社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
KBSE2011-69 (2012-3)

Assurance Case 作成手法（d*）の適用評価

猿渡 卓也[†, †2] 山本 修一郎[‡]

†名古屋大学大学院情報科学研究科〒464-8601 愛知県名古屋市千種区不老町
‡名古屋大学情報連携統括本部〒464-8601 愛知県名古屋市千種区不老町
†2 株式会社 NTT データ 〒135-8671 東京都江東区豊洲 3-3-9

E-mail: †2 saruwatarit@nttdata.co.jp, ‡ yamamotosui@icts.nagoya-u.ac.jp

あらまし 現在，開発されている情報システムの多くは，様々なシステム（要素）が相互に依存しあうオープンシステムである．しかし，代表的な Assurance Case の記法（GSN）では，複数の対象が同じ木構造の中で表現され，システム同士の依存関係が明確にならない．そこで，対象同士の Dependability の依存関係を記述できる d*（d*Framework）手法の導入を，エレベーターの制御システムを例にとり適用評価した．

キーワード Assurance Case，d*Framework，ディペンダビリティ

Evaluation of an Assurance Case development method (d*)

Takuya SARUWATARI[†, †2] Shuichiro YAMAMOTO[‡]

† Graduate School of Information Science Nagoya University Furo-cho, Chikusa-ku, Nagoya, Aichi, 464-8601 Japan
‡ Strategy Office, Information and Communications Headquarters Nagoya University
Furo-cho, Chikusa-ku, Nagoya, Aichi, 464-8601 Japan

†2 NTT DATA CORPORATION 3-3-9, Koto-Ku, Tokyo, 135-8671 Japan

E-mail: †2saruwatarit@nttdata.co.jp, ‡ yamamotosui@icts.nagoya-u.ac.jp

Abstract Many information systems are now developed as open systems which mutually depend on each other. Although assurance cases are expected to confirm dependability of open systems, it is difficult to clearly describe dependency among co-related systems, because systems are not explicitly represented in structured trees of assurance cases. By using the case study of an elevator control system, we evaluated the d* framework which can describe the dependability of inter-related open systems.

Keyword Assurance Case, d*Framework, Dependability

1. はじめに

情報システムの用途が多様化し対象が複雑になるにつれ，情報システムの Dependability の確保が難しくなってきている．例えば，複数の情報システムを連携させて利用する場面に対しての検討が必要である．

近年，システム開発における Dependability の確保は Assurance Case の作成という形で検討されるようになってきている[1][2][3][4][5]. Assurance Case の作成には，目的を構造的に整理できる GSN（Goal Structuring Notation）が利用されている．システム開発工程において，GSN による Assurance Case を作成・管理していくことにより，Dependability の確保が期待できる．しかし，複数のシステムが協調して処理を実行する状況において，1 つの木構造のゴールグラフで Assurance Case を作成した場合，個々のシステムの Dependability 情報が Assurance Case 全体に散在することになるため，システム間の相互作用を明示的に扱うことが困難になる．この問題に対処するために d*Framework(以下 d*)が提唱されている[6]．GSN に対する d*の特徴は，サブシステム等の複数の Actor が協調するようなシステムについて，それら Actor を考慮しながら分析できる点にある．

本稿では，エレベーターシステムを例題として d*を適用することで手法の評価を実施した．また，i*Framework[7]（以下 i*）と SARM[8]に対して d*を比較した．

2. Assurance Case

Assurance Case とは，“システムの特性に関する重要

な要求が，与えられた環境の中の適用で十分に正当化されるという，説得力のある議論を提供する構造化された文書”[3]と定義されている．すなわち，システムの Dependability を保証するために作成するモデルであると考えることができる．Assurance Case の作成は，システムの Dependability を確保するための要求を対象としている点で，要求工学におけるゴールグラフ作成の要素を含んでいる．しかし，Assurance Case の作成はゴールグラフを要求分析のためだけに作成するのではなく，システムの Dependability 要求を保証するソリューションまでを統合的に把握することを目的としている．

現在，Assurance Case のための記法として GSN が広く利用されている．GSN には，「ゴール分割時の Strategy の定義が可能」，「最下層のゴールに対する Solution の付与が可能」，「Goal や Strategy への付加情報（Justification，Assumption，Model，Context）の付与が可能」の 3 つの特徴がある．しかし GSN によるモデル作成では，相互に関係するオープンシステムの Assurance Case を適切に表現することが難しいという問題がある．そこで，本稿では相互に関係するオープンシステムの Assurance Case を作成できる d*を適用評価した．

2.1. d*Framework

Actor を明確に考慮した Assurance Case を作成するために考案された記法が d*である[6]．d*は，ゴール指向要求分析におけるゴールグラフ表記法の一つである i*[7]と同様に，Actor 間の関係すなわち Actor 間の Dependability 情報を記述できる．また，Dependability 情報と Actor の内部を記述するために，GSN を利用する．

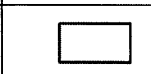
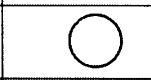
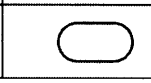
2.1.1. d*Framework の構成要素

d*の構成要素は，“Actor”，“Goal”，“Strategy”，“Evidence”，“Context”である．

1. Actor
   システムを構成する要素を示す．
2. Goal
   システムが充足すべき性質を示す．下位 Goal や下位 Strategy に分解される．
3. Strategy
   Goal の充足を導くために必要となる論証を示す．下位 Goal や下位 Strategy に分解される．
4. Solution
   Goal が充足できることを示す Evidence(証跡)を示す．
5. Context
   Goal や Strategy が必要となる理由としての外部情報を示す．

d*の各構成要素の表記法を表 1 に示す．

表 1 d*Framework の構成要素

Actor	
Goal	
Strategy	
Solution	
Context	

2.1.2. d*Framework の作成手順

d*では，以下の手順を反復的に繰り返すことにより Assurance Case を作成する（図 1）．

1. Actor Elicitation
   システム内の Actor を抽出する．作成の初期段階では，システム全体の構成図等を利用して抽出する．作成途中の段階では，Inter Analysis 手順，Inner Analysis 手順の結果として抽出される可能性がある．
2. Dependability Elicitation
   Actor 間に，相互に関係する Dependability 情報を抽出する．
3. Inter Analysis
   Actor 間の Dependability 情報を，GSN を用いて分析する．最下層の Goal について他 Actor との関係を検討し，Goal を満たすことができる Actor への Dependability 情報とする．満たすことのできる Actor が存在しない場合は，必要に応じて Actor Elicitation 手順に進み，対象としている Goal（Dependability 情報）を充足することのできる新規 Actor を追加する．
4. Inner Analysis
   Actor 内を，GSN を用いて分析する．対象としている Actor に対して他 Actor から依存されている Dependability 情報がある場合，その Dependability 情報に対応する Actor 内の Goal との関係を明確にする．分析中に，対象としている Actor 以外の Actor に依存する Goal が抽出された場合，Actor Elicitation 手順に進み，対象としている Goal（Dependability 情報）を充足することのできる新規 Actor を追加する．

上記 4 つの手順を繰り返すことにより，d*による Assurance Case を作成する．

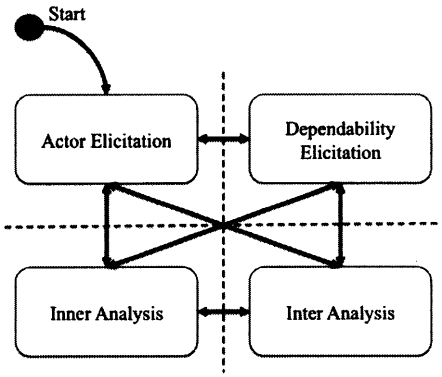


図 1 d*Framework の手順

3. d*Framework の適用評価

以下では適用評価の対象とする“エレベーターシステム”について述べ，次いで d*による Assurance Case の作成手順を示す．

3.1. 対象システム

例題としたエレベーターシステムのシステム構成を図 2 に示す．このシステムは，“巻上機”，“制御盤”，“ロープ”，“かご”，“ドア”，“つり合いおもり”，“のりばドア”，“乗客”，“のりば”で構成されている．複数のサブシステムが相互に協調して動作するシステムである．

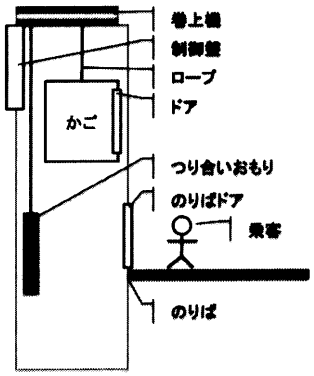


図 2 エレベーターシステム

3.2. Assurance Case の作成

例題としたエレベーターシステムに対して，d*を使った Assurance Case の作成を試みた．以下の節では，図 1 の各手順について，作成した d*図を示す．

3.2.1. Actor Elicitation

d*では，Assurance Case 内に Actor を記述できる．Actor Elicitation 手順では Actor を抽出する．ここでは，Assurance Case 作成を開始するにあたって最初に実行する Actor の抽出について示す．すなわち，“エレベーターシステム”のシステム構成図から Actor を抽出した状況を図 3 に示す．図 2 の“エレベーターシステム”の構成要素の全てを，8 個の Actor として抽出している．

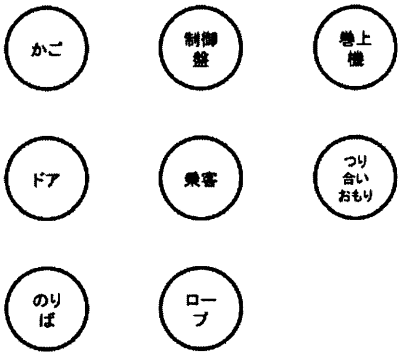


図 3 Actor Elicitation

3.2.2. Dependability Elicitation

d*では，Actor 間に Actor 相互に関係する Dependability 情報を記述できる．ここでは，7 個の Actor の間に 7 個の Dependability 情報を抽出した状況を示している（図 4）．

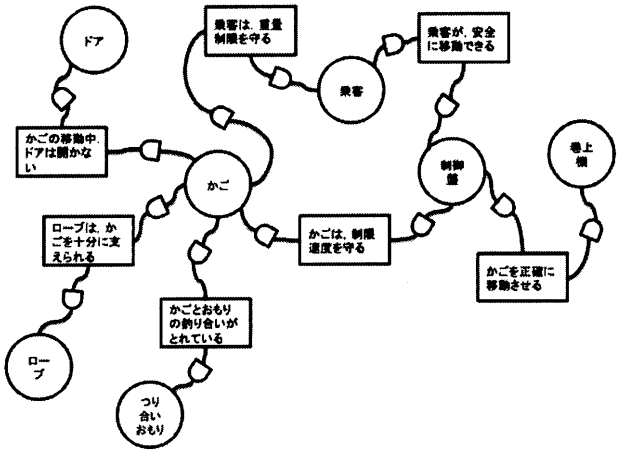


図 4 Dependability Elicitation

3.2.3. Inter Analysis

d*では，Actor 間の Dependability 情報を，GSN を用いて分析（分解）できる．ここでは，“制御盤”と“かご”の間にある Dependability 情報“かごは，制限速度を守る”を分析した状況を示している（図 5）．分析の結果，“調速機”の導入が必要であることがわかったので，図 5 では新規 Actor として“調速機”を追加している．つまり Inter Analysis の結果を受けて Actor Elicitation の手順を実行している．このように，d*の各手順は，相互に関係しており，ある手順の結果を受けて，他の手順を実施する必要が発生する場合がある．

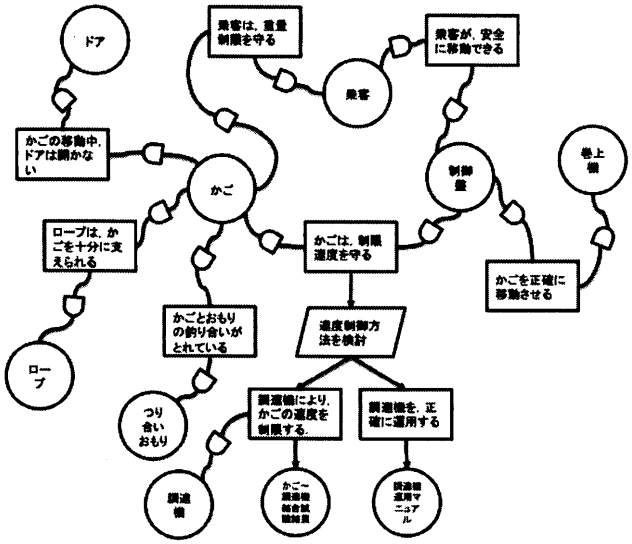


図 5 Inter Analysis

3.2.4. Inner Analysis

d*では，Actor 内の Dependability 情報を，GSN を用いて分析できる．ここでは，それまでの分析結果として抽出されていた Actor の一つである“かご”の内部を分析している．図 6 は，“かご”の内部及びその周辺を拡大した図である．トップゴールの下で 2 つの Strategy“全ての危険な状態について検討”，“全ての操作シナリオについて検討”を考え，ゴール分割を行うことで分析を進めている．

他 Actor から依存されている Dependability 情報についても検討している．分析を行った時点で，このような状態にあった 2 つの Dependability 情報“かごは，制限速度を守る”，“かごは，調速機の制限速度を守る”は，“かご”内部の Goal である“制限速度を守る”に関連づけられている．その後，“かご”内部において保証手段が検討されている．また，内部分析の結果わかった他 Actor に依存する必要がある Dependability 情報についても検討している．このような Dependability 情報 3 つのうち“かごは，地面に衝突しない”は，この時点において依存先となる他 Actor が存在しなかったため，Actor Elicitation 手順に進み新規 Actor“緩衝盤”を抽出している．

“静止中安全である”の下に付与した菱形は，この Goal がまだ分析途中であることを表す GSN の記法である．すなわち，この Goal については，今後も分析を継続する必要があることを表現している．

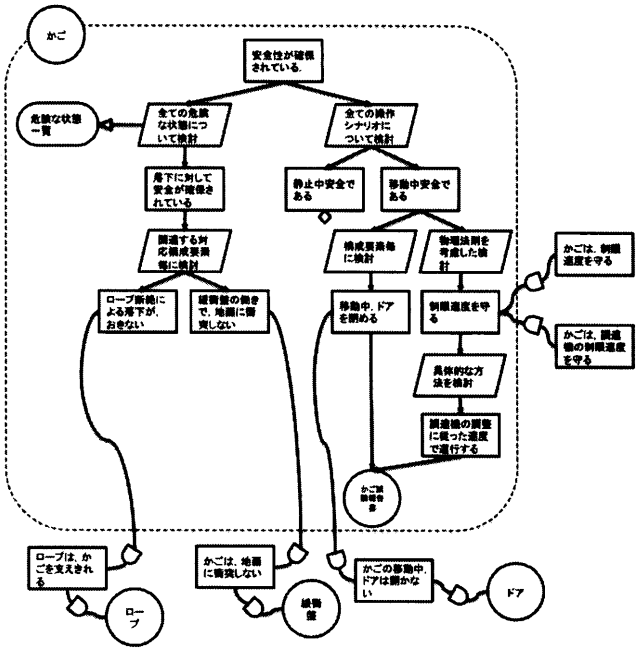


図 6 Inner Analysis（かご内部）

3.2.5. Assurance Case 作成の結果

本稿では，上述したように図 6，図 7，図 8 の d*図を作成した．この結果，表 2 に示すような d*Framework の Actor, Goal, Solution の個数となった．ただし，この個数は Assurance Case 作成の途中段階での結果である．このまま作成を進めればエレベーターシステムについて複数の Actor を考慮しながら分析し，Assurance Case を作成することができることは明らかである．

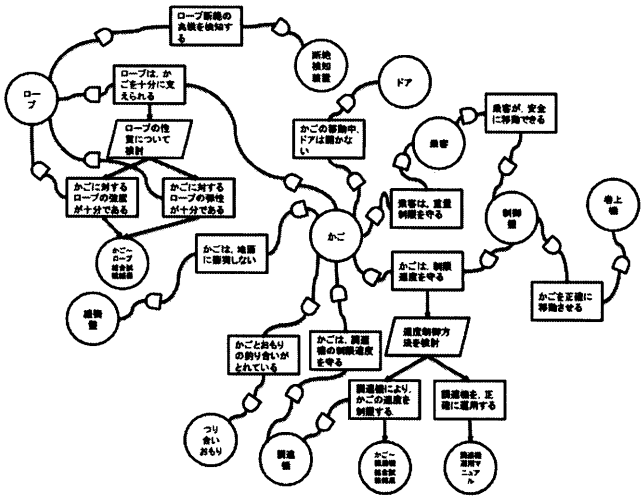


図 7 Assurance Case（全体図）

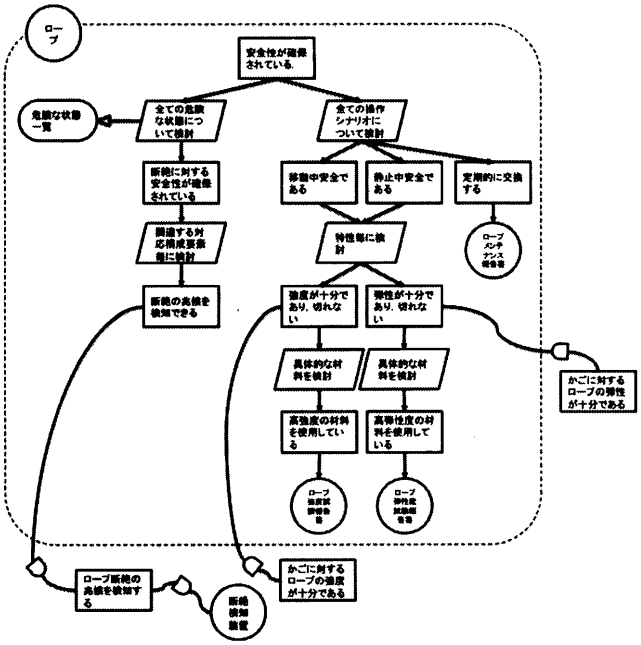


図 8 Inter Analysis（ロープ内部）

表 2 Actor, Goal, Solution の個数

要素	個数
Actor（図 7）	10
Actor 間の Goal（図 7）	14
Actor 間の Solution（図 7）	2
Actor（かご）内の Goal（図 6）	9
Actor（かご）内の Solution（図 6）	1
Actor（ロープ）内の Goal（図 8）	10
Actor（ロープ）内の Solution（図 8）	3

4. 考察

4.1. 有効性

エレベーターシステムの例題に対して d*Framework を適用した結果，d*が持つ“Dependability 情報を，Actor 間と Actor 内の双方で抽出できるという特徴”により，Dependability 情報の検討漏れの防止が期待できることがわかった．また，この特徴は，一般的なシステム開発の試験工程である単体試験，結合試験のような段階的な Dependability の確保と親和性が高いこともわかった．たとえば，単体試験による Dependability の保証に関する情報を Actor 内に記述し，結合試験による保証は Actor 間に記述できる．

4.2. 課題

例題に適用した結果，いくつかの課題があることがわかった．Inter Analysis と Inner Analysis で同じようなゴール分解が行われ，情報が重複しているようにみえる場合がある．Inter Analysis と Inner Analysis で実施すべき分析の質の違いを区別することが難しく，また区別できたとしても同様の分析になってしまうのが原因である．しかし，Dependability 情報に対する分析の抜け漏れを防ぐという観点から考えると，この問題は許容できると考えられる．また，Assurance Case の図が複雑になってしまう問題がある．本稿で示した例題は，Assurance Case 作成の初期段階に過ぎないが，かなり複雑な図になってしまっている．この後作成が進めば，図は一層複雑になっていき視認性なども劣っていくことが想定される．本格的な利用を考えると，今後この問題を解決できるツールの開発が課題になることが考えられる．関連研究に示した，D-Case エディタはこの課題を克服する手段として期待される．

4.3. i*, SARM, d*の比較

本稿で適用評価した d*と，Actor を考慮しながら分析を実施することができる i*ならびに SARM を，1）モデル作成の目的，2）適用工程，3）記法（要素数等），4）Actor 間の依存関係”について比較すると，表 3 に示す結果となった．

表 3 i*, SARM, d*手法の比較

	i*Framework	SARM	d*Framework
モデル作成の目的	システムの要求分析	システムのセキュリティ要求分析	システムの Dependability の確保
適用工程	要求定義	要求定義	要求定義，設計，製造，試験，運用
記法（要素数等）	図 構成要素：5 種類（Actor, Goal, Task, Resource, Soft-Goal）	表 構成要素：5 種類（Actor, Goal, Task, Resource, Soft-Goal）	図 構成要素：5 種類（Actor, Goal, Strategy, Solution, Context）
Actor 間の依存関係	Actor 間の関係に，Goal, Task, Resource, Soft-Goal を記述できる	Actor 間の関係に，Goal, Task, Resource, Soft-Goal を記述でき，Goal, Soft-Goal に関しては分析できる	Actor 間の関係に，Goal を記述でき，その内容を分析できる

表 3 の 3 つの手法は，モデル作成の目的が異なっている．従って，それぞれの手法は互いに補完しあう関係にある．[8]で提案されているスパイラルレビューのように，d*を他の手法と組み合わせて利用するための検討が必要である．

4.4. GSNとの比較

GSNとd*は，共にAssurance Caseの作成に利用することができる．d*の内部ではGSNを利用しているので，特徴に重なっている部分が多い．従って，d*の特徴であるActorの記述と，その間のDependability情報を分析できるという点において違いが出る．GSNでは，複数のActorが関与するシステムにおけるモデル作成においても，Actorの情報を明確に分けて分析することをしない．一方，d*では明確に分けて分析することができる．この特徴により，複数のサブシステムが関係するシステムのAssurance Caseの作成では，d*の特徴が生かせると考えられる．

5. 関連研究

近年，Assurance Caseに関する研究が，数多く実施されている[1][2][3][4][5]．Assurance Caseの作成プロセスに関する研究[1]では，d*でも使用しているContextを含めたGSNの記法や，GSNグラフのパターンを記述するための記法を導入している．また，Assurance Case作成の事例研究も実施されている．[2]は，医療機器分野におけるAssurance Case作成の事例研究である．これらAssurance Caseに関する研究では，Assurance Caseの作成にGSNが利用されている．GSNを利用したAssurance Caseの作成は，システム内における詳細な単位のサブシステム等であるActorを明確に考慮していない．それに対して本稿で適用評価したd*は，システム内のActorを明確に考慮し，Actor間とActor内のDependability情報を記述できる．

要求工学分野では，i*，SARM等のActorを考慮した分析手法が提案されている[7][8]．また，UML（Unified Modeling Language）においても，ユースケース図等でActorが考慮されている．これらの手法と本稿で評価したd*は，モデル作成の目的など異なる点がある．

5.1. Dependability Case

Dependability caseの作成を支援するためにD-Caseエディタ[9]が開発されている．D-Caseエディタではは GSNを拡張した記法を作成できる．今後d*Fの作成を支援するためのエディタの開発が期待される．

6. まとめと今後の課題

本稿では，エレベーターシステムを例題にしたAssurance Caseをd*を使って作成し評価した．d*は，システム内のActorを考慮しながら，分析を進めることが出来る点に特徴がある．この特徴は，サブシステムに分割されているシステムの，Assurance Caseの作成に有用であると考えられる．また，i*，SARM，d*を目的と機能性の観点から比較した．3つの手法は，それぞれモデル作成の目的が異なり，補完し合える関係にある．今後，複数の種法と組み合わせた使い方の検討が必要となる．

本稿の例題適用においてd*によるAssurance Case作成手順を提案するとともに，エレベーター問題に適用することにより，手順の実施可能性を確認した．しかし，まだ一部を記述しただけであり，全体を記述する必要がある．また実際のシステムを対象にした評価も必要である．さらに，d*のより詳細な活用方法についても検討が必要である．

今後は，d*を使った事例研究を継続的に実施することにより，d*の有効性を検証する必要がある．また，d*の形式化も今後の課題である．

文　献

[1] Kelly, Tim P. “Arguing Safety - A Systematic Approach to Safety Case Management,” DPhil Thesis, York University, Department of Computer Science Report YCST, May 1999.

[2] Mark-Alexander Sujan, Floor Koornneef, and Udo Voges, Goal-Based Safety Cases for Medical Devices: Opportunities and Challenges, F. Saglietti and N.Oster (Eds.): SAFECOMP 2007, LNCS 4680, pp. 14-27, 2007.

[3] Ankrum, T. Scott and Alfred H. Krombolz. “Structured Assurance Cases:Three Common Standards, “Slides presentation at the Association for Software Quality (ASQ) Section 509 meeting, the MITRE Corporation, 25, January 2006

[4] Bishop, P., Bloomfield, R., Guerra, S.: The Future of Goal-Based Assurance Cases. In:Proc. Workshop on Assurance Cases, pp. 390-395 (2004)

[5] Thomas Rhodes, Frederick Boland, Elizabeth Fong, Michael Kass , Software Assurance Using Structured Assurance Case Models, NIST Interagency Report 7608,

[6] Shuichiro Yamamoto, “How Can We Cope with the Changing Requirements?”, WOSD 2011,

[7] Eric S. K. Yu. Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering. In 3rd IEEE Int. Symp on Requirements Engineering, pages 226-235, Washington DC, Jan 1997.

[8] 金子朋子，山本修一郎，田中英彦，“アクタ関係表に基づくセキュリティ要求分析手法（SARM）を用いたスパイラルレビューの提案”，2009 Information Processing Society of Japan.

[9] D-Case Editor A Typed Assurance Case Editor, www.il.is.s.u-tokyo.ac.jp/deos/dcase/