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

信学技報
TECHNICAL REPORT OF IEICE.
NS2003-316, IN2003-271(2004-03)

Webサービスアプリケーションのソフトウェアメトリクスに関する考察

串戸 洋平 石井 健一 山内 寛己 井垣 宏 玉田 春昭 中村 匡秀 松本 健一

奈良先端科学技術大学院大学情報科学研究科 〒630-0192 奈良県生駒市高山町 8916-5

E-mail: {youhei-k, keni-i, hiroki-y, hiro-iga, harua-t, masa-n, matumoto}@is.aist-nara.ac.jp

あらまし 近年，Webサービスを用いたシステム開発が注目を集めている．しかし，Webサービスの品質を評価するための方法論は未だに体系化されていない．本論文では，Webサービスアプリケーションの品質特性を計測することを目的に，4つの新たなソフトウェアメトリクスを提案する．具体的には，まずサービス指向アーキテクチャとオブジェクト指向設計の違いに着目し，既存のオブジェクト指向メトリクスの適用可能性を検証する．この検証結果を基に，新たに4種類のWebサービスメトリクス(RFWS,NOWS,EMWS,NHTWS)を提案する．提案したメトリクスを，我々の研究グループが開発した3種類のWebサービスアプリケーションに適用し，実験結果とメトリクスの関連性を考察した．その結果，Webサービスアプリケーションの効率性・保守性と提案メトリクスとの間に関連があることが認められた．

キーワード Webサービス，ソフトウェアメトリクス，サービス指向，オブジェクト指向

A Proposal of Software Metrics of Web Service Application

Youhei KUSHIDO Ken-ichi ISHII Hiroki YAMAUCHI Hiroshi IGAKI Haruaki TAMADA
Masahide NAKAMURA and Ken-ichi MATSUMOTO

Graduate School of Information Science, Nara Institute of Science and Technology

8916-5 Takayama, Ikoma, Nara, 630-0192 Japan

E-mail: {youhei-k, keni-i, hiroki-y, hiro-iga, harua-t, masa-n, matumoto}@is.aist-nara.ac.jp

Abstract Web service applications are one of the most emerging applications in the networked computing. However, there has been no systematic methodology to evaluate the quality of the Web service applications yet. To quantitatively evaluate the quality of the applications, this paper presents four new software metrics. First, we see the difference between service-oriented architecture and object-oriented design, and validate the applicability of the conventional object-oriented metrics. Based on the validation, we propose four new metrics (RFWS, NOWS, EMWS and NHTWS) for Web service applications. We applied the proposed metrics to three versions of a Web service application. The empirical result showed that the proposed metrics have a relevance to performance and maintainability of the Web service application.

Keyword Web Service, software metrics, service-oriented, object-oriented

1. はじめに

ネットワークの進歩とソフトウェアの多様化により，多種多様なサービスが溢れている．このような中，より高度なサービスを効率的に実現するためにサービスの連携が必要になってきており，その一手段としてWebサービスが注目されている[2]．Webサービスはサービス指向アーキテクチャ(Service Oriented Architecture)の考えに基づき，分岐したサービス間の疎結合を標準化された手順(XML, SOAP/HTTP, UDDI)により実現するものである．従来のオブジェクト指向設計のように機能を一つのオブジェクトとして設計するのではなく，より粒度の大きいサービスを一つの部品(コンポーネント)として設計する[4]．これにより，既存サービスの再利用性を向上し，標準化された通信手段によって，異なるシステム間の連携が効率よく実現できるとされている．

近年 Web サービスを用いたシステムがいくつか開発されつつある(例：Google Web APIs[5], Amazon Web Service[1])．しかし，サービス指向アーキテクチャに基づいたWebサービスは，実地運用が開始されてからまだ日が浅く，体系だった開発方法論は未だに提案されていない．また，著者らの知る限りでは，Webサービスアプリケーションの品質特性を計測する手段も存在しない．

そこで本論文では，Webサービスアプリケーションの品質特性を計測することを目的に，4 つの新たなソ

フトウェアメトリクスを提案する．具体的には，まずサービス指向アーキテクチャとオブジェクト指向設計の違いに着目し，既存のオブジェクト指向メトリクスの適用可能性を検証する．この検証結果を基に，新たに 4 種類の Web サービスメトリクス(RFWS, NOWS, EMWS, NHTWS)を提案する．さらに，提案したメトリクスを，我々の研究グループが開発した 3 種類の Web サービスアプリケーションに適用し，実験から得られた結果とメトリクスの関連性を考察する．その結果，提案するメトリクスと Web サービスアプリケーションの品質との間に関連が認められた．

2. ソフトウェア品質特性とメトリクス

2.1. ソフトウェア品質特性

ソフトウェア品質を分析するための定性的な枠組みとして，ソフトウェア品質特性が提案されている(ISO/IEC9126 - JIS X0129 [6]，図 1 参照)．

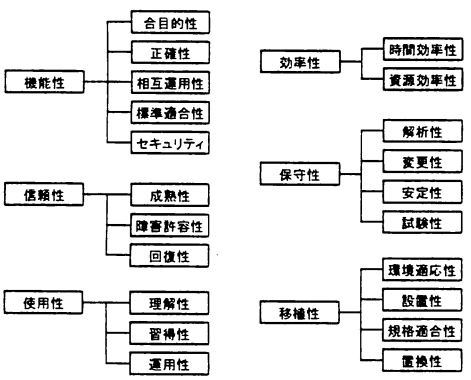


図 1　ソフトウェア品質特性[6]

この規格では，ソフトウェアの 6 種類の品質特性とそれらをさらに詳細化した 21 種類の品質副特性を定めており，一般的にこれらの特性からソフトウェアに応じて必要なものが取捨選択される．以降，代表的な 6 種類の品質特性について簡単に説明する．

「機能性」はソフトウェアがある目的をもって求められる必要な機能を実装している度合いである．ソフトウェアが正しく動くかといった正確性や，相互運用性・セキュリティといったもの等がこれにあたる．

「信頼性」は実装している機能があらゆる条件の下で機能要件を満たして(必要な期間)正常動作し続けることができる度合いである．障害が起こっても表面上正しく動作する度合い(障害許容性)等がこれにあたる．

「使用性」はソフトウェアの使いやすさや使用するのにかかる労力の度合いである．ソフトウェアの使い方の理解しやすさ(理解性)等がこれにあたる．

「効率性」は明示された条件における，ソフトウェアがもつ目的達成の度合いと，使用する資源の量の関係である．ソフトウェアの機能を実行する際の時間的なコスト(時間効率性)等がこれにあたる．

「保守性」はソフトウェアの改訂に付随する一連のタスクに関する労力の度合いである．バージョンアップなどに伴う変更にどれぐらいの労力がかかるか(変更性)等がこれにあたる．

「移植性」はソフトウェアをある環境から別の環境下に移した場合のソフトウェアの能力を推し量る指標である．ソフトウェアが違う環境に移植されても正常に動作する度合い(環境適応性)等がこれにあたる．

2.2. 既存のソフトウェアメトリクス

ソフトウェアメトリクス(以降メトリクスと呼ぶ)とはソフトウェアのさまざまな品質特性を客観的な数学的尺度によって定量的に評価するものである．ソフトウェアの開発において早期に適切なメトリクスを用いて品質や進捗状況などを定量的に評価することは，後の行程での工数割り当てや品質管理のための指標として非常に有益なものとなる[8][9]．

現在まで様々なメトリクスが提案されてきたが，ここではオブジェクト指向メトリクスとして有名な Chidamber らの複雑度メトリクス(C&K メトリクス)について説明する[8]．C&K メトリクスではソースコード中のクラスに注目して，主に保守性に関する 6 つのメトリクスを提案している．

WMC(weighted methods per class). あるクラスのメソッドをアルゴリズムの複雑さに基づき重み付けしその和を計測する．メソッドの重み付けにはそのメソッドの行数，変数の数等を用いるが，クラスのメソッドがどれも同等の複雑さだった場合は単純にクラスのメソッド数である．WMC が高いほど複雑であり保守性が下がる．

DIT(depth of inheritance tree). あるクラスのスーパクラスの数を計測する．DIT が高いほど継承されている変数やメソッドが多いということであり，再利用性は向上するが，使用性や保守性が下がる．

NOC(number of children). あるクラスのサブクラスの数を計測する．NOC が高いほどサブクラスへの影響が高いので保守性が下がる．

CBO(coupling between objects). あるクラスに関係しているクラスの数を計測する．CBO が高いほど他のクラスに依存していることになり保守性が下がる．

RFC(response for a class). あるクラスに関係しているメッセージの数を計測する．RFC が高いほどメッセージの数が多いということであり，テスト・デバッグにコストがかかり保守性が下がる．

LCOM(lack of cohesion of methods). あるクラスの凝集性の欠如を計測する．LCOM が大きいほど変数を共有している部分が多いことを表しメンテナンスにコストがかかり保守性が下がる．

3. Webサービスとサービス指向アーキテクチャ

3.1. Webサービス

Webサービスとは，ソフトウェアをサービスという単位でコンポーネント化し，Webサーバ上でその機能を提供するための標準的な枠組みである．OSやプログラミング言語に依存することなく，異なるアプリケーション間の連携を可能にするサービス指向アーキテクチャの考えに基づいている[3]．図1にWebサービスの利用形態について示す．各Webサービスは，UDDIレポジトリと呼ばれるWebサービスを管理するサーバ上においてWSDLと呼ばれる規格に従いサービスへのインターフェース(即ち，メソッド)を公開する(図1(0))．クライアントアプリケーション(以降CAと呼ぶ)は，WSDLの情報をもとにWebサービスの場所を特定し(図1(1)(2))，公開されたメソッドを通じてWebサービスの機能を利用する(図1(3)(4))．このメソッド呼び出しは，リモートプロシージャコール(RPC)によって遠隔的に行われるが，RPCに伴うメッセージ授受は，全て標準化された手続き(XML, SOAP)によって行われるため，CAは通常のメソッド呼び出しとほぼ同じ方法でWebサービスを利用できる．つまり，CAの開発者は，メッセージの書式や送受信の手順，Webサービスの内部ロジック等を一切気にすることなく(疎結合(loose coupling)と呼ばれる)，Webサービスの機能をアプリケーションに組み込むことが出来る．

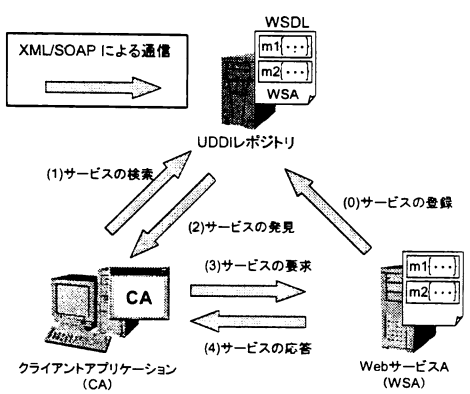


図1 Webサービスの利用形態

3.2. サービス指向とオブジェクト指向の違い

サービス指向では，最も重要なのは疎結合という概念である．Webサービスはクライアントアプリケーション(CA)や他のWebサービスとの連携において疎な結合である．疎結合を実現するために，一旦公開されたWebサービスはそのインターフェースを変更することは原則的に許されない．CAの開発者はWebサービスの実装内部を知ることが出来ないため，勝手なインターフェース仕様の変更はそのWebサービスを利用する全てのCAのテストのやり直しにつながるからである．

一方，疎結合の制約が特に規定されていない一般のオブジェクト指向開発では，オブジェクトが密に結合される場面が少なくない．このような場合ではあるオブジェクトに変更があった場合には開発者はその変更に対してアプリケーションが受ける影響等を考慮しつつプログラムしなおすという多大な労力が必要となる．

また，サービス指向開発において，「サービスの継承」という概念は存在しない．サービスの提供者の立場からは，提供するサービス機能のうちどのサービスが実際に使用されるかはわからないし，使用する側からはそのサービスの全ての機能を継承して新たなサービスを作るというスタイルはとらない．

3.3. オブジェクト指向メトリクスの適用可能性

サービス指向においては，オブジェクト指向では規定されていなかった「サービス間の疎結合」という新たな概念が存在する．従って，従来のオブジェクト指向メトリクスをWebサービスアプリケーションに直接適用することはできないと考えられる．ここでは，2.2節で述べたC&Kメトリクスを取り上げて考察する．

まず，CBOをそのままWebサービスアプリケーションに適用することは問題があると思われる．Webサービスにおいては，疎結合を実現するために，そのインターフェースが変更されることは原則的に許されていない．従って，一つのアプリケーションに関連するWebサービスの数が，必ずしも保守性の低下につながるとは言えない．前節で述べたように，オブジェクト指向での結合とサービス指向での結合には違いがあるため，この事を考慮する必要がある．

次に，オブジェクト指向におけるクラスには継承関係があるが，Webサービスには継承関係はない．従って，DITおよびNOCをWebサービスに適用することはできない．オブジェクトとWebサービスはコード内では同じようにメソッド呼び出しによって使用されるため，一見似ているが，別のものであるので慎重にソフトウェアメトリクスを考える必要がある．

既存のオブジェクト指向メトリクスは，各CAまたはWebサービス単体の計測においては適用可能であるが，CAとWebサービスの連携，およびWebサービス間の連携の計測においては，新たなメトリクスが必要になると考える．

4. Webサービスメトリクスの提案

既存のソフトウェアメトリクスの適用可能性を考慮し，我々は以下の4つのメトリクスを提案する．

- RFWS(response for a Web Service)
- NOWS(Number of Web Services)
- EMWS(effective methods per Web Service)
- NHTWS(number of hop to terminus Web Service)

以降の定義において，CA をメトリクスの計測対象とするクライアントアプリケーション，WS を Web サービスとする．Web サービス WA が別の Web サービス WB を呼び出す場合には，WA を CA とみなしてメトリクスを算出する．

4.1. RFWS(response for a Web Service)

定義：

W1, W2,...,Wn を CA が直接呼び出す全ての WS とする．このとき，CA の RFWS メトリクスを

RFWS = CA と Wi ($1 \leqq i \leqq n$)との間での要求・応答メッセージの総和

と定義する．

説明：

ある CA とその CA とやり取りしている全ての Web サービスについて，CA からのサービスの要求メッセージと Web サービスからの応答メッセージの数を計測する．この数を計測することにより，ネットワークの利用の度合いがわかり効率性を評価できると考える．RFWS が高ければ，つまり処理的にボトルネックとなりうるネットワークの利用の度合いが高いということになり効率性は悪くなると思われる．また，RFWS を計測することにより同時に CA の Web サービスへの依存の度合いがわかり，信頼性について評価できると考える．RFWS が高ければ，ネットワークを通して Web サービスとやり取りをすることが多いということになりネットワークでの障害にあいやすくなるということであり信頼性が下がると思われる．RFWS が高いほどメッセージの数が多いという点で既存の RFC と類似しているが，テスト・デバッグにかかる労力については有用な評価はできないと考える．その理由は CA と Web サービス間は疎結合であるため，もし Web サービスに変更があったとしても CA はインターフェースが変わらない限り影響を受けないからである．

4.2. NOWS(Number of Web Services)

定義：

W1, W2, ..., Wn を CA が関係する全ての WS とする．このとき，CA の NOWS メトリクスを

NOWS = n

と定義する．

説明：

CA と直接的または間接的に連携する全ての Web サービスの数を計測する．この数を計測することにより Web サービスを利用している度合い，つまりどれだけのネットワーク資源を使用するのかがわかり信頼性について評価できると考える．NOWS が高ければそれだけ複数のネットワークを通る可能性が高くなり，ネットワークでの障害にあいやすくなるということであり信頼性が下がると思われる．また，Web サービスを利用する事は CA 内のメソッドを利用するよりオーバーヘッドが大きくなると思われるため，効率性を評価できると考える．NOWS が高ければオーバーヘッドが大きくなり効率性は悪くなると思われる．さらには，Web サービスの疎結合の特徴から NOWS を計測する事により保守性を評価できると考える．NOWS が高ければ，Web サービスの疎結合性より保守性の面において CA が考慮するべき労力が減少し，保守性はよくなると思われる．

4.3. EMWS(effective methods per Web Service)

定義：

W1, W2,...,Wn を CA が直接呼び出す全ての WS とする．このとき，CA の EMWS メトリクスを

EMWS = EM / PM

EM : W1,W2,...,Wn の利用メソッド数の総和

PM : W1,W2,...,Wn の公開メソッド数の総和

と定義する．

説明：

ある CA が利用している Web サービスに着目し，CA がその Web サービスのメソッドを何個使用しているかという事を，その Web サービスの全ての公開メソッド数で割った値を計測する．この数を計測することにより，CA の Web サービスのメソッドの利用の具合がわかり，CA からみて Web サービスがどれだけ CA の目的に合致していたか，つまり機能性が評価できると考える． EMWS が 1 に近いほど Web サービスの利用者側である CA の目的に合致していることになり，機能性がよいことになる．

4.4. NHTWS(number of hop to terminus Web Service)

定義：

W1,W2,...,Wk を WS とする．

今，Wi が Wi+1 ($0 \leqq i \leqq k$)を要求するような系列

ρ = W0(=CA), W1, W2,...,Wk

を CA の WS 系列と定義する．又，ρ の長さ(=k)をホップ数と定義し，hop(ρ)と書く．CA が WS 系列 $\rho 1, \rho 2, \ldots, \rho n$ を持つとき，CA の NHTWS メトリクスを，

NHTWS = $\max_i$ {hop(ρi)}

と定義する．

説明：

ある CA からのサービスの要求・応答メッセージと連携する Web サービスをシーケンスチャートで表したときの最大ホップ数を計測する．この数を計測することにより CA からの要求を受けた Web サービスがどれだけ他の Web サービスに依存しているのかがわかり信頼性を評価できると考える．また，NHTWS は潜在的な Web サービスの利用数とも考えられ効率性も評価できると考えられる．NHTWS が大きいほど，CA の要求はさまざまな Web サービスを経由することに

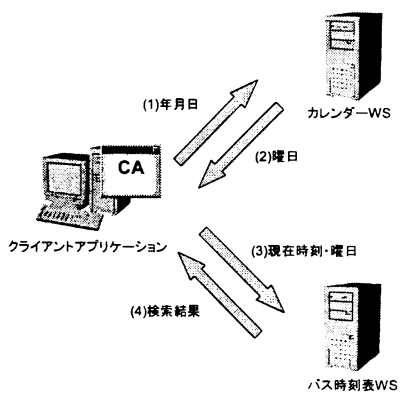
なり，信頼性が悪くなると思われる．また，Webサービスを経由する分だけ効率性が悪くなると思われる．

5. 計測実験及び評価

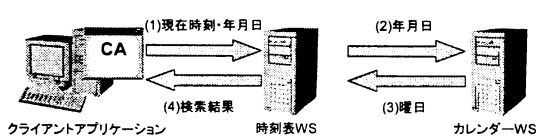
提案した4つのメトリクスを，我々が文献[7]において開発したWebサービスアプリケーションに適用し，評価を行った．

5.1. バス時刻表検索サービス[7]

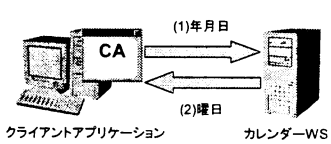
我々は文献[7]において，Webサービスを用いた「バス時刻表検索サービス」を開発した．このサービスは，1つのクライアントアプリケーションCAと，2つのWebサービス(時刻表検索WS，カレンダーWS)から構成され，ユーザが現時刻から最も近いバスの時刻を1クリックで取得できる機能を提供する．我々はこの同一の仕様を基に，図3に示す3種類の実装を行った．



(a)リダイレクト型



(b)プロキシ型



(c)スタンドアロン型

図3　バス時刻表検索サービスの3種類の実装方法[7]

図3(a)のリダイレクト型実装は，CAがカレンダーWSの結果を受け取ってから時刻表WSを要求する方式である．(b)のプロキシ型は，時刻表WSがCAの代わりにカレンダーWSにアクセスする方式である．最後に(c)のスタンドアロン型は，CAが時刻表WSの機能を取り込んだ実装となっている．また，[7]における性能評価実験の結果は，表1，表2の通りとなっており，実行時間からスタンドアロン型，プロキシ型，リダイレクト型の順に効率性が良く，LOC(Line of code)の値からプロキシ型，リダイレクト型が保守性の面においてスタンドアロン型より優れている結果となっている．

表1　実行時間の計測結果(秒)[7]

ループ回数	プロキシ型	リダイレクト型	スタンドアロン型
1	3.80	4.06	2.48
100	5.11	5.83	3.39
1000	16.98	21.65	11.39

表2　LOCの計測結果(行)[7]

	プロキシ型		リダイレクト型		スタンドアロン型	
LOC	CA	WS	CA	WS	CA	WS
	102(8)	145(35)	108(15)	140(30)	199(106)	NA

5.2. 計測実験結果

4つのメトリクスを適用し測定した結果について述べる(表3)．まずRFWSは，リダイレクト型ではカレンダーWSとの間で2，バスWS1との間で2の計4の値となった．プロキシ型ではバスWS2との間での2のみであり，スタンドアロン型ではカレンダーWSとの間での2であった．

NOWSは，CAに関連する全てのWebサービスの数であるのでリダイレクト型，プロキシ型ではともに2となり，スタンドアロン型は1となった．

EMWSは，リダイレクト型はカレンダーWS，バスWS1の両方が公開メソッド数は1でありCA1が使用しているメソッドも1であるので(1+1)÷(1+1)で1となった，プロキシ型についてもバスWS2の公開メソッド数1でCA2が使用しているメソッドも1であるので1÷1となり1であった．スタンドアロン型も同様にカレンダーWSを用いているので1÷1で1となった．

NHTWSは，リダイレクト型ではカレンダーWSと時刻表WSともに連携するWebサービスがないので1である．プロキシ型では時刻表WSがカレンダーWSと連携しているのでCAからのホップ数は2となり2である．スタンドアロン型はカレンダーWSが他のWSと連携していないので1である．

表3　計測結果

メトリクス	リダイレクト型	プロキシ型	スタンドアロン型
RFWS	4	2	2
NOWS	2	2	1
EMWS	1	1	1
NHTWS	1	2	1

5.3. 考察

我々が開発した文献[7]の3種類のWebサービスを利用したアプリケーションに対して提案する4つのメトリクスを適用した結果について考察し，提案するメトリクスの評価を行った(表4)．評価結果において○とは提案メトリクスと品質特性の間に相関が見られた項目で，△は相関があると予想したが今回の計測実験では評価する事ができなかった項目で，×は予想に反して相関が見られなかった項目である．

表 4　提案メトリクスと品質特性の関係

品質特性	RFWS	NUWS	EMWS	NHTWS
機能性			△	
信頼性	△	△		△
使用性				
効率性	○	○		×
保守性		○		
移植性				

RFWS については，信頼性と効率性について相関があると考えていたが，RFWS の測定の結果，リダイレクト型が高い値を示し，プロキシ型とスタンドアロン型が低い値を示した．この事は，表 1 に示される「リダイレクト型はプロキシ型とスタンドアロン型に比べて実行時間が遅かった」という結果に相関が見られ，RFWS は効率性を評価できると思われる結果となった．信頼性については，今回の実験において信頼性についての議論がなされておらず RFWS の信頼性に関する相関については評価できなかった．

NOWS については，信頼性と効率性と保守性について相関があると考えていたが，NOWS の測定の結果，リダイレクト型とプロキシ型が高い値を示し，スタンドアロン型が低い値を示した．この事は，表 1 に示される「リダイレクト型とプロキシ型はスタンドアロン型に比べて実行時間が遅かった」という結果と合致しており効率性について相関があると思われる．また，表 2 に示される「リダイレクト型とプロキシ型はスタンドアロン型に比べて保守性の面で優れている」という結果に対して，相関があると思われる．信頼性については前述の通り評価を行う事はできなかった．

EMWS については，機能性について相関があると考えていたが，今回適用したアプリケーションではそれぞれの Web サービスがそれぞれ目的の機能だけを果たすように開発されていたため EMWS の値としては全てが同じ値としてでてしまった．そのため今回の計測実験では EMWS の機能性に関する評価はできなかった．

NHTWS については，信頼性と効率性について相関があると考えていたが，NHTWS の測定の結果，プロキシ型が高い値を示し，リダイレクト型とスタンドアロン型が低い値を示した．この事は，表 1 に示される「プロキシ型はスタンドアロン型に比べて実行時間が遅かった」という結果に合致したが，「プロキシ型はリダイレクト型とくらべて実行時間が早い」という結果には合致しなかった．よって，NHTWS は我々の予想に反して効率性に関しては評価できないと思われる結果となった．信頼性については前述の通り評価を行う事はできなかった．

今回の計測実験では，計測対象が効率性と保守性についてのみの性能評価であったため，提案メトリクスの機能性と信頼性についての相関が評価できなかった．また，計測実験の回数もまだまだ少なく提案メトリクスの有用性についてはまだまだ議論する必要があると思われる．

6. 終わりに

本論文では既存のオブジェクト指向ソフトウェアメトリクス(C&K メトリクス)を Web サービスに適用可能か考察した．考察の結果，Web サービスの疎結合性と Web サービスでは継承関係が存在しない事からそのまま適用する事に問題があることを指摘し，その問題を考慮した新たな 4 つのメトリクス(RFWS,NOWS,EMWS,NHTWS)を提案した．そして提案したメトリクスを当研究室で開発した Web サービスを利用した 3 つのクライアントアプリケーション(同じ目的を達するが実装の仕方が違う)に適用し，提案メトリクスの検証・考察を行う事によって提案するメトリクスの有用性について一部示した．

今後の予定としては，提案するメトリクスを他の多種多様な Web サービスを用いたクライアントアプリケーションへ適用し，本論文で検証できなかった提案するメトリクスの有用性についてさらに議論を進めていきたいと思う．

文　献

[1] Amazon Web Services, http://www.amazon.com/gp/browse.html/104-0877510-2922306?node=3435361

[2] 青山幹雄, “Web サービス技術と Web サービスネットワーク”, 信学技報, IN2002-163, pp.47-52, Jan. 2003.

[3] David A.Chappel and Tyler Jewell, Java Web サービス, 長瀬嘉秀, オライリー・ジャパン, 東京, 2002

[4] Ethan Cerami, Web サービス, 長瀬嘉秀, オライリー・ジャパン, 東京, 2002

[5] Google Web APIs, http://www.google.com/apis/

[6] 東基衛, ソフトウェア品質評価ガイドブック, 東基衛他編, 日本規格協会, 1994

[7] 石井健一, 異なる連携方式を用いた Web サービスアプリケーションの開発および評価, 信学技報, IN, March. 2004, (to appear)

[8] Shyam R. Chidamber and Chris F. Kemerer, “A Metrics Suite for Object Oriented Design”, IEEE Transactions on Software Engineering, Vol.20, No.6, pp.476-493, June. 1994.

[9] Yacoub S., Ammar H. and Robinson T., “Dynamic Metrics for Object Oriented Designs”, Proc. of the Sixth International Symposium on Software Metrics, pp50-60, Boca Raton, Florida, November, 1999