一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
KBSE2013-46 (2013-9)

SEMAT と保証ケースの関係についての考察

山本　修一郎

名古屋大学 情報連携統括本部 情報戦略室
〒464-8601 名古屋市千種区不老町
E-mail:　syamamoto@ acm.org

あらまし　Jacobson らによる理論に裏打ちされた実践的なソフトウェア開発方式 SEMAT(Software Engineering Method and Theory)が，現代のソフトウェア工学の課題を解決する方法として注目されている．しかし，具体的な有効性については明確ではなかった．本稿では，SEMAT と保証ケースの関係について考察する．

キーワード　SEMAT，ディペンダビリティケース，アシュアランスケース

Considerations on the relationship between SEMAT and Assurance Case

Shuichiro Yamamoto
Nagoya University, Strategy Office, Information and Communications Headquarters
Furo-cho. Chikusa-ku, Nagoya 464-8601 Japan
E-mail: syamamoto@ acm.org

Abstract *Although SEMAT(Software Engineering Method and Theory) has been attracted to solve modern Software Engineering issues, the effectiveness of the method was not sufficiently clear. In this paper, the relationship between SEMAT and dependability cases is considered.*

Keyword SEMAT, Dependability case, Assurance Case

1 はじめに

最近，Use Case 図や UML で知られる Jacobson らが取り組んでいることが，理論に裏打ちされた実践的なソフトウェア開発方式 SEMAT(Software Engineering Method and Theory)である[1][2][3][4].

本稿では要求工学の立場から SEMAT について説明し，次いでシステムの安全性を確認するために注目されている，安全性ケース(Safety case)，アシュアランスケース（Assurance case，保証ケース）やディペンダビリティケース（Dependability case, D-Case）[1][2][3][4][5][6][7]との関係について考察する．

なお本稿では，安全性ケースやアシュアランスケース，ディペンダビリティケースを総称して保証ケースという用語を用いる．

以下では，まず第 2 節で SEMAT を説明する．第 3 節で，SEMAT と保証ケースの関係について述べる．第 4 節で，まとめと今後の課題について述べる．

2 SEMAT の概要

2.1 SEMAT プリンシプル

SEMAT には，活動可能性，拡張性，実務性という 3 つのプリンシプルがある．

(1)活動可能性(Actionable)

SEMAT では，アルファ(alpha, Aspiration Led Progress and Health Attribute)によって，活動すべき目標を指示できる．アルファでは，(1)開発活動の進捗と健全性について，(2)肯定的な成果の達成に着目することにより，(3)個別ではなく全体を検討する必要がある．ソフトウェア開発活動全体の健全な進行状態を示す本質的な要素がアルファである．基本的なアルファとして，SEMAT では，機会，ステークホルダ，要求，ソフトウェア・システム，作業，作業方法，チームを定義している．これらのアルファの内容については後述する．

（２）拡張性(Extensible)

SEMAT では，多様な開発に適用できるだけでなく，必要に応じて実践的手法を追加できる．たとえば，アルファでは，ソフトウェア開発活動の基本的な状態だけを定義しており，成果物や作業の内容については，具体的な活動を選択できるようになっている．これにより，アジャイル開発や大規模開発に対して SEMAT を適用できる．

（３）実務性(Practical)

SEMAT では，実務担当者を支援するために，アルファの状態を定義するとともに，各状態にあることを確認するための検査目標（チェックポイント）を一覧提示するアルファ状態カードを用意することにより，実践的で具体的な思考の枠組みを提供している．

2.2 従来のソフトウェア工学との比較

上述した SEMAT のプリンシプルは，従来のソフトウェア

工学の問題を解決するために，新しいソフトウェア開発方式の再構築に向けた取組み姿勢を表現している．SEMAT カーネルのプリンシプルが，従来のソフトウェア工学と，どのように異なるかを比較した結果を表 1 に示した．

表 1 から分かることは，SEMAT が従来のソフトウェア工学には，3 つの問題があると認識しており，それを解決するのが 3 つのプリンシプルであるということである．

表 1　SEMAT カーネルと従来のソフトウェア工学

プリンシプル	SEMAT カーネル	従来方式の問題点
活動可能性	アルファによって，活動すべき目標を指示できる．ソフトウェア開発活動の健全な進行状態を示す本質的な要素がアルファである．	文書などの成果物
拡張性	多様な開発に適用できる．必要に応じて実践的手法を追加できる．	類似した手法の集まり 手法の変更で，良い手法を悪い手法で置き換えてしまう
実務性	実践的で具体的な思考の枠組みにより，実務担当者を支援	プロセスエンジニアや品質エンジニアを支援

SEMAT では，後述する 7 種類のアルファとその状態と検査目標を定義するだけで，開発プロセスを明示的に定義していない．開発プロセスと工程生産物は，状態間の遷移を実現するために選択されるプラクティスによって具体化されるようになっている．ここで，特定の目的に対して何かを実施するために反復できるアプローチがプラクティスである．プラクティスの組合せがメソッドである．

このように，開発プロジェクトの進捗と健全性を可視化するために，ソフトウェア開発に共通するアルファ状態に着目したゴール指向手法が SEMAT である．

2.3 SEMAT の主な図形要素

SEMAT では，アルファ，状態，活動空間，活動，作業成果物に対して，表 2 に示す 5 種類の図形要素を用いている．

表 2　SEMAT で用いられる主な記号

名称	図形	説明
アルファ		業務を成功させるために，進捗と健全性を監視する必要のある物事．アルファ・カーネルには，機会，ステークホルダ，要求，ソフトウェアシステム，作業，作業方法，チームがある．
状態		ある条件が成立する状況を状態で表現する．アルファごとに状態が定義されている．
活動空間		一つ以上の実施すべき作業があることを明示する．具体的な活動によって活動空間を埋める必要がある．
活動		一つ以上の作業項目の種類を活動によって定義する．作業をどのように遂行すべきかを活動が指示する．
作業成果物		作業によって生産される成果物を定義する．作業成果物によってアルファを記述する．

2.4 カーネル・アルファ

カーネル・アルファの 7 つのアルファは，カスタマ層，ソリューション層，エンデバー層からなる 3 層に分類されている．カスタマ層のアルファには，機会とステークホルダがある．ソリューション層のアルファには，要求とソフトウェア・システムがある．エンデバー層のアルファには，作業，作業方法，チームがある．

◆機会

ソフトウェア・システムの開発・変更を適切化する周辺環境の集合が機会である．

◆ステークホルダ

ソフトウェア・システムに影響を与え，ソフトウェア・システムから影響を受ける人々，集団，組織がステークホルダである．

◆要求

機会に対処してステークホルダを満足させるために，ソフトウェア・システムが実行すべきことが要求である．

◆ソフトウェア・システム

ソフトウェア，ハードウェア，データから構成されたシステムがソフトウェア・システムである．ソフトウェアの実行により，ソフトウェア・システムが価値を提供する．

◆作業

成果を達成するために，実行される精神的，物理的な努力からなる活動が作業である．

◆作業方法

作業を指導し支援するために，チームによって利用される必要に応じた実践的な手法とツールの集合が作業方法である．

◆チーム

ソフトウェア・システムの開発，保守，展開，支援に活発に従事する人々の集団がチームである．

2.4 カーネル・アルファの状態

表 3 に示すように，カーネル・アルファに対して状態が定義されている．

表 3　カーネル・アルファの状態

アルファ	状態
機会	識別，必要解，価値形成，実現可能，対処，利益獲得
ステークホルダ	認識，表現，参画，合意中，展開充足，使用充足
要求	着想，限定，首尾一貫，受容可能，対処，充足
ソフトウェア・システム	構成選択，例証，利用可能，用意，運用可能，廃棄
作業	開始，準備，開始，制御中，完結，終了
作業方法	原則確立，基礎確立，使用中，環境整備，有効，廃棄
チーム	手配，形成，協調，実演，解散

たとえば要求アルファの状態は表 4 に示すように定義されている．

2.5 検査目標

カーネル・アルファの状態を判定するための規則が検査目標である．SEMAT では，すべてのカーネル・アルファの状態に対して，表 5 のような検査目標が定義されている．

表 4　要求アルファの状態

状態	説明
着想 Conceived	新しいソフトウェア・システムのニーズがステークホルダ間で合意された状態
限定 Bounded	システムのスコープと要求管理方式が確立された状態
首尾一貫 Coherent	新システムの本質的特徴が明確に定義された状態
受容可能 Acceptable	初期ソリューションとしてステークホルダが受容できるシステムを要求が定義した状態
対処 Addressed	システムを開放し利用するだけの十分な要求が実現された状態
充足 Fulfilled	システムのニーズを完全に達成していることにステークホルダが合意できる十分な要求が実現された状態

表 5　要求アルファの検査目標

状態	検査目標
着想	① 新システムのニーズが明確である ② ユーザを識別している ③ 初期スポンサを識別している
限定	① 目的と範囲について合意している ② 成功基準が明確である ③ 要求管理方式について合意している ④ 制約と前提を識別している
首尾一貫	① ビッグピクチャが明確・参画者間で共有している ② 重要な利用シナリオについて説明している ③ 優先順位が明確である ④ 矛盾に対処している ⑤ 影響について理解している
受容可能	① ステークホルダが合意可能なソリューションを要求が記述している ② 合意要求の変更率が低い ③ 価値が明確である
対処	① システムが受容可能であるための十分な要求を実現している ② システムを運用することの価値についてステークホルダが合意している
充足	① システムが要求とニーズを完全に充足している ② システムが不完全であることを示す要求が残っていない

2.6 アルファに基づくゴール指向活動定義手法

SEMAT では，アルファの状態に着目してプラクティスを定義することができる．この手法を整理すると，以下のような手順にまとめることができる．

[手順 1] 現在のアルファ状態を確認

[手順 2] 将来のあるべきアルファ状態を明確化

[手順 3] 現在の状態から将来のアルファ状態に移行するために必要な活動空間を定義

[手順 4] 活動空間をプラクティスの活動で具体化することにより，作業を実施

3.SEMAT と保証ケース

以下では，SEMATと保証ケースの関係について議論する．保証ケースの研究動向については既存の報告[7]〜[38]を参照されたい．

SEMAT と保証ケースには， SEMAT によるソフトウェア開発を保証ケースが支援するという関係と，保証ケースを開発するためのプラクティスを SEMAT によって表現して活用するという関係がある．

3.1 保証ケースによる SEMAT の支援

SEMAT ではαカーネルの属性によって，ソフトウェア開発状況を適切に確認することが基本である．たとえば，要求αでは，着想,限定,首尾一貫,受容可能,対処,充足という要求の７つの状態と，要求がその状態であることを確認するためのチェックリストが状態ごとに提示されている．

一方，保証ケースでは，対象に対する主張についての証拠を確認することによって主張の妥当性を保証することができる．

したがって，保証ケースでαカーネル属性状態についての主張を確認することができれば，保証ケースによって SEMAT 活動を支援できることになる．以下では，その方法を考える．

カーネルαの基本的な考え方を整理すると，次のようになる．

カーネルαが事後状態 Q であるためには，カーネルαが事前状態 P にあることと，チェックリスト C が満たされることが必要である．

図 1 に示すような保証ケースを用いることにより，この考え方を確認することができる．カーネルαには開発活動の進行に従った複数の状態があって，開始状態から終了状態に遷移する．図 1(a)では，カーネルアルファに，P と Q という状態があり．状態 P で，チェックリスト C が満たされれば，状態 Q に遷移することを示している．このとき，図 1(b)に示したように，状態 Q に対して，上位主張「αが状態 Q である」を作成する．この主張が成立することを示すためには，「αが状態 P である」ことと「チェックリスト C が満たされていること」が必要であることから，「条件を確認する」という説明ノードを用いて，上位主張を，これらの下位主張に分解している．

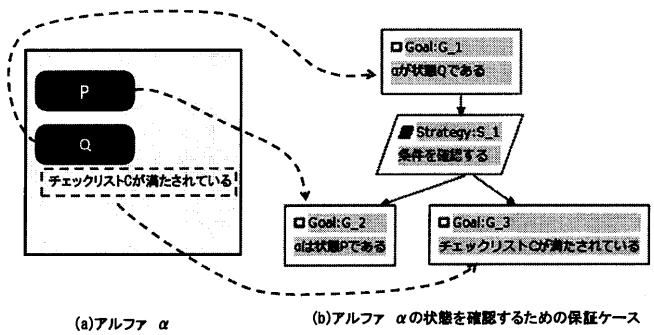

図 1　アルファからの保証ケース作成

アルファ状態が 3 個以上あるときも，同様にしてこの分解を多段階に適用することができる．このとき，カーネルアルファの最終状態から開始状態に向って逆方向に遡りながら，保証ケースを最上位ノードから最下位ノードに向って分解していく．

たとえば，表 4 に示したように，要求アルファの状態は，着想状態から始まり,限定状態,首尾一貫状態,受容可能状態,対処状態を経て,充足状態で終了する．したがって，「要求アルファが充足状態である」を最上位の主張として，状態遷移とは逆に，「要求が対処状態である」,「要求が受容可能状態である」,「要求が首尾一貫状態である」,「要求が限定状態である」と分解していき，最後に，「要求が着想状態である」という主張の証拠を確認するように，保証ケースを作成することになる．要求アルファに対する保証ケースの例を図 2 に示す．

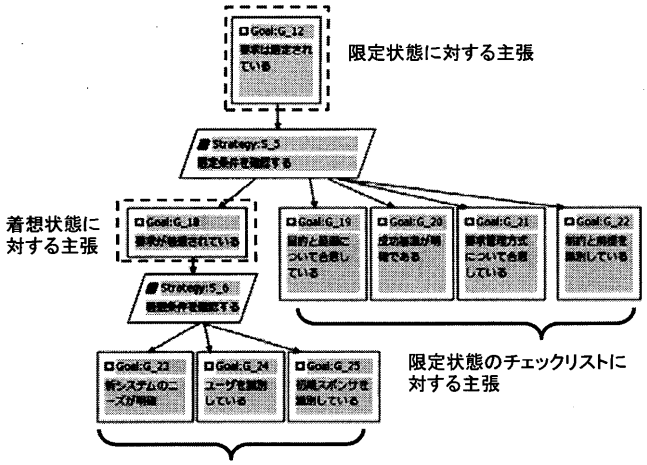

図 2　要求アルファに対する保証ケースの例

他のカーネルアルファに対しても，同様にして保証ケースを作成できることは明らかである．

以下では，保証ケース開発メソッドを SEMAT で定義する方法について考えよう．

3.2 SEMAT による保証ケース開発プラクティス

SEMAT のプラクティスでは，作業対象と作業内容を，それぞれアルファと成果物，活動空間と活動で定義する．

このため，表 6 に示したように，SEMAT ではカーネルアルファに基づいて，メソッドを構築することができる．

表 6　メソッドの構築手順

手順	説明
1	メソッドを構成するための適切なカーネルアルファを選択
2	アルファに作業成果物を追加
3	活動空間を活動で具体化することにより，プラクティスを構成
4	複数のプラクティスを合成することにより，メソッドを構築

以下では，保証ケースを作成するためのメソッド構築手順を説明する．

(1)カーネルアルファの選択

保証ケースでは，機会，ステークホルダ，要求，ソフトウェアに対する主張を確認することになる．保証ケースに関連するアルファとして，機会，ステークホルダ，要求，ソフトウェアがある．これらのアルファを選択した理由は，以下の通り．

【選択理由】機会

ソフトウェア・システムの開発・変更を適切化する周辺環境が機会であることから，保証ケースで記述される主張の中で機会について言及する必要がある．

【選択理由】ステークホルダ

ソフトウェア・システムに影響を与え，ソフトウェア・システムから影響を受ける人々，集団，組織がステークホルダであることから，保証ケースの主張について，ステークホルダと合意する必要がある．

【選択理由】要求とソフトウェア・システム

機会に対処してステークホルダを満足させるために，ソフトウェア・システムが実行すべきことが要求であることから，「ソフトウェア・システムが要求を実現している」という主張を，保証ケースで確認する必要がある．

(2)下位アルファの追加

保証ケースを作成する活動に対する進捗状況とその健全性を確認するためのアルファが必要になる．追加するアルファの候補として，保証アルファとリスク・アルファが考えられる．テストと同じように，保証ケースはソフトウェア・システムの性質を確認するための成果物である．これに対して，リスクはソフトウェア・システムがもたらす環境への脅威と考えられるので，機会アルファがソフトウェア・システムの周辺環境を表すのと同じように，リスクをアルファと考えてよいと判断した．

また保証ケースでは，安全性やディペンダビリティ，セキュリティなど，対象とする関心事に問題がないことを保証する．たとえば，安全性を保証するための安全性ケースでは，対象システムにリスクがないことを保証しようとする．このため保証ケースにおける監視対象として，リスク・アルファを考えることは妥当である．この場合，テスト項目と同じように，ソフトウェア・システムについての生産物が保証ケースであると考える．

次に，リスク・アルファが対応する層として，カスタマ層かソリューション層かを判別する必要がある．リスクを解消するためにはリスク対策が必要となることと，要求やソフトウェア・システムにもリスクが発生すること，リスク対策はソリューションで考慮することになることから，リスク・アルファはソリューション層であるとした．

リスク・アルファの状態として，識別，分析，定義，合意，対処，確認，監視という 7 状態が考えられる．

リスク・アルファの 7 状態に対する状態チェックリストを表 7 に示した．

(3)活動空間の定義

リスク・アルファの活動空間では，現在のリスク・アルファ状態から将来のリスク・アルファ状態を実現するために必要な作業を定義する．現在のリスク・アルファ状態とは，ソフトウェア・システムにリスクが潜在している状態である．将来のリスク・アルファ状態とは，ソフトウェア・システム

の潜在的なリスクを識別して適切な対策をとったソフトウェア・システムが実現できていることが確認されている状態である．このために必要なプラクティスとして，リスク識別とリスク対策保証について検討する．

まずリスク識別に対する活動空間としては，要求理解，リスク分析，ステークホルダニーズ理解が考えられる．要求理解のための活動では，保証ケースの主張をステークホルダが合意する．リスク分析のための活動では，ソフトウェア・システムのリスク分析を実施する．ステークホルダニーズ理解のための活動では，リスク・レビュを実施する．

表 7　リスク・アルファ状態チェックリストの例

状態	チェックリスト
識別	システムの環境と構造が明確 ステークホルダを識別 ハザードを識別
分析	リスクの範囲について合意 影響判断基準が明確 重要な利用シナリオについて説明
定義	リスクが明確かつ参画者間で共有 優先順位が明確 制約と前提を識別 影響について理解
合意	ステークホルダが合意可能なリスク対策を要求が記述 保証価値が明確
対処	システムが受容できる十分なリスク対策を実現 リスクを前提にしたシステム運用についてステークホルダが合意
確認	システムがリスク対策を完全に充足 対処不能なリスクが発生することを示す利用シナリオが残っていない
監視	システムが安全要求を完全に充足 システムのリスク対策が不完全であることを示す要求が残っていない

表 8　リスク識別プラクティスの構成例

プラクティス	要求抽出プラクティス	
作業対象	アルファ	成果物
	リスク	リスク一覧
	要求	保証主張
	ソフトウェア・システム	アーキテクチャ
作業内容	活動空間	活動
	要求理解	保証主張をステークホルダが合意
	リスク分析	ソフトウェア・システムのリスク分析
	ステークホルダニーズ理解	リスク・レビュ

表 9　リスク対策確認プラクティスの構成例

プラクティス	リスク対策確認	
作業対象	アルファ	成果物
	リスク	リスク一覧
	要求	保証主張
	ソフトウェア・システム	保証ケースの証拠
作業内容	活動空間	活動
	要求理解	保証主張とリスク対策をステークホルダが合意
	リスク対策確認	保証ケースによってソフトウェア・システムのリスク対策を確認

次にリスク対策保証に対する活動空間について考えると，要求理解とリスク対策確認がある．要求理解のための活動では，保証ケースの主張とリスク対策をステークホルダが合意する．またリスク対策確認のための活動では，保証ケースによってソフトウェア・システムのリスク対策を確認する．

この結果をプラクティス定義表でまとめると表 8,表 9 に示したようになる．

3.3 留意点

保証ケースを作成するプラクティスを SEMAT で実現することで，その妥当性を保証ケースで確認できる．したがって保証ケース作成プロセスの妥当性を保証ケースによって再帰的に確認できる．

以下では，保証ケースを作成するメソッドを SEMAT で実現する上で明らかとなった留意点について述べる．

(1)アルファの選択と追加

メソッドで必要となるアルファを選択するために，明確な理由が必要である．また選択したアルファだけで十分であることを確認する必要がある．対象とする作業で必要なアルファが不足していれば追加する必要がある．本稿では保証ケース作成作業を対象とするため，リスク・アルファを追加した．このとき，アルファとして，保証ケースなのかリスクなのかについて検討した．本稿では，リスク・アルファだけを選択したが，保証ケースを選択することもできる可能性がある．

この点で，「進捗と健全性を監視する必要のある状態を持つ対象物がアルファである」という定義だけでは，選択したアルファの適切性を判断するには不十分である．したがって，メソッドを定義する場合，より明確なアルファ選択基準が求められる．

(2)活動空間と活動

アルファを現在状態から将来状態に変換するために，活動空間を定義する必要がある．この場合，具体的な活動は考えやすいが活動空間は抽象的なので，工夫が必要である．このため，活動空間の妥当性判断基準が必要である．たとえば，筆者は，活動空間と活動の関係を，目的と手段の関係であると考えている．しかし，より客観的な活動空間の考え方が必要であると思われる．

(3)メソッドとプラクティス

メソッドを何個のプラクティスから構成するかについて検討する必要がある．本稿の例では，保証ケース作成メソッドを，リスク識別プラクティスとリスク対策確認プラクティスから構成したが，もっと多くのプラクティスを追加することもできるだろう．この問題はプラクティスの粒度をどのように判断するかということでもある．プラクティスが，アルファを開始状態から終了状態に変換する作業を支援するという観点からみると，プラクティスとアルファ状態をどのように対応づけるかという問題でもある．この点でプラクティスの最適性を判断する基準が必要であると思われる．

4 おわりに

本稿では，保証ケースと SEMAT の関係について述べた．まず保証ケースによって SEMAT を支援する方法を説明した．次いで，保証ケース作成メソッドを SEMAT アルファとプラクティスで定義した．さらに，SEMAT を用いてソフトウェア開発プラクティスを記述する上での留意点を明らかにした．

今後，実際のソフトウェア開発に SEMAT の適用が進むことによって，本稿で指摘した事項についての理解を進め，ソフトウェア開発プラクティスを統一的に整理する手法を検討していく．

謝辞

本研究は CREST「実用化を目指した組込みシステム用ディペンダブル・オペレーティングシステム」研究領域（DEOS プロジェクト）の支援を受けたものである[39][40][41].

参考文献

[1] Ivar Jacobson, Pan Wei Ng, Paul E. McMahon, Ian Spence, and Svante Lidman, “The Essence of Software Engineering – Applying the SEMAT Kernel”, Addison-Wesley Pearson Education, 2013
[2] Ivar Jacobson, Pan-Wei Ng, Paul E. McMahon, Ian Spence, and Svante Lidman, The Essence of Software Engineering: The SEMAT Kernel, CACM, Vol.55, No.12, pp. 42-49, 2012
[3] SEMAT, http://semat.org/
[4] Ivar Jacobson, Michael Striewe, Ashley McNeile, The ESSENCE Language Concepts, Examples, Insights, http://laser.inf.ethz.ch/2012/slides/Jacobson/The Semat Language-5.pdf
[5] Essence – Kernel and Language for Software Engineering Methods, http://www.omg.org/spec/Essence/1.0
[6] GTSE2013, Technical Program, http://semat.org/?page_id=760
[7] Kelly, T. P, A Six-Step Method for the Development of Goal Structures, York Software Engineering, 1997
[8] T. Kelly. “Arguing Safety, a Systematic Approach to Managing Safety Cases”. PhD Thesis, Department of Computer Science, University of York, 1998
[9] J. A. McDermid. Software safety: where's the evidence? In SCS '01: Proceedings of the Sixth Australian workshop on Safety critical systems and software, pages 1-6, Darlinghurst, Australia, Australia, 2001. Australian Computer Society, Inc.
[10] I. Bate, T. Kelly, Architectural considerations in the certification of modular systems, Reliability Engineering and System Safety 81, pp.303–324,2003
[11] Tim Kelly and Rob Weaver, The Goal Structuring Notation – A Safety Argument Notation, Proceedings of the Dependable Systems and Networks 2004 Workshop on Assurance Cases, July 2004
[12] Despotou G., Kelly T., Extending the Concept of Safety Cases to Address Dependability. In proceedings of the 22nd International System Safety Conference (ISSC), Providence, RI USA, proceedings by System Safety Society 2004.
[13] Jackson, D. et al, Software for dependable systems–sufficient evidence?, NATIONAL RESEARCH COUNCIL, 2008
[14] 山本修一郎, 松野裕, ディペンダビリティケース作成法に関する一考察, KBSE研究会, IEICE-112, vol. IEICE-SS-164, No. IEICE-KBSE-165, pp.61-66, 2012
[15] 松野 裕, 高井利憲, ヴァイセ パテゥー, 山本修一郎, アシュアランスケース構築法の提案, KBSE 研究会, 2012
[16] 松野裕, 山本修一郎,ユースケース分析に基づくディペンダビリティケース作成法の提案, KBSE研究会, IEICE-112, vol. IEICE-KBSE-419, KBSE2012-61, pp.19-24
[17] Vaise Patu, Yutaka Matsuno, Shuichiro Yamamoto, Application of D-Case to the usage flow diagram scenario of the Distributed E-Learning System called KISSEL in Asian Pacific Universities, KBSE 研究会, 2012
[18] 高間翔太, 松野裕, 山本修一郎, スーパーコンピュータ運用手順に対するディペンダビリティの確認手法の提案, 信学技報, vol. 112, no. 165, KBSE2012-18, pp. 37-42 2012
[19] 高間翔太, 松野裕, 山本修一郎, ディペンダビリティ・コンテクストの推定手法の提案, KBSE 研究会, 信学技報, vol. 112, no. 314, KBSE2012-42, pp. 25-30, 2012
[20] 徳野達也, 松野裕, 山本修一郎, エンタープライズアーキテクチャ開発プロセスに対するディペンダビリティケース作成法の提案, 信学技報, vol. 112, no. 165, KBSE2012-36, pp. 145-150 2012
[21] 徳野達也, 松野裕, 山本修一郎, TOGAF NEXT に対する ADM プロセステンプレートの提案, KBSE研究会, 信学技報, vol. 112, no. 314, KBSE2012-55, pp. 103-108, 2012
[22] 山本修一郎, 松野裕, ディペンダビリティケースへの責任属性導入法の検討, KBSE研究会, 信学技報, vol. 112, no. 314, KBSE2012-52, pp. 85-90, 2012
[23] 松野裕, ヴァイセ バトゥ, 山本修一郎, アシュアランスケースへの構造化文書の適用に関する調査, 信学技報, vol. 112, no. 165, KBSE2012-20, pp. 49-54, 2012
[24] Vaise Patu, Yutaka Matsuno, Shuichiro Yamamoto, Research framework for dependability science based on assurance cases, IEICE Tech. Rep., vol. 112, no. 165, KBSE2012-21, pp. 55-59, July 2012
[25] 猿渡卓也, 松野裕, 星野隆, 山本修一郎, Modular GSN の定式化, KBSE研究会, 信学技報 vol.112, No.165, pp.151-156, 2012
[26] Shuichiro Yamamoto, Yutaka Matsuno, d* framework: Inter-Dependency Model for Dependability, DSN 2012
[27] 山本修一郎, 松野裕, ディペンダビリティケース分解パターンについての考察, KBSE研究会, 信学技報, vol. 112, no. 496, KBSE2012-80, pp. 67-72, 2013
[28] 山本修一郎, 議論分解パターンに基づくディペンダビリティケースの作成実験, KBSE研究会, 信学技報, 信学技報, vol. 113, no. 71, KBSE2013-12, pp. 67-72,
[29] 松野裕, 高井利憲, 山本修一郎, D-Case 入門, ～ディペンダビリティ・ケースを書いてみよう！～, ダイテックホールディング, 2012, ISBN 978-4-86293-079-8
[30] 松野裕, 山本修一郎, 実践 D-Case,～ディペンダビリティ・ケースを活用しよう！～, ダイテックホールディング, 2013, ISBN 978-4-86293-091-0
[31] 山本修一郎, 主張と証拠, ダイテックホールディング, 2013, ISBN 978-4-86293-095-8
[32]山本修一郎, 保証ケース作成上の落とし穴：要求工学, (2013/3/17) – Kindle 版 ,ASIN:B00BVT80O2
[33]山本修一郎, 保証ケース議論分解パターン：要求工学, (2013/3/17) – Kindle 版,ASIN: B00BVTF7R0
[34]山本修一郎, システムとソフトウェアの保証ケースの動向：要求工学, (2013/3/18) – Kindle 版,ASIN: B00BWSUHPM
[35] Shuichiro Yamamoto, Yutaka Matsuno , An Evaluation of Argument Patterns to Reduce Pitfalls of Applying Assurance Case, 1st International Workshop on Assurance Cases for Software-intensive Systems (Assure 2013)
[36] D-Case 実証評価研究会, http://dcase.jp/
[37] D-Case エディタ, http://www.dependable-os.net/tech/D-CaseEditor/
[38] 松野裕, 山本修一郎, アシュアランスケースツールへのプログラミング言語技術の適用, KBSE研究会, 信学技報, vol. 112, no. 496, KBSE2012-81, pp. 73-78, 2013
[39] DEOSプロジェクト, http://www.crest-os.jst.go.jp
[40]DEOS プロジェクト, 2011 科学技術振興機構 White Paper DEOS-FY2011-WP-03J, *www.dependable-os.net/ja/topics/file/**White_Paper**_V3.0J.pdf*
[41] Mario Tokoro eds., Open Systems Dependability, Dependability Engineering for Ever-Changing Systems, CRC Press, 2012