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
KBSE2010-55 (2011-3)

情報連携アーキテクチャ分析技法

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　複数のシステム間における情報連携を容易化することは，個別システムの機能を越えた非機能要求である．本稿では，非機能要求シナリオに基づくアーキテクチャ分析手法を情報連携性に適用することにより，情報連携アーキテクチャを分析できることを明らかにする．

キーワード　情報システム連携，情報連携アーキテクチャ，品質特性シナリオ，非機能要求

A Method for Analyzing Information Integration Architecture

Shuichiro Yamamoto

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

Abstract　*Non Functional Requirements is used to describe the easiness of integration between different information systems. In this paper, an architecture analysis method based on non functional requirements is applied to integrate information systems. We also discuss the effectiveness and future issues of the proposed method.*

Keyword　Information system integration, Information Integration architecture, quality characteristic scenarios, Non functional requirements

1 はじめに

組織に対して，これまでに多様な情報システムが提供されているが，必ずしも情報が連携されていないという問題があった．現状では，研究活動を可視化するために必要な情報サービスが部分的に個別の情報システムとして構築されているため，それぞれの情報システムが変更されたり，新たに追加されるたびにその都度，連携手順を再構成する必要があったり，組織の業務活動情報を結合するための作業が特定の組織の中で人手による運用で実行されていて組織全体では利用できていないなどの問題があった．

この問題の原因は，組織における業務活動情報が異なる情報システムの中で断片的に管理されており，本来は，構成員の業務活動として一貫性があるにも関わらず，それぞれの情報システムに一度蓄積されてしまった情報を統合的に再結合できないことにある．

このため，持続的な情報連携サービスを構築するための方法論の研究を進めている[1]．これまでにアクター層（人間による情報連携），コミュニケーション層（C型情報システムによる情報連携），オペレーション層（O型情報システムによる情報連携）からなる3階層情報連携アーキテクチャを提案し情報連携条件を具体化した[2]．

本稿では，複数のシステム間における情報連携を容易化することが，個別システムの機能を越えた非機能

要求であることに着目して，非機能要求シナリオに基づくアーキテクチャ分析手法を提案する．次いでこの手法を情報連携性に適用することにより，情報連携アーキテクチャを分析できることを明らかにする．

2 非機能要求とシナリオ

非機能要求には，次の３つのレベルがある．

①機能レベル　特定の機能に対する特性

②システムレベル　特定のシステム全体で満たす必要のある特性

③複合システムレベル　複数のシステムが満たすべき特性

機能レベルの非機能要求の例としては，計算効率やメモリ効率がある．システムレベルの非機能要求の例には，可用性や柔軟性，正確性などがある．複合システムレベルの非機能要求には相互連携性,情報連携性,全体最適性などがある．

システムレベルや複合システムレベルの非機能要求を満たすために，個別システムの範囲を越えて必要となるソフトウェア構成要素がある．このような構成要素の機能は，非機能要求を前提としていることになる．逆に言えば，非機能要求からアーキテクチャの構成要素とその特性としての機能を抽出することになる．

システムに対する非機能要求を達成するためには，シナリオと，シナリオを実現するためのアーキテクチャ上の判断が必要となる．ここで，どのようなアーキテクチャの構成要素を用意すれば非機能要求を満たすことができるかという意思決定をアーキテクチャ上の判断という．

またシステムレベルの非機能要求はシステムの運用に対して確認する必要があるから，運用シナリオから，アーキテクチャ上の決定を導くことができる．

たとえば，システムの可用性に対する運用シナリオでは，①可用性資源の特定，②可用性資源に対する障害検出,③障害の修復,④再開始が必要になるだろう．この場合，①～④のシナリオに従ったコンポーネントを追加することが考えられる．たとえば可用性資源の障害を監視するための構成要素をアーキテクチャに追加できる．このときの課題は，ある非機能要求を達成するために必要となるこのようなコンポーネントの追加に際して，他の非機能要求に矛盾する可能性があることである．

もし非機能要求ごとに構成要素を追加しても対立しない前提であれば，非機能要求ごとに独立にアーキテクチャ上の判断を分類しておき，選択することができる．しかし，非機能要求のある関心事が，複数の非機能要求に共通することもあるだろうから，関心事が重複する非機能要求間で対立が発生するかもしれない．この場合については対立を解決するように構成要素を追加するか，非機能要求間の対立を緩和するような追加判断が求められる．

非機能要求を達成するためのシナリオを非機能要求シナリオと呼ぶ

非機能要求シナリオでは，非機能要求に対して①発生源（主体），②刺激，③環境，④シナリオ内容，⑤システム，⑥成果物，⑦応答，⑧評価基準を記述する．

ここで，品質特性シナリオ[3]では④⑤は明示されていないことを注意しておく.非機能要求シナリオとして，④⑤を追加した理由は，構成要素として，非機能要求シナリオ自身の内容としての手順が必要であることと，非機能要求が対象とするシステムを要素とする必要があるからである．また品質特性として，可用性，変更容易性，性能，セキュリティ，テスト容易性，操作性が挙げられている[3]ことからも分かるように，品質特性シナリオも非機能要求シナリオと考えることができる．

可用性に対する非機能要求シナリオを記述すると次のようになる．

[非機能要求]　可用性
[主体]　システムの外部環境
[対象]　システム
[事前状況]　通常状態
[契機]　想定外のメッセージを受信
[シナリオ]
1) 受信した想定外メッセージから対象となる可用性資源を特定
2) 可用性資源の復旧通知
3) 可用性資源の動作を継続
[入力]　想定外のメッセージ
[出力]　可用性資源の運用続行通知
[応答]　オペレータへ運用続行を通知
[事後状況]　オペレータが運用続行メッセージを受理
[関係者]　オペレータ
[役割分担]　想定外メッセージ受信時にオペレータが対応する．
[品質測定基準]　想定外メッセージ条件では，可用性を保証する資源についてどのような想定外メッセージが発生する条件を定義する．

通知メッセージ条件では，通知すべきメッセージの条件を定義する．

システム停止時間条件では，システム停止時間の許容限界を定義する．

3 シナリオに基づくアーキテクチャ分析

ATAM では，品質特性に基づいてアーキテクチャの特性を分析することができる[3]. このように ATAM では品質特性という用語を使用しているが，前述したように，内容的に同等だと考えられるので，以下では，品質特性ではなく非機能要求を用いて説明する．

非機能要求に基づいてアーキテクチャを分析する手法には，非機能要求ごとにアーキテクチャを分析する方法と，複数の非機能要求を考慮してアーキテクチャを分析する方法の２つがある．

非機能要求シナリオを用いる場合，非機能要求シナリオの構成要素ごとにアーキテクチャ上の判断を定義する．たとえば，可用性シナリオには，可用性資源の特定，復旧通知，可用性資源の運用継続がある．それぞれに対して,可用性資源の監視,可用性資源の復旧,可用性資源の動作再開などへ対処するためのアーキテクチャの構成要素を追加する必要がある．

複数の非機能要求を考慮してアーキテクチャを設計する方法では，非機能要求間の対立を考慮する必要がある．非機能要求間で対立が生じる例として，柔軟性と性能の対立がある．この場合，これらの非機能要求のいずれか，あるいは双方を具体化して両立できるように非機能要求を緩和する必要がある．また，このような非機能要求はシナリオに関係する主体ごとの関心事が異なることから生じるので，逆に，非機能要求シナリオの主体から非機能要求を抽出できる．ここで，非機能要求シナリオXの主体Aから導かれる非機能要求Nは，非機能要求シナリオXが達成しようとする非機能要求Qの下位となる非機能要求であることに注意する必要がある．

このような非機能要求の対立に基づくアーキテクチャ分析の手順は，次のようになる．

[分析法]非機能要求の対立に基づくアーキテクチャ分析法
[手順 1] 非機能要求シナリオを作成する
[手順 2] 非機能要求シナリオに基づいて必要なコンポーネントを追加
[手順 3] 非機能要求シナリオに関係する主体の関心事に基づいて，非機能要求を抽出
[手順 4] 対立する非機能要求を探索する
[手順 5] 対立がなければ，終了
　　対立があれば，手順５へ．
[手順 6] コンポーネントの追加判断により対立を解消できる場合，コンポーネントを追加して非機能要求の対立を解消する．
もしコンポーネントの追加により対立を解消できない場合，非機能要求を具体化して，手順３へ．
（手順終わり）

この方法に基づいて，非機能要求に基づくアーキテクチャを設計する例を考えよう．

いま，開発担当がシステムに期待する非機能要求は柔軟性で，事業担当がシステムに期待する非機能要求は性能であるとしよう．そこで，開発担当の期待に応えるには，柔軟性を達成するために，「ハードとソフトの分離」というアーキテクチャ判断をして，ハード制御コンポーネントの追加が必要だとしよう．そうすると，このコンポーネント追加はシステムの柔軟性の向上には役立つがシステム性能には寄与しないので対立することになる．そこで性能条件を具体化して，許容性能を見積ることで，許容性能を満足するようにハード制御コンポーネントを実現することにすれば，柔軟性と性能を対立することなく達成できるアーキテクチャを設計できることになる．

4 情報連携性シナリオに基づくアーキテクチャ分析技法

異なるシステム間で情報が連携できるときこれらのシステムの集合は情報連携性を持つという．情報連携性はシステム集合についての非機能要求であると考えられる．

このとき情報連携性シナリオには，図１に示すように，①連携情報抽出，②連携情報管理，③連携情報提示がある．

前述した非機能要求シナリオに従って情報連携性シナリオを作成すると次のようになる(分析法[手順1])．

①情報連携の発生源となる主体は連携先システムである．

②情報連携システムが活動する契機は，連携先システムからの情報提示要求である．

③情報連携契機が発生する条件となる状況は，連携元システムと情報連携システムが存在することである．

④情報連携活動への入力は，連携先システムの認証情報，連携元システムと必要な連携情報項目である．

⑤刺激を受けて実行される情報連携活動は，連携情報抽出，連携情報管理，連携情報提示である．

⑥情報連携活動の対象となるシステムは，連携元システムである．

⑦情報連携活動によって作成される成果物は提示される連携情報である．

⑧システム出力と応答が満たすべき条件は，要求に応じた連携情報が提示されることである．

⑨活動結果としてシステムが生成する通知内容は，連携情報の提示記録と，連携情報が提示できない場合への対応依頼である．

⑩システムからの応答を受理する関係者は，連携情報の運用者である．

⑪生成された応答の妥当性に対する評価基準は，連携情報の提示についての性能や適合性などについての条件である．

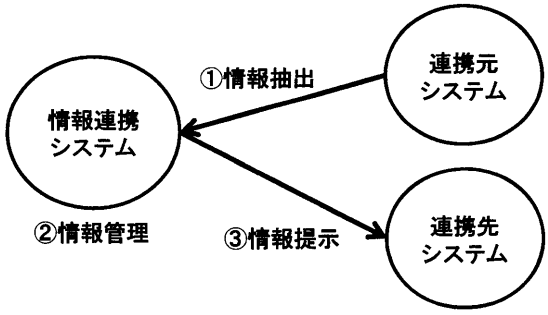

図１　情報連携性シナリオの構成

次に，この情報連携性シナリオ⑤で明らかになった連携情報抽出，連携情報管理，連携情報提示に対してそれぞれ連携情報抽出コンポーネント，連携情報管理コンポーネント，連携情報提示コンポーネントを抽出する．このとき，情報連携の関係者ごとに下位の非機能要求を抽出することができる．ここで情報連携性についての関係者は，連携元システムの管理者，情報連携システムの管理者，連携先システムの管理者と利用者である．

4.1 連携情報抽出

まず連携情報抽出の関係について，連携元システムの管理者と情報連携システムの管理者を考える．連携元システムの管理者は，連携元システムを用いた業務の運用性が連携情報の抽出によって低下しないことを要求することになる．これに対して情報連携システム

の管理者は連携情報の抽出容易性を要求することになる．このとき，運用性と抽出容易性の対立点として，連携情報抽出によって運用性が損なわれること，逆に，運用性を維持することで必要な連携情報が抽出できないことが顕在化する．そこで，運用性を具体化して許容可能な運用条件を明確に定義できれば，運用性を犠牲にすることなく，連携情報を抽出できる．これにより，運用性と抽出容易性の対立点を解消できる．この議論の結果をまとめると，図2に示すようになる．

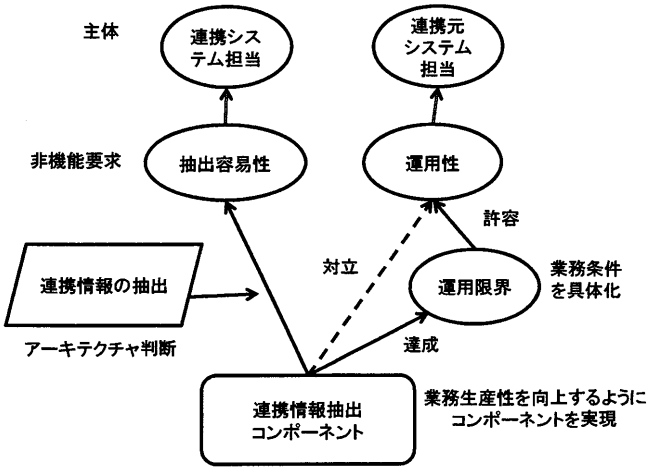

図2　連携情報抽出コンポーネント

また，この結果に基づいて連携情報抽出シナリオを整理すると，表1に示すようになる．

表1　連携情報抽出シナリオ

構成要素	説明
発生源	情報連携管理システム
刺激	抽出依頼
事前状況	連携元システムのデータが更新されている
入力	抽出条件指定
シナリオ	連携情報の抽出
システム	連携元システム
出力	連携情報
事後状況	連携情報が抽出され管理されている
応答	抽出記録．抽出できない場合，連携管理担当に連絡
応答先	情報連携担当
評価基準	抽出性能の許容限界

4.2 連携情報管理

次に連携情報管理の関係者として，連携元システムの管理者と情報連携システムの管理者，連携先システムの管理者を考える．連携元システムの管理者は，抽出された連携情報の安全性を要求するはずである．これに対して連携システムの管理者は，連携情報の管理が容易であることを要求することになる．また連携先システムの管理者は連携情報を柔軟に閲覧するための開放性を要求することになる．このとき，連携情報の安全性が，管理容易性ならびに開放性と対立することになる．そこで，安全性を具体化して，管理容易性と開放性に対する許容可能な条件を満たしながら連携情報へのアクセスを制限することを考える．たとえば連携先システムに関係する情報だけを開放し，関係しない情報については提示しないようにすることで，連携先システムの開放性要求と対立しない安全な情報管理ができる．またアクセス権限管理コンポーネントを用意することで連携情報管理を軽減できる．したがって，これらの条件を満足するように，アクセス権限管理コンポーネントを追加することにより，運用性と抽出容易性の対立点を解消できることが分かる．この議論の結果をまとめると，図3に示すようになる．

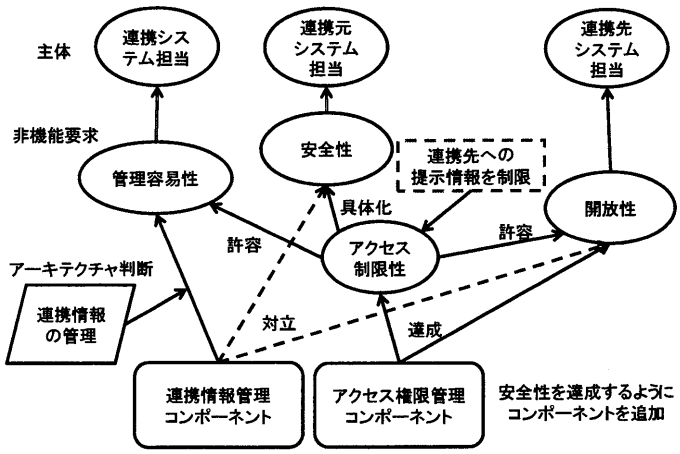

図3　連携情報管理コンポーネント

この結果に基づいて連携情報管理シナリオを整理すると，表2に示すようになる．

表2　連携情報管理シナリオ

構成要素	説明
発生源	連携元システム
刺激	連携情報の抽出
事前状況	連携情報の抽出先があること
入力	抽出された連携情報
シナリオ	連携情報の管理
システム	連携情報管理システム
出力	連携情報
事後状況	抽出された連携情報が管理されている
応答	連携情報の管理記録．連携情報が管理できない場合，管理担当に連絡
応答先	連携情報の管理担当
評価基準	連携情報管理の安全性，管理容易性，開放性の許容限界

4.3 連携情報提示

最後に連携情報提示の関係者として，情報連携システムの管理者，連携先システムの管理者を考える．すでに考察したように，連携先システムの管理者は，管理されている連携情報の開放性を要求する．これに対して連携システムの管理者は，連携情報のアクセス権限を管理する必要がある．そこで，連携情報提示コンポーネントの追加では，連携情報管理の場合と同じように，開放性とアクセス制限性が対立する．

そこで連携先システムが適切なアクセス権限を持つことを判断するために，予め認証された連携先が許された範囲で連携情報を閲覧できるように，連携先シ

ステムの認証条件を具体化して開放性を制約する．これによって開放性とアクセス制限性の対立を解消する．この議論の結果をまとめると，図4に示すようになる．

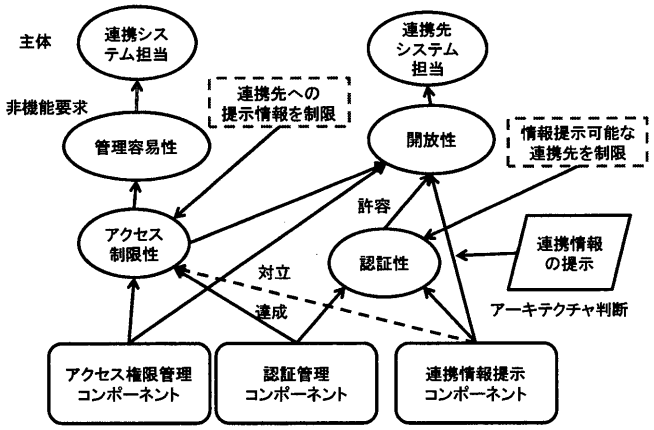

図4　連携情報提示コンポーネント

この結果に基づいて連携情報提示シナリオを整理すると，表3に示すようになる．

表3　連携情報提示シナリオ

構成要素	説明
発生源	連携先システム
刺激	連携情報の提示要求
事前状況	連携先システムが認証可能でアクセス権限があること．提示可能な連携情報が管理されていること
入力	連携先システムの認証情報，提示対象連携情報の指定
シナリオ	連携先の認証，アクセス権限確認，連携情報の提示
システム	連携情報管理システム
出力	提示対象の連携情報
事後状況	指定された連携情報を閲覧可能
応答	閲覧記録．閲覧できない場合，連携管理者に連絡
応答先	連携管理担当
評価基準	連携情報提示についての認証性，アクセス制限性，開放性の許容限界

5　考察

5.1　アーキテクチャ分析技法の有効性

前述した結果から，非機能要求シナリオに基づいて情報連携アーキテクチャを分析できることが明らかになった．また，本技法では，対立する非機能要求を許容可能な非機能要求によって具体化することにより，対立を解消できる方法の有効性を，情報連携アーキテクチャの分析における非機能要求の対立の解消について，確認した．

5.2　動的分析

本提案では，情報連携アーキテクチャの静的な側面に対する分析例を紹介した．情報連携アーキテクチャには動的な特性についても分析する必要がある．情報連携シナリオは動的な特性を分析する上でも有用な情報を記述しているので，本提案手法をアーキテクチャの動的分析に応用できると考えている．この点については，今後の課題である．

5.3　適用性

本手法を，大学における研究費管理システムとの情報連携アーキテクチャに対して，情報連携性シナリオを用いて分析することができる．

この例では予算可視化システムと経理システムとを情報連携しようとしている．このとき関係者として利用者と業務担当，情報連携担当を想定する．この関係者ごとに情報連携性シナリオを作成することで情報連携アーキテクチャを分析できることは明らかである．

このように本手法は多様な情報連携システムに適用できるので，後述するアーキテクチャパターンとして整理できる可能性がある．

また，本稿では，非機能要求のうち情報連携性に対して，非機能要求シナリオに基づくアーキテクチャ分析技法を適用した．今後，情報連携性以外の非機能要求についても，本技法の適用性を評価する必要がある．

5.4　代替案の分析

連携情報アーキテクチャの代替案の分析を考えると，表4のようなアーキテクチャの比較表が用いられることが多い．本提案では，非機能要求の具体化の過程でこれらの代替案が段階的に選択していると考えられる．表形式による代替案の比較法は一覧性が高いという特徴があるので，本提案の結果や途中経過を比較表の形で提示することができれば有効であると思われる．したがって，このような比較表とアーキテクチャ分析図の相互変換法についても今後検討する必要がある．

表4　連携情報管理アーキテクチャの比較分析例

関係者	非機能要求	(案1)一括抽出により連携情報を連携元システムから抽出	(案2)必要な連携情報ごとに抽出する
連携管理者	抽出容易性	○)処理が容易	×)処理件数が増加するだけでなく処理が煩雑
連携元管理者	運用性	×)不必要な情報を抽出すると，運用効率が低下する △)許容性能の上限を設定	○)不必要な情報を抽出しなければ運用効率が低下しない
総合判断		○	×

5.5　アーキテクチャ分析プロセス

本稿では，情報連携性シナリオを構成要素ごとに分解して独立にアーキテクチャを分析する手法を示した．情報連携システムの全体アーキテクチャを構成するためには，このようにして分解されたアーキテクチャを再結合する必要がある．このとき分解時に具体化された非機能要求が，再結合によって新たな対立を生じる可能性があるかもしれない．再結合による非機能要求の対立が発生する条件の明確化とともに，このような対立の解消手法についても今度検討する必要がある．

6　関連研究

6.1　ATAM

ソフトウェアアーキテクチャを評価する技法としてATAM (Architecture Tradeoff Analysis Method) が多くの業界で採用されている[3].

ATAMでは，品質特性に対するシナリオを用いてア

ーキテクチャがどのように反応するかを質問に従って分析することができる．ATAM の実施では，評価チーム，意思決定権者，関係者の参加が求められる．評価チームは品質特性要求に対してシナリオを対応付け，重要性と実現性を三段階で評価する．次に，この評価結果に基づいて優先度の高いシナリオについてアーキテクチャ上の決定事項との関連，トレードオフ，リスクを判別する．ATAM により，アーキテクチャが持つリスク領域を特定することができる．

6.2 アーキテクチャパターン

ソフトウェア開発過程で具体化されたソフトウェアアーキテクチャは，特定の問題に対する解決策であるため，そのままでは他のソフトウェア開発に適用できない．このため，アーキテクチャパターンでは，①解決したい設計上の問題，②アーキテクチャパターンが適用される典型的な条件などのコンテキスト，③設計上の問題に対するアーキテクチャパターンによる解決方法の静的側面と動的側面，④アーキテクチャパターンの特性としての利点と不利な点，⑤アーキテクチャパターンを活用してアーキテクチャを設計実装するときの留意点や適用事例を記述する[4].

アーキテクチャパターンにより,問題のクラスとそれに適用できるソフトウェアアーキテクチャのクラスを対応付けることで，アーキテクチャ上の設計方針と解決策を再利用が期待できる．

6.3 アーキテクチャケース

アーキテクチャ記述言語をなぜ，どのように適用したのか，その結果はどうであったのかを記録，活用するための枠組みとしてアーキテクチャケースが提案されている[5]．アーキテクチャケースでは，①アーキテクチャ記述言語を採用するに至った開発・技術の情勢変化 ，②アーキテクチャ記述言語が対象とした IT システム変容への期待，③アーキテクチャ記述言語が達成しようとしたシステム高信頼化への貢献と解決した技術課題，④構成要素，構成要素間関係，評価特性，⑤アーキテクチャ記述プロセス，⑥アーキテクチャ記述を用いた開発組織，適用組織，⑦アーキテクチャを記述,適用,教育するための具体的取り組みを記述する．このようにアーキテクチャケースでは開発管理面，開発要員スキル面，開発プロセス面，アーキテクチャ記述言語の選択面で具体的な判断結果や，選択したアーキテクチャ記述言語によってどのようにシステムアーキテクチャを記述し，どのような課題がありそれをどのように解決したのかを記述できる.アーキテクチャ・ケースが流通すれば，実践的で具体的なアーキテクチャ記述言語の事例が蓄積活用できるので,アーキテクチャ記述言語の普及を加速できる可能性がある．

6.4 運用要求記述

運用活動を定義する運用要求定義票[6]と非機能要求シナリオの構成要素を比較すると，①非機能要求シナリオの構成要素の評価基準が運用要求定義票にはないこと，②非機能要求シナリオでは，明示的に記述できない運用要求定義票の項目として，規則と役割分担がある．評価基準と規則を対応付けることができると考えられる．役割分担についても非機能要求シナリオに追加することができる．このようにすれば，運用要求定義票と非機能要求シナリオを同様に用いることができる．このことの効果は，以下のようである．アーキテクチャも人工物であることから，想定している事前状況や事後状況に限界がある．したがって想定外のインシデント事象に対しては，アーキテクチャだけでは対応できないことから，アーキテクチャに基づくシステムを運用する体制組織で，インシデント事象に対応するプロセスを明確にする必要がある．このように本手法ではアーキテクチャを分析設計するだけでなく，アーキテクチャを扱う組織体制を分析設計できるという特徴がある．

6.5 SysML

図 2，図 3，図 4 に示したように，今回は直感的な表記法を用いるこよにより，主体，非機能要求，コンポーネントからなる情報連携アーキテクチャを分析した．これらの要素は，SysML[7]でも記述できると思われる．今後，本稿で提案した手法を SysML による情報連携アーキテクチャ分析技法として洗練していく予定である．

7 おわりに

本稿では持続的情報連携サービス分析方法論の構築に向けた研究のロードマップ[1]に基づいて情報連携アーキテクチャ分析技法を提案し有効性を確認した．ロードマップで提示した情報連携サービス分析方法論は，①3 階層情報連携アーキテクチャ[2]，②情報連携コミュニティ分析手法[8]，③情報連携アーキテクチャ分析技法，④情報連携構造分析技法，⑤情報連携メトリクスから構成される．

今後は本稿で提案した情報連携アーキテクチャの課題を解決するとともに，④⑤の研究を進めていく予定である．

参考文献

[1] 山本修一郎，持続的情報連携サービス分析方法論の研究課題，知能ソフトウェア工学研究会，20010,11.24

[2] 山本修一郎，3階層情報連携アーキテクチャの提案，知能ソフトウェア工学研究会，20011,1.24

[3] Len Bass, Paul Clements, Rick Kazman, Software Architecture in Practice,前田他訳，実践ソフトウェアアーキテクチャ，日刊工業新聞社，2005

[4] 中谷多哉子，青山幹雄，佐藤啓太編，ソフトウェアパターン，共立出版，１９９９

[5] 独立行政法人 情報処理推進機構. ソフトウェア・エンジニアリング・センター，高信頼性システム 開発技術の動向，sec.ipa.go.jp/reports/20100331c/20100331c_2.pdf

[6] 山本修一郎、システム運用知識抽出法の提案、知識流通ネットワーク研究会, 2010 .

[7] OMG SysML Specification, www.omgsysml.org/

[8] 山本修一郎，情報連携コミュニティ分析技法，第8回知識流通ネットワーク研究会，http://www4.atpages.jp/sigksn/conf08/index.html