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

2006－SE－151（22）
2006／3／24

メタモデルを用いたセキュアシステム開発支援ツールの開発

小黒 博昭　　市原 尚久　　吉村 俊哉　　明関 恵美

株式会社 NTT データ　技術開発本部

あらまし　セキュアシステムの開発では、開発の初期段階からセキュリティを考慮した要件定義が行われる必要がある。さらに、要件定義から設計、実装、試験へと進む過程において、前工程で定めた定義内容が後工程へ確実に反映される必要がある。我々は、セキュリティ要件定義から設計までの活動を支援するセキュアシステム開発支援ツールのプロトタイプを開発した。提案ツールは、準形式化されたセキュリティ要件定義集から必要な要件を抽出し、GUI を用いて要件記述を完成させていく環境を提供する。さらに、テーラリングされた要件から、セキュリティ要件メタモデルのインスタンスを生成し、対応するセキュリティ設計メタモデルのインスタンスに変換する。得られた設計モデルは UML モデリングツールのアドインから読み込むことが可能であり、要件が反映された設計図の自動生成が可能である。本稿では、提案ツールの評価および考察を行う。

A Development of a Secure System Development Support Tool using Metamodel

Hiroaki Oguro　　Naohisa Ichihara　　Toshiya Yoshimura　　Emi Meiseki

Research and Development Headquarters, NTT DATA Corporation

Abstract—　It is necessary for a secure system development that requirements definition including security consideration should be performed. Besides, in a series of processes advancing from requirements definition to the design, the implementation, and the tests, it is essential that matters defined in the previous process should reflect the next process. We have developed a prototype of a secure system development support tool that helps activities from the security requirements definition process to the design process. The proposed tool provides us with the environment in which we extract and customize security requirements, transform the generated instance of the security requirements metamodel into the instance of the security design metamodel, and acquire the design diagram reflected by the requirements automatically. In this paper, we evaluate and consider the tool.

1.　はじめに

セキュアシステムの開発は、要求されるセキュリティレベルに応じて、システム開発の初期段階からセキュリティを考慮した適切な要件定義から開始される必要がある。さらに、要件定義から設計、実装、試験へと進む過程において、前工程で定めた定義内容が、予め定義されたトレースに従って、間違いなく後工程へ反映されることが望ましい。トレースの確保の必要性はセキュアシステムの開発に限った話ではないが、セキュアシステムの場合、システムの不具合が引き起こす様々なリスク（単にシステムの改修コストにとどまらず、顧客への損害賠償コストや、製造企業に対する信用の喪失なども含まれる）を考慮すると、一般の業務システムと比較して、より重視されなければならない。

一方で、このようなセキュリティ設計の作業品質は、設計者の熟練度やセキュリティに関する知識に依存する部分がかなり大きく、同時に行われる多数のシステム開発の現場で一様な品質を確保しながら実践することが困難であった。そのため、セキュリティ設計に熟練した設計者のノウハウをツール等により

共有化し、一様な設計品質を確保することが望まれている。

セキュリティに関する要件定義においては、システムに依存しない共通的なセキュリティ要件の定義集として、Common Criteria [2] (以下、CC)がある。CC に準拠した開発を行う場合、まず要件定義者は CC のリスク分析結果に応じてセキュリティ要件からシステムに必要な要件を抽出する。CC で定義されているセキュリティ要件は、そのままでは不完全な自然言語で記述されており、それをベースに要件定義者が文字列または数値を加筆したり、列挙された選択項目の中から一つまたは複数を選択したりしながら、要件の記述を完成させるという方法が採られている。本稿では、このような方法で要件の記述を完成させる作業を「要件のテーラリング」と呼ぶ。

CC の要件定義スキームに倣った開発を行うには、以下のような課題が存在する。

[課題]

1. セキュリティ要件定義集の維持管理コスト(要件の追加・削除・修正等にかかるコスト)が大きい。
2. セキュリティ要件定義集の不完全な自然言語が直感的に理解しにくく、解析しながらテーラリングを進める作業が負担となる。
3. セキュリティ要件の依存性を意識した要件定義に係る作業(ある要件 A を定義するためには、他の要件 B が定義済みであることが必要または推奨とされる場合に、その要件 B を定義したり、要件 B を定義しない理由を明示したり、要件 B が依存する他の要件 C を定義したりなど)、およびそれらの設計への反映が複雑である。
4. テーラリング済みの要件を修正する場合、オリジナルの要件記述に含まれていた情報(文字列または数値などを表す型や、非選択要素)が失われているため、オリジナルへの再参照が発生する。

我々は、これらの課題を解決するため、CC をベースとしたセキュリティ要件定義から設計までの一連の活動を支援するセキュアシステム開発支援ツール(以下、提案ツール)のプロトタイプを開発した。提案ツールは 2 つのツールからなり、1 つはセキュリティ要件 DB 管理ツール、もう 1 つはセキュリティ要件テーラリングツールである。

セキュリティ要件 DB 管理ツールは、上記課題 1 を解決する。本ツールでは、後のテーラリングを容易にするため、CC のセキュリティ要件記述を準形式化したものを扱い、XML によるスキーマ定義を与えている。要件の維持管理作業については、GUI による操作環境が提供されている。

セキュリティ要件テーラリングツールは、上記課題 2～4 を解決する。GUI を用いて要件定義者が必要な要件を抽出し、テーラリングする環境を提供する。そして、別途定義されたセキュリティ要件メタモデルを参照し、テーラリング情報を基に、そのインスタンスであるセキュリティ要件モデルを生成する。得られた要件モデルは、予め定義されたトレースの定義に従って設計モデルへ変換される。得られた設計モデルは、外部の UML モデリングツールのアドインを通して読み込まれ、要件定義が反映された設計図として自動的に表示される。

本稿の次節以降の構成は以下の通りである。2 節では、CC の概要を説明し、CC に特有なセキュリティ要件定義手法について述べる。3 節で提案ツールが要件から設計への変換に利用しているメタモデルとその変換について述べる。4 節でセキュアシステム開発支援ツールの詳細を述べ、5 節でその評価と考察を行う。6 節でまとめと今後の課題を述べる。

2. Common Criteria

本節では、CC の概要および CC が規定しているセキュリティ要件のテーラリングについて説明する。

2.1. Common Criteria の概要

Common Criteria とは 1999 年 6 月に ISO/IEC 15408 [4] として標準化された IT セキュリティ評価基準である。2000 年 7 月には、JIS X 5070 としても制定されている。CC の評価機関は、ベンダが申請した IT 製品または IT システムのセキュリティに関して、ベンダが作成したセキュリティターゲット(ST)と呼ばれるセキュリティ基本設計書を基にして、設計書、ソースコード、試験結果、マニュアル、脆弱性分析、さらには開発環境や管理体制などの検査を通じて評価を行い、合格ならば CC 認証を付与する。

CC のドキュメントは 3 つのパートから構成されており、パート 1 は総則及び一般モデル、パート 2 はセキュリティ機能要件、パート 3 はセキュリティ保証要件を定めている。

2.2. セキュリティ機能要件のテーラリング

CC では、評価対象のシステムを TOE(Target of Evaluation)、TOE のセキュリティ機能を TSF(TOE Security Function)と呼ぶ。CC のパート 2 では、セキュリティ機能要件として、表 1に示す 11 クラスが提供されている。

表 1: CC のセキュリティ機能要件

クラス	セキュリティ機能要件
FAU	Security Audit（セキュリティ監査）
FCO	Communication（通信）
FCS	Cryptographic support（暗号サポート）
FDP	User data protection（利用者データ保護）
FIA	Identification and authentication（識別・認証）
FMT	Security management（セキュリティ管理）
FPR	Privacy（プライバシー）
FPT	Protection of the TSF（TSF の保護）
FRU	Resource utilization（資源利用）
FTA	TOE access（TOE アクセス）
FTP	Trusted path/channels（高信頼パス/チャネル）

各クラスはさらにファミリ、コンポーネント、エレメントという階層構造で定義されている。例えば、識別・認証クラス FIA 配下の、ファミリ FIA_AFL には、コンポーネント FIA_AFL.1 が、以下の 2 つのエレメント FIA_AFL.1.1 および FIA_AFL.1.2 を定義している。

- FIA_AFL.1（認証失敗時の取り扱い）
  - FIA_AFL.1.1
    TSF は、[**割付: 認証事象のリスト**]に関して、[**割付: 回数**]回の不成功認証試行が生じたときを検出しなければならない。
  - FIA_AFL.1.2
    不成功の認証試行が定義した回数に達するか上回ったとき、TSF は、[**割付: アクションのリスト**]をしなければならない。

CC に準拠した開発では、セキュリティ要件の中から任意に必要な要件をコンポーネント単位で選択し、太字部分を以下のようにテーラリングしながら要件定義を行う。

（テーラリングの例）
- FIA_AFL.1（認証失敗時の取り扱い）
  - FIA_AFL.1.1
    TSF は、**ユーザのパスワード認証**に関して、**3** 回の不成功認証試行が生じたときを検出しなければならない。
  - FIA_AFL.1.2
    不成功の認証試行が定義した回数に達するか上回ったとき、TSF は、**パスワードのロック**をしなければならない。

3. メタモデル

本節では、提案ツールが要件から設計へのマッピングを実現するために利用している MOF (Meta Object Facility) [5][6]ベースのメタモデルについて説明し、実装した 2 つの適用例について述べる。

3.1. MOF ベースのメタモデル

一般にメタモデルとは、あるモデルをさらに抽象化した概念を整理したものであり、モデルの意味的構造を表す。OMG (Object Management Group)は、オブジェクト指向設計においてメタモデルを定義するための枠組みとして、MOF を提供している。以下で単にメタモデルと言う場合、MOF ベースのメタモデルを表す。

メタモデルのインスタンスとなるモデルは、UML (Unified Modeling Language) [7] のクラス図に限らず、表形式などの特定の表記法に沿って作成された成果物や、自然言語で書かれた文章等も対象となりうる。例えば、システム開発においては、開発工程で作成される成果物全般がモデルであり、これら成果物の意味的な構造を表したものがメタモデルである。

MOF ベースのメタモデルおよびそのインスタンスであるモデルを蓄積し、管理するための機構として、MOF リポジトリがある。MOF リポジトリの API を生成する標準として MOF to IDL Mapping [5] があり、それに準拠した実装として Medini [3] がある。

3.2. 要件メタモデルと設計メタモデル

我々はテーラリングされた後のセキュリティ要件の意味的構造を考察し、要件メタモデルを定義した。さらに、テーラリング情報が反映された要件モデルを、UML で表現された設計図へ変換するための設計メタモデルを定義した。これらは、3.3節で述べる設計変換に利用される。

本開発で実装したメタモデルとして、以下3.2.1節でパスワードロックに関するメタモデルを、3.2.2節で初期パスワードに関するメタモデルを示す。

3.2.1. パスワードロックに関するメタモデル

パスワードロックに関する要件メタモデルを図 1に示す。ここでは、パスワード認証の連続失敗時にシステムがパスワードをロックする条件と、その状態からの回復条件が要件定義の本質であることを抽象クラスで規定し、それらの条件を具体化したものとして、認証失敗回数や、経過時間などをモデル化している。

次に、このパスワードロックに関する要件メタモデルのインスタンスである要件モデルが、設計モデルへマッピングされる際の設計メタモデルを図 2に示

す。ここでは、パスワードロックの実行および回復に関する設計情報を視覚化するのに適した設計図として UML の状態遷移図を採択し、そのメタモデルを拡張することで、設計メタモデルが作成されている。

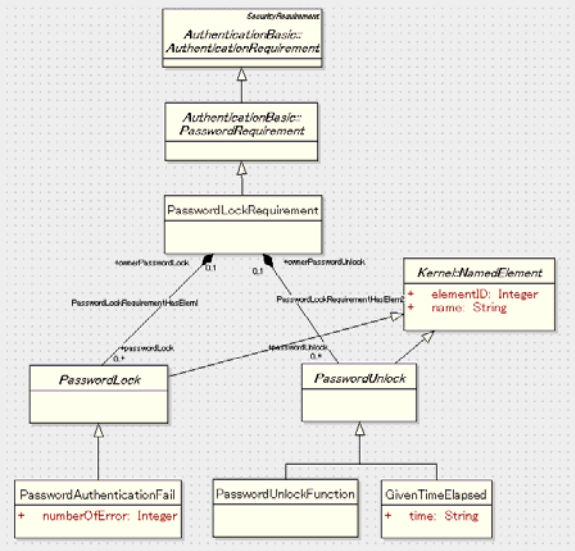

図 1: パスワードロックに関する要件メタモデル

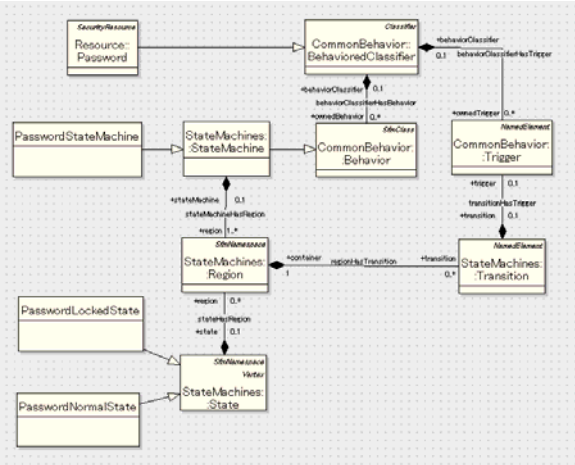

図 2: パスワードロックに関する設計メタモデル
（状態遷移図）

3.2.2. 初期パスワードに関するメタモデル

初期パスワードの設定からユーザの初回利用までに関する要件メタモデルを図 3に示す。ここでは、初期パスワードに係る要件として、その生成方式を抽象クラスで規定し、それを具体化したものとして、初期パスワードをユーザが指定する場合と、システムが自動生成する場合があることを規定している。また、初期パスワードの品質検査や、ユーザによる変更要求がありうることを規定している。

次に、この初期パスワードに関する要件メタモデルのインスタンスである要件モデルが、設計モデルへマッピングされる際の設計メタモデルを図 4に示す。ここでは、初期パスワードの設定からユーザの初回利用開始までの設計情報を視覚化するのに適した設計図として UML のアクティビティ図を採択し、そのメタモデルを拡張することで、設計メタモデルが作成されている。

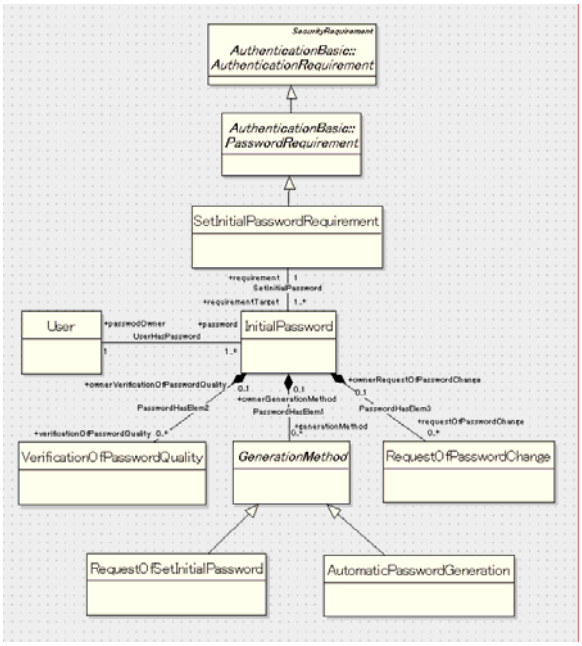

図 3: 初期パスワードに関する要件メタモデル

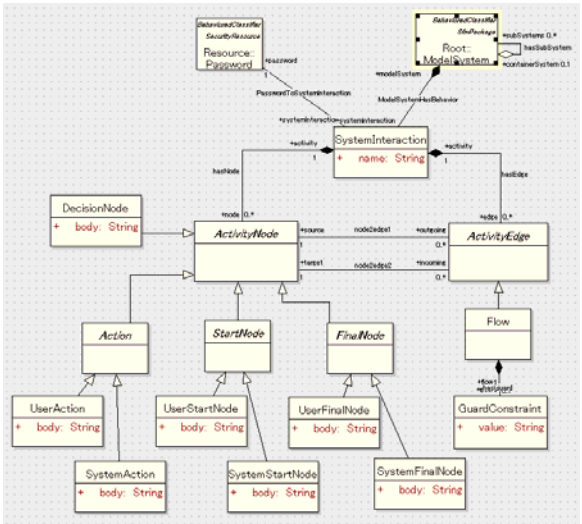

図 4: 初期パスワードに関する設計メタモデル
（アクティビティ図）

3.3. 要件から設計への変換

提案ツールでは、別途定義されたセキュリティ要件メタモデルを参照し、テーラリング情報を基に、セ

キュリティ要件メタモデルのインスタンスであるセキュリティ要件モデルを生成する。得られた要件モデルは、セキュリティ設計メタモデルで規定されたセキュリティ設計モデルへ変換される。例として、図 5にパスワードロックに関する要件から設計への変換ルールを示す。提案ツールでは、図 5のようなトレースの定義をコード内に予め記述することで変換を実現する。

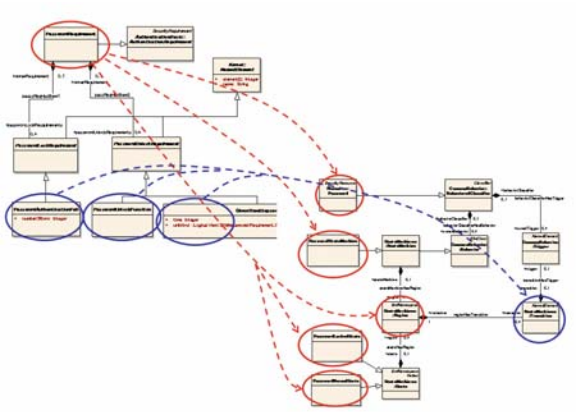

図 5: 要件から設計への変換(パスワードロック)

4. セキュアシステム開発支援ツール

本節では、セキュアシステム開発支援ツールの詳細と動作確認の結果について述べる。

4.1. システム利用形態

提案ツールを含むシステムの利用形態を図 6に示す。提案ツールのユーザは、セキュリティ要件 DB 管理者、メタモデル管理者、システム開発者の 3 名である。

セキュリティ要件 DB 管理者は、セキュリティ要件 DB 管理ツールを用いて、要件の登録、削除、および編集を行い、セキュリティ要件 DB に保存する。この管理者は、準形式的なセキュリティ要件記述に関する十分な知識を持つと仮定する。

メタモデル管理者は、セキュリティ要件 DB 管理者と協調しながらセキュリティ要件の意味を考察し、セキュリティ要件メタモデルおよび設計メタモデルを定義し、MOF リポジトリへ格納する。

システム開発者は、セキュリティ要件テーラリングツールを用いて、セキュリティ要件 DB から必要なセキュリティ要件を抽出し、要件のテーラリングを行う。さらに、このツールから MOF リポジトリに接続し、要件モデルを生成し、対応する設計モデルへ変換する。このとき要件定義が不十分であったり、要件メタモデルに違反するような要件定義がなされていたりする場合は、MOF リポジトリから要件モデルの生成を拒否される。生成された設計モデルは、外部の UML モデリングツールのアドインから読み込まれ、UML 図が自動生成される。システム開発者には、メタモデル定義に関する特別な知識は要求されないと想定する。

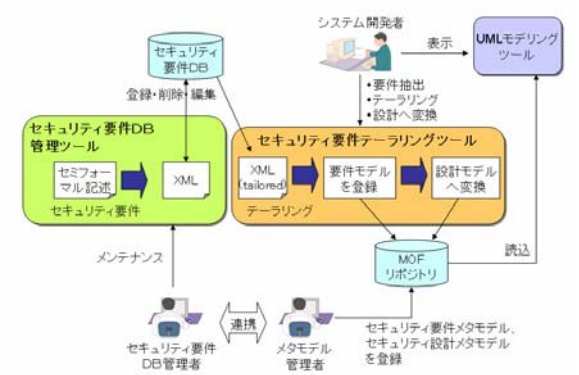

図 6: システム構成

4.2. ソフトウェア構成

提案ツールを含むシステム全体のソフトウェア構成を表 2に示す。セキュリティ要件 DB に関しては、今回の開発では XML ファイルにより実装している。

表 2: ソフトウェア構成

OS	Microsoft®, Windows® XP
開発環境	Microsoft®, Visual C# .NET 2003
MOF リポジトリ	IKV++, Medini MM バージョン 2.0.4
ミドルウェア	Borland®, Janeva® 6.5
UML モデリングツール	Sparx Systems, Enterprise Architect 日本語版 バージョン 5.00.768

4.3. セキュリティ要件 DB 管理ツール

4.3.1. 要件の維持管理

セキュリティ要件 DB 管理ツールのメイン画面を図 7に示す。画面左側に階層化されたセキュリティ要件が表示され、既存の要件を選択し、その編集または削除ができる。新規階層および新規要件の作成も可能である。編集された要件は XML ファイルへ保存される。

4.3.2. 要件記述の準形式化

DB 管理ツールは、準形式化されたセキュリティ要件記述を取り扱う。パスワードロックに関係するセキュリティ要件記述例を表 3に示す。内容の欄の大括弧で囲まれた部分がテーラリングされる領域であり、型(名称、数値、択一、複数選択、任意記述、など)がまず宣言され、コロンで区切られた中括弧が続く。1 番目の中括弧はこのテーラリング領域に記述されるべき内容の意味を表す。2 番目の中括弧は、型が

択一または複数選択時における選択項目のリストである。

我々は、テーラリング領域の構造に対し、XML によるスキーマ定義を与え、一覧として管理できるようにした。これにより、後のテーラリング作業が容易になると同時に、択一または複数選択のテーラリングの際の非選択要素の情報も保持でき、テーラリングの修正を容易にする。

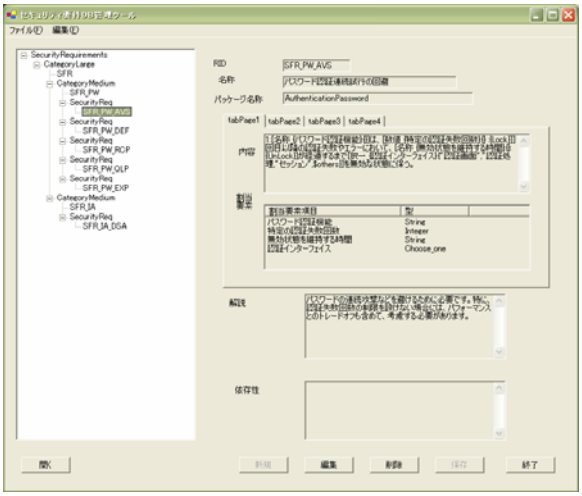

図 7: セキュリティ要件 DB 管理ツール
（メイン画面）

表 3: パスワードロックに関する要件

要件 ID	SFR_PW_AVS
要件名	パスワード認証連続試行の回避
内容	_1:[名称_{パスワード認証機能}:{}]は、[数値_{特定の認証失敗回数}:{}]回目以降の認証失敗やエラーにおいて、[名称_{無効状態を維持する時間}:{}]が経過するまで[択一_{認証インターフェイス}:{"認証画面","認証処理,"セッション",$others}]を無効な状態に保つ。
説明	パスワードの連続攻撃などを避けるために必要。特に、認証失敗回数の制限を設けない場合には、パフォーマンスとのトレードオフも含めて、考慮する必要がある。
依存性	なし

4.4. セキュリティ要件テーラリングツール

4.4.1. 要件のテーラリング

セキュリティ要件テーラリングツールのメイン画面を図 8に示す。画面左側に、管理されているセキュリティ要件が読み込まれ、システム開発者が要件の内容を確認しながら必要な要件を抽出し、図 9の編集画面で要件のテーラリングを行う。

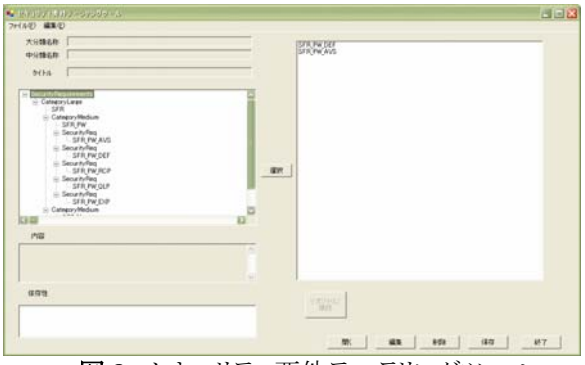

図 8: セキュリティ要件テーラリングツール
（メイン画面）

図 9: セキュリティ要件テーラリングツール
（編集画面）

4.4.2. パスワードロックに関する設計変換・読込

パスワードロックに関する要件として、表 3に示した要件 SFR_PW_AVS（パスワード認証連続試行の回避）をテーラリングし、メイン画面から MOF リポジトリへ接続、要件モデルの登録、設計モデルへの変換を行い、UML モデリングツールから設計モデルの読み込みを行った結果を図 10に示す。パスワードロックに関する要件定義内容が反映された状態遷移図が表示される。

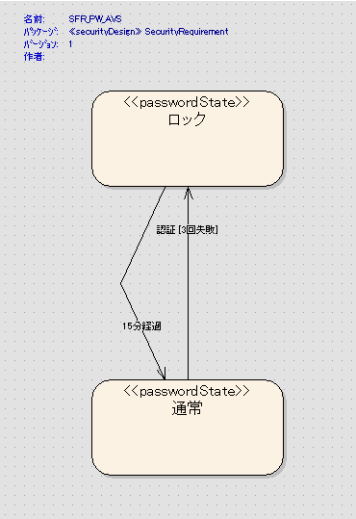

図 10: パスワードロックに関する状態遷移図

4.4.3. 初期パスワードに関する設計変換・読込

初期パスワードに関する要件として、表 4に示す要件 SFR_PW_DEF（パスワード初期値）をテーラリングし、登録、設計変換、および読み込みを行う際の動作を説明する。この要件には、2 つの別の要件への依存性が存在する。1 つは SFR_PW_QLP というパスワード品質の検査に関する要件、もう 1 つは SFR_PW_RCP というパスワード変更要求（初期パスワードを利用者自身に変更させる）に関する要件である。テーラリングツールは、SFR_PW_DEF の内容のテーラリング情報のみならず、依存先の要件のテーラリング有無やその内容を考慮して、初期パスワードに関する要件メタモデルを作成し、設計変換を行う。

表 4: 初期パスワードに関する要件

要件 ID	SFR_PW_DEF
要件名	パスワード初期値
内容	_1: [名称_{パスワード認証機能}:{}]は、[複数選択_{パスワード初期化のタイミング}:{"新規ユーザ登録時",$others}]において、パスワードの初期値を[択一_{初期値生成方法}:{"自動生成","ユーザ指定",$others}]により設定する。
説明	ユーザ初期登録などの際に、初めてパスワードが設定される際の要件。システム内部で自動生成したものを提供する場合と、ユーザ自身が設定できる場合がある。
依存性	SFR_PW_QLP（パスワード品質） SFR_PW_RCP（パスワード変更要求）

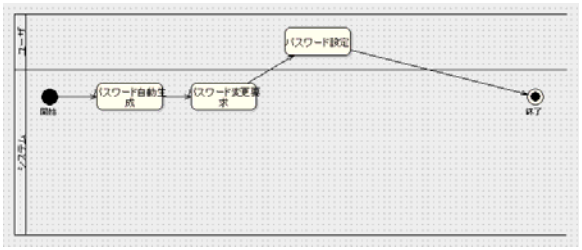

図 11: 初期パスワードに関するアクティビティ図
（自動生成、品質検査なし、変更要求あり）

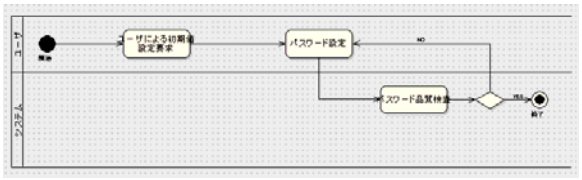

図 12: 初期パスワードに関するアクティビティ図
（ユーザ指定、品質検査あり、変更要求なし）

図 11は、初期値生成方法として「自動生成」を選択し、SFR_PW_QLP を非選択（未テーラリング）、SFR_PW_RCP を選択しテーラリングした場合に得られるアクティビティ図である。

図 12は、初期値生成方法として「ユーザ指定」を選択し、SFR_PW_QLP を選択しテーラリングを行った場合に得られるアクティビティ図である。ユーザ指定の場合は、はじめからユーザがパスワードを設定することが前提となっているため、システム側がユーザに変更要求をかける SFR_PW_RCP の考慮は無意味となる。

5. 評価・考察

本節では、提案ツールの性能、現場適用可能性、および提案ツールが採択したメタモデルアプローチの有利点と課題について評価および考察を行う。

5.1. 処理性能

提案ツールの処理性能については、2006 年時点での一般的な開発用デスクトップ PC（CPU: Intel® Pentium® 4 3.0GHz、メモリ: 2GB）上で、実用上ストレスなく要件のテーラリング、MOF リポジトリへの要件モデルの登録、設計モデルへの変換を行えることを主観評価により確認した。リポジトリ接続から設計モデルへの変換にかかる時間は 1 秒弱であり、UML モデリングツールから設計モデルを読み込み、表示するのにかかる時間も 1 秒弱である。従って、開発の現場においては、処理性能の低さから使用を敬遠されるようなことはないと予想される。

5.2. 現場適用可能性

識に大きく依存するため、一様な品質を維持することが困難であるという根本的な問題があった。提案ツールは、セキュリティ設計の経験が浅い設計者にも GUI によるテーラリングを通して、定義漏れや依存性の見落としを防止することができ、セキュリティ設計の品質向上に寄与することが期待できる。

一方で、セキュリティ要件 DB 管理者およびメタモデル管理者には、セキュリティ設計の熟練性とセキュリティに関する深い知識が要求される。特に、要件の意味的構造を考察しメタモデル化するメタモデル管理者の人材確保と養成が必要となる。企業内で組織的に利用する場合には、社内体制の整備等も必要になると考えられる。

また、1節で提示した課題 1～4 は、提案ツールにより解決される。これに伴い、システム開発者の負担が軽減されることが期待されるが、その検証は今後の課題である。

5.3. メタモデルアプローチの有利点・課題

提案ツールでは、要件を反映させた確実な設計を得るために、メタモデルによるアプローチを採択した。

メタモデルによるアプローチの利点としては、ツールに非依存な領域で、概念を形式化して取り扱うことが可能な点である。従って、今回の提案ツールにおける設計変換先は UML としているが、従来の表形式等の他の設計図に対応させることは、同一概念に対する異なるビューを与えることに対応し、必要最低限の修正しか発生せず、実装も容易である。

また、課題としては、まだ広範な適用実績を持たないことと、ツールを開発する側に高度なモデリング技能が要求されることである。

6. まとめと今後の課題

本稿では、セキュリティ設計の品質が設計者の経験や知識に大きく依存するために一様な品質を確保することが困難であるという根本的な問題意識の下で、設計者のスキルに極力依存せずにかつスキルのある設計者と同等の要件定義および設計が行えるようにすることを目的に、その一つの実現手段として、セキュアシステム開発支援ツールを提案した。提案ツールは、CC の要件定義手法に基礎を置き、その上で発生しうる課題を解決するように構成した。また、要件および設計のメタモデルを作成し、予め定義されたトレースに従って、確実に要件から設計へ変換されることを実現した。

今後の課題として、提案ツールでは、表 1に示すセキュリティ機能要件のうち、概念的に設計に変換しやすい識別・認証に関する要件を対象としたが、1節で述べたように、従来、セキュリティ設計の作業品質は設計者の熟練度やセキュリティに関する知他の要件（プライバシなど）の中には、非機能性の強い要件もあり、設計変換が本質的にあまり意味をなさないものについて整理する必要がある。

また、提案ツールではトレースの定義をコード内に予め記述することで変換を実現しているが、より厳密なトレースの確保と可視化を実現するためには、別途トレースメタモデルを定義した上で要件メタモデルと設計メタモデルの間の対応を取る必要がある[1][8]。

参考文献

[1] 我妻智之, 神谷慎吾, 大平直宏, 松下 誠, 楠本真二, 井上克郎, ‘‘メタモデルに基づくトレーサビリティ技術の提案’’, 電子情報通信学会技術研究報告, KBSE2005-20, Vol.105, No.270, pp.25-30 (2005).

[2] Common Criteria for Information Technology Security Evaluation - Part 1: Introduction and general model, Part 2: Security Functional Requirements, Part 3: Security Assurance Requirements, Version 2.3, Common Criteria Sponsoring Organizations, (2005). http://www.commoncriteriaportal.org/

[3] IKV++ Technologies AG. http://www.ikv.de/

[4] International Standard ISO/IEC 15408:2005 Information technology - Security techniques - Evaluation criteria for IT security - Part 1: Introduction and general model - Part 2: Security Functional Requirements, Part 3: Security Assurance Requirements, International Organization for Standardization, (2005).

[5] Object Management Group, ‘‘Meta Object Facility (MOF) Specification, Version 1.4’’ (2002). http://www.omg.org/docs/formal/02-04-03.pdf

[6] Object Management Group, ‘‘Meta Object Facility (MOF) 2.0 Core Specification,’’ (2003). http://www.omg.org/docs/ptc/04-10-15.pdf

[7] Object Management Group, ‘‘Unified Modeling Language: Superstructure, version 2.0’’ (2005). http://www.omg.org/docs/formal/05-07-04.pdf

[8] 大平直宏, 松下 誠, 岡野浩三, 楠本真二, 井上克郎, 山下裕介, 我妻智之, ‘‘Struts フレームワークにおけるメタモデルを用いた追跡可能性実現手法の提案”, 情報処理学会研究報告, 2005-SIGSE-150, Vol.2005, No.119, pp.33-40 (2005).