Jurnal Teknologi

Full Paper

REVIEW ON STRUCTURAL PROPERTIES METRICS IN SOA DESIGN

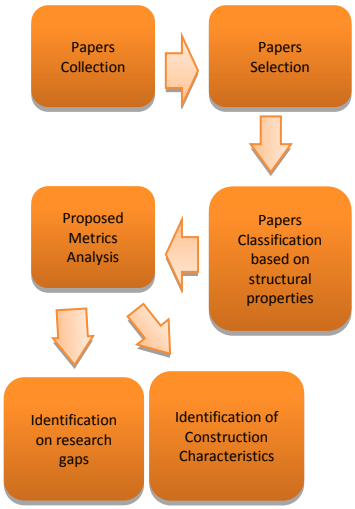
Nik Marsyahariani Nik Daud[a*], Wan Mohd Nasir Wan Kadir[b]

[a]Universiti Teknologi MARA Terengganu, Terengganu, Malaysia
[b]Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia

Article history
Received
2 February 2015
Received in revised form
14 September 2015
Accepted
12 October 2015

*Corresponding author
mndnik2@live.utm.my

Graphical abstract



Abstract

The application of metrics for the purpose of measuring qualities of software has been widely practice within software engineering industry. Though in the beginning most measurement involves artifacts such as source code, quality measurement from the beginning of software lifecycle is slowly gathering attention. Service-Oriented Architecture (SOA) is a new paradigm that introduces way to develop software in a faster way by adopting available services. SOA application is better comprehended using its composition design. Thus in earlier researches, many metrics had been proposed in relation to SOA design. This paper focuses on proposed metrics from earlier research works specifically one that measure the structural properties of SOA application. The structural properties metrics selected are analyzed from both static and dynamic perspectives. Around seventeen papers are selected during the data collection process. The proposed metrics are then separated into its respective properties type and are further analyzed to find required characteristics for each discussed property. The result from the analysis is list of construction characteristics for each structural property that can be used as guidelines for future researchers that incline on proposing new metrics. The metrics classification also indicates on research gap within the area of structural properties metrics in SOA domain.

Keywords: Structural properties metric, service oriented architecture, metrics review

Abstrak

Penggunaan metrik bagi tujuan pengukuran kualiti perisian telah lama diamalkan di dalam industri kejuruteraan perisian. Walaupun pada permulaannya kebanyakan pengukuran yang dibuat melibatkan kod pengaturcaraan, pengukuran kualiti pada peringkat awal kitar hidup perisian mula mendapat perhatian. Senibina berdasarkan servis (SOA) merupakan paradigm baru yang memperkenalkan cara untuk pembangunan perisian dengan lebih pantas menggunakan servis sediaada. Aplikasi SOA lebih mudah difahami menggunakan rekabentuk kompositnya. Oleh itu, dalam penyelidikan terdahulu, banyak metrik yang dicadangkan adalah berkait dengan rekabentuk SOA. Artikel ini memfokus kepada cadangan metrik daripada penyelidikan terdahulu terutamanya metrik yang mengukur struktur aplikasi SOA. Metrik struktur yang dipilih dianalisis dari perspektif statik dan dinamik. Sebanyak tujuh belas artikel telah dipilih semasa proses pengumpulan data. Metrik yang dicadangkan kemudiannya dipisahkan mengikut jenis struktur dan dianalisis bagi mengenalpasti karakter yang perlu ada bagi setiap struktur. Keputusan akhir yang diperolehi adalah senarai bagi karakter pembangunan bagi setiap struktur yang boleh digunapakai sebgai panduan bagi penyelidik yang berminat untuk mencadangkan metrik baru. Pengklasifikasian metrik mengikut struktur juga membolehkan pengenalpastian potensi penyelidikan di dalam bidang metrik struktur SOA.

Kata kunci: Metrik sifat struktur, senibina berorientasikan servis, kaji semula metrik

1.0 INTRODUCTION

Service Oriented Architecture (SOA) is bringing new paradigm within software development. Considered as belonging to distributed system, SOA application uses services to execute its functionalities thus shorten the development time and deliver more reliable output. The involvement of services that are not measurable from maintainer's perspectives however had encouraged on propositions of new type of metrics that will allow for measurement process to take place. Qualities of software are often related to structural design of the software. Thus it is only logical that most of the quality related metrics for SOA usually involves on measuring the structure of SOA application.

SOA application is developed based on concept where application logic is encapsulated as a service and these loosely coupled services interact with each other via messages [1]. Its idea is to encourage on reusability of software component that had been proven successful into other similar software projects. SOS can either be services that are composed together to fulfill specific business goal or it can be software that uses certain services to execute processes. Being loosely coupled, interaction between services in SOS are limited to only messages, thus it is expected to be more maintainable. Services in SOS can be added and removed dynamically during runtime.

As services represent business processes, this making it closely relates to business logic and rules that tends to change rapidly [2], forcing services to be frequently maintained. It also opens up on new perspective on maintaining software as these changes happen dynamically during runtime without human interference. Thus, a highly maintainable SOS is mostly welcomed. Apart from this, most of previous researches in SOS have been concentrating on area such as mechanism for developing and publishing services, services' composition, or invoking and discovering services that other area in SOS such as quality of the software, are underdeveloped [3]

Metrics that measure structural properties are important where they are mostly use as basis to form other complex measurement. Structures of software have direct consequence on external quality of software. As an example, quality attributes such as maintainability and performance, are affected by complexity or cohesion. An early measurement on these structures will prevent from spending lots of time and cost in later stage to improve the software. Identifying what makes SOA different from other types of software has been one of the main interests in SOA's early day. These works had been leading towards identification of SOA characteristics such as loose coupling and high granularity [4, 5]. It seems evidence that one of the strength in SOA is dedicated within the structure of the software itself.

Since the introduction of object oriented programming, codes had progressively been written in modular way. New analysis is introduced as consideration on interactions and relationships within this modular software is taken into account. Properties that related to measurement like cohesion, coupling, size of program, and complexity had been introduced to capture the properties of software. Some of these properties had been re-applied and reuse to measure SOA structure. These include coupling, cohesion and complexity. For this paper, the classification's scope is restricted to these three attributes due to its common usage and wide applicability.

As SOA is getting into the hype these days, many researchers had attempted on introducing new metrics in order to improve the quality of SOA application. Taking the same route as object oriented paradigm, the structural properties of SOA application is being studied and metrics based on its structural properties is proposed. However, a need to identify what makes the new metric acceptable in order to measure these structural properties has risen. The main idea of this paper is to find the characteristics that hold the meaning of the properties and used it as a benchmark in future works. The paper focuses on structural properties as metrics based on the properties are usually used to derive measurement of other quality attributes such as maintainability. Aside from it, this study will also help to identify the research gap within metric propositions in SOA domain.

Coupling was introduced as programmer started to write program code in modular way. It is defined as "the manner and degree of interdependence between software modules"[6]. There had been some categorization done on coupling where it is separated from high coupling to low type of coupling. As suggested in [6] there are six types of coupling; common-environment coupling, content coupling, control coupling, data coupling, hybrid coupling and pathological coupling. [7] however, suggested only five types of coupling which covers data coupling, stamp coupling, control coupling, common coupling and content coupling with data coupling is the lowest type of coupling and content coupling as the highest type of coupling.

Cohesion is introduced as a measure on relationships of elements within a module. Based on standard defined by [6], cohesion is defined as "the manner and degree to which the tasks performed by a single software module are related to one another" which emphasizes on module strength. Cohesion usually fall into one of the following categories: coincidental, logical, temporal, procedural, communicational, sequential, and functional [8]. These categories highlighted on types of cohesion existing in software with coincidental is the weakest type of cohesion up to functional as the strongest type of cohesion.

Complexity stated by [9] as "difficulties to maintain, change and understand software". One of the more famous complexity metrics is the cyclomatic complexity metric [10] where it measures independent path in source code. Complexity is usually derived from other concepts such as coupling, cohesion and size or it can be measured directly from the software artifacts.

All these three structural properties can be defined based on static and dynamic perspectives. Coupling for example, can be measured from static point of view by measuring coupling between services and at the same time dynamic relationship between services can affect the level of coupling. Thus the investigation scope of the paper is seen from both perspectives.

This paper is divided into the following sections: Section 1 briefs on some introduction involving SOA, metrics and structural properties. Section 2 summarizes on related works that can be found within this area. Section 3 presents on the preparation process of data collection by elaborating used method. Section 4 laid an overview where each structural metrics is grouped together and brief explanation on these metrics context is discussed. Section 5 further discussed the selected metrics based on characteristics that are available in the structural context. Finally, some conclusion on works done is presented in Section 6.

2.0 RELATED WORKS

Analysis on existing metrics is not new. Recent years show that many works have been done within this area where these studies varied in their objectives and domains of metrics, but focused on the similar ideas of discovering proposed metrics. Some of these studies focused on doing survey based on specific metrics in particular areas [11, 12] while others concentrated on discussing on trend of metrics in more generic idea [13, 14].

In the area of OO metrics, some studies is done based on general survey on OO metrics [15-17]. Others focused in specific area of OO metrics, such as design metrics [18] and software quality metrics [17, 19, 20]. Within the area of software quality metrics, certain concepts e.g. complexity receives more attentions as there are studies concentrating on complexity in relation to OO metrics [21, 22].

Most of papers that focused on OO metrics reviews use similar ways to present their works. The process usually involves list of related metrics and some brief introduction for each metrics. These pattern can be traced from recent papers such as from [15, 17]. The only differences are on scope of the paper and how this metrics are represented, either by using a template or table.

In SOA, some works related to discussion of existing metrics are found. The earliest works detected are done by [11] where a survey is done on service reusability metrics. This is followed by [12] that focused on survey on coupling measurements in SOA metrics. Each paper listed related metrics in reusability and coupling, respectively. Recent work had been found in [23] that discussed on survey and classification made based on web services metrics proposed in recent years. However the author did not discuss further details of each metrics but were more focused on explaining available metrics in selected classification.

Using this knowledge, it seems that analysis of existing metrics within certain area is not a foreign idea. Even in a new developing area such as SOA, there are already few papers that discusses on this matter as highlighted earlier. However, most of works in SOA is either focused in certain quality attributes metrics or specific technology in SOA. This paper aims on highlighting structural properties metrics from SOA design aspect that hopefully will narrow the gaps of literature in this area.

3.0 DATA COLLECTION

The first step in collecting related papers from recent studies is to conduct a search process using multiple search tools. List of keywords that include the combinations of SOA and metrics within publication title are produced. The reason for insisting on having the keywords within the title is to find papers that concentrate and discuss in details on proposed metrics instead of briefly mentioning the metrics. The search included papers from 2006 and above as to keep it relevant to the topic. The search is conducted using prominent search tools, such as *EngineeringVillage, SCOPUS, Web of Science* and *IEEE Xplore*. The search return total numbers of 603 related papers with about 108 of the papers are repetitive resulting in 495 different papers.

Filtering process is done on these papers by omitting papers that are not fulfilling certain criteria. The criteria are selected papers should discuss on metrics related to at least one of the three structural properties. Papers that discuss on SOA middleware are also excluded as the scope of this paper is on SOA application. The selected papers should at least discuss on the definition of the proposed metrics in details as the information are needed for discussion purposes. The last criteria that the paper need to have is that the proposed metrics are to be implemented during design phase. Due to this restriction, only seventeen papers had been selected for the research purposes. Table 1 lists the selected papers based on years and type of structural property metrics that the papers proposed.

Table 1 Selected papers based on structural properties

Year	Structural Properties Metrics		
	Coupling	Cohesion	Complexity
2006	[24]	-	
2007	[25]	[26]	-
2008	[27]	-	[28, 29]
2009	[30] [31]	-	[32]
2010	[33]	[34]	[35]
2011	-	[36] [37], [38]	-
2012	[39]	-	-
2014	[40]	[40]	[40]

4.0 METRIC CLASSIFICATIONS

This section briefly introduces each selected metric based on its structure properties type. Apart from that, an analysis is done to identify the characteristics for each group. The purpose is to get the overview of what these metrics cover and to identify characteristics while constructing metrics based on each type.

4.1 Coupling

Until recently, there are around eight works proposed for coupling metrics in SOA. Some of the earliest works on coupling begun at 2006 and most recent coupling metrics are proposed in 2012. One of the earliest work for coupling metric in SOA is proposed by [24] describing on decoupling of web services. The proposed metrics focused on measuring coupling within service context. Metrics are constructed based on relationship of service with environments such as its resources, required services, and connection with other services. The coupling is divided into two perspectives, dependency and invocation of services. Service dependency is furthered divided into resource dependency and required service dependency.

Metrics proposed are based on finding averages of coupling. He proposed four coupling metrics, three belonging to dependency relationship while one is based on service invocation. As can be seen from the definitions below, all four proposed metrics are calculated by finding the average of relationship to number of services. All these metrics returned value where lower computed value means lower coupling of the services.

In 2007, [25] proposed on coupling metrics for service oriented design based on formal model for SOA software. These metrics includes both static and dynamic aspects of the software. Measurement is applied during design phase where service oriented system is represented using graph model. This graph model represents implementation elements within services and outside services where it is divided into implementations such as business script, procedure and class, and service is defined using service interface and a set of implementations.

For the proposed metrics, weight is assigned for different type of relationship to emphasize on strength of each coupling. Apart from that, the construction of the metrics is basically counting number of relationship between elements. The metrics calculate number of basic coupling metrics aggregated into the metrics in order to represent coupling metrics at system and service level.

As part of a design quality model that being proposed, [27] proposed coupling metric that is based on number of provider and consumer services over software size. The measurement for coupling is made based on services interaction within a service composition. The design is based on three layers of services, process layer, intermediary layer and basic layer. The metric is constructed based on division of services over software size.

[30] proposed coupling metrics based on information theoretic principles in order to construct the coupling metrics. The metrics constructed based on dependencies between service consumer and provider. In this case, the dependencies are defined as call invocation between services where the relationship is described using dependency graph. Service coupling metric is constructed using information entropy where the computation is made on dependency graph between complex service with atomic service and the probabilities of changes that relate between these two elements. Low value of the metric indicates low coupling and vice versa.

[31] presented metrics that measure dynamic coupling in service oriented software. The measurement took place during runtime and considered dependencies at runtime only. Couplings are measured from services perspectives and some of the metrics are derived from object oriented metrics such as fan in and fan out metrics. Coupling metric is measured by counting number of calls made between two services over total number of calls made by one of the service.

Another study [33] proposed on coupling metric based on risk formula that involves criticality, probability of failure occurrence and probability of non-detected failure. The computation is made from service composition aspect which takes into account semantic coupling, syntactic coupling and physical coupling.

[39] suggested metrics suite analyzed by using fuzzy model. The metric suite extends existing works in coupling by considering dependency factors such as IO dependency, delayed dependency and indirect dependency between services. The coupling is measured directly as it rely on dependency between services directly. The relationships taken into consideration for metrics construction involve both

dependency and invocation relationship. Direct Dependency, DD(S) compute on number of relationship where two way relationships is counted as two. Delayed message dependency metric is counted as number of synchronous call over total number of message.

The most recent proposed coupling metrics are done by [40] that proposed on direct and indirect coupling metrics. The metrics is calculated based on architectural level design which involves consumer and provider dependency level in order to calculate coupling. Coupling is also computed by dividing it to size metric.

Table 2 summarizes on related works on coupling metrics that have been explained in this section. One work focused solely on dynamic aspect of coupling. Other works such as [33] and [39] concentrated more on static aspect of software with only one metric that measure coupling in dynamic aspect.

Table 2 Coupling metrics classifications

Coupling	View		Applied Phase	Measured artifacts
	Static	Dynamic		
[24]	Yes	Yes	Design	WSDL
[25]	Yes	Yes	Design	SOA application design model
[27]	Yes	No	Design	Architecture design
[30]	Yes	No	Design	Operations
[31]	No	Yes	Runtime	SOAP message
[33]	Yes	No	Design	SCA assembly model
[39]	Yes	No	Design	Direct
[40]	Yes	No	Design	Architecture design

4.2 Cohesion

Around seven works are found that proposed on cohesion metrics. Cohesion mostly related to coupling thus aside from proposing coupling metrics [26] had also proposed cohesion metrics for service oriented design. The design level metrics measure cohesion based on formal model of service oriented design which are related to categories of cohesion. These categories are coincidental cohesion, logical cohesion, communicational cohesion, external cohesion, sequential cohesion, implementation cohesion and conceptual cohesion. In reference towards service design, two new additional categories in cohesion, external cohesion and implementation cohesion is suggested. The cohesion metrics are derived based on assumptions made on these categories. Four characteristics of cohesion within SOA context, where the cohesion is investigated from data, usage, flow of operations and implementation is introduced. In this case, service cohesion is measured based on service operations.

[34] proposed on finding service cohesion based on operations sharing resources or business entity. The work can be categorized in communicational cohesion where the measurements are done based on service operation access on business entities. Cohesion between two operations is computed based on their shared and non-shared entities. The non-shared entities calculation is made based on connection available between these entities in other operations.

[36] proposed on measuring service cohesion based on conceptual relationships retrieved from business process. Though the study emphasizes more on method to identify and constructing service, a service cohesion degree metric is proposed to calculate data gathered from the proposed method.

Measuring cohesion in service design are also the intention of work proposed by [37]. Communicational cohesion is being the focus in the paper where the cohesion is measured based on operations between entities within service. The metrics proposed is developed based on entity relationship diagram and distance calculates between entities. Cohesion for services is computed by finding average of cohesions for all services.

One of the latest proposed works in cohesion metrics [38] focused on measuring lack in cohesion. These metrics measure on communicational and sequential cohesion aspect of operations listed on service interface. This work was mostly inspired by cohesion lack metrics proposed in OO domain. The metrics operate by measuring operations of services based on its input message and output message. Data for metrics is obtained from WSDL specification where message similarity is checked to calculate the cohesion presence specifically communicational cohesion and sequential cohesion. The metrics measure lack of cohesion within the provided services. [40] proposed on cohesion metrics that calculates cohesion based on number of consumer and provider belonging to a service. This is then divided with size of service to derive other cohesion metric.

A summary on discussed cohesion metrics can be found in Table 3. Discussion on cohesion metrics seems to involve business process as well as operation listed on service interfaces. To achieve better understandings on cohesion, semantics of the operations needs to be available.

Table 3 Cohesion metrics classifications

Cohesion	View		Applied Phase	Measured artifacts
	Static	Dynamic		
[26]	Yes	No	Design	Service interface operations
[34]	Yes	No	Design	Service operations
[36]	Yes	No	Design	Service operations
[37]	Yes	No	Design	Service operations
[38]	No	Yes	Runtime	Service interface operations
[40]	Yes	No	Design	Number of consumer and provider

4.3 Complexity

Complexity had always been considered as one of the key property to be taken into account for measurement purposes. One of the earliest attempts on constructing complexity metric in SOA domain is made by [41], which calculated complexity for SOA by constructing metric to calculate distributed operations within SOA. The proposed metric, balance factor, highlight on whether service resources distribution are well balanced in SOA in order to avoid problems such as bottleneck. The calculation is based on number of operations distributed among web services in SOA. Data are taken from WSDL documents and analysis done at architectural level.

From the modifiability perspectives of SOA, [28] had proposed on metrics that measured complexity as factor that contribute to modifiability. To calculate complexity, the coupling property is used as a measure for complexity. Apart from it, four metrics are introduced where complexity is addressed from system perspectives. The metric computes on system aggregation between consumer and provider. This work used generic architectural model to represent composition of SOA and speaks more on structure of the software by taking into account the calculation of size and coupling of the composition.

Another work that proposed complexity metric is [27]. The work proposed that complexity of service oriented system is measured based on number of operations metric. This is calculated by adding total number of synchronous calls and asynchronous calls times by 1.5. However no further explanation is done on how the metric is constructed which limited the usage of this metric.

Using SOMA lifecycle as the core work process, [32] proposed metric suite where metrics are divided into two levels, system and service level. These metrics are aggregated to form two metrics that measure complexity for both levels. The metrics used SOA artifacts such as WSDL files and SOAP messages to calculate the complexity of SOA. These metrics are evaluated using case study of real life project. The proposed metrics covered both runtime and design time phase. The calculation involved structure of SOA artifacts available during design and runtime. [40] calculated complexity of services by deriving it from cohesion and coupling metrics.

In 2010, an approach is made to view web service composition using Petri Net representation by [35]. As web service composition is represented using Petri net, complexity metrics related to control flow is introduced where it is evaluated using two case studies. The metrics introduced are count based metric and execution path metrics. This is, until recently, the one and only work on complexity metrics that calculated complexity from behavior view. Table 4 summarizes proposed complexity metrics and its scope and area of application.

Table 4 Complexity metrics classifications

Complexity	View		Applied Phase	Measured artifacts
	Static	Dynamic		
[41]	Yes	No	Design	WSDL
[28]	Yes	No	Design	Architectural model
[32]	Yes	No	Design and Runtime	WSDL, CDL
[35]	No	Yes	Design	Petri net business process
[40]	Yes	No	Design	Architectural model

5.0 RESULTS AND DISCUSSIONS

This section dedicated on discussing finding from previous sections. The discussion is divided into two separate views. The first part is dedicated on discussing construction properties patterns that are identified from these papers. Identifying the patterns can be used to guide researchers who are interested to do works related to metrics construction in these metrics. The second part of the discussion focused on classification of these proposed metrics into static and dynamic classes.

Using the proposed coupling metrics as references, pattern on properties that are used in constructing coupling metrics is identified. Table 5 listed identified properties for coupling and how many

coupling metrics are constructed based on these design properties. Most coupling metrics are constructed based on these two relationships, invocation calls and its dependency on other services. This relationship can take place within service level and the service level values are then aggregated in order to find system level coupling.

As coupling property deals with interaction of artifacts within an application, in design phase the level is divided into two which are service and system. This to ensure it covers both view of the application. Coupling measure taken from service level can affect the level of coupling at system level. The relationship of the coupling is divided into dependency and invocation. Mostly, dependency is coupling relationship that needs to be considered during static design while invocation is a relationship that occurs during dynamic interaction.

Table 5 Coupling metrics construction properties

	Relationship		Level	
Proposed Coupling metrics	**Dependency**	**Invocation**	**System**	**Service**
[24]	✓	✓	-	✓
[25]	✓	✓	✓	✓
[30]	-	✓	✓	✓
[31]	✓	✓	✓	-
[33]	✓	-	✓	-
[39]	✓	✓	-	✓
[40]	✓	-	-	✓

Table 6 listed properties used to construct cohesion metrics. All authors implied on the use of cohesion categories proposed by [42] to construct their metrics. Many authors aimed on quantifying sequential cohesion and communicational cohesion as this type of cohesion indicates good cohesion. And even though functional cohesion is the most indicates high order cohesiveness in module, it is hard to quantify this type of cohesiveness thus not many of the authors approached this type of cohesion. [26] proposed on four metrics that can be aggregated in order to measure on level of cohesion that a service belonged to.

As cohesion is a measure that involves internal aspects, service operations is the only indications on how cohesive is a service can be. At a higher level of design, some of the cohesion measurement took place at architectural level by using business entities as the artifacts. In SOA context, all of the proposed cohesion metrics used service operations to measure the cohesiveness of a service. Cohesion is usually measured based on operations and methods of components or classes. However, as service exposes its functionalities using service operations, the computation of perceived from business process perspective as three out of five authors used business entities as part of the measured artifacts.

Table 6 Cohesion metrics construction properties

	Measured Artifacts		Categories of Cohesion						
Proposed Cohesion metrics	**Service operations**	**Business entities**	**Co[a]**	**Lo[b]**	**Tp[c]**	**Pr[d]**	**Cm[e]**	**Sq[f]**	**Fng**
[26]	✓	-	✓	✓	✓	✓	✓	✓	✓
[34]	✓	✓	-	-	-	-	✓	-	-
[36]	✓	✓	-	-	-	-		✓	✓
[37]	✓	✓	-	-	-	-	✓	-	-
[38]	✓	-	-	-	-	-	-	✓	-
[40]	-	✓	-	-	-	-	-	-	-

[a]coincidental cohesion [b]logical cohesion [c]temporal cohesion [d]procedural cohesion [e]communicational cohesion [f]sequential cohesion [g]functional cohesion

Properties for complexity metrics construction are listed in Table 7. Complexity seems to be associated with number of services as most authors mentioned the relation between Number of Service (NoS) metric with complexity. Another property that is frequently mentioned within these papers is service composition. Authors that used service composition as ground work for complexity measures suggested that composition contributed to complexity of services as service that composed of aggregated services is more complicated compared to service that constitutes atomic service.

Table 7 Complexity metrics construction properties

		Type of complexity		Service Level	
Proposed Complexity Metrics	**Size**	**Structure**	**Dynamic**	**Aggregation**	**Atomic**
[41]	-	✓	-	-	✓
[28]	✓	✓	-	✓	✓
[27]	✓	✓	-	-	-
[32]	✓	✓	-	✓	✓
[35]	✓	✓	✓	✓	-
[37]	✓	✓	-	-	-
[40]	✓	✓	-	✓	✓

In previous section, discussion is done based on each structural property. Explanation is given on definition for each metrics before data extracted according to criteria. Selected criteria are type of SOA software views, phase where these metrics are applicable and type of artifacts chosen to be measured by the metrics. These criteria are selected as it helps to derive the final conclusion on the metrics classifications process.

The proposed SOA metrics are classified based on *when* and *what* perspectives. There are four types of class established: static structure, static behavior, dynamic structure and dynamic behavior. These four classes are used to explain on *when* and *what* perspectives for each metric. Table 8 represents available data using these four classes against the three structural properties metrics.

On classifying these metrics, criteria such as applied phase and measured artifacts are used to indicate on which class those metrics belong to. As explained previously, static refer to artifacts of software that are taken from design phase and level, while dynamic represent artifacts that considered time as one of elements. By comparing these factors with software views, the classifications are done on proposed metrics.

There are six works proposed on coupling metrics. Most of the works belong to static structure class with total number of five works. Two of the proposed works are classified in static behavior while only one work concentrates on coupling in dynamic behavior class. No work on measuring coupling in dynamic structure had been proposed yet. Some works belong to more than one type of classes such as [21] and [22]. This is due to the metrics that they proposed cover both structure and behavior views.

Works on cohesion metrics are scarcer as they belong to only two classes, static structure and dynamic behavior. There are no proposed works on both static behavior and dynamic structure. Three quarters of the proposed works concentrate on measuring cohesion based on its static structure. Only one work found in dynamic behavior. For complexity, there had been at least one work proposed in each class except for the dynamic behavior class. Similar to the other properties, most of the works are focusing on static structure class with one work for each static behavior and dynamic structure classes.

Using this data, it can be concluded that most metrics fall into static structure class. This can be due to the fact that measuring software at the early stage is way cheaper and less consuming as opposed to evaluating software during its operation time. Apart from that, there are less works proposed in static behavior and dynamic structure classes as the blurred line exist between these classes. For example, behaviors that are represented during static time are not expected to be much different from behavior on dynamic time. Thus, metrics that are proposed for static behaviors can be applied with some cautions, during dynamic behaviors.

Most works for all three structural properties fall into static structure categories. However there are lacking of works can be found within other categories. This opens up to possibilities for works to be done in other categories especially from dynamic perspective of the system. SOA application, being categorized as dynamic, distributed system thus studying its behavior from this aspect will improve on the quality of the system. Researchers can find opportunities to measure the dynamic aspect of SOA applications as it is still considered lacking in this particular area.

Table 8 Structural properties classifications

Class\Properties		Coupling	Cohesion	Complexity
Static	**Structure**	5	3	4
	Behavior	2	0	1
Dynamic	**Structure**	0	0	1
	Behavior	1	1	0

6.0 CONCLUSION

Structural properties such as coupling, cohesion and complexity play some major role in the level of qualities of software. Finding out what characteristics need to be highlighted when developing structural metrics is important as not to develop unusable metrics. This paper concentrates on collecting and classifying proposed structural properties metrics in recent years in relation to SOA domain. Identification of construction characteristics for coupling cohesion and complexity is made based on proposed metrics. Research gaps within this area are also recognized by classifying proposed metrics into static and dynamic perspectives of SOA application structures. The result shows that there is still lacking of studies within dynamic perspectives of SOA application. This might be due to its complexities as services are changeable and replaceable during SOA application runtime hence making the behavior more unpredictable.

The paper serves as two-folds for future works as it prepares on the list that can be used as guideline while constructing metrics for SOA design and to identify area where existing metrics could be used to derive new metrics in order to measure quality attributes in SOA application.

Acknowledgement

The authors would like to express their deepest gratitude to Research Management Center (RMC), Universiti Teknologi Malaysia (UTM) and Ministry of Education Malaysia for their financial support under Fundamental Research Grant Scheme (Vot number R.J130000.7828.4F216).

References

[1] Papazoglou, M. P. 2003. Service-oriented Computing: Concepts, Characteristics and Directions. *In Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE'03)*.

[2] Erl, T. 2005. *Service-Oriented Architecture: Concepts, Technology, and Design*. New Jersey: Prentice Hall.

[3] Athanasopoulos, D., Zarras, A., and Issarny, V. (2009, 26th May). Maintenance of Service Oriented Software. Available: https://www.rocq.inria.fr/arles/index.php/ongoing-research-projects/122-maintenance-of-service-oriented-software.

[4] Papazoglou, M. P. 2003. Service-oriented Computing: Concepts, Characteristics and Directions. In Web Information Systems Engineering, 2003. *WISE 2003. Proceedings of the Fourth International Conference on*. 3-12.

[5] Erl, T., Soa. 2008. Principles of Service Design. vol. 1. Prentice Hall Upper Saddle River.

[6] ISO. 2010. Systems and Software Engineering Vocabulary. ISO/IEC/IEEE 24765. 2010(E): 1-418.

[7] Mall, R. 2009. *Fundamentals of Software Engineering*. PHI Learning.

[8] Yourdon, E. and Constantine, L. L. 1979. *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design.*Prentice-Hall, Inc.

[9] Zuse, H. 1991. Software Complexity: Measures and Methods. Walter de Gruyter & amp; Co.

[10] McCabe, T. J. 1976. A Complexity Measure. *Software Engineering, IEEE Transactions on*. SE-2: 308-320.

[11] Karthikeyan, T. and Geetha, J. 2012. A Study and Critical Survey on Service Reusability Metrics. *International Journal of Information Technology and Computer Science (IJITCS)*. 4: 25.

[12] Babu, D. V. and Darsi, M. P. 2013. A Survey on Service Oriented Architecture and Metrics to Measure Coupling. *International Journal on Computer Science & Engineering*. 5.

[13] Kitchenham, B. 2010. What's Up with Software Metrics? – A Preliminary Mapping Study. *Journal of Systems and Software*. 83: 37-51.

[14] Gómez, O., Oktaba, H., Piattini, M., and García, F. 2008. A Systematic Review Measurement in Software Engineering: State-of-the-Art in Measures. In *Software and Data Technologies*. vol. 10, J. Filipe, *et al*. Eds. ed. Springer Berlin Heidelberg. 165-176.

[15] Jamali, S. M. January 2006. *Object Oriented Metrics: A Survey Approach*. Tehran.

[16] Xenos, M., Stavrinoudis, D., Zikouli, K., and Christodoulakis, D. 2000. Object-oriented Metrics-A Survey. In *Proceedings of the FESMA*. 1-10.

[17] Dubey, S. K. and Sharma, A. 2012. Comparison Study and Review on Object-Oriented Metrics. *Global Journal of Computer Science and Technology*. 12.

[18] Abreu, F. B. August 1995. Design Metrics for Object Oriented Software Systems. In *ECOOPS'95: Quantitative Method Workshop, Portugal*.

[19] Rosenberg L. H. and Hyatt L. E. April 1997. Software Quality Metrics for Object-oriented Environments. *Crosstalk Journal*. 10.

[20] El-Ahmadi, A. 2006. Software Quality Metrics for Object Oriented Systems. BSc Degree Engineering, Informatic and Mathematical Modelling, Technical University of Denmark, Kongens Lyngby.

[21] Henderson-Sellers, B. 1996. *Object-oriented Metrics: Measures of Complexity*. Prentice-Hall, Inc.

[22] Kumar R. and Kaur G February 2011. Comparing Complexity in Accordance with Object Oriented Metrics. *International Journal of Computer Applications*. 15: 42-45.

[23] Ladan, M. I. 2012. Web Services Metrics: A Survey and Classification. *Journal of Communication and Computer*. 824-829.

[24] Qian, K., Liu J., and Tsui, F. 2006. Decoupling metrics for services composition. In Computer and Information Science, and 2006 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse. ICIS-COMSAR 2006. *5th IEEE/ACIS International Conference on*. 44-47.

[25] Perepletchikov, M., Ryan, C., Frampton, K., and Tari, Z. 2007. Coupling Metrics for Predicting Maintainability in Service-Oriented Designs. In *Software Engineering Conference, 2007. ASWEC 2007. 18th Australian*. 329-340.

[26] Perepletchikov, M., Ryan, C., and Frampton, K. 2007. Cohesion Metrics for Predicting Maintainability of Service-Oriented Software. In Quality Software, 2007. *QSIC'07. Seventh International Conference on*. 328-335.

[27] Bingu, S., Siho C., Suntae K., and Sooyong P. 2008. A Design Quality Model for Service-Oriented Architecture. In Software Engineering Conference, 2008. *APSEC '08. 15th Asia-Pacific*. 403-410.

[28] Hofmeister, H. and Wirtz, G. 2008. Supporting Service-Oriented Design with Metrics. In Enterprise Distributed Object Computing Conference, 2008. *EDOC'08. 12th International IEEE*. 91-200.

[29] Dimitoglou, M. J. M. G. 2009. A Service Oriented Architecture Complexity Metric, Based on Statistical Hypothesis Testing.

[30] Wang, X.-J. 2009. Metrics for Evaluating Coupling and Service Granularity in Service Oriented Architecture. In Information Engineering and Computer Science. *ICIECS 2009. International Conference on*. 1-4.

[31] Quynh, P. T. and Thang H. Q. 2009. Dynamic Coupling Metrics for Service–Oriented Software. *International Journal of Computer Science and Engineering*. 3: 46-46.

[32] Hirzalla, M., Cleland-Huang, J., and A. Arsanjani. 2009. A Metrics Suite for Evaluating Flexibility and Complexity in Service Oriented Architectures. In Service-Oriented Computing – ICSOC 2008 Workshops. vol. 5472, G. Feuerlicht and W. Lamersdorf, Eds. ed: Springer Berlin Heidelberg. 41-52.

[33] Hock-Koon, A. and Oussalah M. 2010. Defining Metrics for Loose Coupling Evaluation in Service Composition. In Services Computing (SCC). *2010 IEEE International Conference on*. 362-369.

[34] Rostampour, A., Kazemi A., Shams F., Zamiri A., and Jamshidi, P. 2010. A Metric for Measuring the Degree of Entity-centric Service Cohesion. In *Service-Oriented Computing and Applications (SOCA), 2010 IEEE International Conference on*. 1-5.

[35] Mao, C. 2010. Control Flow Complexity Metrics for Petri Net-based Web Service Composition. *Journal of Software*. 5: 1292-1299.

[36] Kazemi, A., Rostampour A., Zamiri A., Jamshidi P., Haghighi H., and Shams F. 2011. An Information Retrieval Based Approach for Measuring Service Conceptual Cohesion. In Quality Software (QSIC). *2011 11th International Conference on*. 102-111.

[37] Daghaghzadeh, M., Dastjerdi A. B., and Daghaghzadeh H. 2011. A Metric for Measuring Degree of Service Cohesion in Service Oriented Designs. *International Journal of Computer Science Issues*.

[38] Athanasopoulos, D. and Zarras A. V. 2011. Fine-Grained Metrics of Cohesion Lack for Service Interfaces. In *Web Services (ICWS), 2011 IEEE International Conference on*. 588-595.

[39] Karhikeyan, T. and Geetha J. 2012. A Metrics Suite and Fuzzy Model for Measuring Coupling in Service Oriented Architecture. In *Recent Advances in Computing and Software Systems (RACSS), 2012 International Conference on*. 254-259.

[40] Elhag, A. A. M. and Mohamad R. 2014. Metrics for Evaluating the Quality of Service-oriented Design. In

Software Engineering Conference (MySEC), 2014 8th Malaysian. 154-159.

[41] Dimitoglou, M. J. M. G. 2008. A Service Oriented Architecture Complexity Metric, Based on Statistical Hypothesis Testing.

[42] Stevens, W., Myers, G., and Constantine, L. 1979. Structured Design. In Classics in *Software Engineering,* Y. Edward Nash, Ed. ed. Yourdon Press. 205-232.