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
KBSE2012-55 (2012-11)

TOGAF NEXTに対するADMプロセステンプレートの提案

徳野　達也† 松野　裕†† 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 情報戦略室 〒464-8601 愛知県名古屋市千種区不老町
E-mail: †tokuno@nagoya-u.jp, ††{matsu,yamamotosui}@icts.nagoya-u.ac.jp

あらまし TOGAFでは，ディペンダビリティ概念を包含するTOGAFの次期版TOGAF NEXTを開発中である．本稿では，前回の提案手法の問題点を改善する手法を提案し，実際にディペンダビリティケースを作成することで，その改善手法の有用性を確認する．そしてTOGAF NEXTにおけるアーキテクチャ開発プロセス(ADM)に対するディペンダビリティの確認を容易化するテンプレートを提案する．

キーワード エンタープライズ・アーキテクチャ，TOGAF NEXT，ディペンダビリティ，ディペンダビリティケース

A proposal on ADM process template for TOGAF NEXT

Tatsuya TOKUNO†, Yutaka MATSUNO††, and 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
E-mail: †tokuno@nagoya-u.jp, ††{matsu,yamamotosui}@icts.nagoya-u.ac.jp

Abstract TOGAF NEXT is the next version of TOGAF which includes a dependability. It is now under development. In this paper, the previous method is improved to new one. It is also confirmed the effectiveness of the improved method by creating dependability cases. We propose a template based on the created dependability cases to facilitate the validation of dependability in architecture development process(ADM) in TOGAF NEXT.

Key words Enterprise Architecture, TOGAF NEXT, Dependability, Dependability Case

1. はじめに

近年，システムの安全性を確認する手法としてアシュアランスケースやディペンダビリティケースが注目されている．ディペンダビリティケースによってディペンダビリティを保証する対象は数多く存在するが，ほとんどが特定のシステムを対象として行われている．しかし企業などの組織では特定のシステムではなく複雑化した複合的なシステム群を対象とすることが多い．そのため複数の文書と複数のディペンダビリティケースが集合レベルで対応することになることが多い．個々のシステムのディペンダビリティが検討されていても，それら全体を連携させた場合のディペンダビリティは検討が行われていないことがある．

そこで組織内の複数のシステムを最適化する手法であるエンタープライズ・アーキテクチャのディペンダビリティを確認することで，複雑化した組織のシステムのディペンダビリティを保証する．エンタープライズ・アーキテクチャがディペンダブルであるには最適化プロセスと最適化されたアーキテクチャがディペンダブルでなければならない．そのためまず最適化プロセスに注目する．しかしエンタープライズ・アーキテクチャと言ってもその構造は組織ごとに全く異なる．またその組織ごとにディペンダビリティケースを作成していては非常に手間がかかってしまう．そこで，エンタープライズ・アーキテクチャ・フレームワークの最適化プロセスのディペンダビリティケースを作成する．このディペンダビリティケースをテンプレートとして利用することで個々の組織の最適化プロセスのディペンダビリティケースを容易に作成できる．

そこで前回の報告ではエンタープライズ・アーキテクチャ・フレームワークとして代表的なTOGAF [6]を対象とし，TOGAFアーキテクチャ開発プロセスを構造化したディペンダビリティケースを作成する手法を提案した．本稿では，前回提案した手法での問題点を改善する規則を提案し，引き続きディペンダビリティの確認を容易化するテンプレートの提案を行う．またその改善手法で浮かび上がった新たな問題点についても議論する．

以下に本論文の構成を述べる．2章では，ディペンダビリティケースとその記述方法について説明する．3章では TOGAF と，そのアーキテクチャ開発手法である ADM と ADM 内の記述項目について説明する．4章では，ディペンダビリティケースを記述する上での問題点について説明する．5章では，提案手法として，上記の問題点を解決する規則を提案する．6章では，それぞれの規則を実際にフェーズ A とフェーズ B に対して適用し，ディペンダビリティケースを作成する．7章では実際に作成したディペンダビリティケースについて考察し，最後に今後の課題について述べる．

2. ディペンダビリティケースについて

アシュアランスケース [3] [4] は，主に欧米で普及しているシステムの安全性などを確認するために用いられる方法である．また主にディペンダビリティについて議論する場合は，ディペンダビリティケースとも呼ばれる．そこで本稿ではこれらを総称してディペンダビリティケースという用語を用いる．

ディペンダビリティケースの記法の一つに Goal Structuring Notation(GSN) [5] がある．これは Tim Kelly らによって提唱された要求を木構造に分解し，議論を容易にする表記方法である．議論すべきゴールをトップゴールとし，ゴール分解の理由などが記述されるストラテジに基づいてトップゴールを複数のサブゴールに分解する．またコンテキストとしてゴールを議論する際の条件などの情報が記述される，そして最下層のゴールにエビデンスを与えることでそのゴールを保証している．このように抽象的なトップゴールを分解し，各サブゴールを保証することで，トップゴールを保証することが可能である．

本稿では DEOS プロジェクト [1] の一環として開発された D-Case Editor [2] と呼ばれるディペンダビリティケース作成を支援するツールを用いて，ディペンダビリティケースを GSN で記述する．

3. TOGAF とは

TOGAF(The Open Group Architecture Framework) [6] とは，エンタープライズ・アーキテクチャの導入，作成，利用，維持を支援するための手法と支援ツールを提供するエンタープライズ・アーキテクチャを開発するためのフレームワークである．

TOGAF は，ベスト・プラクティスおよび既存アーキテクチャ資産の再利用可能なセットによって支えられた，反復型プロセス・モデルに基づいている．また TOGAF の重要な要素は手法であるということである．これはつまりアーキテクチャ開発手法 (ADM) によって経営ニーズに合致したエンタープライズ・アーキテクチャを策定することができるということである．

また TOGAF ではディペンダビリティは考慮されていないが，現在 TOGAF の次期版としてディペンダビリティ概念を包含する TOGAF NEXT が開発中である．

3.1 アーキテクチャ開発手法

TOGAF の中核には前述したアーキテクチャ開発手法 (ADM) が存在する．ADM はアーキテクチャ開発のための，検証済みの反復可能なプロセスを提供する．ADM は，アーキテクチャ・フレームワークの確立，アーキテクチャ・コンテンツの開発，トランジションの実行，アーキテクチャの実現のガバナンスを含んでいる．これらのアクティビティは全て，継続的なアーキテクチャの定義と実現の反復サイクルの中で実行される．これにより，ビジネス・ゴールと機会に応じて，コントロールされた方法で，組織がエンタープライズを変革することが可能となる．

ADM には以下の複数のフェーズが存在する．

- 初期フェーズ
- フェーズ A：アーキテクチャ・ビジョン
- フェーズ B：ビジネス・アーキテクチャ
- フェーズ C：情報システム・アーキテクチャ
- フェーズ D：テクノロジ・アーキテクチャ
- フェーズ E：機会とソリューション
- フェーズ F：移行プランニング
- フェーズ G：実践ガバナンス
- フェーズ H：アーキテクチャ変更管理
- 要件管理

3.2 アーキテクチャ開発手法の記述項目とその分析

ADM はプロセス全体にわたって，上記のフェーズの間，あるいはフェーズの中で反復される．ADM の各フェーズはそれぞれ目的，入力，ステップ，出力で構成される．そこで ADM の各フェーズにおける目的，入力，ステップ，出力に注目することで，各フェーズに対してディペンダビリティケースを作成する．本手法では，前回の手法を適用して問題点が残ったフェーズA：アーキテクチャ・ビジョンと以降のフェーズを対象とする．

4. 記述上の問題点

TOGAF ADM からディペンダビリティケースを作成するにあたって，いくつかの問題点がある．

4.1 関係矢の重なり

前回の提案手法では，アクティビティや目的など上位でのゴール分解が，ステップによる分解で混ざってしまうという問題点があった．このため上位で分解した意味がなくなってしまい，また作成したディペンダビリティケースの関係が重なりあうことで見にくくなり，見やすさ・分かりやすさという観点からも問題があると言える．

4.2 複合したエビデンス

前回の提案手法では，「承認されたアーキテクチャ策定作業計画書」をエビデンスとするステップが五つ存在した．そのため「承認されたアーキテクチャ策定作業計画書」という一つの文書に複数のステップに関連することが記述されていることが分かる．これをテンプレートととして実際の組織に適用した際に，この複合文書と実際の文書との対応が分かりにくいという問題があることが予想できる．

4.3 広すぎるエビデンスの意味

フェーズ A には，「コミュニケーション計画」というエビデンスが存在するが，この「コミュニケーション計画」とは何に対する誰とのコミュニケーション計画であるのか分かりにくく，言葉の意味が広くなりすぎてしまっている．

4.4 対応するステップがない目的の存在

フェーズAには,「並行している他のアーキテクチャ開発サイクルに対する影響と他のサイクルからくる影響の理解が妥当である」という目的によるゴールが存在するが，このゴールに対して特定の対応するステップがない状況となってしまう.

4.5 記述のない文書への対応

ディペンダビリティケースを作成する上で，エビデンスやコンテクストとして必要な文書が入出力には記述されていないことがある.

5. 提案手法

これまで提案した規則に加え，上記の問題点を解決できるように規則を修正または新たに提案する．それぞれの規則は以下のようになっている.

5.1 これまで提案した規則

5.1.1 トップゴールの設定

フェーズのトップゴールを設定する．フェーズAを例にすると，フェーズのトップゴールは「フェーズA：アーキテクチャ・ビジョンがディペンダブルである」である.

5.1.2 要素間の対応関係とゴール分解方法

TOGAF文書の記述からディペンダビリティケースを作成する．前述のようにADMの各ステップには目的，入力，ステップ，出力が存在する．またディペンダビリティケースには，ゴール，サブゴール，コンテクスト，エビデンスといった要素が存在する．それらの要素を表1のように対応させる．ゴールやコンテクストとなる文書記述を命題や状態に変換する．まずアクティビティでトップゴールを分解する．次にアクティビティに対応する目的でゴール分解し，目的に対応するステップでゴール分解を行う．入力をコンテクスト，出力をエビデンスとしてゴールに付加する．ディペンダビリティケースの構造は図1のようになる．対応関係は同じ単語が含まれる場合や，文書内の記述から判断する.

表1 TOGAF ADMとディペンダビリティケースの対応関係

活動要素	ディペンダビリティケース要素
アクティビティ	上位ゴール
目的	中位ゴール
ステップ	下位ゴール
入力	コンテクスト
出力	エビデンス

5.1.3 入力を付加するゴールの選択

原則として入力が対応するコンテクストは具体的なエビデンスとともにステップが対応する下位ゴールに付加する．しかし個別のステップではなくアクティビティやフェーズ全体にかかるような入力の場合は，コンテクストをステップより上位のゴールに付加する.

5.1.4 記述のない入出力の対応

フェーズ内部の入出力の存在など，フェーズの入出力の記述以外にステップに必要な入出力があると判断した場合に，文書の記述から適宜コンテクストやエビデンスを付け加える.

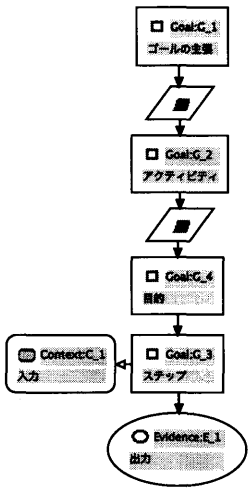

図1 ADM要素を利用したディペンダビリティケースの作成例

5.2 既存の規則を修正または新たに提案する規則

4.に記述した問題点とそれを解消する5.2の規則との対応は表2のようになっている．以下に規則の詳細を述べる.

表2 問題点と規則の対応関係

問題点	規則
4.1	5.2.1，5.2.2
4.2	5.2.3
4.3	5.2.4
4.4	5.2.1，5.2.5
4.5	5.2.5

5.2.1 目的をコンテクストに変換

問題点4.1と4.4を解決するための方法として規則5.1.2を修正する．表3のように目的をコンテクストとして扱い，上位ゴールをアクティビティ，下位ゴールをステップとして対応させ，図2のようにディペンダビリティケースを作成する.

表3 提案手法における対応関係

活動要素	ディペンダビリティケース要素
アクティビティ	上位ゴール
目的	コンテクスト（ディペンダビリティ要求）
ステップ	下位ゴール
入力	コンテクスト（必要な入力情報）
出力	エビデンス

5.2.2 アクティビティや目的をand分解

問題点4.1を解決するための方法として新たに規則を作成する．アクティビティと目的の文章がandで接続されているような場合，その文章を複数のゴールに分解し，ディペンダビリティケースを作成する．フェーズAの例では,「ビジネス要件および制約要件を定義する」という目的を「ビジネス要件を定義する」と「制約要件を定義する」の二つのゴールに分解する.

5.2.3 複合エビデンスの分解

問題点4.2を解決するための方法として新たに規則を作成する．複合した文書を一つのエビデンスとするのではなく，その

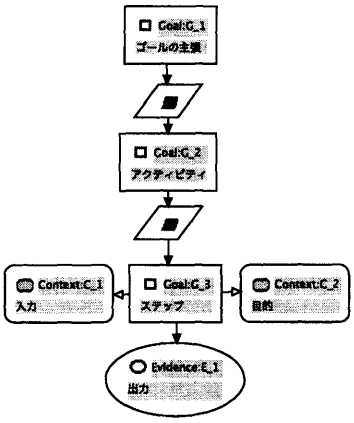

図 2　提案手法のディペンダビリティケース作成例

複合した文書内の関連するサブセクションごとにエビデンスを分割し，それぞれのステップのエビデンスとして対応付ける．フェーズ A の例では，「承認されたアーキテクチャ策定作業計画書」というエビデンスを文書内記述より関連する六つのサブセクションで分解し，六つのエビデンスとする．

5.2.4　エビデンスの具体化

問題点 4.3 を解決するための方法として新たに規則を作成する．対応する文書のステップの記述からエビデンスの内容を具体化する．例えばフェーズAの「コミュニケーション計画」の場合，対応するステップ内の記述より，「ステークホルダとのエンタープライズ・アーキテクチャ開発の進捗についてのコミュニケーション計画」であることが分かるため，このように具体化したものをエビデンスとする．

5.2.5　対応する記述が入出力にない項目を undeveloped に

規則 5.1.4 では入出力に対応する記述がない場合，ステップの内容から判断して適宜コンテクストやエビデンスを付け加えていた．しかし TOGAF NEXT に対するテンプレートの提案であることを考えると，必要な記述がないと判断した場合は undeveloped とすることで TOGAF 文書の構造と対応させ，規則 5.1.4 を修正する．

6.　フェーズへの適用結果

フェーズ A に規則 5.2 を適用する．4.1 を解決する方法として規則 5.2.1 と 5.2.2 の二つを提案しているため，それぞれを適用した場合と同時に適用した場合の三つのディペンダビリティケースを作成する．また三つの場合それぞれに対して規則 5.2.3，5.2.4，5.2.5 を適用する．

目的をコンテクストとして扱ったディペンダビリティケースは図 3，アクティビティや目的が and で接続されている場合，その文章を複数のゴールに分解したディペンダビリティケースは図 4，両方の規則を同時に適用したディペンダビリティケースは図 5 のようになる．

図より複合したエビデンスをサブセクションごとに分解することで，ステップが文書のどの部分をエビデンスとしているのかが明確になった．同様にエビデンスを具体化することで，文書内のステップ記述を読まなくても，どのようなエビデンスであるのかが分かりやすくなったと言える．対応する記述が入出力にない項目を undeveloped にすることにより TOGAF 文書と対応したディペンダビリティケース構造となり，文書のどの部分を改善する必要があるかが明確になったと言える．

次に，図より全ての適用方法で問題点であった関係矢の重なりがなくなっていることが分かる．まず目的をコンテクストとして扱う方法は，アクティビティやステップといった行動をゴールとし，目的はゴールを議論する前提を記述したディペンダビリティ要求であるコンテクストとして扱うことでディペンダビリティケースが作成しやすくなった．対応するステップがない目的も上位ゴールのコンテクストとして付加することで問題点を解消することができた．しかし C_13 のようにコンテクストが別のアクティビティ間で同様にコンテクストとなる部分が存在した．また目的内に複数のステップに対応する記述がされていたため上位のゴールであるアクティビティにコンテクストとして付加した．そのためどのステップに対応するコンテクストなのか明確ではなくなってしまった．

次にアクティビティと目的を and 分解する方法はきれいな階層構造となり見やすい構造となっていることが分かる．アクティビティごとの分解以下で，目的やステップが重複して他のアクティビティに対応することも発生しなかった．しかし G_21「並行している他のアーキテクチャ開発サイクルに対する影響と他のサイクルからくる影響の理解が妥当である」という目的に対して特定の対応するステップがない状況は改善することができなかった．また分割することで G_35「アーキテクチャ作業の優先度付けが妥当である」という新たな対応するステップがないゴールが発生してしまった．

最後にアクティビティと目的を and 分解し目的をコンテクストとして扱う方法では，目的をコンテクストとして扱う方法で問題となった，複数のステップに対応するため上位のゴールであるアクティビティに追加する問題を，目的を分解することで対応するステップが単一とすることができた．原則として目的をコンテクストとしてステップに対応付けることで，どのステップに対応するコンテクストか明確でないという問題点を解決することができた．

アクティビティと目的を and 分解し目的をコンテクストとして扱う方法をフェーズ B：ビジネス・アーキテクチャにも適用した．適用したディペンダビリティケースは図 6 となる．

7.　考　　察

本稿の提案規則を適用することで，前回の問題点を解決したフェーズ A とフェーズ B のディペンダビリティケースを作成することが可能であった．また問題点 4.1 を解決する方法として規則 5.2.1 と 5.2.2 の二つを同時に適用する最後の方法が最もよいディペンダビリティケースを作成することができたと言える．課題としては以下が挙げられる．

7.1　目的に対応するステップが存在しない

提案手法をフェーズ A に適用すると次の二つのコンテクスト C_12「アーキテクチャ作業の優先度付けを行っていること」，C_21「並行している他のアーキテクチャ開発サイクルに対す

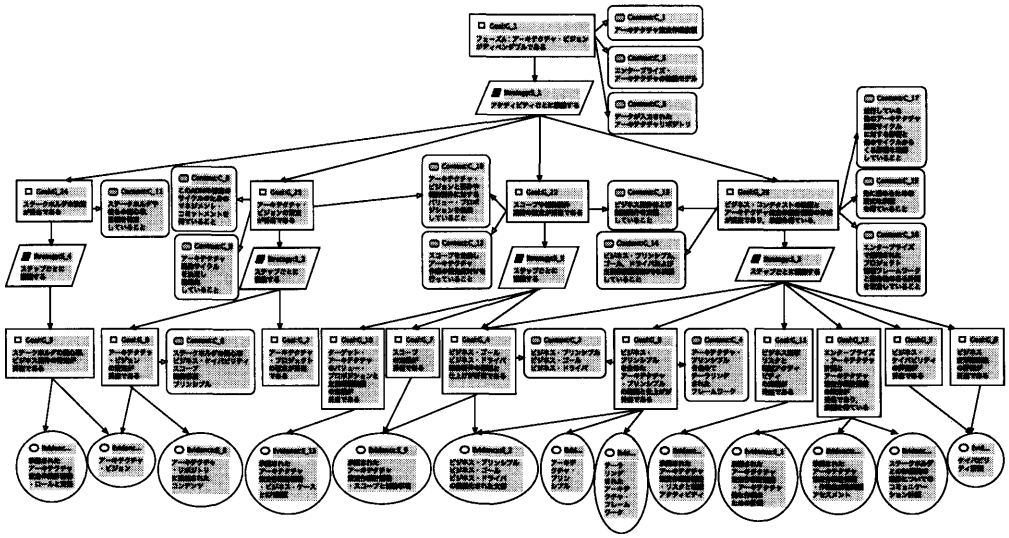

図 3　フェーズ A の目的をコンテクストとしたディペンダビリティケース図

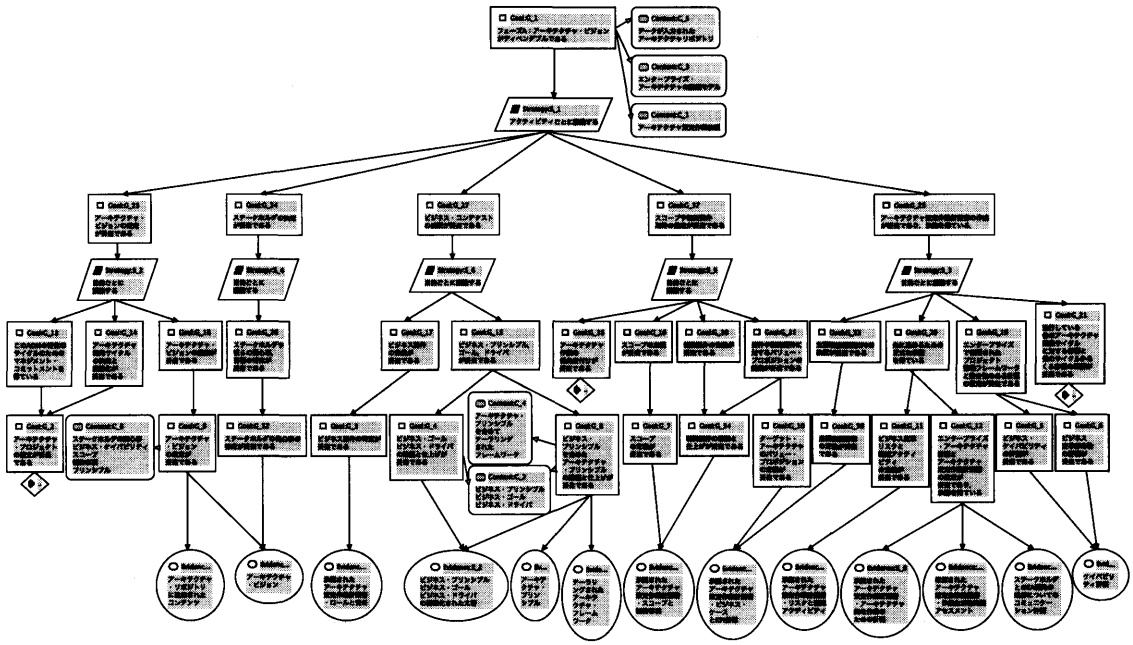

図 4　フェーズ A の and 分割したディペンダビリティケース図

る影響と他のサイクルからくる影響を理解していること」に明確に対応するステップを見つけることができなかった．C_12 は and 分解することで生成された目的の一つであり，分解することで明確に対応するステップがないことを発見した．そのため TOGAF 文書に対応するステップが必要であると言え，TOGAF NEXT では文書の記述を改善する必要があると言える．また C_21 も対応するステップが存在しないと判断したが，これは特定のステップで満たされる目的ではないと判断し，上位のゴールにあたるアクティビティのコンテクストに対応させるのが妥当であるとした．

7.2　コンテクストとなる目的が少ない

目的をコンテクストとしてステップで満たすべきものとして付加することで，ステップの妥当性を確認する上で有用である．しかしフェーズ B では TOGAF 文書に記述される目的が少なく，コンテクストがないステップが多く存在し，ステップごとにどのような条件を満たすべきか分かりにくくなってしまった．そのため TOGAF NEXT ではこのフェーズの目的の記述を増やすべきであると言える．

7.3　具体的なエビデンスがないステップの存在する

フェーズ A でのステップ「アーキテクチャ・プロジェクトの確立が妥当である」やフェーズ B のステップ「アーキテクチャ・ランドスケープにわたる影響の解決が妥当である」では，出力に直接関係のある文書が存在せず，またステップに具体的なエビデンスとなる記述が見られなかった．そのため，ステップ内または出力に具体的なエビデンスを記述する必要があると言える．

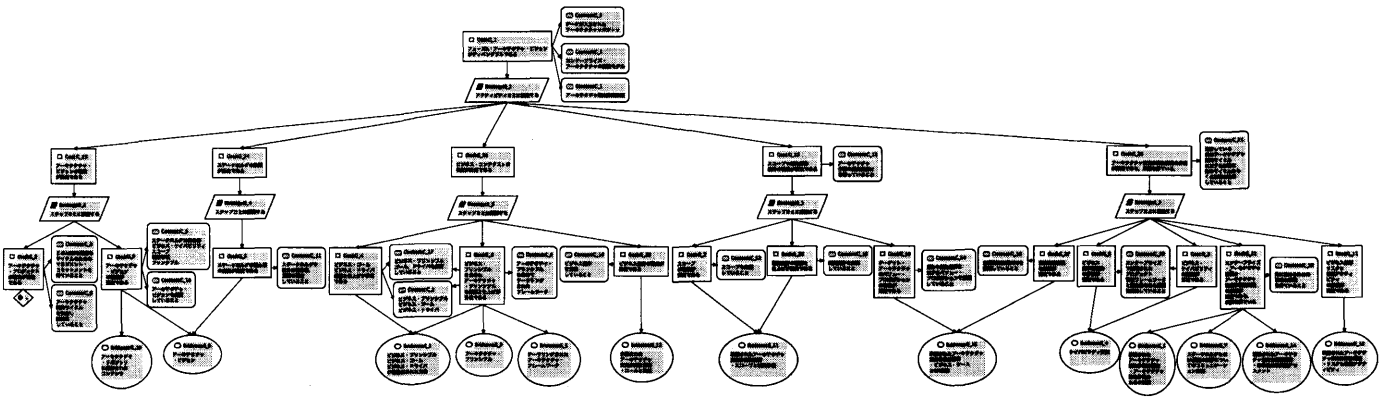

図 5　フェーズ A の目的をコンテクストとしてステップに対応付けして，and 要素を分割したディペンダビリティケース図

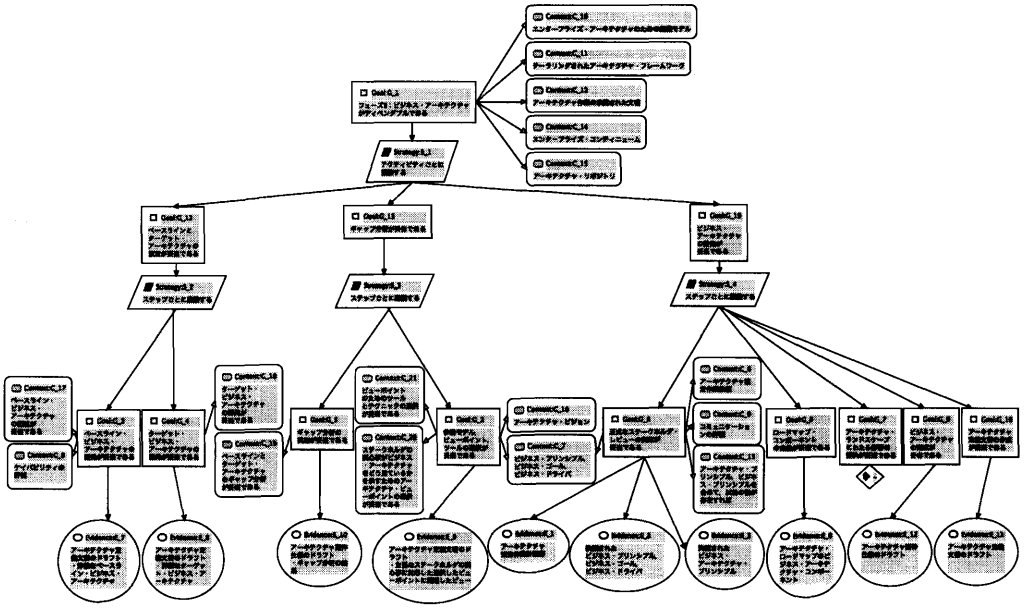

図 6　フェーズ B：ビジネス・アーキテクチャのディペンダビリティケース例

8. おわりに

本稿では，問題点を解消するために新たな規則や修正した規則を実際に適用し，新たな TOGAF ADM のディペンダビリティケース作成法を提案した．また実際にフェーズ A：アーキテクチャ・ビジョンとフェーズ B：ビジネス・アーキテクチャに対して適用し，その適用結果に基づき，以下の課題を明らかにした．

- 目的に対応するステップが存在しない
- コンテクストとなる目的が少ない
- 具体的なエビデンスがないステップが存在する

また今後 ADM の残りの 7 フェーズにも同様に本手法を適用することにより，実際に TOGAF NEXT に対する ADM プロセステンプレートを提案していく予定である．

謝　辞

本研究は CREST「実用化を目指した組み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．

文　献

[1] DEOS プロジェクト http://www.crest-os.jst.go.jp
[2] D-Case Editor http://www.dependable-os.net/tech/D-CaseEditor/
[3] Peter Bishop, Robin Bloomfield, Sofia Guerra, The future of goal-based assurance cases, DSN, 2004
[4] T. Scott Ankrum, Alfred H. Kromholtz, Structured Assurance Cases: Three Common Standards, IEEE International Symposium on High — Assurance Systems Engineering, 2005
[5] Tim Kelly and Rob Weaver. The goal structuring notation - a safety argument notation. In Proc. of the Dependable Systems and Networks 2004, Workshop on Assurance Cases, 2004.
[6] TOGAF Version 9 日本語訳版 http://www.opengroup.or.jp/togaf.html
[7] 山本修一郎，要求工学基礎知識，名古屋大学情報連携統括本部情報戦略室，2012