KAOSによるセキュリティ要件の獲得・分析

田原 康之[*1]　Axel van Lamsweerde[*2]　Emmanuel Letier[*3]

[*1] 電気通信大学　[*2] Université catholique de Louvain　[*3] University College London

近年システムに対するセキュリティ上の脅威が多様化・複雑化し，変化も急速になりつつあるため，現在採用されているようなセキュリティ要件の分析手法では，セキュリティ要件を抜け・漏れなく的確に分析し，迅速にシステム開発・運用に反映させることが困難になってきている．一方，システム・ソフトウェアへの一般的な要求に関する同様な課題への解の１つとして，ゴール指向要求分析手法が注目されている．本稿では，代表的なゴール指向要求分析手法の１つであるKAOS（Knowledge Acquisition in autOmated Specification）に基づいた手法を解説する．KAOSは，システムゴール，すなわち抽象レベルがさまざまに異なっている要求を系統的に分析する手法である．本稿では，システムゴールとしてセキュリティゴール，あるいは逆に攻撃者のゴールを分析することにより，セキュリティ要件の獲得・分析を行う手法を扱う．

セキュリティ要求工学の課題とKAOSへの期待

インターネットに代表される，急速に大規模化・複雑化・グローバル化しつつある今日のコンピュータネットワークシステムにおいては，セキュリティの問題はますます重大なものになってきている．たとえば，セキュリティが侵害された場合の被害はますます増大している．一方で，脆弱性を発見・修正したり攻撃を防止したりするために必要なコストも増大している．またシステムの機能の高度化に伴い，潜在的なセキュリティ攻撃の種類も，特にWebベースのシステムでは急増している．さらにセキュリティの強化を図る場合，性能や使いやすさといった他の非機能的要求とのトレードオフも考慮しなければならない．このような状況においては，従来のような専門家の経験と感覚に頼ったセキュリティ対策では，もはや対応しきれなくなってきている．

このようなセキュリティ問題の原因には，人間系の問題や物理的なものもあるが，ソフトウェアの欠陥が大きな割合を占める．ソフトウェアの欠陥によるセキュリティの問題に対しては，稼働中のソフトウェアの更新といった，その場しのぎの対策が中心となっているのが現状である．しかし，急速に進展しつつあるコンピュータシステムにおいて，日々新たなセキュリティ問題が発生している状況の下では，そのような対策では限界がある．可能な限り根本的な対策が必要である．

セキュリティに限らず，ソフトウェアの欠陥への対策は，ソフトウェア工学の中心課題の１つである．基本方針として，可能な限り開発プロセスの上流の段階で，潜在的な欠陥を少なくすべきである，という認識が浸透している．特に，最も上流の部分に属する要求分析工程が重視されている．すなわち，ユーザを中心としたステークホルダ（システムに対する利害関係者）の要求を，可能な限り抜け・漏れなく抽出し，要求間の因果関係や矛盾を明確にすることにより，要求を満たさないことに起因するソフトウェアの欠陥の防止を目指している．

以上の考えに基づき，セキュリティに関する要求工学（以下，単に「セキュリティ要求工学」と称する）の研究が近年盛んである．セキュリティは，要求分析工程においては，利用容易性，信頼性，性能などと並んで，非機能的要求の１つとして数えられている．しかし，セキュリティが他の非機能的要求と決定的に異なる点がある．それは，セキュリティを侵害する主体としての攻撃者の存在である．攻撃者は，独自の意図を持ち，対象システムのセキュリティを阻害するための独自の手段（攻撃手段）を行使する．したがって，セキュリティ要求分析においては，通常の要求分析が対象とするシステム構成要素やステークホルダのほかに，攻撃者と攻撃手段のモデルを構築し，本来の要求モデルとの関係性を明確にしなければならない．

このような考察により，セキュリティ要求工学においては攻撃者の意図が重要となる．一方，前述したコスト・性能・使いやすさといった，他の非機能要求とのトレードオフの解決などの課題を考慮すると，どの程度のセキュリティを望んでいるのかという，ステークホルダの意図も同時に重要である．そこで，要求の背後にある人間の意図を重視した要求工学手法として，ゴール指向

要求工学が注目されている．本章では，ゴール指向要求工学の代表的な手法の１つである KAOS に基づいたセキュリティ要求の獲得・分析手法について解説する．

KAOSとは？

KAOS は，i* と並んで代表的なゴール指向要求工学手法の１つである．そこで本章では，ゴール指向要求工学について概説した後，KAOS について説明する．

◆ゴール指向要求工学とは？

通常要求工程は，ステークホルダ（利害関係者）による会議，あるいはヒアリングやインタビューなどによって得られる，システムへのニーズに関する断片的な情報が入力となる．そして最終的な成果物として，次工程である設計への入力として適切に記述された要求仕様書が出力となる．「適切に記述された」というのは，設計を開始するのに十分な，システムへのニーズに関するデータが，明確かつ詳細に記述されていることを意味する．すぐ分かるように，これらの入力と出力の間には，大きなギャップがある．

要求工程においては，入力から出力を導出するために，このギャップを埋める必要があるが，そのための概念としてゴールが重要であることが，研究者の間で広く認識されてきた．なぜならば，上記の入力と出力は無から生じたわけではなく，システムへのニーズに関する，より抽象度が高い要求や，より上位の目的が背後にあるはずなので，そのような要求や目的をゴールとして明確にすることが，ギャップを埋めることにつながるからである．

しかしほとんどの要求工学の技術は，下位レベルの要求を中心に扱うような，要求工程の後半部分を対象としてきた．一方ゴール指向要求工学は，要求工程の前半部分として，ステークホルダのゴールを洗い出し，そのゴールを満たすようなシステム提案の選択肢を明らかにすることを目標とするものである．

◆ KAOSとは？

KAOS は，ゴール指向要求工学の枠組みの１つであり，抽象度の高いレベルのゴールから，システム設計の選択肢を生成することを主に目指したものである．KAOS では，ゴールを AND/OR 詳細化リンクによりサブゴール（部分ゴール）と関係付ける．ゴールがサブゴールの集合に AND- 詳細化されるのは，全サブゴールが満たされると親ゴールも満たされる場合である．一方 OR- 詳細化リンクは，ゴールを別の AND- 詳細化の集合と関係付ける．その意味は，複数の詳細化のうちの１つでも満たされれば，親ゴールも満たされる，ということである．また，同時に満たすことができないゴールを関係付けるために，ゴール間競合リンクも使用する．

次に，ゴール指向要求工学において，ゴールと並んで重要な概念がエージェントである．なぜなら，ゴールはさまざまなエージェントの協調により達成されるものだからである．エージェントは，環境における，既存の，および開発対象のソフトウェアコンポーネント，外部のデバイス，ならびに人間を含む．その結果，要求工学プロセスにおいて扱われるシステムは複合的なものとなる．なぜなら，そのようなシステムは開発対象のシステムと環境との両方を含むからである．

KAOS では，ゴールに基づいて複数のエージェントを含む設計の選択肢を洗い出し，またエージェントの誤った振舞いに対処する．前者においては，さまざまなゴールのサブゴールへの詳細化の仕方や，エージェントへのゴール達成責任のさまざまな割り当て方により異なる設計が得られる．この場合，自動化されたシステムとその環境との境界がまったく異なることもあり得る．またゴールは，システムにおいてある役割を果たすべきエージェントの特定にも使用される．システム全体に関するゴールの達成が必要な場合には，新しいエージェントを導入する．エージェントの誤った振舞いへの対処については，まずゴールのサブゴールへの詳細化が，環境内のエージェントに対する仮定をおいていることに注意する．すなわち，ゴールと仮定はエージェントの理想的な振舞いを定義している．エージェントの誤った振舞いは，これらのゴールと仮定の侵害を引き起こす例外として定義される．さらに，ゴールレベルでエージェントの誤った振舞いを扱うことにより，そのような例外の解決がより自由に可能となる．たとえば，ゴールの詳細化や責任割り当ての選択肢を考慮することにより，異なるシステム設計を採用することができる．

KAOS では，次の４種類のモデルを作成する（**図 -1**）[☆1]．

ゴールモデルは KAOS の主導的なモデルである．本モデルは複合的なシステムのゴールを規定する．ゴールは，複合的なシステムが，通常は複数のエージェントの協調により，達成すべき目的を定義する．ゴール詳細化リンクは，１つのゴールをサブゴールの集合に結び付ける．また，ゴール間競合関係も規定する．

オブジェクトモデルはアプリケーションドメインにおいて扱うオブジェクトを規定する．オブジェクトは，自律的か，従属的か，瞬間的か，能動的か，といった性質に応じて，実体，関係，イベント，あるいはエージェン

☆1 文献 1）では，さらにエージェントインタフェースモデルと呼ばれるものを追加し５種類としている．

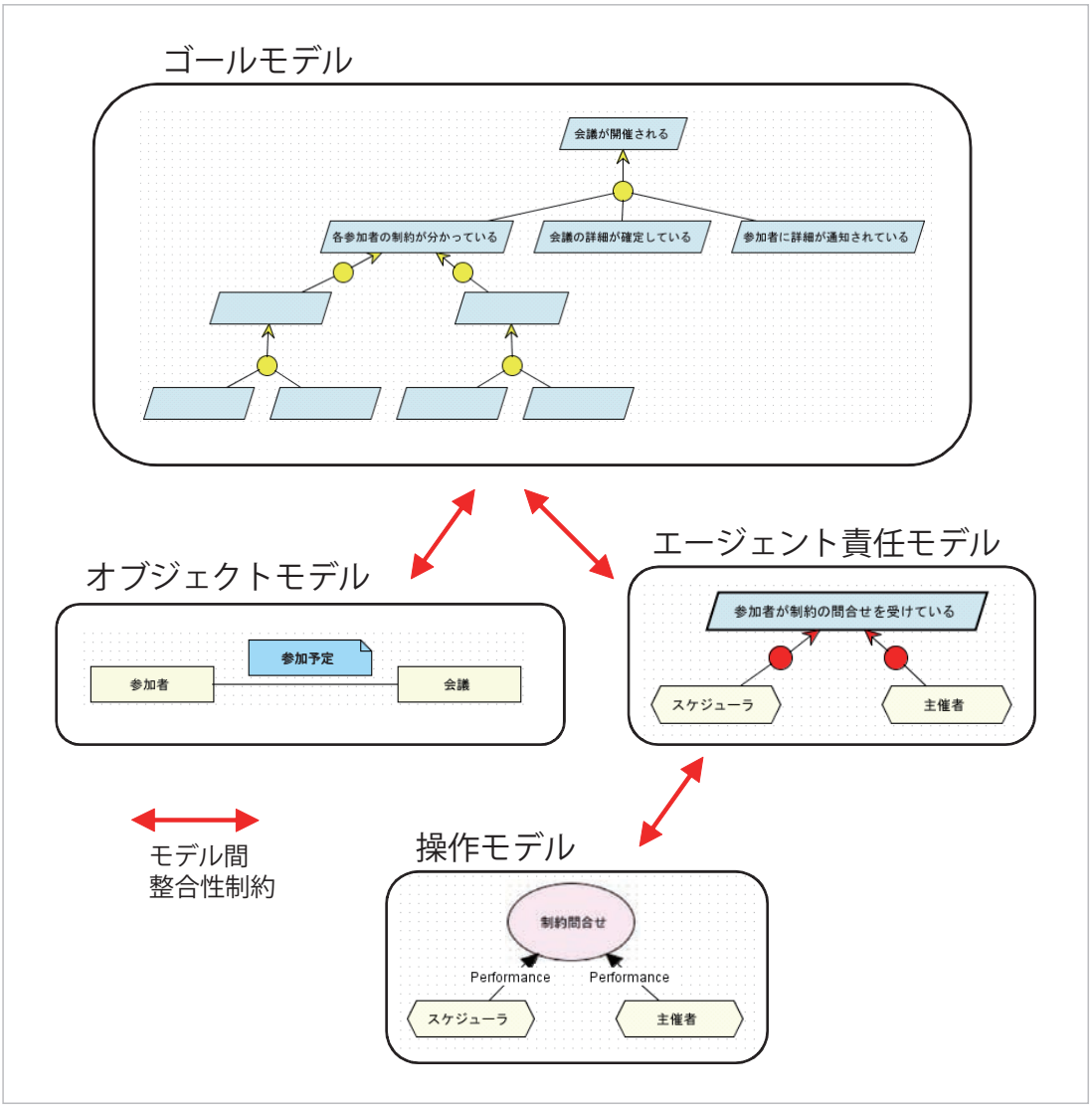

図 -1
KAOS で作成するモデル

トに分類される．オブジェクトの特性は，属性および他のオブジェクトの関係リンクにより表される．オブジェクト間の継承は IS-A 関係で表される．

エージェント責任モデルはエージェントへのゴール達成責任の割り当てを規定する．責任割り当ては，ゴール詳細化プロセス停止の基準となる．単一のエージェントに責任を割り当てられたゴールはそれ以上詳細化しない．責任割り当ての選択肢は，OR 責任リンクとして扱われる．責任を割り当てるというのは，ゴールに対し責任を持つエージェントが，そのゴールの達成を保証するために，振舞いの制限を要求される唯一のエージェントとなる，ということを意味する．

操作モデルはアプリケーションドメインにおける状態遷移を定義する．操作はドメイン事前・事後条件により定義される．操作には，さらに個々のエージェントに割り当てられたゴールの達成を保証するために必要な要求☆2 が与えられる．このような要求は，要求事前条件，要求トリガ条件，および要求事後条件として定義される．これらの要求は操作化リンクにより達成を保証されるゴールと結び付けられる．さらに各操作は実行リンクによりその操作を開始するエージェントと結び付けられる．

KAOS 反モデルによるセキュリティ要求の獲得・分析☆3

ゴール指向要求分析手法，特に KAOS が持つ上述のような特徴を活かすことにより，セキュリティ要求の獲得・分析において必要な特性を持つ手法の創出が期待できる．たとえば，設計の選択肢の洗い出しが可能である，といった特性が実現できる．そこで本章では，KAOS における反モデルの概念に基づく手法を紹介する．本手法の概要は，次の 2 種類のモデルを反復的かつ並行して構築する，というものである．

☆2 この「要求」は，一般的な用語ではなく KAOS 固有の概念である．
☆3 本章の内容は文献 2）に基づく．

1. ソフトウェアとその環境をともに包含し，それらのゴール，エージェント，オブジェクト，操作，要求，および仮定の間の相互関係を規定する，開発対象システムのモデル．
2. 1つ目のモデルから導出され，モデル要素の仕様が，どのように，なぜ，誰によって悪意に脅かされるかを示す反モデル．

セキュリティ要求は次の手順の反復により体系的に詳細化される：(a) 機密性やプライバシーなどの性質の分類と結びついた仕様記述パターンのインスタンスを作成し，(b) 作成した仕様記述の脅威となる反モデル仕様記述を導出し，(c) そのような脅威への対策の選択肢を導出し，モデルから得られる他の品質要求に最もよく合致する選択肢を選ぶことにより，新たな要求を定義する．

本手法は，KAOS において要求達成に対する**障害**を特定し解消する枠組みを拡張したものである．障害とは，ゴールの侵害のシナリオを特定する手段として導入された概念である．あるゴールに対する障害とは，ある条件であって，それが満たされると元のゴールが達成できなくなるようなものである．障害の特定手法として，ゴールの否定を AND/OR 詳細化する方法と，障害パターンの適用による方法がある．障害の解決手法もさまざまなものが提案されている．

本章の手法において，セキュリティ要求に関する概念は次のように扱われる．

- セキュリティ関心事はセキュリティゴールにより表され，ソフトウェアに対する**セキュリティ要求**，または環境に対するセキュリティに関する期待（後者は環境における**セキュリティポリシー**として規定される）にまで精緻化・詳細化される．
- **攻撃者**は環境における悪意のあるエージェントにより表される．
- **脅威**は攻撃者が意図的に成立させる障害により表される．
- 脅威に対して防御すべき**資産**（Asset）は受動的あるいは能動的なオブジェクトで表す．

このように，セキュリティを扱うために追加の抽象概念を持ち込む必要はなく，KAOS の枠組みをそのまま使用することができる．

以降で本手法の詳細を説明する．

反モデルの構築フェーズでは，セキュリティゴールを脅かす障害を特定する．障害を詳細化する木構造は，安全性重視システムにおいて危険をモデル化したり文書化するための故障木（Fault Tree）や，セキュリティ重視システムにおいて潜在的攻撃をモデル化したり文書化したりするための脅威木（Threat Tree）といった，広く知られた手法に似ているが，システムゴールとの結び付きや，体系的な方法論としての特徴において異なっている．特に，攻撃者とそのゴールや能力，および監視や制御が必要なソフトウェアの脆弱性といったものを扱える程度に充実したモデルを対象としている．

そこで，そのようなモデルを**反モデル**と呼び，セキュリティゴールへの悪意のある障害を含む攻撃者自身のゴールを**反ゴール**と呼ぶ．当然ながら反ゴールは，対象システムが満たすべきゴールと区別する必要がある．反モデルは，より詳細な脅威を特定し，そのような脅威への期待される対策としてより強固なセキュリティ要求を導出できるようなものでなければならない．

本フェーズは以下のステップから構成される．

1. セキュリティゴールの仕様記述パターンを，オブジェクトモデル中の保護すべきオブジェクトを用いてインスタンス化し，その否定として初期反ゴールを特定する．
2. 前ステップで特定した各反ゴールに対し，「この反ゴールから誰が利益を得るのか？」といった質問により，潜在的な攻撃者を抽出する．
3. 各反ゴールと，前ステップで特定した対応する攻撃者クラスに対し，「この攻撃者クラスのインスタンスはなぜこの反ゴールを達成したいのか？」という質問により，攻撃者のより高レベルの反ゴールを特定する．本ステップは再帰的に反復する．
4. 選択肢の分枝に沿って，反ゴールを AND- 詳細化・抽象化することにより，反ゴールの AND/OR グラフを作成し，最終的には特定された攻撃者エージェント，または攻撃を受けるソフトウェアエージェントにより実現可能な末端反ゴールを特定する．前者は攻撃者に割り当てられた反要求であり，後者は被攻撃者に割り当てられた脆弱性である．本ステップを進める手法はいくつか提案されている．
5. 反ゴールの仕様からオブジェクト反モデルとエージェント反モデルを導出する．
6. すべての反要求を，対応する攻撃者エージェントの潜在能力により AND/OR- 操作化する．潜在能力というのは，盗聴やなりすましなどのことである．

なお，ステップ4において反ゴールがあるエージェントにより「**実現可能である**」ということは，エージェントにより監視可能，および制御可能な条件により定義される（実現可能性のより技術的な定義は文献1）に記述されている）．また，ステップ4で反ゴール詳細化により導出される脆弱性は，コモンクライテリア（文献3））における「**脆弱性**」の定義，すなわち「脅威と結びつくことにより，セキュリティ要求の侵害が可能となるような，エージェントの状況」と適合する．

次に対策の作成フェーズでは，特定されたさまざまな脆弱性と反要求に対する対策の選択肢を検討する．本

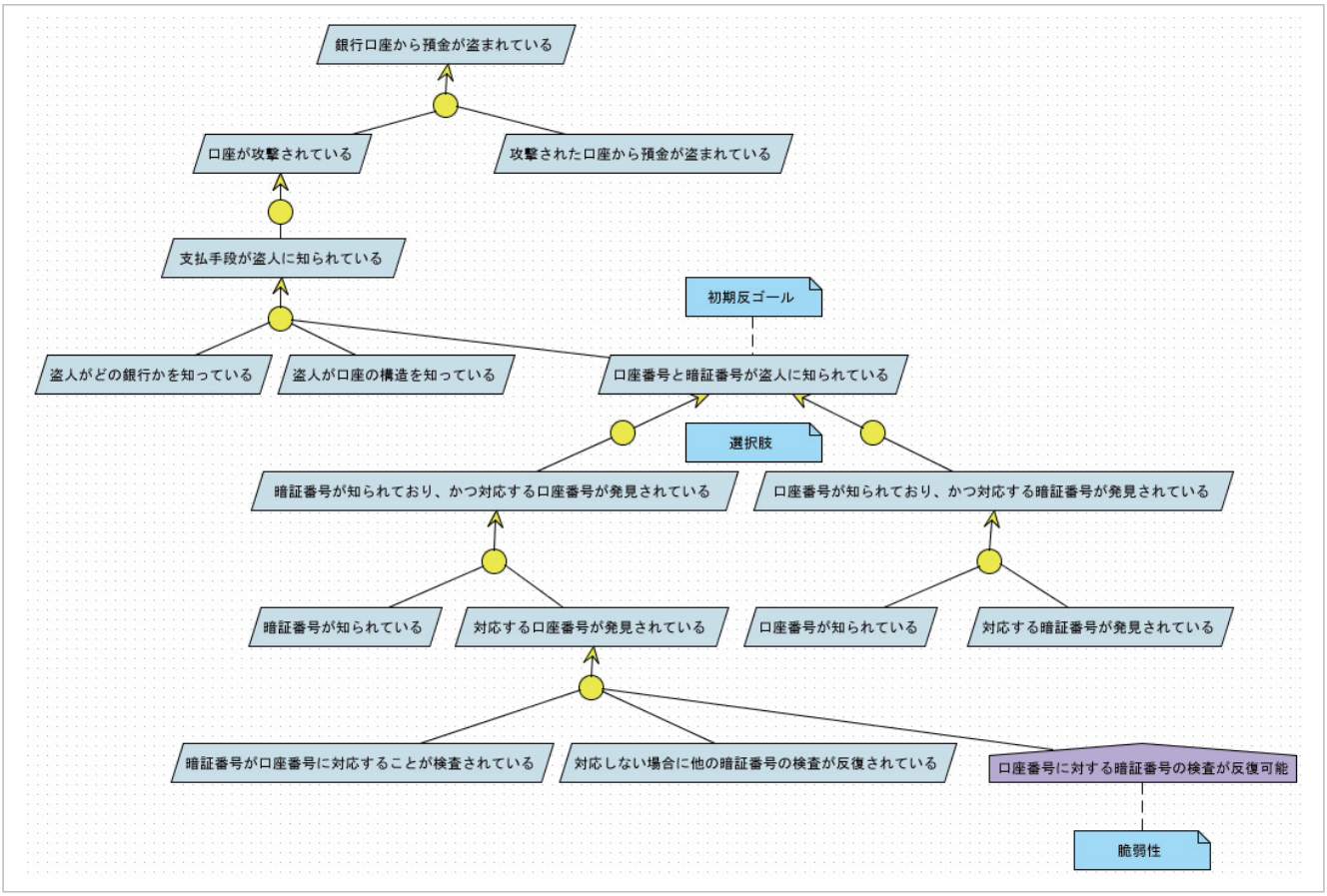

図 -2　KAOS 反モデルの例

フェーズでは，文献 4）に記述された障害解決のためのものと同様な手法を使用することにより，対策の選択肢を生成することが可能である．また，そのほかにセキュリティ特有の手法もある．対策の選択肢の生成後，前述した他の品質要求との適合性に基づいて，最適な対策を選択する．選択された対策は新たなセキュリティゴールとして AND/OR 詳細化される．ここで新たなモデル・反モデル構築サイクルが必要となる．

あまり深刻でない反ゴールについては，反ゴール監視技術や侵害検出技術を用いて，その解決を仕様策定時から実行時に先送りすることも可能である．

次に，本手法の適用例として，Web 上の銀行サービスを取り上げる（**図 -2**）．具体的には，ユーザが口座にアクセスする際に用いる情報，すなわち口座番号と暗証番号の漏洩防止というセキュリティ要求の分析を行う．まずステップ 1 において，仕様記述パターンとして機密性ゴールパターンの適用を行う．本パターンは，「機密情報が許可されないエージェントに知られることの防止」という抽象的なゴールをインスタンス化する．本例では，インスタンスとして「口座番号と暗証番号が許可されないエージェントに知られることの防止」というゴールの仕様記述を得る．そして，その仕様記述を否定した「口座番号と暗証番号が許可されないエージェントに知られている」を反ゴールとする．なお，KAOS，および本章の手法では，さまざまな数理論理学的な分析も含み，本ステップでもパターンで規定されるゴールの論理式表現のインスタンス化と否定も行うが，このような分析には大規模な背景知識が必要なため，本章ではその詳細を省略する．

次にステップ 2 において，「誰が上述の反ゴールから利益を得るのか？」という質問により，潜在的な攻撃者エージェントクラスとして「盗人」，「クラッカー」，「銀行の品質保証担当」などを抽出する．

ここで，例として「盗人」エージェントの場合を考える．するとステップ 3 において，親反ゴールとして「支払手段が盗人に知られている」，および（さらにその親の）祖父母反ゴールとして「銀行口座から預金が盗まれている」を抽出することができる．

ステップ 4 から 6 で作成される反モデルの例を図 -2 に示す．たとえば反ゴール「口座番号と暗証番号が許可されないエージェントに知られている」の（OR 詳細化による）サブゴールとして，2 つの選択肢「暗証番号が知られており，かつ対応する口座番号が発見されている」，および「口座番号が知られており，かつ対応する暗証番号が発見されている」を得ている．詳細化プロセスは，脆弱性，および「盗人」エージェントが実現可能

な反要求を得るまで続けられる．

その結果，「盗人」エージェントに割り当て可能なものとして導出された反要求は，1つ目の選択肢

「暗証番号が口座番号に対応することが検査されている」および

「対応しない場合に他の暗証番号の検査が反復されている」

またはもう1つの選択肢

「口座番号が暗証番号に対応することが検査されている」および

「対応しない場合に他の口座番号の検査が反復されている」

となる．

これらの反要求は，Web技術により操作可能である．2つ目の反要求はより巧妙であり，実際の高度な攻撃に対応したものとなっている．

次にこれらに対応する脆弱性として，

「口座番号に対する暗証番号の検査が反復可能」

および

「暗証番号に対する口座番号の検査が反復可能」

がそれぞれ導出されている．

以上の反モデルを構築すると，新たなセキュリティゴールとなるセキュリティ対策の導出に移る必要がある．本例では，脆弱性の重要性を考慮すると，解決手法として「脆弱性の修正」が必要であると考える．すると2つの新たな，同時に満たすべきセキュリティゴール

「口座番号に対する暗証番号の検査の反復可能性の防止」

「暗証番号に対する口座番号の検査の反復可能性の防止」

が生成される．

最初の新たなゴールは，暗証番号の入力回数に対する通常の制限（3回のみ許可，など）に対応する．2つ目の新たなゴールは，1つの暗証番号に対し，対応する口座番号を網羅的に調べ上げる可能性を回避するものである．この後は，ソフトウェアへの要求と環境に対する期待に至るまで，これらのゴールを詳細化する必要がある．必要に応じて，これらの新たなゴールに対する反モデル構築サイクルを実施することもあり得る．

まとめ，およびセキュリティ要求工学におけるKAOSの将来展望

本稿では，ゴール指向要求工学手法KAOSに基づくセキュリティ要求の獲得・分析手法の例として，反モデルを用いた手法を解説した．本稿で示した銀行システムの例では，攻撃手段の選択肢とそれに対する対策の導出について詳述したが，ゴール指向手法の特徴である，より上位のゴールによる攻撃者やステークホルダの意図の表現についてはあまり触れなかった．しかし，導出したセキュリティ対策をどの程度まで（たとえば暗証番号の検査の反復可能回数）施すかについては，コストやユーザの利便性といった他の非機能要求を考慮する必要があるので，その際に有用となると考えられる．

以上のようにKAOS，あるいはゴール指向要求工学手法全般について，セキュリティ要求工学における有用性が期待されているが，一方で現実的な状況で適用するにはまだ課題も多い．たとえば具体的にゴールの詳細化や抽象化を行うのは容易ではなく，可能な限り網羅的なモデルを構築しようとすると規模が厖大なものになる，といった課題がある．このような課題を解決するためには，実際の適用事例の増加や，それを通じたノウハウの蓄積，および可能な限り作業を自動化するツールの開発などが必要である．したがって，ゴール指向要求工学の分野に携わる研究者や実務者の増加を期待したい．

参考文献
1) Letier, E. : Reasoning about Agents in Goal-Oriented Requirements Engineering, Ph.D. thesis, Université catholique de Louvain (2001).
2) van Lamsweerde, A. : Elaborating Security Requirements by Construction of Intentional Anti-models, In Proc. of ICSE '04, pp.148-157 (2004).
3) Common Criteria for Information Technology Security Evaluation Part 1 : Introduction and General Model, http://www.commoncriteriaportal.org/thecc.html (Sep. 2007).
4) 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).

（平成20年11月28日受付）

田原 康之（正会員）　▶tahara@is.uec.ac.jp
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．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）．

Axel van Lamsweerde　▶avl@info.ucl.ac.be
蘭フィリップス研究所，Namur大学教授，Bruxelles大学教授を経て，現在Université catholique de Louvain（以上ベルギー）教授．ACM Transactions on Software Engineering and Methodology 誌編集長．2000年ACMフェロー．

Emmanuel Letier　▶e.letier@cs.ucl.ac.uk
2001年Université catholique de Louvain（ベルギー）博士．同大ポスドク研究員，英Imperial College 客員研究員歴任．2006年より英University College London 講師．