ゴール指向要求工学の体系化のための共通用語

田原康之[††] 長野伸一[†,††]
吉岡信和[††] 本位田真一[††,†††]

ゴール指向は，要求工学の最も重要なアプローチの 1 つである．ゴール指向を効率的に採用しようと考える技術者のために，ゴール指向要求工学の体系化が必要である．しかし，既存の研究は，体系化のための共通用語がなく，したがって共通用語を用いて方法論を比較することも行っていないため，体系化への基盤としては十分でない．本論文では，このような体系化を可能とするための共通用語を提案する．さらに，共通用語を用いて，代表的なゴール指向要求工学手法である，KAOS と i*のメタモデルをそれぞれ作成し，メタモデルを利用して，KAOS と i*の比較を行う．共通用語は，ゴールやエージェントといった，各方法論で使用される構成単位と，責任や操作可能化といった，構成単位間の関係を含む．共通用語においては，ゴールとは異なる単位として「要件」と呼ばれる概念を導入する．また，仮説の検証を通じて，本論文で提案する共通用語が，ゴール指向要求工学の体系化の基礎となりうるかどうかを確認する．

Common Terminology for Systemitization of Goal-oriented Requirements Engineering

Yasuyuki Tahara,[††] Shinichi Nagano,[†] Nobukazu Yoshioka[††]
and Shinichi Honiden[††,†††]

Goal-orientation is one of the most important approaches to requirements engineering. Systematization of goal-oriented RE is needed to make effective the activities of engineers who want to adopt goal-orientation. However, the existing efforts are not sufficient as bases of the systematization because they lack common terminology needed to the systematization and they do not make comparison of the methodologies using the terminology. In this paper, we provide a common terminology that has the potential to enable the systematization. We create metamodels of the KAOS and i* methodologies individually in the terminology, and then make a comparative study of these two methodologies using the metamodels. The terminology includes individual elements used in the methodologies such as goals and agents, and relations between the elements such as responsibility and operationalization. We provide a new concept called a requisite as a different element from a goal. We check by validations of hypotheses if our terminology can be a basis of the systematization of goal-oriented RE.

1. はじめに

ゴールは要求工学において非常に重要な概念であると考えられている．要求工学の方法論には，ゴールを中心に考えるものがあり，ゴール指向要求工学方法論と呼ばれる．その中でも，KAOS[12),17)] と i*[21)] は最も広く知られ，適用されている．ゴール指向が，ソフトウェア開発の目標や，ステークホルダの意図を分析するための，要求工学のアプローチとして有効であると認識されるにつれて，ソフトウェア技術者が習得すべき技術の 1 つになろうとしている．

我々は，次のような理由により，ゴール指向要求工学の体系化が必要であると考えている．van Lamsweerde[18)] が指摘するように，ソフトウェア要求をステークホルダに説明することは，重要な課題である．ソフトウェア技術者は，開発の途中において，要求分析のプロセスや成果物について，ステークホルダとコミュニケーションをとる必要がある．さらに，方法論について，様々な選択肢から 1 つを選んで習得しようとする場合に，選択基準と容易な習得方法が必要である．このようなコミュニケーション，選択，および習得を効率的に行うには，ゴール指向要求工学を

† 株式会社東芝研究開発センター
Corporate R&D Center Toshiba Corporation
†† 国立情報学研究所
National Institute of Informatics
††† 東京大学
The University of Tokyo

体系化することにより，その様々な概念の意味，ゴール指向要求工学に固有の特徴，および各方法論に特有の特徴を明確にする必要がある．

体系化の第 1 段階に必要なステップとして，体系を説明するための共通用語の確立が必要である．前述のように，ステークホルダとのコミュニケーションにおいては体系化が重要であり，コミュニケーションのための言葉として共通用語が必要である．Anton ら[1)]は，共通用語は技術者とステークホルダの双方が分かりやすく使い慣れた概念に基づいていなければならない，と論じている．このような用語を使用することにより，方法論の特徴を記述できると考えられる．

次の段階の 1 つとして，共通用語が体系化の基礎として適切かどうかのチェックが必要である．そのためには，少なくとも次の 2 点を検討しなければならない．1 点目は，用語を用いてそれぞれの方法論を説明できるかどうかである．2 点目は，用語がゴール指向要求工学の本質的な特徴を表しているかどうかである．たとえば，従来の研究[1),16),18)]で，ゴールに関係する概念の抽象レベルについて論じられている．詳しくは，ゴールには抽象度に関して高レベルなものと低レベルなものがあり，後者は「要求」とも呼ばれる．このような区別は，ゴール指向要求工学の本質的な特徴と考えられ，したがってそれを表す用語が必要である．

従来ゴール指向要求工学では多くの研究がなされているが，上記の観点から体系化の基礎として十分とはいえない．Kavakli[11)]はゴール指向方法論のメタモデルを構築し，KAOS と i*を含むいくつかの方法論の方法モデル（method model）を定義することにより，ゴール指向要求工学の統一的なフレームワークを与えようとしている．この研究は，戦略的観点からの方法論のモデル化に絞っている．このような観点から，いくつかのゴール指向要求工学方法論に共通して現れる手順をメタモデルに基づいて特定し，各手順で使われる戦略によって方法論を区別している．しかし，そのメタモデルは抽象的で，各方法論の詳細は検討の対象外である．たとえば，KAOS と i*では，ゴールやエージェントは異なる意味で用いられている．しかし，この研究でのメタモデルは，そのような違いを扱っていない．したがってそのメタモデルは，一方の方法論には精通していない人とのコミュニケーションに利用したい場合，その相手は，精通している方の意味のみで理解しようとし，誤解を生じる結果に陥りやすい．そこで，適切なコミュニケーションを実現するためには，共通の概念を扱うようなさらに粒度の細かいメタモデルを検討する必要がある．

従来研究のほとんどは，各方法論に特化した用語とモデルでゴール指向要求工学の特徴を説明している．一方，異なる方法論を比較し，各方法論に共通な特徴と個別の特徴を明らかにしようとする研究もある．たとえば，文献 21) は i*を KAOS を含む他の方法論と比較している．本文献の主張は，KAOS がゴールに関する幅広い還元を前提とし，基本的にトップダウンな方法でゴールを分割するのに対し，i*は分散した志向性（intentionality）の概念を前提とし，志向的な関係を再構築することによって分析を進める，というものである．しかしこの比較では，志向性などの i*特有の用語を使用している．したがって共通用語を定義しておらず，共通用語を用いた比較により各手法の特徴を明らかにするものではない．

本論文では，ゴール指向要求工学に関し，代表的な方法論である KAOS と i*に絞り，上述したような体系化を可能にするような共通用語を提案する．用語として，方法論で使用する概念的な構成要素として，ゴール，要件，操作，エージェント，およびオブジェクトの 5 種類を用意し，また構成要素の間の関係を含める．そして，それらの用語を使用して，KAOS と i*のメタモデル☆を構築し，2 つの方法論の比較を行って，ゴール指向要求工学に固有の特徴と，各方法論に特化した特徴とを議論する．メタモデルは 2 種類のグラフから構成される．1 つは概念的な構成要素をノードとし，関係をアークとする．このようなグラフは，方法論の静的な構造を検討するのに有用である．もう 1 種類のグラフは，要求工学プロセスにおける，モデルの部分間の依存関係を表す．本グラフは，プロセスの 1 ステップによって，モデルのある部分を表す構成要素の関係から，モデルの別の部分が導出されることを表す．このようなメタモデルは，方法論の動的な特徴を検討するのに有用である．本論文では，これらのメタモデルにより，提案する共通用語が各方法論を適切に説明できることを示す．

共通用語がゴール指向要求工学の体系化の基礎となりうるかどうかをチェックするために，既存の文献で議論されてきたような，KAOS と i*の特徴の比較を，仮説として提示し，メタモデルを使用してその仮説を検証する．そしてその検証結果を調べることにより，共通用語がゴール指向要求工学に固有の特徴と，各方法論に特化した特徴のどちらも表現できていることを示す．

☆ Kavakli[11)] のメタモデルが，ゴール指向要求工学方法論一般に対する統一的なものであるのに対し，本論文のメタモデルは，各手法ごとに個別に構築する点が異なる．

なお，本論文で提案する共通用語は，KAOS と i* の両者に共通する概念を表す用語として抽出したものであるが，必ずしも意味内容に関してまで，両者で同一の概念を表すとは限らない．なぜなら，用語において同じ言葉で表される概念でも，方法論ごとに異なる扱いを受ける場合があるからである．さらに共通用語は，KAOS と i*に対してのみ行った分析に基づいており，ゴール指向手法全般に関する十分性，必要性の検討は今後の課題である．ただし，KAOS と i*は，現在最も普及した手法であるため，この 2 手法を検討したことにより，少なくとも必要性は認められると考えられる．

本論文の構成は以下のとおりである．2 章では，KAOS と i*の特徴について概説する．3 章では，共通用語，およびその用語から構成される KAOS と i*のメタモデルを提示する．4 章では，KAOS と i*の異なる特徴を比較して議論した例として仮説を示し，メタモデルに基づいた検証を行う．5 章では，仮説検証の結果を示し，共通用語とメタモデルが，ゴール指向要求工学の体系化の基礎となりうることを確認する．6 章では従来研究を紹介し，7 章では結論と今後の課題を述べる．

2. KAOS と i*の特徴

本章では，KAOS と i*の特徴について概説する．両方法論は，前述のようにゴール指向方法論であるので，ゴールの概念を中心にした要求モデルを構築することにより，要求分析作業を進めるものである．そこで，各方法論について，どのようなモデルをどのような観点で構築するのかを説明する．

まず KAOS は，システムゴール，すなわち，開発対象システム（to-be system）への要求に対する，様々な抽象レベルのゴールを，系統的に分析することを特徴とする．したがって，あくまでもシステムゴールの詳細化の作業を中心とし，ある段階でシステム境界や機能要件の特定を行う．また，そのような系統的な分析に対し，時相論理（temporal logic）に基づく理論的背景を定めている．

次に i*は，開発対象システムの分析の前段階として，現状（as-is）の分析を行い，システムゴールを特定するフェーズを含む点が特徴である．しかも現状分析においては，開発対象システムに関係する登場人物（ステークホルダや外部システムなど）をアクタとして定式化し，アクタ間の依存関係を分析する．したがって，構築するモデルは，依存関係を表す「戦略的依存関係モデル」（Strategic Dependency model，SD model）と，各アクタ内部の関心事を表す「戦略的根拠モデル（Strategic Rationale model，SR model）」の 2 種類がある．

以上のように，KAOS と i*は，分析の基本的な観点が異なっており，体系的な比較・検討が困難であると考えられる．本論文では，両方法論の共通用語を定義することにより，そのような比較・検討を行うことを目的とする．

3. KAOS と i*の共通用語

3.1 共通用語の定義

本論文で提案する共通用語は，2 種類の概念から構成される．1 つ目はゴール，エージェント，および操作などといった，方法論で使用される個別の要素を表す．2 つ目は要素間の関係を表す．関係として，たとえば，一般にゴール指向要求分析プロセスは，ゴール間の論理的関係を表すゴールグラフを扱う．そのほかにも要素間の関係として様々なものがある．本論文では，共通用語をこの 2 種類に分ける．**表 1** に要素の詳細な説明，および **表 2** に関係の詳細な説明を示す．

用語における 5 つの構成要素を，次のように，単語での表現と，3 文字以内の省略形とともに定義する．ゴール（G）は，望ましい状態に関する戦略的な関心事である．エージェント（Ag）は，ゴールを達成する責任を持つ実体である．要件（R）は，開発すべきシステムのソフトウェアと環境に関する，複合的かつ技術的な関心事である．操作（OP）は，R を達成する手段である．オブジェクト（OB）は，物理的，または情報的実体である．以下，要素は省略形で表すことにする．なお，省略形は KAOS の用語を元にしているが，これは要素の意味が KAOS のものと同じだということではない．

これら 5 種類の要素の根拠は以下のとおりである．実際のゴール指向要求工学においては，ゴールの検討だけではなく，ゴールを達成する責任を持つシステムコンポーネントやステークホルダ，実行すべき操作，およびゴールの達成に必要な資源や情報の検討も必要である．したがって，後者の 3 つを，Ag，OP，および OB という 3 種類の要素でモデル化する．

また次の理由により，関心事を表す要素として G と R の 2 種類を用意する．van Lamsweerde [18] が指摘するように，ゴールの抽象レベルは，高レベルで戦略的な関心事から，低レベルで技術的な関心事まで広がっている．本文献によれば，前者はより安定的であり，後者はより揮発性が高く，安定的な関心事を揮発性の高い関心事から分離することは重要であるため，抽象

表 1 構成要素の共通用語
Table 1 Common terminology for the elements.

記号	要素の定義	KAOSにおける定義（文献12), 19), 20)	i*における定義(文献4)	KAOSとi*の違い
G	**ゴール**:望ましい状態に関する戦略的な関心事	高レベルで戦略的な関心事を表す**ゴール**	**ゴール**:アクタの戦略的な利害関係	システム以外のAgが関心を持つか(i*)否か(KAOS)
R	**要件**:開発対象システムを実現するためのソフトウェアと環境に関する複合的な技術的関心事	低レベルで技術的関心事を表す**ゴール**	何かを実行する方法を表し,さらに分割すべき抽象的タスク	i*ではタスクとして扱い,KAOSではゴールとして扱う i*では入出力関係を定める場合もあるが,KAOSでは,入力関係と出力関係を定めない システム以外のAgが関心を持つか(i*)否か(KAOS)
OP	**操作:Rを達成する手段**	**操作**:オブジェクトに対する入出力関係.操作の適用は状態遷移を定義する.	何かを実行する方法を表し,これ以上分割する必要のない具体的タスク	i*では入出力関係は必ずしもない場合もあるが,KAOSでは,入力関係と出力関係を定める必要がある システム以外のAgが関心を持つか(i*)否か(KAOS)
Ag	**エージェント**:ゴールを達成する責任を持つ実体	**エージェント**:能動的な,すなわち,操作を実行する能力を持つオブジェクト.エージェントは,ソフトウェアエージェント,ハードウェア装置,または人間である.	**アクタ**:システムや組織的環境において,戦略的ゴールや志向性を持つ実体	志向性を持つか(i*)否か(KAOS)
OB	**オブジェクト**:物理的,あるいは情報的実体	**オブジェクト**:明確に特定できる「もの」	**実体**:物理的,あるいは情報的実体	構造や属性を持ちうるか(KAOS)否か(i*)

レベルの区別も重要である．R は低レベルな抽象度のゴールに対応し，技術的関心事であると考える．また後述のように，要素間の関係を検討することによっても，このような区別の重要性を認識できる．

関係を表す共通用語として，次の 6 種類の概念を導入する．まず「貢献」は，ある要素に関し，G または R で表される関心事の達成に対する影響を表す．貢献には，正負の区別がある.「データフロー」は，R または OP が OB を入力，または出力とすることを表す.「依存」は，エージェントやゴールが他の要素に関心を持っていることを表す.「ゴール洗練」は，G または R が部分的関心事に洗練（詳細化）されることを表す.「操作可能化」は，G または R が操作実行により達成されることを表す.「関係性」は，オブジェクトが他のオブジェクトと関係を持つことを表す．これらの関係は，表 2 の最初の 3 列に示すように，さらに詳細に分類される．たとえば，貢献関係には正負があり，負の貢献関係は，関係を持つ要素に応じてさらに 3 種類に分類される．関係の各分類は，R1，R2 というような記号のラベルで示される．以下，要素と同様に，関係もこれらの記号で表すことにする．表 2 の 3 列目において，R->G のような矢印は,「R は G に対し正の貢献を持つ」といったように，関係の主語・述語を表す.

表 2 からは，次のように，R を導入して 2 種類の関心事を区別した最も重要な理由が分かる．G と OB は 1 つの関係（R13）しか持たないのに対し，R と OB は 2 つの関係（R7 と R8）を持つことに注意する．したがって，G が技術的関心事とは関係がないのに対し，R は技術的関心事を示すため，G は入出力の仕様を持たないのに対し，R はそのような仕様を持つことが分かる．

以上の関係の根拠は次のとおりである．

- 前述の 5 種類の要素に対し，KAOS，i*それぞれにおいて，対応する要素に対して定義されている関係を，既存の文献（KAOS [12),19),20)]，および i* [3),4),21)]）から抽出できる．その結果が，表 2 の KAOS における定義，および i*における定義の欄に示されている．
- 次に，抽出した各手法における関係に対し，同じ共通用語としてまとめられるものとそうでないものとを峻別した結果，24 種類の関係が定義できる．たとえば，KAOS における「オブジェクトが操作への入力である」という関係と，i*における「リソースがタスク（ただし共通用語における OP）への入力である」という関係は，同じ内容を表していると考えられるので，同じ共通用語 R5 としてまとめられる．一方，i*の手段・目的関係は，戦略的な関心事としての G とその実現手段（OP または R）との関係としても考えられるが，KAOS では G の実現手段を考えることはない（R に対しては操作可能化として実現手段を考えることができる）ので，KAOS には対応する概念のない共通用語 R23 として定義している．

3.2 KAOS と i*の違い

以上の共通用語を用いて，KAOS と i*を比較し，そ

表 2 関係の共通用語
Table 2 Common terminology for the relations.

用語	記号	関係の定義	KAOSにおける定義（文献12), 19), 20)	i*における定義（文献3), 4), 21)	KAOSとi*との違い
貢献	R1	ゴールの達成に対する正の貢献(G->G, R->G,O P->G,O B->G)	該当なし	ソフトゴールの達成に対する**正の貢献**	i*では,一方は必ずソフトゴール
	R2	ゴールの達成に対する負の貢献(G->G, R->G)	ゴールの集合は,同時に実現できない場合**競合する**	ハード,またはソフトゴールの達成に対する負の貢献	i*では,一方は必ずソフトゴールだが,KAOSではハードゴール同士の場合もある
	R3	ゴールの達成に対する負の貢献(G->R, R->R)	ゴールの集合は,同時に実現できない場合**競合する**	該当なし	i*では,一方は必ずソフトゴールだが,KAOSではハードゴール同士の場合もある
	R4	ゴールの達成に対する**負の貢献**(OP->G, OB->G)	該当なし	ハード,またはソフトゴールの達成に対する**負の貢献**	i*では,一方は必ずソフトゴール
データフロー	R5	オブジェクトが**操作への入力**(OP->OB)	オブジェクトは**操作への入力**	リソースは**タスクへの入力**	i*では,供給要望と同じ表記で表す
	R6	オブジェクトが**操作への出力**(OP->OB)	オブジェクトは**操作への出力**	リソースは**タスクからの出力**	i*では,供給被要望と同じ表記で表す
	R7	オブジェクトが操作への入力(R->OB)	該当なし	リソースは**タスクへの入力**	i*では,実現手段が指定されているので,その実現手段への入力がありうる
	R8	オブジェクトが操作への出力(R->OB)	該当なし	リソースは**タスクからの出力**	i*では,実現手段が指定されているので,その実現手段からの出力がありうる
依存	R9	**供給要望**:エージェントが他のエージェントにオブジェクトの**供給を要望する**(Ag->OB)	エージェントはオブジェクトの属性値を直接**監視する**	アクタは他のアクタにリソースを**供給することを要望する**	i*では,供給要望関係と供給被要望関係とをペアで使用する．一方,KAOSにはそのような制約はない.
	R10	**供給被要望**:エージェントがオブジェクトの**供給を要望される**(Ag->OB)	エージェントはオブジェクトの属性値を直接**制御する**	アクタはリソースを**供給することを要望される**	KAOSには，制御するエージェントは1つであるという制約がある.
	R11	**実行要望:エージェントが他のエージェントに操作または要件の実行を要望する**(Ag->OP, Ag->R)	該当なし	アクタはタスクが**実行されることを要望する**	i*では,実行要望関係と実行被要望関係とをペアで使用する．一方,KAOSにはそのような制約はない.
	R12	**実行被要望:エージェントが要件の実行を要望される**(Ag->R)	**責任**リンクは,ゴールがエージェントに割り当てられる可能性のある責任を意味する	アクタはタスクが**実行することを要望される**	i*では,エージェントはRの達成責任と操作責任の両方を負う．KAOSでは，OPの達成に関する責任のみを表す.
	R13	ゴールが**オブジェクトに関心を持つ**(G->OB)	該当なし	**手段・目的関係は,アクタがなぜタスクを実行するかについて理解させるもの**	i*はエージェント間の依存関係を分析する過程でオブジェクトを抽出する.
	R14	要件がオブジェクトに関心を持つ(R->OB)	ゴールにおいてオブジェクトが特定された場合,ゴールは**オブジェクトに関心を持つ**	該当なし	KAOSは，ゴールを形式かしたときに特定されたオブジェクトをオブジェクトモデルとして抽出する.
	R15	**対エージェント依存**:エージェントがゴールの達成に関し他のエージェントに**依存する**(Ag->G)	該当なし	**依存**:アクタがゴールの達成に関し他のエージェントに依存する	i*にはエージェント間の依存関係を陽に表すモデルがある.
	R16	**対エージェント被依存**:エージェントがゴールの達成に関し他のエージェントに**依存される**(Ag->G)	該当なし	**依存**:エージェントがゴールの達成に関し他のエージェントに依存される	i*にはエージェント間の依存関係を陽に表すモデルがある.
ゴール洗練	R17	**AND-OR分割によるゴール洗練**(G->G)	**ゴール洗練**	AND−OR**分割によるゴール洗練**	同じ
	R18	**手段・目的分析によるゴール洗練**(G->G)	ゴール洗練方法の1つである**マイルストーン駆動洗練パターン**	**手段・目的分析によるゴール洗練**	同じ (i*はR17とは異なる洗練方法を持つ．KAOSではこれらの関係をR17の特殊なパターンとして扱う)
	R19	**操作分割によるゴール洗練**(R->G, R->R)	ゴール洗練方法の1つである**事例駆動洗練パターン**	**タスク分割によるゴール洗練**	
操作可能化	R20	エージェントが操作を**実行する**(Ag->OP)	エージェントが操作を開始できるとき,エージェントは操作を実行する	エージェントがタスクの**実行を要望される**	i*では,操作関係の存在は,必ず実行要望関係の存在を仮定している
	R21	**操作可能化:操作の実行により要件が達成される**(OP->R)	**操作可能化**は,操作に対し要求された条件が満たされれば,ゴールの満足が保証されることを意味する	該当なし	KAOSでは，複数の選択肢の操作可能化をOR関係で表せる(R22と比較)．KAOSの操作可能化関係は,OPのRに対する関係を表す(R23と比較).
	R22	**操作分割:複合的タスクの階層的記述**(OP->R)	該当なし	**タスク分割**:志向的要素の階層的記述	i*では，タスクのAND-OR分割がある(R21と比較).
	R23	**操作の手段・目的分析:OPまたはRの実行によりゴールが達成される**(OP->G, R->G)	該当なし	**手段・目的関係**は,アクタがなぜタスクを実行するかについて理解させるもの	i*では，操作可能化関係は，OPまたはRのGに対する関係を表す（R21と比較）.
関係性	R24	オブジェクトが他のオブジェクトと**関係性を持つ**(OB->OB)	**関係性:オブジェクト間の継承関係を,IS−A関係によって宣言する.**	該当なし	静的な構造のみモデル化できる.

の違いを検討する．検討結果を，要素については表 1 に，関係については表 2 に示す．表 1 に示すように，KAOS のゴールについて，抽象ゴールは G に，具体ゴールは R に対応し，i*のタスクについて，抽象タスクは R に，具体タスクは OP に対応する．なお，KAOS の障害モデルにおける要素と関係の扱いは省略する．

KAOS と i*の主な違いの 1 つは，R の扱いである．R と OP は，次の点により KAOS では区別する．R は技術的観点を含み，ある動作を「どう」実行するかの情報を暗黙のうちに含んでいるが,「どう」実行するかの情報を明示してはいない．一方 OP は，具体的な操作の方法に関する情報を明示的に含んでいる．しかも，R と OP の間には，R が要求にあたり，OP は，その要求を「どのように」満たすか，についての仕様である，という関係がある．一方 i*においては，G と

R が次の点で区別される．G は関心事や利害関係といった「なぜ」の情報のみ示すが，R はある動作，あるいは手順（routine[21]）を「どのように」実行するかという情報を明示する．まとめると，KAOS と i* における R の扱いの主な違いは，R が「どのように」実行するかの情報を，暗黙のうちに示すか，明示的に示すか，ということである．

このような KAOS の R と i*の R との違いは，関係に関する違いに結び付いている．表 2 に示すように，KAOS では R と OB の間の依存関係 R14 があるのに対し，i*ではそれらの間に入出力関係（それぞれ R7 と R8）がある．KAOS の R は「どのように」実行するかを明示しないので，オブジェクトが R の入力か出力かを指定しない．

さらに，R の導入に対応して，用語の調整が必要である．すなわち，R により各方法論における関係の分類を詳細化しなければならない．たとえば，KAOS のゴールとして，R は操作可能化できるが，G はできない．i*のタスクとしては，R は分割できるが OP はできない．また，KAOS と i*の違いによる調整も必要である．すなわち，用語において同じ言葉で表される概念でも，各方法論において対応する概念を考慮すると，異なる意味を表すことがある．たとえば,「負の貢献」という言葉で 3 つの異なる意味の関係を表している．また，ゴール洗練は 3 種類あるが，KAOS では 1 種類のゴール間の関係として扱っている．

3.3 KAOS と i*のメタモデル

KAOS と i*のメタモデルを，共通用語を用いて構築する．メタモデルは 2 種類のグラフから構成される．1 つ目は要素・関係グラフと呼ばれ，**図 1** に示すような，ノードとアークがそれぞれ要素と関係でラベル付けされるグラフである．たとえば，G と Ag は，i*のメタモデルではそれぞれ関係 R16 と R17 で結ばれるが，KAOS のメタモデルでは，それらの間にアークはない．

要素・関係グラフは，KAOS と i*のそれぞれに対し，表 2 に示されている関係のアークにより，要素のノードを接続したグラフである．すなわち，本グラフは，ちょうど KAOS と i*のそれぞれに対する表 2 のグラフ表現であるため，3.1 節で示した表 2 の正当性により，グラフの正当性も確認できる．

Kavakli[11] もゴール指向要求工学のメタモデルを定義している．本論文のメタモデルとの違いは，ある特定の方法論を表現しようとしているのではなく，様々な方法論に共通して適用できるものを構成している点である．本論文のメタモデルは，KAOS と i*のそれぞれに対し個別に構成する．

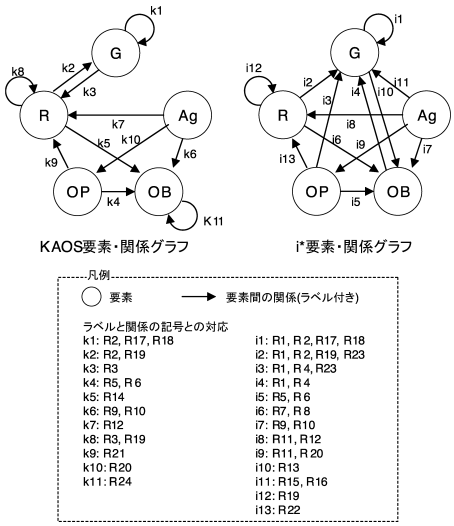
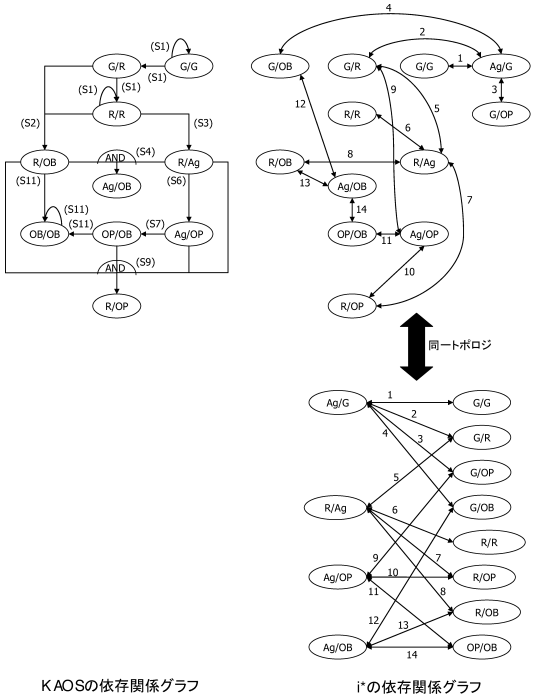


図 1 要素・関係グラフ
Fig. 1 Element-relation graph.

もう 1 種類のグラフは依存関係グラフと呼ばれ，**図 2** に示すような，分析プロセスにおける中間成果物としての，モデルの部分間の依存関係を表す．ノードのラベルは，関係で結ばれる要素の対を表し，アークはプロセスのステップにおいて注目される要素や関係を含むノードから，そのステップで生成される要素や関係を含むノードに向かって描かれ，そのステップでラベル付けされる．たとえば，KAOS において R/Ag から Ag/OP へ向かうアーク（S6）は，R，Ag，およびその間の関係（この場合は R12）に注目し，Ag，OP，およびその間の関係（この場合は R20）に関する情報を生成する．依存関係グラフは，方法論のプロセスそのものを示すものではないことに注意する．本グラフは，プロセスに対する制約のみを表す．なお，図 2 において，i*については同一トポロジの 2 通りの表記を示しているのは，次の理由による．まず上側の表記において，KAOS との比較のため，対応するノードを同じ位置に配置している．しかし，後述する i*の特徴を明確にするために，下側のような左右の 2 部に分かれた表記も合わせて示した次第である．

（S1）や（S2）といった，依存関係グラフのアークのラベルは，4 章における仮説に関する議論で用いる．

依存関係グラフの対象となるプロセスの範囲を明確にするために，プロセスへの入力データを定義する必要がある．KAOS では，通常トップレベルのシステムゴールと，環境において前提とされる事実（「前提」と呼ばれる）を入力として仮定する．一方 i*プロセスは，開発しようとするシステムだけでなく，現在の状態を



凡例

(S1) ゴール洗練	(S8) オブジェクトを読み書きする操作の特定
(S2) ゴールが関心を持つオブジェクトの特定	(S9) 操作の操作可能化
(S3) エージェントへの責任割当	(S10) 操作の分割・統合
(S4) エージェントからの供給要望・被要望関係の特定	(S11) オブジェクト間の関係性の特定
(S5) オブジェクトへの供給要望・被要望関係の特定	(S12) 実行要望・被要望関係の特定
(S6) エージェントが実行する操作の特定	(S13) エージェントが関心を持つゴールの特定
(S7) 操作が読み書きするオブジェクトの特定	(S14) 操作により達成されるゴールの特定

i*の依存関係グラフにおける、番号と依存関係のラベルとの対応表(ただし、方向は下側の表記のもの)

番号	1	2	3	4	5	6	7	8	9	10	11	12	13	14
右方向の依存関係	(S1)	(S6)	(S6)	(S2)	(S14)	(S10)	(S10)	(S7)	(S14)	(S10)	(S7)	(S13)	(S8)	(S8)
左方向の依存関係	(S3)	(S3)	(S3)	(S3)	(S12)	(S12)	(S12)	(S12)	(S12)	(S12)	(S12)	(S5)	(S5)	(S5)

図 2 依存関係グラフ
Fig. 2 Relation dependency graph.

分析し，KAOS における前提を，アクタ間の依存関係とともに分析する，初期フェーズ[21] を含んでいる．この初期フェーズの出力は戦略的依存関係（Strategic Dependency，SD）モデルと呼ばれる．本論文では，現在の状態に，トップレベルのシステムゴールと，関連する依存関係を加えた，開発しようとするシステムの SD モデルに対応する情報を，要求分析プロセスの入力データとする．

KAOS の依存関係グラフは，文献 12) に示されているプロセスにより正当化できる．すなわち，**図 3**，および 4.1 節の仮説 1 の検証において示したプロセスの，各ステップにおいて順次構築される，モデル間の依存関係をグラフ化することにより，本依存関係が得られる．ただし，R の導入は，グラフの最上部において，G が最初に R に洗練される部分にのみ影響する．一方 i*の依存関係グラフは，文献 2) の図 3（本論文の**図 4**）に示される，プロセスを表すアルゴリズムにより正当化される．このアルゴリズムは，i*のプロセスが，無限回の可能性があるループとなることを示している．1 回のループは，要素の導入，関係の導入，および要素に対し責任を持つエージェントを特定する責任分析の 3 種類のステップの逐次実行である．アクタと関係以外の要素の導入は，文献 3) で論じられているように，1 ステップと見なせる．本論文では，分析プロセスへの入力データに関する仮定により，システムを含む必要なアクタは分析の前にすでに検討が完了しているので，分析プロセスにおいて新たなアクタは導入されない．したがって，アクタ以外の導入ステップと責任分析ステップのみ考えれば十分である．依存関係グラフにおいては，導入ステップは右向きのアークに対応し，責任分析ステップは左向きのアークに対応する．なお，Bresciani ら[2] はエージェント指向開発方法論である Tropos について論じているが，その要求分析フェーズは i*そのものであることに注意する．また，i*は，OP がこれ以上分割できないことを除き，R と OP を区別せず，どちらもタスクとして扱う．依存関係グラフにおいては，R が関係する部分と，OP が関係する部分は，ノード OP/OP がないことを除き，正確に対応している．たとえば，Ag/G から G/R と G/OP に向かうアークは 2 つあり，どちらも（S6）のラベルが付いている．

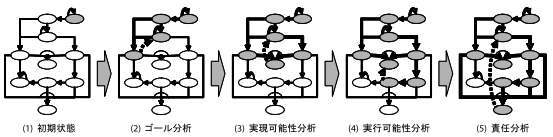


図 3 KAOS プロセスの構造
Fig. 3 Organization of the KAOS process.

```
BEGIN
‘ initialize ’graph G (* in general empty *)
REPEAT
WHILE G <> ‘ desired graph ’AND
‘ there is at least one applicable rule in the INTRODUCTION RULES set ’;
DO
‘ chose an applicable rule ’r=<L,R> ‘ in the INTRODUCTION RULES set ’;
‘ chose an occurrence isomorphism ’i‘ for the application of ’r;
G := (G \ i(L\R)) + i(R\L)
DONE;
WHILE G <> ‘ desired graph ’AND
‘ there is at least one applicable rule in the ANALYSIS RULES set ’;
DO
‘ chose an applicable rule ’r=<L,R> ‘ in the ANALYSIS RULES set ’;
‘ chose an occurrence ’i‘ for the application of ’r;
G := (G \ i(L\R)) + i(R\L)
DONE;
WHILE G <> ‘ desired graph ’AND
‘ there is at least one applicable rule in the DELEGATION RULES set ’;
DO
‘ chose an applicable rule ’r=<L,R> ‘ in the DELEGATION RULES set ’;
‘ chose an occurrence ’i‘ for the application of r ’;
G := (G \ i(L\R)) + i(R\L)
DONE
UNTIL G = ‘ desired graph ’OR‘ no applicable rules left ’;
IF G = ‘ desired graph ’THEN RETURN(G)
ELSE FAIL
END
```

図 4 文献 2) の図 3 で示されるアルゴリズム
Fig. 4 Algorithm shown as Figure 3 in literature 2).

4. 仮説と検証

本論文で提案する共通用語の目的は，ゴール指向要求工学の体系化の基礎として，KAOS と i*の特徴を明確にし，それらの比較・検討を可能にすることである．そこで本章では，そのような目的が達成されたかどうかをチェックするために，既存の文献で議論されてきたような，KAOS と i*の特徴の比較に関する主張を，3 件の仮説として提示し，共通用語とメタモデルを利用して検証する．

4.1 分析プロセスの構造に関する仮説

Yu[21] P.8「4. Related Work」では，「(KAOS など) i*以外のほとんどの手法は，ゴールに対する全体的な観点を仮定し，主にトップダウンな方式で分析する（が，i*はそうではない）」と主張している．本仮説は，その主張を部分的に定式化するものである．

仮 説 1

KAOS のプロセスは，順に実行する 4 フェーズに分かれ，i*のプロセスは，交互に実行する 2 フェーズに分かれる．

検 証

KAOS プロセスの構造については，次のように検証できる（図 3 参照）．図 3 では，各ステップにおいて，依存関係グラフのどのノードが作成されるかを示すことにより，プロセスを表現している．すなわち，太実線で描かれたアークによって，灰色で示されたノードが新規に作成されることを表している．これらのフェーズは，図 3 において太点線で示された，文献 12) において説明されている各 KAOS モデルに対するメタ制約に関係している．

- ゴール分析フェーズ（図 3 の (2)）では，ステップ S1 において，エージェントに割り当てることができるようになるまでゴールを要件に分割し，ステップ S2 において要件の関心事からオブジェクトを特定する．その結果，関係 R14，R17，R18，および R19 を得る．本フェーズは，ゴールモデルとオブジェクトモデルの間のメタ制約[12] に関係する．すなわち，ゴール定義で使用されたすべての用語は，オブジェクトモデルで宣言されなければならないことを検証する．
- 実現可能性（realizability）分析フェーズ (3) では，ステップ S3 において各要件 R に責任を持つ可能性のあるエージェント Ag を割り当て，ステップ S4 において Ag が監視，または制御できるオブジェクトを特定する．その結果，関係 R9，R10，および R12 を得る．本フェーズは，実現可能性メタ制約（realizability meta-constraint）と，単独制御メタ制約（unique control meta-constraint）[12] に関係する．実現可能性メタ制約は，ゴールに対してエージェントの責任を割当て可能か否かをそのエージェントの監視と制御の能力に基づいて特定するための基準を与えるものである．一方，単独制御メタ制約は，各オブジェクトの属性はたかだか 1 つのエージェントによって制御されることを表す．
- 実行可能性（executability）分析フェーズ (4) では，ステップ S6 において Ag が実行できる操作 OP を特定し，ステップ S7 において OP が読み取り，または書き込みを行うオブジェクトを特定する．その結果，関係 R5，R6，および R20 を得る．本フェーズは，入出力メタ制約（input/output meta-constraint）[12] に関係する．すなわち，操作の入力（出力）となる任意のオブジェクト属性は，操作を実行するエージェントによって監視（制御）されなければならないことを検証する．
- 責任分析フェーズ (5) では，Ag が，OP の実行によって確実に R を満足する責任を実際に負えるかどうかをチェックする．その結果，関係 R21 を得る．本フェーズは，責任メタ制約（responsibility meta-constraint）[12] に関係する．すなわち，ゴールは，責任あるエージェントによって実行される操作のみを強調することによって操作可能化されることを検証する．

以上のように，各フェーズのマイルストーンは，フェーズの出力がメタ制約を満たすかどうかのチェックに応じて設定される．

i*プロセスの構造は次のように検証できる．i*の依存関係グラフ（図 2）は，内部にアークを含まない 2 つの部分に分かれる．一方は Ag が関係しているノードのみから構成され，もう一方は Ag が関係していないノードのみから構成されている．プロセスのステップも，グラフのどちらの部分からもう一方へ向かっているかによって，2 種類に分かれる．いい換えると，図 2 において左方向か右方向か，の 2 種類である．左方向のステップは S3，S5，および S12 で，残りは右方向である．したがって，プロセスは 2 つの交互に実行されるフェーズに分かれ，各フェーズは片方向のステップのみから構成される．**図 5** は，i*プロセスが依存関係グラフによりどのように表されるかを示す．図 3 と同様に，図 5 はプロセスの進行を，各ステップにおいて作成される依存関係グラフのノードにより示している．ただし，本図は 1 例のみ示している．太実線で描

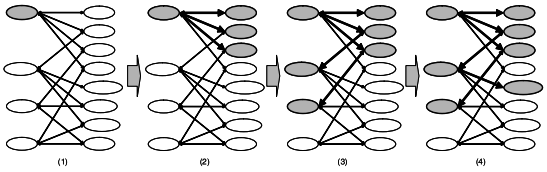


図 5 i*プロセスの構造
Fig. 5 Organization of the i* process.

かれたアークで示されたステップにより，灰色で示されたノードが新規に生成される．このように，左右の各部分の関係を，新規に生成する一連のステップが交互に繰り返される．

4.2 分析対象モデルに関する仮説

一般に要求工学方法論は，システム要求の分析を目的とするが，分析の対象となるモデルの詳細は，個々の方法論により異なる．KAOS と i*に関しては，次のような違いがある．まず KAOS について，Letier[12] P.2 第 2 段落では，「本論文における我々の一般的な目的は，複合的なシステムの高レベルなゴールから，そのゴールを満たすエージェントの割当ての選択肢を求めることである」と説明している．また i*について，Yu[21] では，i*はステークホルダの関心を記述すると主張しており，さらに Yu[21] P.8 左第 3 段落では，「ゴールを達成する責任はあるエージェントから別のエージェントに委譲されるかもしれない」とある．このように，i*が Ag（アクタ）の関心・責任を記述・分析するものであるとしている．本仮説は，これらの主張を定式化する．

仮 説 2

- KAOS はプロセスの中間成果物の間の「実装」関係，すなわち，ある中間成果物が他の成果物の実装である，という関係のモデルを持っている．KAOS は，分析プロセスにおいて，そのような関係が満たされているかどうかをチェックする方法論である．
- i*は，Ag 間の関心・責任関係のモデルを持ち，そのような関係を分析する．

検 証

KAOS に関する主張の検証のために，図 2 の各フェーズにおける実装関係を，文献 12) で説明されている，要素と関係の間のメタ制約の観点からチェックできることを示す．

- ゴール分析フェーズは，ゴール，要件，およびオブジェクトを特定し，後のフェーズで実装関係をチェックする際に参照する．
- 実現可能性メタ制約の目的は，要件がエージェントに対し責任として割り当てられるかどうかを，エージェントの監視・制御能力に基づいて判断する正確な基準を与えることである[12]．すなわち，ソフトウェアエージェント Ag とオブジェクト OB の間の関係 R9 と R10 は，Ag が要件 R を満たすという要求でなければならない，という制約である．これにより，R9 と R10 が R の実装か否かをチェックできる．
- 入出力メタ制約は，操作の入力（または出力）の部分となるすべてのオブジェクトが，操作を実行するエージェントにより監視（または制御）できる，という事実を示している[12]．すなわち，操作 OP とオブジェクト OB の間の関係 R5 と R6 が OP の仕様であり，ソフトウェアエージェント Ag と OB の間の関係 R9 と R10 が OP への要求でなければならない，という制約である．これにより，R5 と R6 が R9 と R10 の実装か否かをチェックできる．
- 責任メタ制約では，責任を持つエージェントにより実行される操作を詳細化するだけで，ゴールを操作可能化できることが要求される[12]．すなわち，OP とオブジェクト OB の間の関係 R5 と R6，および OP とソフトウェアエージェント Ag の間の関係 R20 が，要件 R を満たすための仕様である，という制約である．これにより，R5，R6，および R20 が R を実装しているか否かをチェックできる．

i*に関する主張は次のように検証できる．

- 図 2 において右方向のステップ（S1，S2，S6，S7，S8，S10，S13，S14）はすべて，G，R，OP，または OB（すなわち，Ag 以外の要素）の間の関係を分析し，関係，あるいは必要に応じて要素を生成する．ただし，これらのステップはつねに Ag の関心の観点を考慮している．
- 左方向のステップ（S3，S5，S12）はすべて，Ag の，G，R，OP，または OB への関心，または責任を特定し，関係を生成する．
- まとめると，i*プロセスのすべてのステップは，Ag の関心を考慮し，左方向のステップはすべて Ag の責任も考慮する．このように i*は，関心・責任関係を分析する方法論と見ることができる．

Yu[21] が指摘するように，i*は志向的関係を再構成することにより進められる方法論である．本仮説の検証により，このような主張が次のように表現できることが分かる．すなわち，i*プロセスは，アクタ以外の要素を詳細化する部分と，新規に作成された要素をアクタに割り当てる部分から構成される．その結果，志

向的関係が再構成されることになる．

4.3 G，R，および OP の区別に関する仮説

Yu[21] P.7 第 2 段落では，「これらのソフトゴールの発見は，手段・目的分析（すなわち，「どのように」実行するか，の分析）によっても支援できる」と主張している．Letier[12] P.2 最後の段落では，「ゴール指向詳細化プロセスの次のステップは，ゴールを満たすために，エージェントが実行する操作を導出することである」と主張している．本仮説は，これらの主張を定式化し，G，R，および OP の区別に基づいて検証する．

仮 説 3

- i*において，要素が「どのように」実行するか，の情報を含むか否かの区別は，要素のソフトゴールへの影響を評価するのに有用である．
- KAOS において，要求と「どのように」実行するかに関する使用との区別は，両者の間の満足可能性（satisfiability）の分析に有用である．

検 証

i*に関する主張については，G，R，および OP のソフトゴールに対する影響の違いを検討する．G の影響については，G で表現されている「なぜ（why）」に関する情報しか関係しないので，影響の分析には「どのように」実行するかを考慮する必要はない．一方，R と OP の影響の分析については，ソフトゴールの満足度の評価において「どのように」実行するかの情報が非常に重要である．たとえば，「なぜ」に関する情報として同じものを考えた場合でも，信頼性やセキュリティの強度は大きく異なる可能性がある．このように，「どのように」実行するかの情報に関する区別は有用である．KAOS と i*のメタモデルを比較すると，i*のメタモデルには OP の G に対する負の貢献関係（R4）があるのに対し，KAOS のメタモデルには対応するものがない．これは，ソフトゴールへの影響の分析における「どのように」実行するかの情報の重要性を示す証拠であると考えられる．

KAOS に関する主張は，仮説 1 と 2 の結論から導くことができる．

5. 考 察

5.1 仮説の結果のまとめ

仮説の結果を以下のようにまとめ，本論文の議論により KAOS と i*の比較検討が実施できたことを示す．

- 分析プロセスの構造について
  KAOS プロセスは順次実行される 4 フェーズに分かれ，i*プロセスは交互に実行される 2 フェーズに分かれる（仮説 1）．KAOS に関する結果は，KAOS プロセスがおおむねトップダウンに進むという主張に対応している．
- 分析対象モデルの特徴について
  KAOS は，モデル間の関係に要求仕様・実装関係を持っており，分析プロセスにおいてその関係が満たされるか否かをチェックする方法論である．一方 i* は Ag 間の関心・責任関係のモデルを持っており，その関係を分析する（仮説 2，3）．

5.2 用語とメタモデルの評価

本節では，仮説検証の結果を評価する．本論文の目的は，方法論の比較により，共通用語を提案し，ゴール指向要求工学一般に共通した特徴と，各方法論に特化した特徴の両方を議論することである．したがって，共通用語とメタモデルがそのような特徴を表していて，仮説検証によりそれらの特徴に関する比較研究が適切に実施されているかどうかを評価する．

5.2.1 用語とメタモデルの適切さ

まず，用語とメタモデルが，ゴール指向固有の特徴を表しているかどうかをチェックする．用語により，ゴールの抽象レベルの区別が表現されており，その区別が固有の特徴として利用可能であることは，次のようにして分かる．用語において要件 R の概念が導入され，これにより，KAOS のゴールと操作，および i*のゴールとタスクが区別される．i*の R と G の違いは，前者が「どのように」何かを実行するかの情報を持っているのに対し，後者は持っていない，という点であることが，R の導入により明確になっている．仮説 3 の検証において，この区別を利用して，G と R のソフトゴールに対する影響の違いを議論している．

次に，共通用語が KAOS と i*に特化した特徴を表しているかどうかをチェックする．

表 1，表 2 に示すように，ほとんどの用語は両方法論の構成において共通に使用されている．一方，共通用語により 2 つの方法論の違いを多数見つけることができる．すなわち，用語の概念について，どちらか一方の方法論には存在しないものがある．また，両方法論に存在する概念についても，対応する言葉の意味は通常異なっている．したがって，共通用語は KAOS と i*の比較の基礎として十分有用であるといえる．

本論文では，方法論の構成要素間の関係を定義し，メタモデルにおいてその依存関係を示した．仮説 1 の検証では，依存関係グラフがプロセスの制約を表しているという事実に基づいて，プロセスの構造を論じた．仮説 2 の検証においては，要素間の論理的関係を表す依存関係の意味を利用した．このように，用語とメタモデルは，プロセスの構造に関して，KAOS と i*の

異なる特徴を表している．

5.2.2 仮説の適切さ

仮説の目的は，用語とメタモデルを利用して，代表的なゴール指向要求工学方法論である KAOS と i*の特徴を比較し議論することである．議論においては，両方法論の表面的な違いを扱うだけでなく，ゴール指向固有の特徴も論じなければならない．このような観点から，仮説が適切かどうかをチェックする．

仮説 1 はプロセスの構造を論じた．実際に方法論を適用する際にプロセスの構造を考慮することは，分析工程の作業負荷を評価し，マイルストーンをチェックすることにより進捗を管理するための基礎となるので重要である．したがって，本仮説の検証は，プロセスの観点から方法論の特徴を明確にするのに有用である．

仮説 2 については，KAOS ではゴールモデルやオブジェクトモデルなど，モデル間のメタ制約を規定しており[12]，仮説 2 で扱っている要求・実装関係はメタ制約により定式化できる．さらに，分析ステップの実施順序が，仮説におけるメタ制約のチェック順序に従うような，基本的にトップダウンの方向に進むことも明らかにしている．i*については，プロセスにおいて交互に実施されるステップの集合の存在，および関心・責任関係がどのように分析されるかを明らかにしている．したがって本仮説は，Yu が文献 21) で述べたような比較を，本論文の用語とメタモデルにより明確にしている．

仮説 3 は KAOS における要求仕様と実装仕様の間の関係を論じている．この関係は，上述のゴール・要求の関係と同様に重要であると考えられる．一方，i*はタスクとソフトゴールの間の関係を重視する．たとえば，Yu[21] はソフトゴールへの影響の観点から，タスクの実行可能性（workability）と実現可能性（viability）の区別を論じている．仮説 3 の検証により，i*におけるこのような影響と，関連する概念を容易に検討できることを示している．

5.2.3 仮説検証の正しさ

本論文では，仮説検証により用語とメタモデルの適切さを示したと主張しているので，検証が用語とメタモデルに基づいて実施されており，前提から導出できない仮定を新たに導入してはいないことを確認する必要があるが，これは検証の内容を調べることによりチェックできる．

6. 関 連 研 究

Yu[21] は i*の基本的な概念を紹介している．本文献は，アクタの利害関係のモデル，アクタの志向性，および手段目的分析といった，i*の重要な特徴をいくつか示している．i*および Tropos に関する文献の中にも，同様な特徴を論じたものがある[4)~6),14)]．しかし，これらの議論は共通用語に基づいてなされていない．さらに，これらの特徴は i*を KAOS から区別するのに重要な点であるが，比較検討が十分でない．本論文では，それらの特徴の比較による議論を可能にするための基礎として，用語とメタモデルと提案している．

KAOS と i*に特化した様々な特徴を明らかにする研究がいくつかある．i*については，モデル変換によるプロセスの定式化[2),3)]，ソフトゴールの満足度の分析[8)]，アスペクトの概念の取り込み[23)]，およびアクタ依存関係モデルのメトリクス[7)] の研究がある．KAOS についても，ソフトゴールの満足度の分析の研究[13)] がある．これらの研究は，各方法論個別になされているが，ほとんどの観点は他の方法論にも適用でき，ゴール指向要求分析の本質の構成要素の候補ともなりうると考えられる．本論文の用語とメタモデルは，そのような適用の基盤として利用できるといえる．たとえば，モデル変換の枠組みは，用語を用いることによってさらに一般的な状況に拡張できる．

KAOS と i*のメタモデルはすでにいくつか提案されている．特に，UML に基づくメタモデルが近年検討されており[4),9),15)]，前の 2 つが i*に対するもので，最後の 1 つが KAOS に対するものである．しかし，これらの研究は，メタモデルを提示し，そのメタモデルを利用して例をいくつか記述しているのみである．一方本論文では，KAOS と i*の特徴を比較検討するための基礎としてメタモデルを構築している．したがって，そのためにまず共通用途を提示し，その用語を利用してメタモデルを構成している．またメタモデルでは，プロセスの特徴の議論のために依存関係グラフを用意している．

方法論工学[24)] は，ソフトウェア設計方法論に対し，メタモデリングなどの手段により，ソフトウェア要求分析・設計工程全体，あるいは部分的に，新規構築・比較・分析・統合などの作業の支援を行う．従来の方法論工学では，構造化手法やオブジェクト指向方法論といった，一貫性を考慮して提案されている方法論を対象としている．一方本論文は，ゴール指向要求分析手法という，要求分析工程に特化した手法に対し，比較・分析を行うものである．そのため，メタモデルの分析という方法論工学と同様の手法を用いてはいるものの，ゴール指向要求分析における共通用語の抽出や，依存関係グラフの作成などを実施している点が特徴である．

7. おわりに

本論文では，ゴール指向要求工学に関し，代表的な方法論である KAOS と i*に絞り，体系化を可能にするような共通用語を提案した．用語は，方法論で使用するものと，要件と呼ばれる新規の概念を含む 5 種類の構成要素を用意し，また構成要素の間の関係を含めた．それらの用語を使用することにより，KAOS と i* のメタモデルを構築し，2 つの方法論の比較を行って，ゴール指向要求工学に固有の汎用的特徴と，各方法論に特化した特徴とを議論することができる．さらに 3 件の仮説を検証することにより，共通用語とメタモデルが，ゴール指向要求工学の体系化の基礎となりうることを示した．

本論文で提案した用語は一般的なものであり，問題フレーム手法[10] など，ゴール指向に限らず，他の要求分析手法にも適用できると考える．このように，今後は検討の範囲を拡大することを目指す．また，1 章で述べたように，本論文で提案した共通用語は，KAOS と i*に対してのみ行った分析に基づいており，ゴール指向手法全般に関する十分性，必要性の検討は今後の課題である．

謝辞 研究の機会を与えてくださいました，国立情報学研究所坂内正夫所長には，深く感謝いたします．

参考文献

1) Anton, A.I., Dempster, J.H. and Siege, D.F.: Managing Use Cases During Goal-Driven Requirements Engineering: Challenges Encountered and Lessons Learned, Technical Report TR-99-16, North Carolina State University (1999).
2) Bresciani, P. and Giorgini, P.: The TROPOS Analysis Process as Graph Transformation System, *Proc. Workshop on Agent-Oriented Methodologies* (2002).
3) Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F. and Mylopoulos, J.: Modeling Early Requirements in Tropos: A Transformation Based Approach, *Proc. AOSE 2001*, pp.151–168 (2001).
4) Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F. and Mylopoulos, J.: Tropos: An Agent-Oriented Software Development Methodology, *Autonomous Agents and Multi-Agent Systems*, Vol.8, No.3, pp.203–236 (2004).
5) Castro, J., Kolp, M. and Mylopoulos, J.: A Requirements-Driven Development Methodology, *Proc. CAiSE'01*, pp.108–123 (2001).
6) Castro, J., Kolp, M. and Mylopoulos, J.: Towards Requirements-Driven Information Systems Engineering: The Tropos Project, *Information Systems*, Vol.27, No.6, pp.365–389 (2002).
7) Franch, X., Grau, G. and Quer, C.: A Framework for the Definition of Metrics for Actor-Dependency Models, *Proc. RE'04*, pp.327–328 (2004).
8) Giorgini, P., Mylopoulos, J., Nicchiarelli, E. and Sebastiani, R.: Reasoning with Goal Models, *Proc. ER 2002*, pp.167–181 (2002).
9) Heaven, W. and Finkelstein, A.: A UML Profile to Support Requirements Engineering with KAOS, *IEE Proceedings Software*, Vol.151, No.1, pp.10–27 (2004).
10) Jackson, M.: *Problem Frames: Analyzing and Structuring Software Development Problems*, Addison-Wesley (2000).
11) Kavakli, E.: Goal-Oriented Requirements Engineering: A Unifying Framework, *Requirements Engineering*, Vol.6, No.4, pp.237–251 (2002).
12) Letier, E.: Reasoning about Agents in Goal-Oriented Requirements Engineering, Ph.D. Thesis, Université Catholique de Louvain (2001).
13) Letier, E. and van Lamsweerde, A.: Reasoning about Partial Goal Satisfaction for Requirements and Design Engineering, *Proc. FSE'04*, pp.53–62 (2004).
14) Mylopoulos, J. and Castro, J.: Tropos: A Framework for Requirements-Driven Software Development, *Information Systems Engineering: State of the Art and Research Themes*, Brinkkemper, J. and Solvberg, A. (Eds.), pp.261–273, Springer (2000).
15) Mylopoulos, J., Kolp, M. and Castro, J.: UML for Agent-Oriented Software Development: The Tropos Proposal, *Proc. UML2001*, pp.422–431 (2001).
16) Nuseibeh, B. and Easterbrook, S.: Requirements Engineering: A Roadmap, *The Future of Software Engineering*, ACM (2000).
17) van Lamsweerde, A.: Requirements Engineering in the Year 00: A Research Perspective, *Proc. ICSE 2000*, pp.5–19 (2000).
18) van Lamsweerde, A.: Goal-Oriented Requirements Engineering: A Guided Tour, *Proc. RE'01*, pp.249–263 (2001).
19) van Lamsweerde, A., Darimont, R. and Letier, E.: Managing Conflicts in Goal-Driven Requirements Engineering, *IEEE TSE, Special Issue on Manging Inconsistency in Software De-*

velopment, Vol.24, No.11, pp.908–926 (1998).
20) van Lamsweerde, A. and Letier, E.: Handling Obstacles in Goal-Oriented Requirements Engineering, *IEEE TSE, Special Issue on Exception Handling*, Vol.26, No.10, pp.978–1005 (2000).
21) Yu, E.: Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering, *Proc. RE'97*, pp.226–235 (1997).
22) Yu, E. and Mylopoulos, J.: Why Goal-Oriented Requirements Engineering, *Proc. REFSQ'98*, pp.15–22 (1998).
23) Yu, Y., Leite, J. and Mylopoulos, J.: From Goals to Aspects: Discovering Aspects from Requirements Goal Models, *Proc. RE'04*, pp.33–42 (2004).
24) 鯵坂恒夫，佐伯元司：方法論工学と開発環境，共立出版 (2001).

(平成 17 年 4 月 27 日受付)
(平成 17 年 11 月 1 日採録)

田原 康之（正会員）

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 年より国立情報学研究所に勤務．2004 年より同研究所特任助教授．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等の研究に従事．情報処理学会，日本ソフトウェア科学会会員．

長野 伸一（正会員）

1971 年生．1996 年大阪大学大学院基礎工学研究科物理系専攻博士前期課程修了．1999 年大阪大学大学院基礎工学研究科情報数理系専攻博士後期課程修了．博士（工学）．同年（株）東芝入社．現在，同社研究開発センター知識メディアラボラトリー所属．2004 年より国立情報学研究所特任講師を兼任．主に，ソフトウェア工学，エージェント技術の研究に従事．電子情報通信学会，IEEE CS 各会員．

吉岡 信和（正会員）

1993 年富山大学工学部電子情報工学科卒業．1995 北陸先端科学技術大学院大学情報科学研究科博士前期課程修了．1998 年同大学院大学情報科学研究科博士後期課程修了．博士（情報科学）．同年（株）東芝入社．2002 年より国立情報学研究所に勤務，2004 年より同研究所特任助教授，主にエージェント技術，およびソフトウェア工学の研究に従事．現在に至る．日本ソフトウェア科学会会員．

本位田真一（正会員）

1953 年生．1976 年早稲田大学理工学部電気工学科卒業．1978 年同大学大学院理工学研究科電気工学専攻修士課程修了．(株）東芝を経て 2000 年より文部科学省国立情報学研究所．現在，研究主幹・教授，2001 年より東京大学大学院情報理工学系研究科教授を併任，現在に至る．2002 年 5 月〜2003 年 1 月英国 UCL ならびに Imperial College 客員研究員（文部科学省在外研究員）．2005 年度パリ第 6 大学招聘教授．工学博士（早稲田大学）．1986 年度情報処理学会論文賞受賞．エージェント技術，オブジェクト指向技術，ソフトウェア工学の研究に従事．IEEE，ACM，日本ソフトウェア科学会等各会員．本学会理事．