optical Network) がある．GE-PON は，PON システムという図 5 のようなスター状のネットワークトポロジを有する．このようなネットワークシステムを管理する OSS の機能間の類似度を求める．

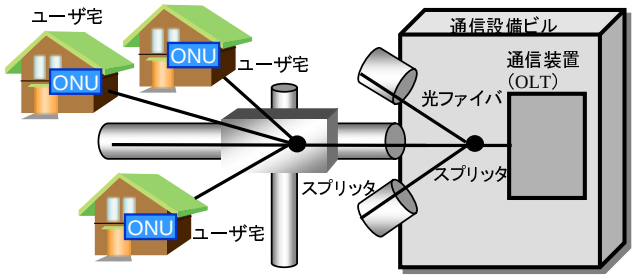


図 5 PON システム

そこで，この OSS の設計書に上述のオントロジー類似度計算方法を適用し，GE-PON の「OLT を登録する」と「PON-IF を登録する」という機能が類似しているかを本稿で提案する類似度計算方法を用いて判断する．それぞれの機能の is-achieved-by 構造によるタスクオントロジーから一部のタスクフローを抽出すると図 6 ようになる．これらのタスクフローを縦軸と横軸上に配置し，

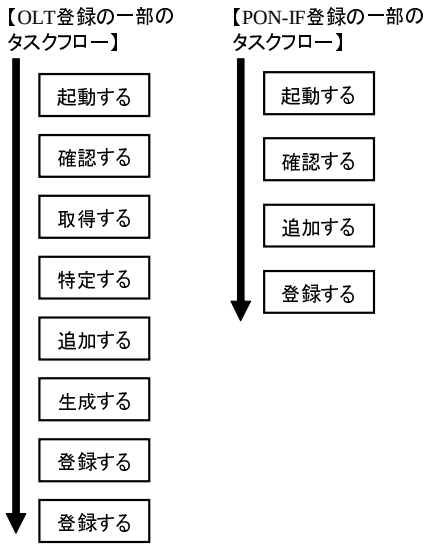


図 6 タスクフロー例

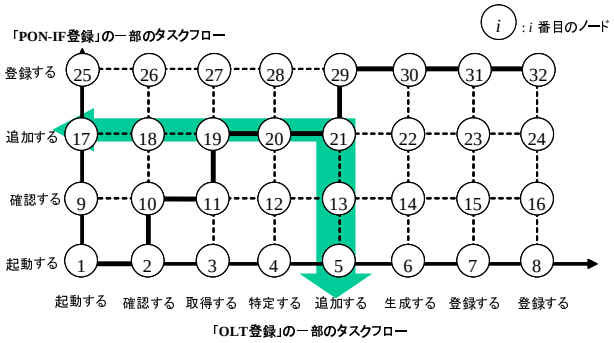


図 7 マッチングさせた経路の結果

各タスクアクティビティの組をノードとする有向グラフを定義し，有向グラフ上の経路を求めると図 7 のようになる．

以上より，2 つのタスクフローについて類似度計算を行なった．それぞれのタスクフローには 4 つの同一のタスクアクティビティが同じ順序で含まれていた．図 7 から，2 つの「追加する」というタスクアクティビティの組で表される 21 番目のノードのように「追加する」というタスクアクティビティはそれぞれマッチングされて太線で示した経路上にあることが分かった．同様に,「起動する」「確認する」「登録する」というタスクアクティビティについてもそれぞれ 1,10,31 番目のノードのところでマッチングしていることが分かる．このことからタスクフローの類似している部分を反映させた類似度を計算できると判断できる．

5 まとめ

タスクオントロジーを構成するタスクフロー間の類似度を定量的に求める方法を提案した．これにより，OSS などのソフトウェアシステムの機能タスクが類似しているかの判断が可能となった．

今後の課題は，ソフトウェア資産の再利用などの検証を通してタスクオントロジーの類似度が妥当であるかの検討を行なっていく．また，類似度計算方法や単語間の類似度の計算方法についてより効果的な方法の確立に努める．更に，ソフトウェア資産の再利用に向けて，オントロジーの OWL[13] による表現及び類似度の判定などを自動化するツールの構築も今後の課題である．

参考文献

[1] 江尻正義, “IP/e ビジネス管理における業務プロセスの共有化・コンポーネント化,” 電子情報通信学会誌, Vol. 87, No. 7, pp.570-576, 2004.

[2] OMG, “OMG Unified Modeling Language Specification, Version 1.5,” http://www.uml. org/.

[3] Ivar Jacobson, Grady Booch, James Rumbaugh, The Unified Software Development Process, Addison Wesley Longman, Inc., 1999.

[4] David S. Frankel, MDA モデル駆動アーキテクチャ, エスアイビー・アクセス, 2003.

[5] 溝口理一郎, 知の科学 オントロジー工学, オーム社, 東京, 2005.

[6] 堀内信吾, 井上貴司, 松雪康巳, 山村哲哉, “オントロジーを用いた OSS 設計法の提案,” 信学技報 TM2004-65, 2004.

[7] 石川誠一, 久保成毅, 古崎晃司, 來村徳信, 溝口理一郎, “タスク・ドメインロールに基づくオントロジー構築ガイドシステムの設計と開発,” 人工知能学会論文誌, Vol. 17, No. 5, pp.585-597, 2002.

[8] UML PRESS, 技術評論社, Vol. 3, 2004.

[9] TeleManagement Forum, MTNM v. 3.0, http://www.tmforum.org/.

[10] 伊藤聡, 山口高平, “オントロジーとエンタープライズモデル,” 人工知能学会誌, Vol. 13, No. 6, 1998.

[11] 崔進, 小松英二, 安原宏, “EDR 電子化辞書を用いた単語類似度計算法,” 情報処理学会 研究報告「自然言語処理」, No. 093, 1992.

[12] NTT-AT, ドキュメントマイニングシステム, http://www.ntt-at.co.jp/product/docmsys/ index.html.

[13] W3C, “OWL Web Ontology Language Reference,” http://www.w3.org/TR/2004/REC-owl-ref-20040210/.