リファインメントパターンを利用した KAOS ゴールモデルから BPMN モデルへの変換

堀田 大貴　本田 耕三　平山 秀昭　清 雄一
中川 博之　田原 康之　大須賀 昭彦

ソフトウェア開発において，ビジネスプロセスのモデル化は重要である．適切なビジネスプロセスモデルを構築するためには，ステークホルダの要求をモデルへ反映する必要がある．そこで本研究ではリファインメントパターンに基づいて構築されたゴールモデルをビジネスプロセスモデルへ変換する手法を提案する．リファインメントパターンに基づいたゴール分解の関係をビジネスプロセスモデルへ変換するためのルール及びアルゴリズムを示す．本手法によって，リファインメントパターンに基づいてステークホルダの要求を形式的に捉えることで，ビジネスプロセスモデル構築に役立てることができる．複数の事例に本手法を適用すること及び，モデル検査による検証によって本手法の有効性を評価する．

In the software development, modeling business process is important. In constructing business process model appropriately, stakeholder's requirements should be reflected in the model. Therefore, in this research, we propose transformation approach from goal models using refinement pattern to business process models. It is denoted that rules of transformation and algorithm. Using our approach supports constructing business process models by specifying stakeholder's requirements formally using refinement patterns. We evaluate the effectiveness of our approach through applying our approach for a number of cases and using model-checking techniques.

1 はじめに

近年，ビジネス環境は，コスト，内部統制，サービス改善等様々な理由で変化する．そのような状況下では，ステークホルダの要求を的確に捉え，変化に対応する必要がある．更に，ビジネス活動において，情報システムは幅広く用いられており，ビジネスプロセスにおいて必要とされるソフトウェアの開発が求められている．

変化への対応や，システム，ソフトウェアに対する要求を効率的に分析するためには，ゴール指向要求分析法や，ビジネスプロセスモデルを利用することができる．ゴール指向要求分析法は要求をゴールモデルとして表現し，システムやソフトウェアに対する個別の要求を表すゴールの分解を繰り返すことによって，要求とその達成手段を詳細化していく手法である．ゴール指向要求分析法の中でも，KAOS [30] はリファインメントパターンを用いてゴール分解を系統的・論理的に行えることが知られている．一方，ビジネスプロセスモデルはビジネスプロセスの流れや条件分岐を表すモデルであり，その中でも BPMN が普及している．Vara らはゴールモデルとビジネスプロセスモデルを併用することによって，情報システムの要求を獲

Transformation of KAOS Goal Models to BPMN Models Using Refinement Patterns.

Hiroki Horita, Kozo Honda, Hideaki Hirayama, Yuichi Sei, Yasuyuki Tahara, Akihiko Ohsuga, 電気通信大学大学院情報システム学研究科, Graduate School of Information Systems, The University of Electro-Communications.

Hiroyuki Nakagawa, 大阪大学大学院情報科学研究科, Graduate School of Information Science and Technology, Osaka University.

コンピュータソフトウェア, Vol.32, No.4 (2015), pp.141–160.
[研究論文] 2014 年 2 月 10 日受付.
本論文は第 20 回ソフトウェア工学の基礎ワークショップ (FOSE2013) の発表論文を発展させたものである．

得する手法を提案している [21]．ゴールモデルによってなぜゴール (目的) を達成しなければならないのかを分析し，ビジネスプロセスモデルを用いて何をどのように行うか分析するという両モデリング手法の利点を活かすことによって多角的に要求を捉えることができる．また，Vara らはシステム開発においてゴールモデルやビジネスプロセスモデルを用いる際，どのような場合にそれらを用いるべきなのか論じている．Vara らは組織が明確な作業手順を定めていない場合はビジネスプロセスモデルよりも先にゴールモデルを作るほうが良いと主張している [21]．その理由として，(1) ビジネスプロセスを一から作ることの困難性，(2) ゴールモデルを作成することによってシステム要求が組織の目標を満たしているか評価できる点，(3) ゴールモデルが作業手順の可変性分析に利用しやすい点を挙げている．

そこで，本研究では，ステークホルダの要求を，的確にビジネスプロセスモデルへ反映するために，ゴールモデルにより記述されたステークホルダの要求を，BPMN モデルへ変換する手法を提案する．ゴールモデルをビジネスプロセスモデルへ変換する研究は複数存在する．しかし，リファインメントパターンを利用したゴールモデルから BPMN モデルへの変換方法はいまだ論じられていない．リファインメントパターンに基づいてゴール分解を行うことで，ゴールモデルにおける要素間の関係を論理的に記述することができ，その関係を BPMN モデルへ変換することで，BPMN モデルにおいて行われる処理の位置づけを明確化するとともに，詳細なモデルを構築できる．更に，近年，リファインメントパターンを利用したソフトウェア開発におけるモデル構築 [8] [13]，形式仕様記述 [22]，リスク評価方法 [6] が提案されており，ゴールモデルをリファインメントパターンに基づいて構築することがソフトウェア開発における様々な工程に活用できることが示されている．そのため，KAOS ゴールモデルにおけるゴール分解ガイドラインであるリファインメントパターン [30] に基づいて分解されたゴールを，BPMN モデルへ体系的に変換するための手法を提案する．6 種類のリファインメントパターンを用いて定義されるゴール分解の関係と BPMN モデルにおける要素との関係を表すルールを作成し，手続き的な変換アルゴリズムを用いることで，KAOS ゴールモデルを BPMN モデルへ変換する．本提案手法により， KAOS ゴールモデルによって系統的に記述された要求と，その実現手順の整合性の確保が可能となる． 複数のケーススタディを行うことで，KAOS モデルにおけるゴール間の関係を，適切に BPMN モデルへ反映できることを示した．

本稿では，まず 2 章で KAOS，BPMN について説明し，3 章で提案手法について説明する．4 章で提案手法のケーススタディを行う．5 章で考察を記述し，6 章で関連研究を示す．最後に 7 章でまとめと今後の課題を示す．

2 背景

この章では本研究において用いるゴール指向要求分析法である KAOS と BPMN について概説する．

2.1 KAOS

ゴール指向要求工学は要求工学の研究において潜在するシステム要求に関する動機の理解や適切な問題に対処するために，適切なシステムができているか確かめる手段として大きな注目を浴びている [14]．

ゴール指向要求工学では，ステークホルダの要求をゴールとして捉え，より具体的なゴールへと分解していくことで要求を導出する．正しい環境仮定において，正しいソフトウェア要求を得ることは，正しいソフトウェアを開発するための重要な条件である [29]．KAOS は代表的なゴール指向要求分析法の 1 つであり，ゴールモデル，オペレーションモデル，エージェントモデル，オブジェクトモデルの 4 モデルを活用し，要求を実現するための操作，操作を行うエージェント，入出力されるオブジェクトを記述し，要求分析を行う．本研究においてはゴールモデルのみを取り扱うため，ゴールモデルに関してのみ以下に記述する．

ゴールモデルの構築方法には親ゴールに HOW と問うことによって，どのような手段 (子ゴールの組み合わせ) によって親ゴールを達成するのか確認する方法と子ゴールに対して WHY と問うことによって子ゴールを何故達成しなければならないのか確認し親

ゴールを導出する方法がある [31]．本研究ではリファインメントパターンを KAOS ゴールモデルのゴール分解に利用しており，これは HOW と問うことによって手段となる子ゴールを導出する方法である．

ゴール分解には 2 種類ある．AND 分解は親ゴールを達成するためにはその子ゴールがすべて達成される必要がある分解法である．AND 分解に求められる性質として，完全性，整合性，最小性 [30] がある．

完全性は以下の式によって表される．

$$\{G_1, G_2, ..., G_m, DOM\} \models G \quad (1)$$

全ての子ゴール $(G_1, G_2, ..., G_m)$ と，全てのドメインプロパティやドメイン仮定 (DOM) が満たされると，親ゴール (G) も満たされる．ここで，ドメインプロパティとは，環境において常に成り立つことが分かっている性質である．ドメイン仮定とは，環境において成り立つことが望ましい性質である．

整合性は以下の式によって表される．

$$\{G_1, G_2, ..., G_m, DOM\} \not\models false \quad (2)$$

子ゴール，ドメインプロパティ，ドメイン仮定は互いに矛盾しないことを表す．

最小性は以下の式によって表される．

$$\{G_1, ..., G_{j-1}, G_{j+1}, ..., G_m, DOM\} \not\models G \quad (3)$$

子ゴールのうち 1 つでも欠落している場合，親ゴールの満足が保証されないことを表す．

リファインメントパターンに基づいてゴール分解を行うことは，ゴールモデルがこれらの性質を持つことに繋がる．

一方，OR 分解はその子ゴールのいずれか 1 つが達成されれば親ゴールが達成されるような分解法である．

ゴールを達成するためにはエージェントの協調が必要である．KAOS におけるエージェントとはソフトウェアコンポーネントやデバイス，人間などである．ゴールとエージェントを関連付けて記述することによって，ゴール達成責任を持つエージェントが明確化される．

2.2 リファインメントパターン

適切なゴール分解は難しい事が知られている．何らかの支援なくゴール分解を行うことは一般的に不十分になりがちであり，矛盾を含むことさえある [31]．ゴール分解を適切に行う手段としてフォーマルメソッドの利用が考えられる．しかし，フォーマルメソッドの利用は難しくコストが高い [10]．このような課題を解決するために KAOS では 6 種類のリファインメントパターンを用いることで，より適切なゴール分解を行うことができる．リファインメントパターンは [6], [20], [29] において時相論理を用いて形式的に記されている．各パターンは正しく，完全であることが証明されている [20]．パターンを再利用することはパターンの正しさ，完全さの再利用を伴う．そのためリファインメントパターン利用者は数学的な側面をあまり意識することなく適切なゴール分解を行うことができる [10]．以下において図 1 を用いつつ各リファインメントパターンを説明する．

2.2.1 Milestone-driven refinement pattern

Milestone-driven refinement pattern は，前提条件 C が成り立つ場合に，最終的にある条件 T を満足させる，すなわち時相論理では「(C⇒◇T)」と表されるゴールに対し，マイルストーン条件 M_1,… ,M_n を設定して，これらを順に満足させる，すなわち「C⇒◇M_1」，「M_1⇒◇M_2」,…,「M_{n-1}⇒◇M_n」，「M_n⇒◇T」に AND 分解するリファインメントパターンである．親ゴール達成 (ターゲット条件への到達) のためのマイルストーン条件がある場合に用いる．図 1 は時相論理によって記述されたマイルストーン条件が 1 つの場合における，Milestone-driven refinement pattern によるゴール分解である．

2.2.2 Decomposition-by-cases pattern

Decomposition-by-cases pattern はターゲット状態に到達する異なるケースがある場合に用いられるリファインメントパターンである．これらのケースは互いに素であり，状態空間全体を網羅している．Decomposition-by-cases pattern によって親ゴールに対し分解条件 CS が設定され，親ゴールを分解した子ゴールがそれぞれ「CS⇒◇T」,「¬CS⇒◇T」となる．分解条件 CS が成り立つ場合も，成り立たない場合もターゲット条件を達成できることがわかる．これらの子ゴールは互いに素であるためどちらかだけが

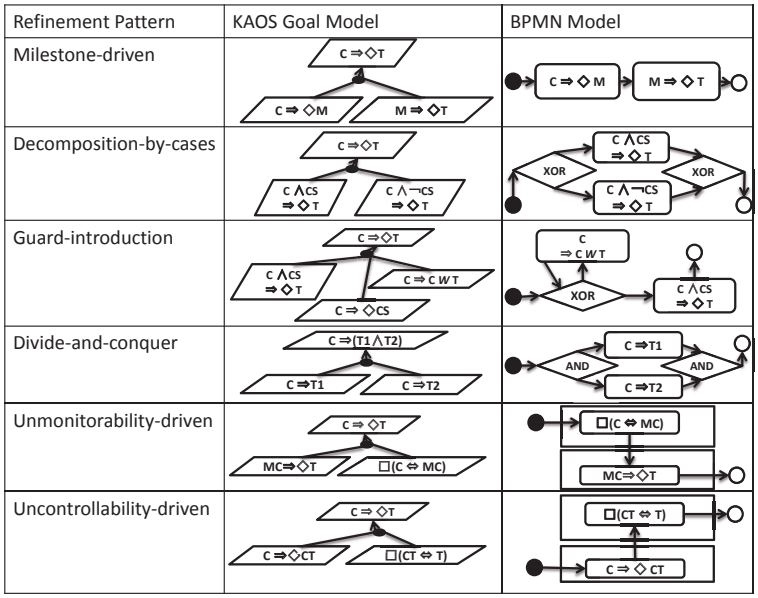


図 1　KAOS モデルから BPMN モデルへの変換パターン

達成される．

2.2.3　Guard-introduction pattern

Guard-introduction pattern は，ターゲット状態へ到達するために，ガード条件が必要な異なるケースへ場合分けされる場合に用いられるリファインメントパターンである．Guard-introduction pattern によって，親ゴールはガード条件達成ゴール，ガード条件ゴール，ガード条件未達成ゴールに分解される．

図 1 は時相論理に基づいて Guard-introduction pattern を記述したものである．「C∧CS⇒◇T」は current condition(C) とガード条件 (CS) が成り立つときそのうちターゲット条件 T が成り立つことを表している．「C⇒◇CS」は current condition(C) が成り立つときそのうちガード条件 (CS) が成り立つことを表している．「C⇒C WT」は T が成り立つまでは C が成り立ち続けることを表している．つまり，T が成り立っていなくても，C が成り立っていれば，そのうち CS が成り立って結果的に T も成り立つことを意味する．

2.2.4　Divide-and-conquer pattern

Divide-and-conquer pattern は，結合しているゴールがより小さいゴールに AND 分解できる場合に用いられるリファインメントパターンである．Divide-and-conquer pattern によって結合した状態にある親ゴールを子ゴールに分解される．これらはどちらも達成されなければいけないゴールである．

図 1 に時相論理によって Divide-and-conquer pattern によるゴール分解を記述している．親ゴールである「C⇒(T_1∧T_2)」が達成されるためには 1 つ目のターゲット条件 (T_1) と 2 つ目のターゲット条件 (T_2) が両方成り立つ必要がある．このゴールがそれぞれ 1 つ目のターゲット条件 (T_1) と 2 つ目のターゲット条件 (T_2) が成り立つためのゴールである．「C⇒T_1」と「C⇒T_2」が両方達成されれば，親ゴールである「C⇒(T_1∧T_2)」も達成される．

2.2.5　Unmonitorability-driven refinement pattern

Unmonitorability-driven refinement pattern は，ゴールの達成を担当しているエージェントでは，監視すべき状態を監視することができない場合に，そのような状況を解決するために用いられるリファインメントパターンである．親ゴールの達成を担当しているエージェントがゴール達成の条件を監視できない場合，ゴール達成の部分的な責任が監視できるエー

ジェントに割り振られる．Unmonitorability-driven refinement pattern によって親ゴールが子ゴール, 期待に分解される. 期待は Software-to-be の環境におけるエージェントの責任のもとにあるゴールである．その場合環境エージェント (センサエージェント) が期待, ソフトウェアエージェントが子ゴールの達成責任を負う．このようにゴールの責任が複数のエージェントに分割される．モニタリングした情報を活用してソフトウェアが動作することになる．

図 1 に時相論理によって Unmonitorability-driven refinement pattern によるゴール分解を記述している．親ゴールである「C⇒◇T」が達成されるためには子ゴールである,「□(C⇔MC)」と「MC⇒◇T」が両方達成される必要がある．ここで MC とは監視すべき対象が監視可能である状態を表す．C の成否と MC の成否が一致すれば (□(C⇔MC))，C が成り立てば MC も成り立つので，そのうち T が達成され，親ゴールである「C⇒◇T」が達成される．

2.2.6 Uncontrollability-driven refinement pattern

Uncontrollability-driven refinement pattern は親ゴールの達成を担当しているエージェントでは制御すべき状態を制御することができない場合において，そのような状況が解決されることを狙いとしたパターンである．親ゴールを Uncontrollability-driven refinement pattern によって分解した場合，ソフトウェアエージェントにより達成される子ゴールと，アクチュエータエージェント等の環境エージェントによって達成される期待になる．アクチュエータエージェントが動作するためにはソフトウェアエージェントによる指示が必要である．

図 1 に時相論理によって Uncontrollability-driven refinement pattern によるゴール分解を記述している．親ゴールである「C⇒◇T」が達成されるためには子ゴールである,「□(CT⇔T)」と「C⇒◇CT」が両方達成される必要がある．ここで CT はコントロールすべき対象がコントロール可能である状態を表す．C が達成されたとき，そのうち CT が達成され，CT の成否と T の成否が一致すれば (□(CT⇔T))，親ゴールである「C⇒◇T」が達成される．

2.3 BPMN

BPMN(Business Process Model and Notation) [27] は，ビジネスプロセスを表記する OMG 標準化仕様の表記法である．BPMN は主に処理や分岐を表すフローノードとそれらの順序関係であるシーケンスフローによって構成される．アクティビティによってビジネスプロセスにおいて行われる処理を表し，シーケンスフローによってアクティビティが行われる順序を定める．更に，アクティビティが記述されるスイムレーンを分けることで，処理を実行する主体ごとに行われるアクティビティを分けて記述できる．処理の分岐はゲートウェイによって記述する．AND Gateway は並列して行う処理を表し，XOR Gateway はいずれか 1 つの処理を実行することを表す．BPMN を用いればビジネスにおいていつ (when)，誰が (who)，何を (what) 行うのか記述することができる．一方で，なぜ (why) 処理を行うのか記述することはできない．この問題はなぜ (why) という側面を記述できるゴールモデルを用いることで，対処することができる．本手法により，リファインメントパターンによりゴールの関係を明確化することで，BPMN モデルにおけるアクティビティが実行されるべき根拠を明確にできる．

図 2 は本論文のケーススタディに用いている BPMN モデルを簡素化したものである．通報のあった場所へ救急車が到達するまでの流れを表している．まず,「通報を受ける」というアクティビティが実行され，その後,「現場へ向かう救急車を選択する」というアクティビティが実行される．アクティビティの間に記述されているシーケンスフローによって，順序関係を表す．その後，XOR Gateway によって，行うべき処理が分岐し，アクティビティである「ステーションにいる救急車が現場へ向かう」,「路上にいる救急車が現場へ向かう」のいずれかが実行される．

3 提案手法

本章では KAOS ゴールモデルにより記述された要求を BPMN モデルへ体系的に変換する手法を説明する．それによって BPMN モデル構築の難しさを軽減し，要求 (KAOS) とシステムプロセス (BPMN) の

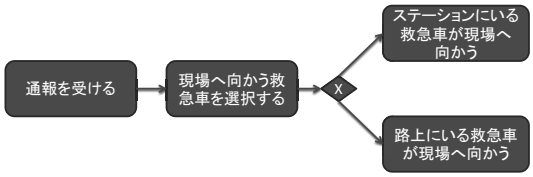


図 2　BPMN モデル記述例

整合性を確保する．3.1 において，本提案手法の適用対象を示す．どのようなゴールモデルに対して本手法が適用可能であるのかを記す．3.2 において，提案手法の概要を記す．3.3 において，複数のリファインメントパターンが用いられた KAOS ゴールモデルをどのように BPMN モデルへ段階的に変換していくのか記す．3.4 においてリファインメントパターンによって分解された KAOS ゴールモデルをいかに BPMN モデルの要素と対応付け，変換するのか記す．

3.1　提案手法適用対象

本研究では OR 分解による要求選択肢を選択済み，かつ，すべてのゴール分解がいずれかのリファインメントパターンを用いて行われているゴールモデルを対象としている．OR 分解による要求選択肢を選択済みとは，同一のゴールを達成するための手段が複数あり，選択可能であるとき，いずれかの達成手段が選択されていることを表す．あるゴールを達成するために手動で行う達成手段とソフトウェアを用いて自動で行う達成手段がある場合，手動で行うか，自動で行うか決定すると，OR 分解による要求選択肢を選択したことになる．

3.2　提案手法概要

図 3 に提案手法の概要を示す．図のゴールモデルは親ゴール A が Milestone-driven refinement pattern によって子ゴール B と C に分解されている．変換ルールによって子ゴール B，C はそれぞれアクティビティB，C に変換される．アクティビティ実行順序は milestone-driven refinement pattern によって示されるゴール達成順序と同じく B，C である．更に子ゴール B は Uncontrollability-driven refinement pattern

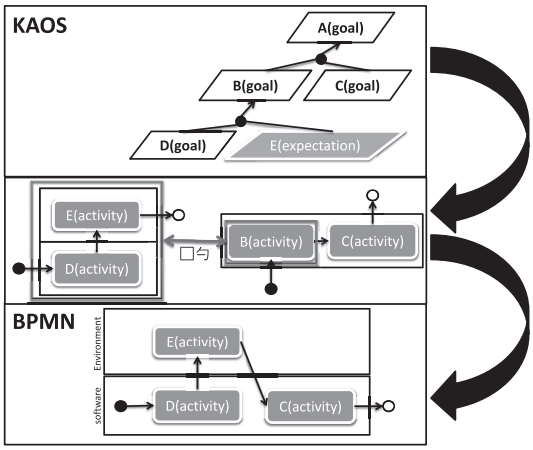


図 3　提案手法概要

によって子ゴール D と期待 E に分解されている．ゴールを達成する部分的な順番は D, E となる．この関係が変換ルールを用いて変換されアクティビティ実行順序は D, E だと示される．その後，ゴール B の達成はその子ゴール D, E を達成することに等しいため変換先の BPMN モデルにおいてアクティビティB と D, E を置換する．その結果アクティビティ実行順序は D → E → C となる．下の BPMN モデルはその様子を表している．アクティビティD，C と E を実行するエージェントは異なるため，エージェントごとにレーンを分けて記述する．本研究では KAOS ゴールモデルから BPMN モデルに変換するにあたってゴール分解をガイドするリファインメントパターンを利用している．図 1 は各リファインメントパターンによるゴール分解を BPMN モデルにおけるアクティビティ実行順序，分岐と対応付けたものである．

3.3　変換アルゴリズム

本節では複数のリファインメントパターンが利用されているゴールモデルを BPMN モデルへ段階的に変換するアルゴリズムを説明する．変換は図 4 のアルゴリズム T() にしたがって行う．まずトップゴールから開始して (図 4 の 1 行目)，各ゴールのリファインメントパターンに着目しルールに従って BPMN モデルへの変換を行う (図 4 の 4 行目)．変換されたアクティビティと対応するゴールが葉ゴールでない

```
1 T(): T1(G0:トップゴール)
2 T1(G): if (G: 葉ゴール) return G
3       else {
4         b = Gの分解に変換ルールを適用した結果;
5         for each G': Gの子ゴール
6               b中のG'をT1(G')で置換
7           }
```

図 4 変換アルゴリズム

ならば，再度ルールに従って変換を行った後アクティビティを置換する (図 4 の 5，6 行目)．ここで,「置換」とは「親ゴールに関するシーケンスフローの接続関係を子ゴールへ継承すること」を指す．これを変換されていないゴールがなくなるまで繰り返すことで KAOS ゴールモデルを BPMN モデルへ変換する．段階的変換の例として図 3 を用いて説明する．図 3 のゴールモデルのトップゴールは A であり，これは milestone-driven refinement pattern で B，C に分解されているため，変換すると B, C の順でアクティビティが実行される．次にゴール B は葉ゴールではないため更に変換を行う．ゴール B は Uncontrollability refinement pattern によって D，E に分解されており，ルールによって BPMN モデルへ変換したものをアクティビティB と置換する．その結果が図 3 の BPMN モデルである．アクティビティB がアクティビティD，E に置換されていることがわかる．

3.4 各リファインメントパターンを利用した変換

本節ではリファインメントパターン 6 種類を用いたゴール分解と BPMN モデルとの対応付けを図 1 を用いつつ説明する．なお，この節による変換は図 4 の 4 行〜6 行目に対応している．

3.4.1 Milestone-driven refinement pattern を用いた変換

図 1 のように KAOS ゴールモデルにおける子ゴールがそれぞれ BPMN モデルにおけるアクティビティへ変換される．アクティビティの実行順序は KAOS ゴールモデルにおいて定められたマイルストーン条件の達成順序と同一である．図 1 においてはマイルストーン条件が 1 つの場合であるため，マイルストーン条件達成のアクティビティとターゲット条件達成のアクティビティが順に実行される．

3.4.2 Decomposition-by-cases pattern を用いた変換

図 1 のように KAOS ゴールモデルにおける子ゴールがそれぞれ BPMN モデルにおけるアクティビティへ変換される．decomposition-by-cases pattern はケース条件の成否によって場合分けするリファインメントパターンであるため，アクティビティは XOR gateway によって分けられ，どちらかのみが実行される．

3.4.3 Guard-introduction pattern を用いた変換

図 1 のように KAOS ゴールモデルにおけるガード条件達成時ゴール (C∧CS⇒◊T)，ガード条件未達成時ゴール (C⇒C $\mathcal{W}$T) はそれぞれ BPMN モデルにおけるアクティビティへ変換される．これらのアクティビティの実行は XOR gateway を用いて表されるガード条件の成否によって分岐する．C 及び CS が成り立っていれば，ガード条件達成時ゴール (C∧CS⇒◊T) が変換されたアクティビティが実行され，C 及び CS が成り立っていない場合はガード条件未達成時ゴール (C⇒C $\mathcal{W}$T) が変換されたアクティビティが実行される．こちらの場合は，CS が成り立つまで XOR gateway の前へ戻り，CS の達成を待つ．

3.4.4 Divide-and-conquer pattern を用いた変換

図 1 のように KAOS ゴールモデルにおける子ゴールがそれぞれ BPMN モデルにおけるアクティビティへ変換される．Divide-and-conquer pattern は結合している状態にあるゴールをより詳細なゴールへ分解する場合に用いられるリファインメントパターンであり，分解された子ゴールは両方達成されなければならない．そのため，この関係を BPMN モデルにおいて AND gateway によって表す．すなわち，アクティビティは両方実行される．

3.4.5 Unmonitorability-driven refinement pattern を用いた変換

図 1 のように KAOS ゴールモデルにおける子ゴールがそれぞれ BPMN モデルにおけるアクティビティ

へ変換される．Unmonitorability-driven refinement pattern は監視すべき状態が監視できない状態を解決するために，ゴール達成の責任が分割されるリファインメントパターンであり，子ゴールを達成するエージェントはそれぞれ異なる．その関係を BPMN モデルにおいては環境エージェントとソフトウェアエージェントへレーンを分割することによって表す．ソフトウェアエージェントは環境エージェントによって監視された情報を利用して動作するため，環境エージェントによるアクティビティ(□(C⇔MC)) が実行された後に，ソフトウェアエージェントによるアクティビティ(MC⇒◊T) が実行される．

3.4.6 Uncontrollability-driven refinement pattern を用いた変換

図 1 のように KAOS ゴールモデルにおける子ゴールがそれぞれ BPMN モデルにおけるアクティビティへ変換される．Uncontrollability-driven refinement pattern は制御すべき状態が制御できない状態を解決するために，ゴール達成の責任が分割されるリファインメントパターンであり，子ゴールを達成するエージェントはそれぞれ異なる．その関係を BPMN モデルにおいては環境エージェントとソフトウェアエージェントへレーンを分割することによって表す．環境エージェントはソフトウェアエージェントから出力される情報を基に動作するため，ソフトウェアエージェントによるアクティビティ(C⇒◊T) が実行された後に，環境エージェントによるアクティビティ(□(CT⇔T)) が実行される．

4 ケーススタディ

提案手法の有効性を検証するためにケーススタディを行った．題材として Cailliau らの研究 [3] で用いられている London Ambulance Service (LAS) のゴールモデルと Cailliau らの研究 [4] [5] で用いられている barbados Car crash Management System (bCMS) のゴールモデル，卸–メーカー間取引におけるビジネス・プロセス・モデル調査研究所レポート [33] より，著者が作成したゴールモデルを用いる．

評価方法として，変換結果である BPMN モデルがゴールモデルにおいて想定されているシナリオを満たせるのか確認すること，及び，BPMN モデルに対してモデル検査を行うことによって，ゴールモデルにおいて想定される性質を満たせるのか形式的に検証することで評価を行う．後者においてはモデル検査ツールである SPIN を用いて，仕様記述言語である PROMELA によって記述された BPMN モデルが，LTL (Linear Temporal Logic) によって記述される性質を満たせるのか検証する．BPMN モデルを PROMELA 化するためには Janssen らの研究 [15] を少し修正して適用した．[15] ではアクティビティを表す変数の真偽を変更することによってアクティビティの実行を表す．検証に用いる性質はトップゴールを直接的に表す性質，及び重要な性質を用いる．

4.1 LAS 概要

LAS は目的地に救急車が到着することを目的とするシステムである．そのためにはまず，事故の状況等を救急隊員に知らせる必要がある．救急隊員は聞き取った情報を定められたフォームに落とし込む．その後，どの救急車が現場へ向かうのか決定される．現場へ向かう救急車は路上にいる場合やステーションにいる場合が考えられる．ステーションにいる救急車が現場へ向かう場合は，その救急車に乗り込む救急隊員に対して指示が伝えられ，プリントアウトされる．その後，救急車が現場へ到着することで目的が達成される．

4.2 LAS におけるケーススタディ

まず，LAS に関するゴールモデルの変換について記述する．このゴールモデルは [11] より構築されたものである．このゴールモデルは何らかの事故が起きた時，それを解決できるということに関するものであり，時相論理を用いて記述されている．

IncHappened⇒◊IncResolved というトップゴールは 7 回分解されている．1，3，4，5，7 回目の分解は Milestone-driven refinement pattern に基づいて分解されており，2，6 回目の分解は Decomposition-by-case pattern に基づいて分解されている．

このゴールモデルを提案手法を用いて BPMN モデルへ変換した．図 5 は変換結果である．アクティビ

図 5 LAS に関するゴールモデルを提案手法により変換した BPMN モデル

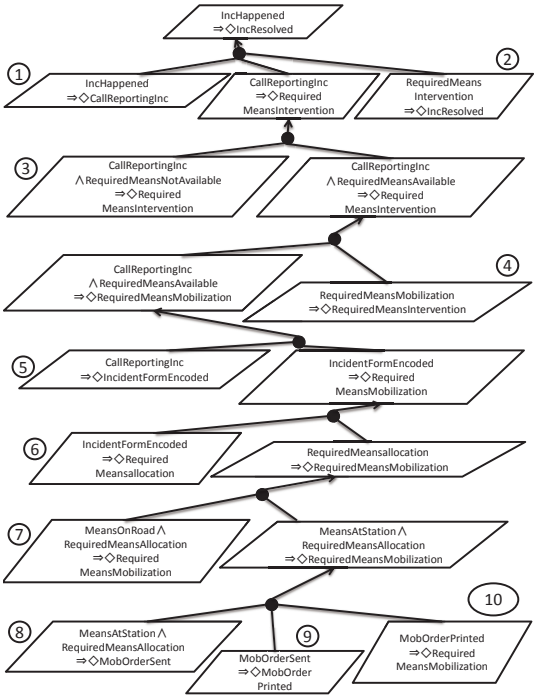


図 6 LAS に関するゴールモデル

ティの数字は図 6 のゴールモデルにおける数字と対応している．生成された BPMN モデルと LAS に関するシナリオである「要請があったとき, 救急車が路上にいる場合はそのまま現場に向かい, ステーションにいる場合は order に従って現場に向かう」との整合性を検証する．救急車が路上にいる場合は BPMN において「(1) 事故が起こり通報される，(5) 通報された内容をフォームに落とし込む，(6) フォームに落とし込まれたものを確認し必要な手段を選択する，(7) 路上にいる救急車が動員される，(4) 動員されることが決まった救急車が現場に向かう」の順でアクティビティを実行することで現場の状況が伝えられた路上を走っている救急車が現場に向かえることを示している．救急車がステーションにいる場合は BPMN において「(1) 事故が起こり通報される，(5) 通報された内容をフォームに落とし込む，(6) フォームに落とし込まれたものを確認し必要な手段を選択する，(8) ステーションにいる救急車が選ばれ，order が送られる，(9)order の内容が印刷される，(10)order に従った救急車が動員される，(4) 動員されることが決まった救急車が現場に向かう」の順でアクティビティを実行することで現場の状況を伝えられたステーションにいる救急車が order にしたがって現場に向かえることを示している．以上の結果から，提案手法はゴールモデルが表している事故が起きた時，救急車が現場へ向かい解決するまでの流れや条件分岐を BPMN モデルへ適切に反映していると考えられる．

次に，図 5 の BPMN モデルを PROMELA 化し，SPIN を用いてモデル検査を行った．検証には以下に示す 2 つの性質を用いた．

性質 1. 起きた事故はそのうち解決される．

この性質は以下の LTL 式で記述できる．

□((事故が通報される)⇒◇(起きた事故が解決される))

性質 2. ステーションにいる救急車は order に従わなければ現場へ向かわない．

この性質は以下の LTL 式で記述できる．

!((ステーションにいる救急車が選択される)&&(order を手にする)) W (救急車が現場へ向かう)

以上の性質を検証した結果，図 5 の BPMN モデルはゴールモデル中の性質を満たしたモデルとなっていることを確認できた．

4.3 bCMS 概要

bCMS は fire station coordinator (FSC) と police station coordinator (PSC) の間で行われるコミュニケーションを支援することにより事故に対処するシステムである．FSC，PSC 内部のコミュニケーションはこのシステムの対象外である．coordinator のタスクに関係する事故の情報は常に最新の状態に保たれなければならない．FSC，PSC は協調していつ，どこに，どのように消防車を送るのか決定し，事故に対処する．

4.4 bCMS に関するケーススタディ

次に，barbados Car crash Management System (bCMS) に関するゴールモデルの変換について記述する．[5] には bCMS に関するゴールモデル全体が記されている．本研究で対象としているのは behavior goal の変換であるため，[5] から該当する部分のみを抽出し整形した．[5] のゴールモデルは OR 分解によって manual system, automated system (centralized system, distributed system) の 3 つに分かれている．Manual system は system-as-is に相当し，ソフトウェアによる支援は行わず，人間が作業を行うシステムである．一方，automated system はシステムにおける一部をソフトウェアにより支援・自動化したものである．本提案手法は OR 分解による選択肢が選択されたゴールモデルを対象としているため，automated system である distributed system をすべての OR 分解において選択した．

[5] に記載されている bCMS のゴールモデルは [7] を基に作られたものである．このゴールモデルはリファインメントパターンが明示的に記されておらず，そのまま各リファインメントパターンをすべてのゴール分解へ当てはめることはできない．そのため，著者がゴールモデルをすべてのゴール分解がいずれかのリファインメントパターンを用いて分解されたものとなるように意味が変化しないよう留意して整形した．

[5] にはまとまった大きなゴールモデルは記載されていない．しかし，断片的な複数のゴールモデルにおける同一の親ゴールと葉ゴールを繋ぎ合わせることによって大きなゴールモデルが得られる．繋ぎ合わせたゴールモデルを整形したものが付録に示された図 9 である．図 9 のゴールモデルの親ゴールが Achieve [Crisis Resolved When Reported] であり，Achieve [Crisis Resolved When Reported] というゴールが各リファインメントパターンを用いて分解されている．図 9 のゴールモデルを提案手法によって変換した結果が図 7 である．

LAS のケーススタディと同様に，まず，ゴールモデルにおけるシナリオが BPMN モデルにおいて満たせるのかを確かめることによって有効性を検証する．シナリオは [7] に記載されている bCMS のメインシナリオを使用する．メインシナリオは [7] のとおり，7 つのパートにより構成されている．メインシナリオは事故が発生し，PSC と FSC がコミュニケーションを確立し，coordinator を同定してから，PSC と FSC が事故が終息したことを同意するまでの流れを表している．メインシナリオ 1, 5 に関しては今回使用したゴールモデルに含めていない部分であるため，検証に使用するシナリオから除外する．図 7 におけるアクティビティの流れを確認すると，適切にゴールモデルにおけるシナリオを反映できていることが確認できる．Unmonitorability-driven refinement pattern, Uncontrollability-driven refinement pattern によって，ソフトウェアエージェントと環境エージェントにレーンをわけることによって，ソフトウェアと外部環境のインタラクションを表現している．

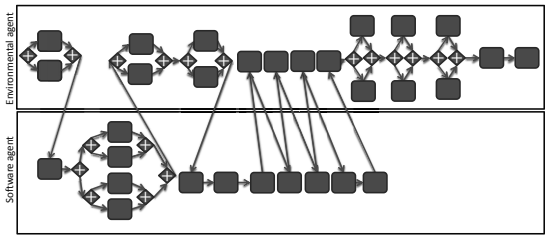


図 7　変換結果まとめ

次に，図 7 の BPMN モデルを PROMELA 化し，SPIN を用いてモデル検査を行った．検証には以下に示す 2 つの性質を用いた．

性質 1. 事故の情報が消防と警察の間で共有され，事故が解決される．

この性質は以下の LTL 式で記述できる．

□((警察の情報を登録する)&&(消防の情報を登録する)&&(両者の情報が共有される)⇒◇(事故が解決される))

性質 2. 消防と警察の間でルートの合意がなければ，現場へ向かわない．

この性質は以下の LTL 式で記述できる．

!(消防がルートに合意する&&警察がルートに合意する) W(現場へ向かう)

以上の性質を LTL によって記述し，検証した結果，図 7 の BPMN モデルはゴールモデル中の性質を満た

したモデルとなっていることを確認できた.

4.5 卸–メーカー間における取引業務概要

卸–メーカー間における取引業務では卸–メーカー間の商品の受発注，物流，請求・支払い等が行われる．製造業や流通業における情報システムを用いた業務支援は未だ不十分であり，迅速な取引情報の交換や情報の共有を低コストで行える情報システムの開発が望まれている.

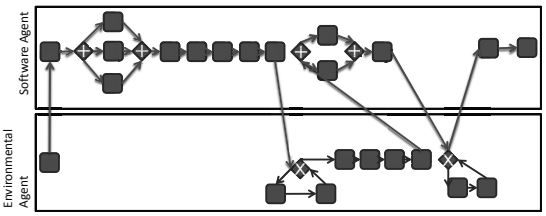


図 8 卸–メーカー取引業務プロセス

4.6 卸–メーカー間における取引業務に関するケーススタディ

次に，卸–メーカー間における取引業務に関するケーススタディを行った．既存のゴールモデルを用いずに，実際に企業が直面している課題である [33] に記載されている情報から著者がリファインメントパターンに基づいてゴール分解を行うことで，KAOS ゴールモデルを構築した．このゴールモデルに対して提案手法を用いて変換した結果が図 8 である．LAS や bCMS のケーススタディと同様に，まず，ゴールモデルにおけるシナリオが BPMN モデルにおいて満たせるのかを確かめることによって有効性を検証する．[33] において卸–メーカー間におけるビジネスプロセスに関して，全体的な流れや受発注や物流等に分割した流れが記載されている．これらと図 8 の BPMN モデルを照らし合わせると適切にシナリオを満たせることが確認できた．また，[33] に記載されているプロセスモデルと同様の流れであることが確認できた．次に，図 8 の BPMN モデルを PROMELA 化し，SPIN を用いてモデル検査を行った．検証には以下に示す 2 つの性質を用いた.

性質 1. 欠品連絡がなされたら，注文が修正されなければならない.

この性質は以下の LTL 式で記述できる.

□((欠品連絡を行う)⇒◇(注文が修正される))

性質 2. 仕入計上処理と売上計上処理が両方実行され，照合チェックが行われなければならない.

この性質は以下の LTL 式で記述できる.

□((仕入計上処理を行う)&&(売上計上処理を行う)⇒◇(照合チェックを行う))

以上の性質を LTL によって記述し，検証した結果，図 8 の BPMN モデルはゴールモデル中の性質を満たしたモデルとなっていることを確認できた.

4.7 ケーススタディにおける考察

LAS，bCMS，卸–メーカー取引業務いずれの場合においても，リファインメントパターンを用いて記述された KAOS ゴールモデルにおけるゴール間の関係を，順序や条件分岐の関係を保ち，適切に BPMN モデルへ反映することができた．リファインメントパターンはゴール分解において頻出するゴール間の関係性をパターン化し，時相論理によって記述したものである．ゴールモデルをリファインメントパターンを用いて構築することは，ゴールモデルが 2.1 節で述べた AND 分解の性質を持つと共に他のモデルに変換する際にも有効に活用できると考えられる．更に LAS に関するゴールモデル，bCMS に関するゴールモデル，卸–メーカー取引業務に関するゴールモデル等，大きさの異なるゴールモデルの内容を適切に BPMN モデルへ反映することができた．このことからモデルの規模に関わらず，本手法によってゴールの関係を BPMN モデルへ反映できるといえる.

5 考察

5.1 生成できる BPMN モデルの特徴

本手法ではリファインメントパターンによってゴールが分解されているゴールモデルを対象としている．そのため，生成した BPMN モデルにはゴールモデルにおけるゴール間の関係が反映されている．ゴールモデルはシステムの目的とその実現手段を記述するた

めに用いるものであり，システムの流れを記述するためのものではない．ゴールモデルにおいては，特に葉ゴールにおいて上位の要求を表すゴールを実現するための手段が記載されており，それは本研究のようにBPMN モデルにおいて記述すべきアクティビティと同等のものだと考えることができる．しかし，ゴールモデルは直接的にシステムの振る舞いを記述するために用いるモデルではないため，複雑な流れを記述することはできない場合が考えられる．本研究はゴールモデルによって記述した要求を漏れなく BPMN 初期モデルに反映することを目的としている．そのため，複雑な BPMN モデルを構築する際は本手法によって生成された BPMN モデルを更に精緻化したほうが合理的である．ゴールモデルはシステムの流れの記述には向かないからである．ゴールモデルは上記のように，システムの目的とその実現手段を表すモデルであり，構造的にシステムの流れを明示的に記述することはできない．一方，ビジネスプロセスモデルはシーケンスフローやゲートウェイを用いてシステムの流れを明示的に記述することが可能である．

5.2 ゴールとアクティビティの同等性

5.1 節において記述したように，ゴールモデルにおけるゴール，特に葉ゴールはアクティビティと同等のものだと考えることができる．これは，BPMN モデルにおけるアクティビティとは，ゴールモデルにおけるオペレーションと同等のものだという仮定を置いているからである．オペレーションとは目的を満たすためにシステム (人やソフトウェアを含む) が提供すべきサービスである．つまり，ゴールを達成するためにはオペレーションを実行する必要がある．十分に分解されたゴール (葉ゴール) であれば，ゴールとアクティビティは一対一で関連付けることができる．ゴールをアクティビティへ変換するとはゴールを達成するためのオペレーションはアクティビティと同一であるという仮定を行っている．

5.3 変換ルールの妥当性

本節では，ゴール達成・開始の前後関係，アクティビティの開始・終了の前後関係が，変換前後において同一のものであるのかについて記述する．本手法は変換ルールを利用することで KAOS ゴールモデルを BPMN モデルへの変換を行う．KAOS ゴールモデルにおいてリファインメントパターンを用いてゴール分解が行われることで記述されるゴールを達成・開始する順番は，変換後である BPMN モデルにおけるアクティビティを終了・開始する順番と同様だと考えられる．Milestone-driven パターンを例に説明する．Milestone-driven パターンは親ゴール達成 (ターゲット条件 T への到達) のために必要な中間的な条件 (マイルストーン条件 M1，M2，M3…) がある場合に用いる．図 1 のように親ゴール (C⇒◊T) が成り立つためには，子ゴール 1 (C⇒◊M), 子ゴール 2 (M⇒◊T) が順に達成される必要がある．上記の時相論理で記述したゴールは，親ゴール (C⇒◊T)(現在条件 C が成り立つとき，そのうちターゲット条件 T が成り立つ.) 達成のために，子ゴール 1 (C⇒◊M)(現在条件 C が成り立つとき，そのうちマイルストーン条件 M が成り立つ.) が実行された後に，子ゴール 2 (M⇒◊T)(マイルストーン条件 M が成り立つときそのうちターゲット条件 T が成り立つ.) が達成される必要がある．時相論理式を見ればわかるように，子ゴール 2 はマイルストーン条件 M が成り立っている場合にそのうちターゲット条件 T が成り立つというものである．つまり，これはゴールの終了に関する前後関係が規定されるとともに，開始に関する前後関係も規定されることを意味する．アクティビティに関しても同様である．Milestone-driven パターン以外の場合であっても時相論理式で記述されているように，終了に関する前後関係，開始に関する前後関係が共に規定される．しかし，規定された前後関係の通りに必ずしもアクティビティが実行されるとは限らない．現実には，あるゴールの達成を待たずに，アクティビティを開始することは起こりうるが，本稿においてはゴールモデルにおける特定の前後関係を仮定してアクティビティの順序へ反映している．例として Milestone-driven パターンを取り上げる．上記のように Milestone-driven パターンにおける変換では順序があると仮定しているが，場合によっては各アクティビティの実行が前後することはありうる．そのような場合においては，あ

る望ましい順序がゴールモデル上に記述されていると仮定して，その順序を BPMN モデルへ反映するとみなす．

5.4 OR 分解の選択

3.1 節において記述したように本研究では OR 分解による要求選択肢を選択済みのゴールモデルを対象としている．しかし，通常開発すべきシステムのステークホルダは複数存在し，各ステークホルダによってシステムに搭載すべき要求の優先順位が異なる場合が考えられる．そのような場合，ステークホルダ間の利害関係の対立によって，ゴール選択を適切に行うことは難しい．ゴールの選択方法に関しては本研究の範囲外であるが，ゴールを選択するための手法はいくつか提案されている．適切にゴール選択を行うための手法として Kaiya らの AGORA (Attributed Goal-oriented Requirements Analysis) [16] が挙げられる．AGORA はゴール指向要求分析法の一種であり，ゴールの属性値として貢献度と満足度を持つ．満足度を用いてステークホルダ間の要求の不一致を同定することでゴール選択を行うことができる [17]．また，斎藤らの手法 [32] ではゴールに寄与度と有効度を割り振ることでどのような理由でゴールが絞り込まれていったのか根拠を表すことができる．

5.5 提案手法適用範囲

本節では，本手法を適用すべき対象と適用できない場合について記述する．

本手法はゴールモデルを変換することによりビジネスプロセスモデルを構築している．よって，本手法の適用対象となりうるのはゴールを達成するために手順を踏む必要があるシステムである．ゴールを達成するために人やソフトウェアが行うべき動作をゴールモデルによる要求の洗練によって獲得し，それらをビジネスプロセスモデルへ変換することで，ゴール達成のための手順を明示的に示せるとともに，プロセスにおいて行うべきことをソフトウェアと人等の環境エージェントで分割することができる．本手法はそのような目的での使用に適していると考えられる．

次に本手法を適用できない場合について記述する．

5.5.1 リファインメントパターンのいずれにも当てはまらない場合

本手法は KAOS ゴールモデルにおけるリファインメントパターンに着目し BPMN モデルへ変換を行う．そのため，リファインメントパターンが明示されている KAOS ゴールモデルのみを変換対象としている．KAOS と，リファインメントパターンの考案者である Lamsweerde は 6 種類のリファインメントパターンを定めている [30]．しかし，ゴール分解において考えられるパターンを 6 種類で網羅することはできず，どのリファインメントパターンにも当てはまらない分解が起こりうる．そのような場合，本手法では BPMN モデルへ変換することができず，ゴール達成の順番はゴールの意味を考慮して決定する必要がある．

5.5.2 ハードゴール以外の要素の変換

本手法では変換対象をハードゴールに限定している．これはリファインメントパターンによるゴール分解はハードゴールのみを対象としているからである [30]．その他の KAOS ゴールモデルにおける要素としては，ドメインプロパティや非機能要求などがある．共に振る舞いを表す要素ではなく，BPMN モデルにおいて対応付けられる要素がないため本手法においては変換しない．しかし，非機能要求はプロセスに影響を与える場合がある．非機能要求には時間的・空間的計算量や入出力サイズ等の性能に関する要求がある．例として書籍検索システムの非機能要求である「書籍検索の応答時間は 1 秒より短くなければならない」が挙げられる．このような非機能要求は BPMN モデルにおいてアクティビティとしては表せないが，アクティビティの制約として表すことができる．このように非機能要求を BPMN モデルへ反映することは，より詳細なシステムプロセスの構築に繋がると考えられ，今後の課題として挙げられる．

6 関連研究

関連研究としてはゴールモデルとビジネスプロセスモデルに関する研究 4 種類及び，企業における取り組みを取り上げる．

6.1 ゴールモデルからビジネスプロセスモデルへの変換

Sun らはゴールモデルをビジネスプロセスモデルに変換する手法を提案している [28]．Sun らの手法ではビジネスプロセスモデルにおけるアクタを単一のものとして扱っているが，本手法ではリファインメントパターンを用いてゴールの達成責任を分割することによって，レーンを分けることができる．

Nagel らは KAOS モデルをビジネスプロセスモデルへ変換する手法を提案している [24]．KAOS モデルを拡張し，オペレーションを実行する順番に関する制約を付加することで，KAOS モデルにおける要素の関連をビジネスプロセスモデルへ反映している．Nagel らの手法では KAOS モデルにおけるゴールを達成するエージェントが考慮されていないため，ビジネスプロセスモデルにおいて複数のレーンを使い分けることはできないが，本手法ではリファインメントパターンを用いてゴールの達成責任を分割することによって，レーンを分けることができる．

Koliadis らはゴールモデルの一種である i*フレームワークにおける変化を BPMN モデルへ反映する手法を提案している [19]．Koliadis らの手法ではゴールモデルの意味をくみ取って変換しなければならない．一方で本手法ではリファインメントパターンを用いることでより効率的な変換が可能である．

6.2 ビジネスプロセスモデルからゴールモデルへの変換

Vara らはビジネスプロセスモデルをゴールモデルへ変換するガイドラインを提案している [21]．ビジネスプロセスモデルにおける要素パターンとゴールモデルとのマッピングを，22 個のガイドラインを用いることによって行う．

Boness らはビジネスプロセスモデルと類似している UML アクティビティ図を用いて，ゴールモデルの作成支援に関する研究を行っている [2]．これらの研究は本研究とは異なりゴールモデルからビジネスプロセスモデルへの体系的な変換手法を提案してはいない．

6.3 ゴールモデルとビジネスプロセスモデルの関連付け

Koliadis ら及び，Cornax らは BPMN と KAOS を関連付ける方法論を提案している [18] [9]．これらの研究は BPMN と KAOS を関連付けることを対象としており両者のギャップ削減を図っている．

Gröner らは記述論理を用いて自動でゴールモデルとビジネスプロセスモデル間の妥当性を検証する研究を行っている [12]．それによってユーザの要求を満たした実行可能プロセスを検知できる．これらの研究は本研究とは異なりゴールモデルからビジネスプロセスモデルへの体系的な変換手法を提案してはいない．

Nagel らは KAOS モデルから検証可能なビジネスプロセス品質制約を自動で生成するパターンベースの手法を提案している [23]．生成された制約を用いてビジネスゴールとビジネスプロセスの整合性を検証することができる．

6.4 リファインメントパターンの活用

リファインメントパターンには様々な利用法がある．Sombat らは KAOS モデルを UML クラス図へ変換する手法を提案した [8]．この研究ではリファインメントパターンに基づくゴールモデルを，OCL 制約が記述された UML クラス図へ変換している．

Honda らは KAOS モデルを用いて定義した要求をユースケースモデル，ロバストネスモデルへ変換する方法を提案した [13]．それにより要求モデル・設計モデル間のギャップの削減を支援する．

Cailliau らはリファインメントパターンに基づくリスク評価フレームワークを提案している [6]．この研究ではリファインメントパターンを用いて分解された確率的に達成されるゴールに対してパターンごとに確率を計算することで，親ゴールの達成確率を計算することができる．

Matoussi らはリファインメントパターンが用いられた KAOS モデルを Event-B モデルへ変換するアプローチを提案している [22]．それによってソフトウェア開発にフォーマルメソッドを活用する際，困難な点である初期フォーマルモデル構築の困難さを軽減することができる．

6.5 企業における取り組み

企業におけるゴールモデル，ビジネスプロセスモデルに関する取り組みとしては，NTT データが提唱した MOYA [34] があげられる．MOYA は NTT データにおけるシステム開発標準である TERASOLUNA の要求定義におけるガイドラインの 1 つである．MOYA の狙いはステークホルダの気付きを導き，要求定義の品質を向上させることであり，関係者，課題，目的，手段，業務をゴールモデルや UML を用いて分析し可視化することがより品質の高い要求定義に繋がると考えられている．MOYA は大きく 2 つのステップに分けられる．ステップ 1 ではゴールモデル等を用いてビジネス目的の確認を行う．ステップ 2 では UML を用いてビジネスのモデル化を行う．ステップ 1 で構築したゴールモデルの情報をステップ 2 へ反映させることで，ゴールを達成できるビジネスプロセス，ユースケース等のモデル構築を支援することができる．MOYA は実際のプロジェクトにおいて多く利用されており，ゴールモデルによって手段と目的を構造化・可視化することで，全体最適となる解決策を選定することができた，各要件間の関連性や達成しようとしている課題と目的との関係を明確にすることができたという評価が実際に MOYA を適用した担当者から得られている．我々の提案手法によってリファインメントパターンを利用したゴールモデルをビジネスプロセスモデルの構築に用いることで，MOYA におけるビジネスプロセスモデル構築を体系的に行うことができると考えられる．

7 まとめと今後の課題

本研究では，KAOS ゴールモデルのリファインメントパターンに基づいて記述された要求を，システムの流れを表す BPMN の初期モデルへ体系的に変換する手法を提案した．またケーススタディを行い提案手法の有効性を示した．その結果 KAOS モデルで記述した要求を達成するべき順番が BPMN モデルにおいて適切に記述できることを示した．これにより，適切な要求定義のための支援ができると考える．今後の課題としては，本手法をツールとして実装することが挙げられる．それによって変換の際に人為的なミスを排除するとともに短い時間で BPMN モデルを生成することができる．すべてのゴール分解がリファインメントパターンを用いて行なわれ，図 4 の変換アルゴリズムを実装すれば自動で BPMN モデルへの変換を行うことが可能だと考える．評価に関しては分野や規模などが異なる様々なケーススタディに本手法を適用することで有効性を検証したい．また，モデル変換の過程を OMG が定めたモデル変換の標準 QVT [26] を用い，Nwokeji らの提案した KAOS メタモデル [25] を利用することで変換をより俯瞰的に示したい．また，形式手法 [12] を用いることで本手法の有効性を様々な角度から検証していきたい．適用範囲の拡大に関しても課題として挙げられる．非機能要求の利用は検討すべき課題である．モデル駆動開発における非機能要求の利用方法は [1] においてまとめられている．

謝辞 本研究は JSPS 科研費 24300005, 26330081, 26870201 の助成を受けたものです．

参 考 文 献

[1] Ameller, D., Franch, X. and Cabot, J.: Dealing with Non-Functional Requirements in Model-Driven Development, in *Proc. of 18th IEEE International Requirements Engineering Conference* (*RE'10*), IEEE, 2010, pp. 189–198.

[2] Boness, K. and Harrison, R.: Goal Sketching with Activity Diagrams, in *Proc. of 3rd International Conference on Software Engineering Advances*(*ICSEA '08*), IARIA, 2008, pp. 277–283.

[3] Cailliau, A.: Model Transformations and Refactoring for Goal-Oriented Models, Master's thesis, UCLouvain, Louvain-la-Neuve, 2012.

[4] Cailliau, A., Damas, C., Lambeau, B. and van Lamsweerde, A.: Modeling car crash management with KAOS, in *Comparing Requirements Modeling Approaches Workshop* (*CMA@RE'13*), IEEE, 2013, pp. 19–24.

[5] Cailliau, A., Damas, C., Lambeau, B. and van Lamsweerde, A.: Modeling Car Crash Management with KAOS, 2013, http://kaos.info.ucl.ac.be/bcms.html.

[6] Cailliau, A. and van Lamsweerde, A.: A probabilistic framework for goal-oriented risk analysis, in *Proc. of 20th IEEE International Requirements Engineering Conference* (*RE'12*), IEEE, 2012, pp. 201–210.

[7] Capozucca, A., Cheng, B., Georg, G., Guelfi, N., Istoan, P. and Mussbacher, G.: Requirements Definition Document for a software product line of car

crash management systems, 2012.

[8] Chanvilai, S., Honda, K., Nakagawa, H., Tahara, Y. and Ohsuga, A.: Goal-oriented approach to creating class diagrams with OCL constraints, in *Proc. of the 27th ACM Symposium on Applied Computing*(*SAC'12*), 2012, pp. 1051–1056.

[9] Cortes Cornax, M., Matei, A., Letier, E., Dupuy-Chessa, S. and Rieu, D.: Intentional Fragments: Bridging the Gap between Organizational and Intentional Levels in Business Processes, in *Proc. of the 10th Confederated International Conferences On the Move to Meaningful Internet Systems* (*OTM'12*), Springer, 2012, pp. 110–127.

[10] Darimont, R. and van Lamsweerde, A.: Formal Refinement Patterns for Goal-driven Requirements Elaboration, in *Proc. of the 4th ACM SIGSOFT Symposium on Foundations of Software Engineering* (*FSE'96*), ACM, 1996, pp. 179–190.

[11] Directorate, T. C.: Report of the Inquiry Into The London Ambulance Service, 1993.

[12] Gröner, G., Asabi, M., Mohabbati, B., Gasevic, D., Silva Parreiras, F. and Boskovic, M.: Validation of User Intentions in Process Models, in *Proc. of the 24th International Conference on Advanced Information Systems Engineering* (*CAiSE'12*), Springer, 2012, pp. 366–381.

[13] Honda, K., Nakagawa, H., Tahara, Y. and Ohsuga, A.: Goal-Oriented Robustness Analysis, in *Proc. of the 10th Joint Conference on Knowledge — Based Software Engineering* (*JCKBSE'12*), IOS Press, 2012, pp. 171–180.

[14] Horkoff, J. and Yu, E.: Analyzing goal models: different approaches and how to choose among them, in *Proc. of the 26th ACM Symposium on Applied Computing* (*SAC'11*), ACM, 2011, pp. 675–682.

[15] Janssen, W., Mateescu, R., Mauw, S. and Springintveld, J.: Verifying Business Processes using SPIN, in *Proc. of 4th International SPIN Workshop* (*SPIN'98*), Springer, 1998, pp. 21–36.

[16] Kaiya, H., Horai, H. and Saeki, M.: AGORA: Attributed Goal-Oriented Requirements Analysis Method, in *Proc. of 10th Anniversary IEEE Joint International Conference on Requirements Engineering* (*RE'02*), IEEE, 2002, pp. 13–22.

[17] Kaiya, H., Shinbara, D., Kawano, J. and Saeki, M.: Improving the detection of requirements discordances among stakeholders, *Requir. Eng.*, Vol. 10, No. 4(2005), pp. 289–303.

[18] Koliadis, G. and Ghose, A.: Relating Business Process Models to Goal-Oriented Requirements Models in KAOS, in *Proc. of the 1st Pacific Rim Knowledge Acquisition Workshop* (*PKAW'06*), Springer, 2006, pp. 25–39.

[19] Koliadis, G., Vranesevic, A., Bhuiyan, M., Krishna, A. and Ghose, A.: Combining *i** and BPMN for Business Process Model Lifecycle Management, in *Proc. of the Business Process Management Workshops*, Springer, 2006, pp. 416–427.

[20] Letier, E. and van Lamsweerde, A.: Agent-based Tactics for Goal-oriented Requirements Elaboration, in *Proc. of the 24th International Conference on Software Engineering* (*ICSE'02*), ACM, 2002, pp. 83–93.

[21] Luis de la Vara, J., Sánchez, J. and Pastor, O.: On the Use of Goal Models and Business Process Models for Elicitation of System Requirements, in *Proc. of Enterprise, Business-Process and Information Systems Modeling — 14th International Conference, BPMDS 2013, 18th International Conference ENNSAD2013* (*BPMDS'13/EMMSAD'13*), Springer, 2013, pp. 168–183.

[22] Matoussi, A., Gervais, F. and Laleau, R.: A Goal-Based Approach to Guide the Design of an Abstract Event-B Specification, in *Proc. of the 16th IEEE International Conference on Engineering of Complex Computer Systems* (*ICECCS'11*), IEEE, 2011, pp. 139–148.

[23] Nagel, B., Gerth, C., Engels, G. and Post, J.: Ensuring Consistency among Business Goals and Business Process Models, in *Proc. of the 17th IEEE International Enterprise Distributed Object Computing Conference* (*EDOC'13*), IEEE, 2013, pp. 17–26.

[24] Nagel, B., Gerth, C., Post, J. and Engels, G.: Kaos4SOA — Extending KAOS Models with Temporal and Logical Dependencies, in *Proc. of the CAiSE'13 Forum at the 25th International Conference on Advanced Information Systems Engineering*, CEUR-WS.org, 2013, pp. 9–16.

[25] Nwokeji, J., Clark, T. and Barn, B.: Towards a comprehensive Meta-Model for KAOS, in *Proc. of the 3rd International Workshop on Model-Driven Requirements Engineering* (*MoDRE'13*), 2013, IEEE, 2013, pp. 30–39.

[26] OMG: Meta object facility (mof) 2.0 query/view/transformation. Specification Version1.0, April 2008.

[27] OMG: Business Process Model and Notation (BPMN), Version 2.0, January 2011.

[28] Sun, Z., Wang, J., He, K., Xiang, S. and Yu, D.: A Model Transformation Method in Service-Oriented Domain Modeling, in *Proc. of the 21st Australian Software Engineering Conference* (*ASWEC'10*), IEEE Computer Society, 2010, pp. 107–116.

[29] van Lamsweerde, A.: Requirements Engineering: From Craft to Discipline, in *Proc. of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering* (*FSE'08*), ACM, 2008, pp. 238–249.

[30] van Lamsweerde, A.: *Requirements Engineering — From System Goals to UML Models to Software Specifications*, Wiley, 2009.

[31] van Lamsweerde, A., Darimont, R. and Massonet, P.: Goal-directed elaboration of require-

ments for a meeting scheduler: problems and lessons learnt, in *Proc. of the 2nd IEEE International Symposium onRequirements Engineering* (*RE'95*), IEEE, 1995, pp. 194–203.
[32] 斎藤忍，山本修一郎: 属性値に基づくゴール選択手法の提案と考察, 経営情報学会誌, Vol. 15, No. 3(2006), pp. 37–50.
[33] 財団法人流通システム開発センター: 酒類・加工食品と日用品・化粧品業界の卸–メーカー間取引におけるビジネス・プロセス・モデル調査研究報告書, 2005.
[34] 萩原淳，斎藤忍: 目的と手段の断絶を解消するビジネスモデリング方法論の実践—気づきから要求を導く—, 情報処理学会デジタルプラクティス, Vol. 4, No. 2(2013), pp. 169–177.

A 付録: bCMS のゴールモデル

図 9 はケーススタディで用いた bCMS のゴールモデルである．ゴールを分解する際に用いるリファインメントパターンがゴール間に記述されている．黒の菱形はゴール，白の菱形は期待，黒の六角形は環境エージェント，白の六角形はソフトウェアエージェントを表す．

B 付録: 卸–メーカー間取引業務のゴールモデル

図 10 はケーススタディで用いた卸–メーカー間取引業務のゴールモデルである．

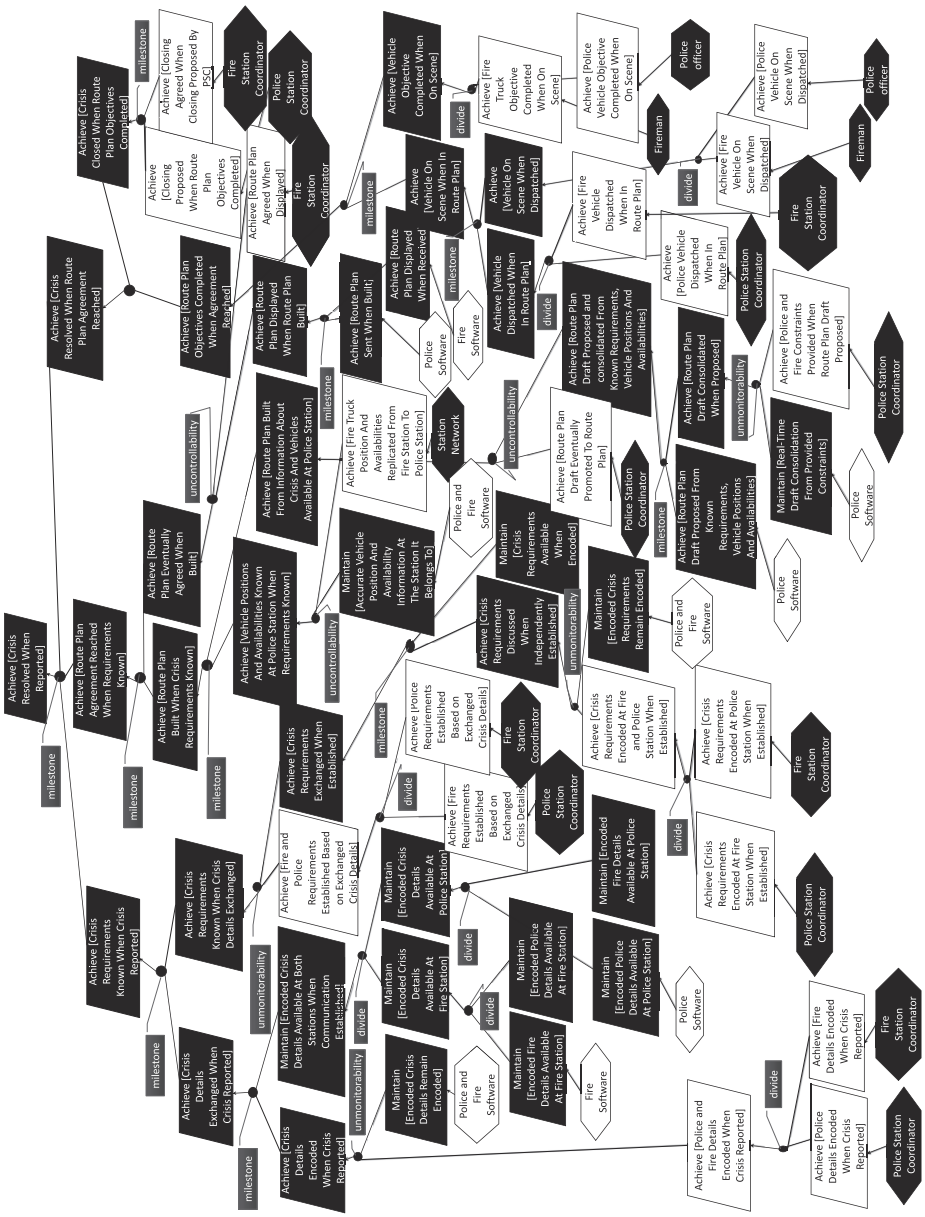


図 9　bCMS のゴールモデル

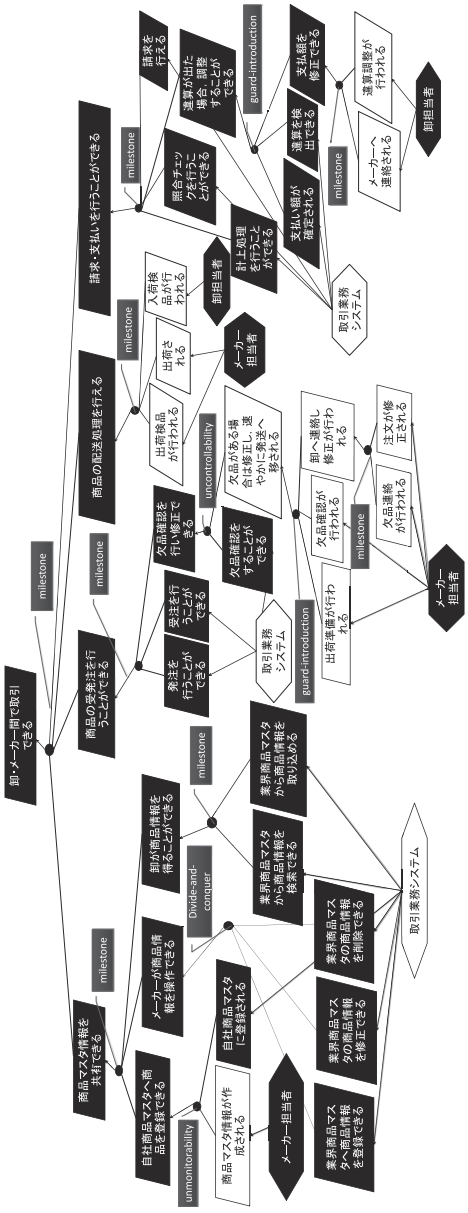


図 10 卸–メーカー間取引業務のゴールモデル

堀田大貴

1989 年生．2012 年青山学院大学社会情報学部社会情報学科卒．2014 年電気通信大学大学院情報システム学研究科博士前期課程修了，現在，電気通信大学大学院情報システム学研究科博士後期課程に在学．ソフトウェア工学，要求工学，プロセスマイニングの研究に従事．

本田耕三

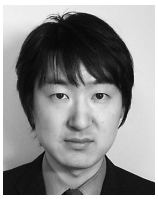
1953 年生．1976 年九州大学工学部電気工学科卒業．同年日本電気 (株) に入社．2011 年電気通信大学大学院情報システム学研究科博士前期課程修了，現在，電気通信大学大学院情報システム学研究科博士後期課程に在学．情報処理学会学生会員．

平山秀昭

1958 年生．1981 年慶應義塾大学工学部管理工学科卒．同年 (株) 東芝入社．2001 年電気通信大学大学院情報システム学研究科博士後期課程了．2003 年より東芝ソリューション (株)，2015 年より東芝ソリューション販売 (株) に所属．2014 年より電気通信大学協力研究員．2015 年より東京電機大学情報環境学部非常勤講師．博士 (工学)(電気通信大学)．並列分散処理，ソフトウェア工学等の研究に従事．情報処理学会，電子情報通信学会会員．

清　雄一

1981 年生．2009 年東京大学大学院情報理工学系研究科博士後期課程修了．同年 (株) 三菱総合研究所入社．同社情報技術研究センター，金融ソリューション本部等に所属．2013 年より電気通信大学助教，現在に至る．分散コンピューティング，セキュリティ，プライバシ保護技術等の研究に従事．情報処理学会，電子情報通信学会，IEEE Computer Society 各会員．



中川博之

1974 年生．1997 年大阪大学基礎工学部情報工学科卒業．同年鹿島建設 (株) に入社．2007 年東京大学大学院情報理工学系研究科修士課程修了，2008 年同大学院博士課程中退．同年電気通信大学助教，2014 年大阪大学大学院情報科学研究科准教授，現在に至る．2013 年早稲田大学博士 (工学)．要求工学，形式手法，エージェントおよび自己適応システム開発手法の研究に従事．情報処理学会，電子情報通信学会，IEEE CS 各会員．

田原康之

1966 年生．1991 年東京大学大学院理学系研究科数学専攻修士課程修了．同年 (株) 東芝入社．1993〜1996 年情報処理振興事業協会に出向．1996〜1997 年英国 City 大学客員研究員．1997〜1998 年英国 Imperial College 客員研究員．2003 年国立情報学研究所入所．2008 年より電気通信大学准教授．博士 (情報科学)(早稲田大学)．エージェント技術，およびソフトウェア工学などの研究に従事．情報処理学会，日本ソフトウェア科学会会員．

大須賀昭彦

1981 年上智大学理工学部数学科卒．同年 (株) 東芝入社．同社研究開発センター，ソフトウェア技術センター等に所属．1985–1989 年 (財) 新世代コンピュータ技術開発機構 (ICOT) 出向．2007 年より，電気通信大学大学院情報システム学研究科教授．2012 年より，国立情報学研究所客員教授兼任．工学博士 (早稲田大学)．主としてソフトウェアのためのフォーマルメソッド，エージェント技術の研究に従事．1986 年度情報処理学会論文賞，2013 年度人工知能学会研究会優秀賞受賞．IEEE Computer Society Japan Chapter Chair，人工知能学会理事，日本ソフトウェア科学会理事，同学会監事等を歴任．情報処理学会，電子情報通信学会，人工知能学会，日本ソフトウェア科学会，電気学会，IEEE Computer Society 各会員．