Journal of Web Engineering, Vol. 13, No.3&4 (2014) 302--332

A CONCEPTUAL COHESION METRIC FOR SERVICE ORIENTED SYSTEMS

ALI KAZEMI

Faculty of Electrical and Computer Engineering, Shahid Beheshti University G. C., Tehran, Iran
ali.Kazemi@mail.sbu.ac.ir

ALI ROSTAMPOUR

Department of Computer Engineering and Information Technology, Payame Noor University

PO BOX 19395-3697, Tehran, Iran
ali.rostampour@kshpnu.ac.ir

HASSAN HAGHIGHI

Faculty of Electrical and Computer Engineering, Shahid Beheshti University G. C., Tehran, Iran
h_haghighi@sbu.ac.ir

SAHEL ABBASI

Computer Faculty, Islamic Azad University, Kermanshah Branch, Iran
Sahel.abasi@gmail.com

Received March 11, 2013
Revised December 20, 2013

Service conceptual cohesion has an incredible impact on the reusability and maintainability of service-oriented software systems. Conceptual cohesion indicates the degree of focus of services on a single business functionality. Current metrics for measuring service cohesion reflect the structural aspect of cohesion and therefore cannot be utilized to measure conceptual cohesion of services. Latent Semantic Indexing (LSI), on the other hand, is an information retrieval technique widely used to measure the degree of similarity between a set of text based documents. In our previous work, a metric, namely SCD (Service Cohesion Degree), has been proposed that measures conceptual cohesion of services based on the LSI technique. SCD provides a quantitative evaluation to measure how much a service concentrates on a single business functionality. In addition, SCD is applied in the service identification step, i.e., when services are not yet available, and the designer plans for developing services with high cohesion. This paper has two contributions in comparison to our previous work. At first, it resolves two anomalies occurring in our previous method when calculating conceptual relationship between service operations. Secondly, as the main contribution of the paper, it presents details of a theoretical validation and an empirical evaluation of SCD. By using a small-scale controlled study, the empirical evaluation demonstrates that SCD could measure conceptual cohesion of services acceptably.

Keywords: Software Metrics, Service Conceptual Cohesion, Service-Oriented Design Principle, Latent Semantic Indexing
Communicated by: D. Schwabe & E.P. Lim

1 Introduction

Service-orientation brings specific constraints and requirements to the design of software systems [1]. In contrast to earlier paradigms which treat an application as a collection of inter-connected procedures or objects, service-oriented software applications are developed by means of reusable and stateless services which aim to demonstrate autonomy from other services in the system [2]. Although service-orientation and its associated computing paradigm, Service-Oriented Computing (SOC), are making an increasingly popular paradigm for the implementation of enterprise software [3], service-oriented software applications are often designed in an ad hoc manner [1] [4], with little consideration given to the underlying design structures; this issue potentially decreases the maintainability of the developed software.

On the other hand, service–orientation urges services to be reusable regardless of the fact that there might be no direct requirement for service reuse. To increase the chance of a service to accommodate future requirements with minimum development efforts, designers are to use design standards which make every service potentially reusable [2]. As one of these standards, high cohesion increases clarity and comprehension of the design and thus simplifies software maintenance [5]. Moreover, putting related operations in one service improves reusability in different contexts since it enables the service to focus on a single business functionality [6].

Service cohesion can be defined as a degree to which the operations of a service contribute to either a single business functionality or some other semantically meaningful concept, such as an abstraction in the problem domain [6]. In this regard, a cohesive service is a right-level abstraction of concept or feature from the problem domain, usually described by business processes. Such definition is quite vague and makes cohesion as one of the most complex software attributes to be quantified [1]. Therefore, current metrics only measure service cohesion from structural point of view [1, 6, 7] and ignore conceptual aspects while cohesion has a conceptual nature. These metrics mainly use the structural information from the source code, such as references to attributes in methods, to measure cohesion. However, since each service encapsulates specific business functionality, service cohesion should be defined as the level of concentration of a service on a single functionality or semantically related business functionalities. To communicate more easily, we use term “conceptual cohesion” throughout the paper when cohesion is measured considering conceptual aspects.

In [8], we proposed a metric, called SCD (Service Cohesion Degree), to measure the degree of conceptual cohesion in a service. This metric can be used in service identification methods effectively. To achieve our goal, we used the LSI technique. LSI was first used for Information Retrieval (IR) purposes [9]. One of the applications of this technique is the calculation of text cohesion [9]. Based on a powerful mathematical method, called Singular Value Decomposition (SVD), LSI provides a completely automatic approach that compares information units in order to measure conceptual relationships. Thus, utilizing LSI is suitable to define an automatically measurable cohesion metric.

On the other hand, we use business processes in order to measure service conceptual cohesion; this approach is aligned with the nature of service as a business concept. The proposed method uses business processes as input and then computes the relationships between business entities using a matrix structure. By applying both SVD and domain reduction algorithm to this matrix, SCD could be calculated. This paper first resolves two anomalies occurring in our previous method [8] when calculating conceptual relationship between service operations. Then, as its main contribution, it presents details of a theoretical validation and an empirical evaluation of SCD.

SCD is validated theoretically using the property based software engineering measurement framework of Briand et al. [10]. Moreover, it is evaluated empirically using a small scale controlled experiment designed specifically for this research. The results show that there is a significant relationship between SCD values and cohesion values acquired from participants in the experiment.

The rest of this paper is organized as follows. In the next two sections a summary of related work together with preliminaries of this work are provided. Section 4 explains our overall approach to use LSI for the cohesion metric definition. Detailed description of the proposed metric is provided in Section 5, followed by an evaluation in Section 6. Section 7 deals with threats that might have an impact on the validity of our results. Conclusions of the work and future research directions are given in the final section.

2 Related Work

In this section, a brief overview of the metrics proposed for measuring cohesion in object-oriented and service-oriented paradigms is presented.

2.1. Object Oriented Cohesion Metrics

Many metrics have been proposed for the cohesion measurement in the object-oriented design. These metrics could be applied during either high-level or low-level design stage. At the high-level design stage, metrics should use information like classes and method interfaces since in this level only such information is available. Metrics in the low-level design make use of fine-grained information obtainable from algorithm and source code analysis; hence, at the low-level design stage all method-method, method-attribute and attribute-attribute associations could be used. On the other hand, metrics applicable in the high-level design are based on characteristics related to the cohesion at early stages of the development. Class cohesion improvement at the high-level design stage reduces development cost and increases software quality.

Cohesion metrics in the object-oriented paradigm are usually based on shared class attributes. For instance, Kemerer and Chidamber propose a metric, called LCOM1 (LCOM abbreviates for Lack of Cohesion of Methods) [12], that counts the number of method pairs without any shared variable. They also provide another version of this metric, namely LCOM2 [13], which calculates the difference between the number of method pairs that do and do not share instance variables.

Li and Henry propose a graph based approach in which each method is considered as a graph node. There is an edge between two nodes if their corresponding methods have at least one shared variable [14]. Their metric, called LCOM3, is defined as the number of connected components of the graph. For LCOM4 which is an extension of LCOM3, there is an edge between two nodes if one of their corresponding methods invokes the other [15]. In addition, Henderson-Sellers [16] proposes LCOM5 that considers the number of methods referencing each attribute.

In [17] Bieman and Kang proposed two cohesion metrics, namely TCC and LCC. According to TCC, two methods are related if they share the use of at least one attribute. However, based on LCC, two methods are related if they share the use of at least one attribute directly or transitively.

In [18] Weighted Transitive Cohesion measurement (abbreviated as WTCoh) is calculated based on the similarities of class methods. Two methods are similar when the variable sets that they can access are overlapped. This study categorizes the cohesion of class methods to either direct or indirect. The transitivity property has been used to measure the indirect cohesion of classes in this study.

The semantic information shared between the source code elements is the basis of conceptual cohesion measurement through a metric, called C3, in [19]. Comments and identifiers as semantic information in IR methods are used to calculate conceptual cohesion. The research assumes that each method owns a set of explanatory comments. Moreover, the use of meaningful identifiers to implement classes is another pre-assumption in this research. As reviewed above, most metrics in the object-oriented paradigm measure cohesion only from the structural point of view and do not consider its conceptual nature. On the other hand, although C3 is a conceptual cohesion metric, it could not be used in the service identification step since C3 is calculated using source code information which is not available for services in this step, i.e., when services are not yet available. Instead, service is a notion at the business level; hence, the required information can be only obtained using business level artifacts, such as business processes.

Although numerous metrics have been proposed to measure class cohesion, the number of similar researches for computing service cohesion is scarce. These studies are briefly reviewed in the next subsection.

2.2. Service Oriented Cohesion Metrics

In [1, 6] eight categories of service-oriented cohesion are defined: Coincidental, Logical, Temporal, Communicational, External, Implementation, Sequential and Conceptual. Four of these categories (Communicational, External, Implementation and Sequential) are quantifiable using structural constructs; see Table 1 for more details. On the other hand, four other categories, i.e., Coincidental, Logical, Temporal and Conceptual, are called purely semantic cohesion categories because they cannot be mapped directly to the structural constructs of service oriented designs without examining service semantics. In summary, the latter four categories require service semantics while the former four categories could be measured without having any problem-domain-level semantics. This paper shows that conceptual cohesion could be indirectly reflected by the quantifiable cohesion categories. In this way, structural properties of services are utilized to measure service conceptual cohesion.

In [5], service cohesion is measured based on the complexity of data flow between service operations. More precisely, cohesion calculation is based on the complexity of information entities transmitted through data flows. The proposed metric in [5] only calculates service cohesion from structural point of view. Moreover, the complexity of information entities is considered based on the designer's experience.

In [7], a metric, namely DM_IAUM, is defined to measure the cohesion of a system of services. Authors consider DM_IAUM as inversely proportional to the average number of messages in a service because they claim that using smaller number of messages in a service implies higher level of data cohesion. Not only no semantics is examined for measuring the degree of service cohesion in [7], but also the authors do not present any justification or evaluation for their claim.

Two metrics are presented in [20] which calculate sequential and communicational cohesion based on the W3C standard architecture. A service is sequentially cohesive if it has a pair of operations such that the input message of one operation and the output message of the other one have shared elementary or complex entities. In contrast, a service has communicational cohesion if it involves a pair of operations whose input messages (or output messages) have shared entities. Roughly speaking, the similarity between messages as a structural property affects the degree of cohesion.

Table 1 Summary of Cohesion Metrics in the Literature.

Name	Definition	Structural or Conceptual
LCOM1 [12]	The number of method pairs without any shared variable	Structural
LCOM2 [13]	The difference between the number of method pairs that do and do not share instance variables	Structural
LCOM3 [14]	The number of connected components in a graph whose vertices are methods and whose edges link similar methods	Structural
LCOM4 [15]	The LCOM3 metric is extended here by adding one edge between a pair of methods if one of them invokes the other one	Structural
LCOM5 [16]	The number of methods referencing each attribute	Structural
TCC, LCC [17]	Ratio of the number of related method pairs to total number of method pairs in the class. In contrast to TCC, LCC considers indirect relationships as well as direct ones.	Structural
WTCoh [18]	The number of shared data entities used by methods; this metric takes the transitive cohesion into account.	Structural
C3 [19]	This metric uses the semantic information shared between the source code elements (comments and identifiers).	Conceptual
DM_IAUM [7]	The number of system services divided by the total number of used messages	Structural
SIDC [6]	A service is considered to be Communicationally cohesive when all of its operations use common parameter and return types.	Are used to calculate conceptual cohesion indirectly
SIUC [6]	This metric indicates that a service is deemed to be Externally cohesive when all of its operations are invoked by all of its clients.	
SIIC [6]	By this metric, a service is deemed to be Implementation cohesive when all of its operations are implemented by the same implementation elements, such as business process scripts and OO classes.	
TICS [6]	A service is deemed to be Sequentially cohesive when all of its operations have sequential dependencies, i.e., when a post condition/output of a given operation satisfies a precondition/input of the next operation.	
V_{COHES} [5]	The data flow complexity across activities in the service which reflects the functional relevance among these activities.	Structural
AC [21]	This metric takes into account both Entity-Entity and Entity-Activity relationships.	Structural
LoCC and LoCS [20]	Two fine-grained metrics that measure the lack of communicational and sequential cohesion.	Structural

Authors in [21] introduce a metric to measure service cohesion based on the relationship between entities processed by operations of the service. The relationship between a weak entity and its identifying entity has been considered as the strongest relationship. Subtyping, constraint, unary, one-to-one binary, one-to-many binary, and finally, many-to-many binary relationships are considered from the second to seventh levels, respectively.

A brief representation of the mentioned cohesion metrics has been shown in Table 1.

Except metrics of [1, 6], the provided metrics for service cohesion in previous researches have considered this quality attribute from structural point of view. These metrics do not indicate the degree of service concentration on a domain-level concept while a service that implements one domain concept could be easily reused and maintained. Moreover, unlike [1, 6] which measure conceptual cohesion from structural properties of services, we intend to quantify conceptual cohesion based on semantics extracted from business processes directly. In other words, [1, 6] can be only applied when services are already available; instead, the present paper aims at supporting the designer in early stages of the service oriented software development life cycle when she plans for developing services with high cohesion. Consequently, our metric is not comparable with metrics like what introduced in [1, 6] because such metrics (in comparison to ours) tries to meet different goals and be applicable in different situations.

3 Setting the Scene

In this section, we first present the basic concepts related to the service oriented paradigm and referenced throughout the paper. Then we give a brief description of the LSI technique.

3.1. Preliminaries

We describe the required concepts using a real sales scenario from the sales department of a “sales and goods distribution company”, called “Behpakhsh” [25, 29]. We will use the same scenario to show the applicability of the proposed metric in subsection 6.2. The following paragraphs describe the scenario:

To attract more orders from customers, the company sends its agents to customers’ locations to receive new orders. The sales scenario begins with customers’ orders. Every agent delivers her orders to the boss of the sales department at the end of each working day. Now, these orders should be investigated and processed. This process which is done by the sales department includes the following steps for each order:

1. Every good may involve some discounts in terms of the quantity ordered by the customer. Good suppliers notify the inventory department of discount policies. Thus, the next step is to receive such information from the current inventory system and calculate goods discounts.
2. The available quantity of each ordered good is compared with the ordered quantity of that good.
3. Customer’s ID, cost of ordered goods, discounts and other details about the order are considered, and then, based on these information, the total price is calculated.
4. The credit information of the customer is received from the accounting department and is compared with the order total price. Credit information is maintained in the current accounting system.
5. The customer is notified of the final price and confirms/discards the order. If she/he affirms the order, subsequent steps will be followed.

6. At this step, the order process is terminated, and the sales department boss can issue a sale invoice for the customer.
7. Orders are ready to be carried when affirmed by customers. To carry the ordered goods, the sales department has some vehicles whose drivers are responsible to deliver goods to customers and receive cheques from them.
8. The boss of the sales department must schedule carries. Also, for every carry which may include one or more sale invoice, he issues an inventory bill (or draft) to the driver. The cost of the carry is calculated according to the distance and other related parameters.
9. The driver goes to the inventory, and after giving the inventory bill to the storekeeper, loads the cargo up.
10. The carrier delivers the goods to the customers, gets cheques from them and gives cheques to the financial unit. The financial unit, then, registers the related receipt document in the accounting system.

In order to simplify the problem description and its scope in both this subsection and subsection 6.2, we restrict this business scenario to main and general (and yet real) issues and ignore detailed exceptions (for example, when a customer does not pay whole of the calculated price, returns some goods, does not affirm the order, changes the order, or cancels it after conformation). In addition, we concentrate on the activities of the sales department.

In our approach, we represent business models through the notion of the CRUD matrix. The CRUD matrix of the above example is shown in Figure 1. A CRUD matrix is defined as $M_{RowNum \times ColumnNum}$, where RowNum is the number of rows, and ColumnNum is the number of columns of the matrix. Business Entities (BEs) are placed in the columns, and Elementary Business Processes (EBPs) are placed in the rows of this matrix. A BE is a dominant information entity with an associated data model and an associated behaviour model in the context of a process scope. Formally, we define an EBP as a set of pairs like (BE, OP) where BE is a business entity processed by the EBP through an operation OP. OP can be "C", "R", "U", or "D" where C, R, U, and D refer to Create, Read, Update and Delete operations, respectively [23]. In this way, each cell indicates the relationship type (C, R, U, or D) between the corresponding BE and EBP; see Figure 1.

The Kth cluster of the CRUD matrix can be defined as cluster K = $\{(EBP_i, BE_j) \mid i=l_1..h_1, j=l_2..h_2\}$, where $1 \le l_1 \le h_1 \le$ RowNum, and $1 \le l_2 \le h_2 \le$ ColumnNum. Each cluster plays the role of a service identified using the CRUD matrix; the cluster notion in our method corresponds to the service interface notion in the W3C standard [20]. Figure 1 indicates four services corresponding to four clusters with different colours. For instance, the left most cluster shown by blue colour indicates a service with 4 BEs and 4 EBPs. In this service, for example, “Receive Order” as an EBP reads (R) “Customer” BE and creates (C) “Order” BE.

Before ending this subsection, it is worth justifying why the CRUD matrix is suitable for service measurement. One of the important applications of metrics in service oriented projects is in the service identification phase. On the other hand, when services are supposed to be identified automatically, the metrics should be automatically measurable. Thus, the business model from which services are identified should make automatic architectural decisions possible. Especially, the definition of quantitative metrics on this model should be feasible.

Using matrices helps design a computational model to improve the formalization of the metrics for service identification. The reason is simply as follows: a matrix is often a good choice for defining concisely the relationships between two different items. Moreover, as the content structure of the matrix is constant, it is possible to analyse it in different ways. The types of analysis that can be performed on a matrix include arithmetic and cross-checking operations (For example, Traceability and Coverage checking for rows or columns). However, in the context of this work we shall focus on an operation, called clustering, in which each cluster plays the role of a service identified using the CRUD matrix. If the rows and columns of a matrix are rearranged, clusters of cells may emerge. A cluster of cells indicates that the entries in the rows and columns have something in common and have an affinity for each other (for example, "rows 1, 2, 3, 4" and "columns 1, 2, 3, 4" in Figure 1).

BE / EBP	Customer	Credit	Account Receivable Note	Order	Discounts	Invoice	Shipping Schedule	Draft	Inventory	Warehouse Voucher
Add Customer	C	C								
Add an Account Receivable Note	R	U	C			R				
Check Credit	R	R		R						
Receive Order	R			C						
Calculate Discounts				R	R					
Check Inventory				R					R	
Calculate Price				R	R					
Add Discounts					C					
Issue Invoice	R	R		R		C				
Schedule Shipping						R	C			
Issue Draft						R	R	C		
Add an Item									C	
Add a Warehouse Voucher	R					R			U	C

Figure 1 The CRUD Matrix for Sales Department Scenario [25].

3.2. Overview of Latent Semantic Indexing

In the traditional approach to IR, a vector space is defined for a collection of documents such that each dimension of the space is a term occurring in the collection, and each document is specified as a vector with a coordinate for each term occurring in the documents. The value of each coordinate is a weight assigned to the corresponding term, or a measure of the importance of that term in characterizing the given document and distinguishing it from other documents in the given collection [9]. LSI is a vector space approach that attempts to capture term-term statistical relationships. In LSI, the document space in which each dimension is an actual term occurring in the collection is replaced by a much lower dimensional document space called k-space (or LSI space) in which each dimension is a derived concept (a conceptual index), called a LSI factor. Hence, LSI factors are information rich in the sense that they capture the term-term relationships that ordinary term-based document space does not. The following steps describe the LSI technique in more details:

1. A matrix is constructed in which each row corresponds to a term that occurs in a document, and each column corresponds to a document. Each element (*m, n*) in the matrix corresponds to the weight of term *m* in document *n*.
2. A weight is assigned to each term in the document. Different weighting methods are proposed in [9]. The simplest weighting model could be achieved by counting the number of term occurrence in a document. In this paper we propose a new weighting model since IR related models in [9] could not be applied directly in our approach.
3. Using the SVD method [9], the term-document matrix **A** is decomposed to three matrices **S**, **T** and **D** which contain the information on terms, unique values and documents, respectively. The original matrix **A** could be obtained using the formula $\mathbf{A}=\mathbf{TSD}^{\mathbf{T}}$ where $\mathbf{D}^{\mathbf{T}}$ is the transpose of matrix **D**.
4. The matrices **T**, **S**, and **D** are reduced to k domains. The value for k is considered 2 in our approach, as suggested in [9]. With k=2, the reduced matrix $\mathbf{M}= \mathbf{T_2S_2(T_2S_2)^T}$ can be created ($\mathbf{T_2}$ and $\mathbf{S_2}$ are resulted from the reduction of **T** and **S** to 2 domains, respectively). Here, **M** is a term-term matrix which indicates relationships between terms. Similarly, since **A** is BE-EBP matrix in our approach, **M** will be in fact the BE-BE matrix, or in other words, will demonstrate the relationships between business entities.

The LSI technique uses the term co-occurrence to obtain relationships between terms. Two terms *a* and *b* co-occur if there is at least one document which contains both of them. We say that, in this case, *a* and *b* have a first order co-occurrence. Similarly, a second order co-occurrence path between two terms *a* and *b* is made through a term *c* such that *a* co-occurs with *c*, and *b* co-occurs with *c*, too. Obviously, there may be more than one second order co-occurrence path between *a* and *b*. Higher order co-occurrences are defined in a similar form; LSI considers all co-occurrence orders and paths.

4 Our Overall Approach to Measure Service Conceptual Cohesion

To achieve a method to measure the conceptual cohesion of services, we map the mentioned concepts in subsection 3.2 to SOA counterparts. Similar to LSI, we define a $m \times n$ Business Entity - Elementary Business Process (BE-EBP) matrix, where *m* is the number of business entities, and *n* is the number of elementary business processes. For example, each EBP in the CRUD of Figure 1 is placed in one of the columns of the BE-EBP matrix. Similarly, each BE is placed in one of the rows of this matrix. Having determined the columns and rows of the BE-EBP matrix, we should give values to its cells using a weighting model. We will use a specific weighting model in this paper. This model must be able to reflect semantic relationships that exist in business processes. In the following paragraphs, key aspects of conceptual cohesion are investigated for understanding their effects on our weighting approach.

In the LSI technique, term co-occurrence information is used; two terms that co-occur several times in documents are more related. We call two business entities are related if both are accessed or processed by at least one business process. The relationship degree between business entities can be specified based on the type of the actions performed by processes on these entities. For example, the action of creating a business entity has a higher affinity to the business process in comparison to a reading action. Generally, we can consider priorities C=4>U=3>D=2>R=1 for actions which are performed on business entities, where C, R, U, and D refer to Create, Read, Update and Delete, respectively [9]. Using this idea, we propose a weighting model that enables us to calculate the relationship degree between business entities.

To explain the weighting model used to calculate values in the BE-EBP matrix, consider the CRUD matrix shown in Figure 1. Values of the first column of the BE-EBP matrix that corresponds to the first EBP, i.e., Add Customer, has been shown in Figure 2. Since each of "Customer" and "Credit" is created one time by "Add Customer", their associated weights in the corresponding cells are 4. Other BEs are not accessible by "Add Customer" at all; hence, their associated weight in the matrix is zero.

BE	EBP		
Customer	4		
Credit	4		
Account Receivable Note	0		
Order	0		
Discounts	0		
Invoice	0		
Shipping Schedule	0		
Draft	0		
Inventory	0		
Warehouse Voucher	0		

Figure 2 The BE-EBP Matrix.

The above matrix can be filled for all of EBPs as well. Now, performing SVD, three matrices **T**, **S**, and **D** can be obtained, where $\mathbf{A}=\mathbf{TSD}^{\mathbf{T}}$ (**A** is the BE-EBP matrix). Then, considering k=2, the reduced matrix $\mathbf{M}= (\mathbf{T}_2\mathbf{S}_2)(\mathbf{T}_2\mathbf{S}_2)^{\mathbf{T}}$ is in fact the BE-BE matrix and indicates relationships between business entities. Values of the cells in this matrix are not normalized and even can be negative values. Since negative values are not meaningful as relationships degrees, the least negative value, e.g., "-a" (where $a\geq 0$), should be first added by value "a" to be equal to zero; zero means lack of relationship. To preserve initial difference between values, other cells of the matrix are then added by "a" as well. Finally, to normalize values, we multiply this matrix by **1/max** where max is the maximum value in the matrix.

In summary, using the LSI concept as above, it is possible to show semantics existing in business processes in the form of BE-BE matrix. In the next section, we introduce SCD metric which is calculated based on this matrix.

5 The Proposed Service Cohesion Metric

At first, it should be mentioned that the proposed metric is defined based on an absolute scale where the metric takes values in the range of 0 to 1. Value 1 indicates the strongest possible cohesion while value 0 indicates lack of cohesion. In our previous work [8], to formulize our metric, we used a graph-based approach as follows: Consider service S with a set of operations $\boldsymbol{O} = \{\boldsymbol{O_1}, \boldsymbol{O_2}, \ldots, \boldsymbol{O_m}\}$. Each operation $\boldsymbol{O_i}$ of service S accesses a set of business entities that are shown as $\boldsymbol{BE_j} = \{ \boldsymbol{BE_{j,1}}, \boldsymbol{BE_{j,2}}, \ldots, \boldsymbol{BE_{j,n}}\}$. For each pair of operations $\boldsymbol{O_i}$ and $\boldsymbol{O_j}$ in service S, a complete graph G = (**V**, **E**) is formed such that $\mathbf{V} = \boldsymbol{BE_i} \cup \boldsymbol{BE_j}$. A value is then assigned to each edge *e* in set *E* that shows the degree of relationship between those business entities connected by *e*. This degree of relationship could be obtained from the BE-BE matrix described in the previous section. The conceptual relationship between two operations $\boldsymbol{O_i}$ and $\boldsymbol{O_j}$ is then calculated as follows:

$$ORD(i,j) = \begin{cases} \dfrac{\sum_{BE_{i,p} \in V,\, BE_{j,q} \in V,\, p \neq q} ([BE-BE])_{p,q}}{|V| \times (|V|-1)/2} & |V| > 1 \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

where $\boldsymbol{BE} - \boldsymbol{BE_{p,q}}$ is the degree of relationship between business entities $\boldsymbol{BE_{i,p}}$ and $\boldsymbol{BE_{j,q}}$ based on the BE-BE matrix. Also, |**V**| is the cardinality of set **V**.

This graph-based approach suffers from the following two anomalies:

1. If there is a common BE in $\boldsymbol{BE_i}$ and $\boldsymbol{BE_j}$, only one vertex is considered in the mentioned graph for that BE. On the other hand, we did not consider loops (edges from a vertex to itself) in the graph. In this way, the effect of this common business entity is discarded when calculating conceptual relationship between two operations $\boldsymbol{O_i}$ and $\boldsymbol{O_j}$ while common entities should obviously increase the degree of relationship between operations.
2. The relationship between two entities should be considered only when they belong to different operations, but we are considering the relationship between two business entities accessed by only one of the operations $\boldsymbol{O_i}$ and $\boldsymbol{O_j}$, too.

To resolve the second mentioned anomaly, we propose to use a bipartite graph G = (**V**, **E**), instead of the previous complete graph, such that $\mathbf{V} = \boldsymbol{BE_i} \cup \boldsymbol{BE_j}$, and every edge of **G** is of the form $\{a,b\}$ with $a \in \boldsymbol{BE_i}$ and $b \in \boldsymbol{BE_j}$. To resolve the first mentioned anomaly, when ***EB*** exists in both $\boldsymbol{BE_i}$ and $b \in \boldsymbol{BE_j}$ we consider two copies of it in V, one plays its role in $\boldsymbol{BE_i}$ and the other one plays its role in $\boldsymbol{BE_j}$. Similar to the previous method, a value is then assigned to each edge *e* in set **E** that shows the degree of relationship between those business entities connected by *e*.

Now, the conceptual relationship between two operations $\boldsymbol{O_i}$ and $\boldsymbol{O_j}$ is calculated as follows:

$$ORD(i,j) = \begin{cases} \dfrac{\sum_{BE_{i,p} \in BE_i,\, BE_{j,q} \in BE_j} ([BE-BE])_{p,q}}{(|BE_i| \times |BE_j|)} & BE_i \neq \emptyset \text{ and } BE_j \neq \emptyset \\ 0 & \text{Otherwise} \end{cases} \quad (2)$$

Having the conceptual relationship degree between operations in place, the degree of cohesion of a service is defined as follows:

$$SCD(S) = \begin{cases} \dfrac{\sum_{o_i \in O,\, o_j \in O,\, i>j} ORD(i,j)}{m \times (m-1)/2} & m > 1 \\ 1 & m \leq 1 \end{cases} \quad (3)$$

Where, m and **O** is the number and the set of operations in service S, respectively. In summary, the measurement of the proposed metric involves the following functions:

```
Function SCD (S)
Op_Relationship=0
        // O  is the set of operations of S
if m < =1 then
        SCD = 1
else
begin
        Op_Relationship=0
        for i=1 to m
          for j=1 to m
               if i>j then
                  Op_Relationship += ORD(i,j)
        SCD = Op_Relationship / (m × (m − 1)/2)
end
End SCD
```

```
Function ORD (i,j)
BE_Relationship=0

Create bipartite graph G=(V,E) where

V = BE_i ∪ BE_j
if Not(BE_i ≠ ∅ and BE_j ≠ ∅) then
     ORD(i,j)=0
else
begin
     BE_Relationship=0
     for p=1 to #BE_i
        for q=1 to #BE_j
            BE_Relationship += BE-BE(p,q)
     ORD = BE_Relationship / (|BE_i| × |BE_j|)
end
End ORD
```

6 Empirical Validation of the Proposed Metric

In this section, an empirical validation of SCD is provided. At first, the applicability of the proposed metric is shown by applying it on a case study. Then, using the cohesion principles, it is shown that our metric is a valid measure of cohesion from the measurement theory point of view. Finally, a controlled experiment is done, and the results are analysed.

6.1. The Used Model for Validation of SCD

The Model for Empirical Validation of SCD based on [24] is shown in Figure 3.

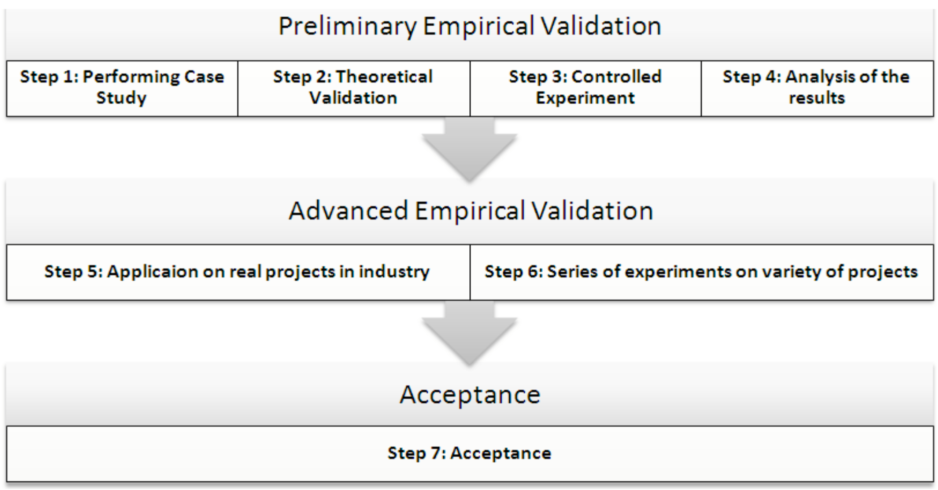

Figure 3 The Model for Empirical Validation of SCD.

As it can be seen in Figure 3, the first stage for empirical validation aims at preliminary empirical validation. Preliminary empirical validation is divided into four steps: Performing Case Studies, Theoretical Validation, Controlled Experiment and Analysis of the results. The second stage includes the application of the metric on real projects from industry. After a series of experiments, the results should be analysed and compared for metric acceptance. Performing experiments in industry is going to be done as a future work of this paper once the authors can obtain relevant information from a real service-oriented project. Thus, only the first stage in the mentioned model is performed in this paper. On the other hand, internal validity of a study must be done before doing its external validity, and controlled experiments have better supports for controlling instrumentation effects which can affect internal validity of a study [1], [24]. In other words, laboratory validation (controlled experiment) allows us to have a much more control on parameters such that we can control the effects of these parameters on software development.

In the following subsections, we perform four steps of the preliminary empirical validation on our metric. More precisely, first, applicability of the proposed metric is shown based on a case study. Then, using the cohesion principles, it is shown that our metric is a valid measure of cohesion from the measurement theory point of view. Finally, a controlled experiment is done, and the results are analysed.

6.2. Case Study

Our case study is related to the processes of the sales department of "Behpakhsh" company [25, 29] as the most important department of this company. The CRUD matrix and the steps of that part of the sales scenario concerned in this paper have already been presented in subsection 3.1. This case study was chosen due to the following reasons:

1. *Company activities and its sales scenario are typical, clear and well-known.*
2. *Since the mission of the company is to offer and give suitable services to customers, SOA can be a proper technology for this company. Moreover, the company has several software systems with diverse platforms; hence, its IT decision makers would rather integrate existing systems based on the service oriented technology instead of buying new expensive systems.*
3. *Authors are familiar with the procedure of the sales scenario because it has been used in several papers and studies in the ASER Group [26]. Also, one of the authors participated in the enterprise architecture and IT master plan projects of this company. Thus, she knows the business of the company well.*

"Behpakhsh" has 6 sub-companies, 26 branches, more than 3000 official employees and more than 90000 sales agents that are directly or indirectly involved in the selected sales scenario. Required information about the selected scenario was acquired from the IT manager of this company. In fact, only the IT manager and one of the authors perform the case study together. Persons who are directly involved in this scenario do not have any technical expertise and are only familiar with the software system they use. They are also experts in the "sales and distribution" business area.

In Figure 1, four identified services were shown as clusters with different colours on the CRUD matrix. We assign numbers to these services beginning with 1, and from top and left. Figure 4 indicates the business entities accessed by the operations of service 1 (the blue cluster):

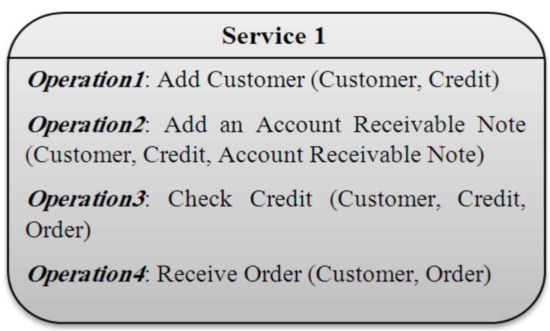

Figure 4 The Interface for Service 1.

At this time, the BE-EBP matrix could be formed and filled using the method described in Section 4; the result is shown in Table 2.

Once the BE-EBP matrix is formed, the SVD algorithm could be applied. In order to run SVD, we implemented the measuring process in Matlab software version 7.6.0.324. The BE-BE matrix is then constructed. The resulting matrix is shown in Table 3. This matrix is normalized, and its negative values are replaced by zero.

Table 2 The BE-EBP Matrix.

BE \ EBP	Add Customer	Add an Account Receivable Note	Check Credit	Receive Order	Calculate Discounts	Check Inventory	Calculate Price	Add Discounts	Issue Invoice	Schedule Shipping	Issue Draft	Add an Item	Add a Warehouse Voucher
Customer	4	1	1	1	0	0	0	0	1	0	0	0	1
Credit	4	3	1	0	0	0	0	0	1	0	0	0	0
Account Receivable Note	0	4	0	0	0	0	0	0	0	0	0	0	0
Order	0	0	1	4	1	1	1	3	1	0	0	0	0
Discounts	0	0	0	0	1	0	1	4	0	0	0	0	0
Invoice	0	1	0	0	0	0	0	0	4	1	1	0	1
Shipping Schedule	0	0	0	0	0	0	0	0	0	4	1	0	0
Draft	0	0	0	0	0	0	0	0	0	0	4	0	0
Inventory	0	0	0	0	0	1	0	0	0	0	0	4	3
Warehouse Voucher	0	0	0	0	0	0	0	0	0	0	0	1	4

Table 3 The BE-EBP Matrix.

BE \ BE	Customer	Credit	Account Receivable Note	Order	Discounts	Invoice	Shipping Schedule	Draft	Inventory	Warehouse Voucher
Customer	1									
Credit	1	1								
Account Receivable Note	0.60	0.65	1							
Order	0.64	0.73	0.48	1						
Discounts	0.42	0.47	0.37	0.53	1					
Invoice	0.65	0.69	0.47	0.43	0.32	1				
Shipping Schedule	0.34	0.34	0.31	0.30	0.29	0.32	1			
Draft	0.33	0.33	0.31	0.30	0.29	0.32	0.29	1		
Inventory	0.48	0.42	0.35	0.00	0.05	0.50	0.33	0.32	1	
Warehouse Voucher	0.46	0.42	0.35	0.07	0.11	0.47	0.32	0.32	0.86	1

The cohesion of the first service in Figure 1 (with blue colour) is calculated as follows. This service has four operations: Add Customer ($\boldsymbol{O_1}$) Add an Account Receivable Note ($\boldsymbol{O_2}$), Check Credit (($\boldsymbol{O_3}$) and Receive Order ($\boldsymbol{O_4}$). The operation set and business entities related to each operation are as follows:

$\boldsymbol{O} = \{\boldsymbol{O_1}, \boldsymbol{O_2}, \boldsymbol{O_3}, \boldsymbol{O_4}\}$

$\boldsymbol{BE_1}$ = {***Customer, Crecit, Account Receivable Note***}

$\boldsymbol{BE_2}$ = {***Customer, Crecit***}

$\boldsymbol{BE_3}$ = {***Customer, Crecit, Order***}

$\boldsymbol{BE_4}$ = {***Customer, Order***}

In order to obtain the conceptual cohesion of this service, function SCD written in section 5 should be executed. Then, ORD function is invoked for each pair of operations. In function ORD, for each pair of operations, a bipartite graph is constructed. For example, this graph for operations $\boldsymbol{O_1}$ and $\boldsymbol{O_2}$ is depicted in Figure 5. Set **V** is formed with the union of business entities related to $\boldsymbol{O_1}$ and $\boldsymbol{O_2}$. Of course, for every shared business entity, i.e., Customer and Credit, a distinct copy is considered in the graph.

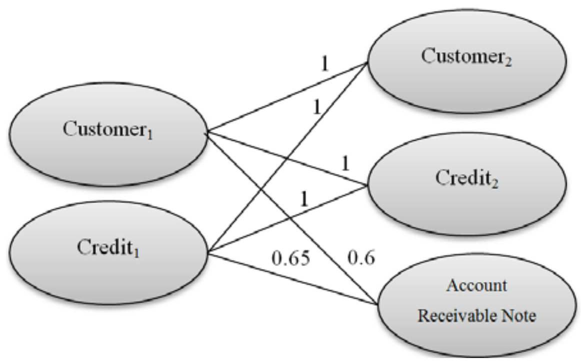

Figure 5 Business Entities Graph for Operations O_1 and O_2.

The label of each edge of the mentioned graph shows the degree of conceptual relationship between each pair of business entities. Now, the conceptual relationship between $\boldsymbol{O_1}$ and $\boldsymbol{O_2}$ is calculated as follows.

$$\mathbf{ORD(1,2)} = \frac{\mathbf{BE_{Relationship}}}{(|\mathbf{BE_1}| \times |\mathbf{BE_2}|)} = \frac{\mathbf{1+1+1+1+0.60+0.65}}{\mathbf{6}} = \mathbf{0.87}$$

The result of calculations for other pairs of operations is shown in Table 4:

Table 4 ORD Values for Service 1.

	O_1, O_2	O_1, O_3	O_1, O_4	O_2, O_3	O_2, O_4	O_3, O_4
ORD	0.87	0.89	0.84	0.78	0.74	0.83

Finally, the conceptual cohesion of this service is calculated as follows:

$$SCD(S_1) = \frac{0.87 + 0.89 + 0.84 + 0.78 + 0.74 + 0.83}{6} = 0.82$$

Table 5 contains the calculated conceptual cohesion values for all identified services:

Table 5 SCD Values for All Identified Services.

Service / Metric	Proposed Cohesion Metric Value (SCD)
S_1	0.82
S_2	1
S_3	0.66
S_4	0.52

6.3. Theoretical Validation

There are different methods for metric validation from theoretical point of view. Some of them have subjective nature whereas others have a basis from axiomatic or calculation theory. Among them, the property-based software engineering measurement framework [10] has been widely used in the literature [1], [6]; hence, we use it to validate SCD.

Property1: Non-negativity and Normalization is satisfied since the metric can only take values between 0 and 1.

Property 2: Null Value is satisfied since SCD is 0 when the degree of conceptual relationship between business entities accessed by the operations of a service is 0.

Property 3: Monotonicity is satisfied because the overall cohesion is not decreased by adding a new BE to a set of BEs that are accessed by a specific pair of operations.

Property 4: Cohesive Modules is satisfied because by joining two unrelated services, the resulting cohesion would not be bigger than the maximum of original cohesion values. Roughly speaking, unrelated services include unrelated operations, i.e., those operations accessing unrelated business entities; therefore, the degree of the conceptual relationship (which we calculate by ORD function) between operations of two unrelated services cannot be greater than the degree of the conceptual relationship between operations of each of them.

Regarding the above illustration, SCD satisfies all of the cohesion properties, and therefore, it is a valid measure of cohesion from theoretical point of view.

6.4. Controlled Experiment

Although the proposed metric is theoretically valid, it has not been yet shown that there is a meaningful relation between metric values and empirical values of cohesion. Therefore, we must ensure that SCD is well explaining the cohesion of a service. This subsection presents a controlled experiment that authors have conducted to empirically validate SCD.

According to [6], there are several steps to perform an experiment. These steps were done in turn as follows:

6.4.1. Objective of Study

The objective of this study is an empirical analysis of SCD in order to illustrate that this metric shows the degree of conceptual cohesion well.

6.4.2. Hypotheses

The definition of a clear hypothesis is a crucial precondition for conducting an experiment. A hypothesis is a tentative assumption made in order to draw out and test its empirical consequence. Our hypothesis is as flows:

"SCD is a proper metric for measuring the degree of conceptual cohesion of a service at the service design stage".

6.4.3. Experimental Protocol

The procedure of the experiment and the way of controlling data are explained in this subsection.

6.4.3.1. Selecting Variables

In experimental studies, there are two types of variables, Dependent and Independent, which are both evaluated by the hypotheses test. In this paper, the CRUD matrix and identified services are independent variables, and conceptual cohesion is the dependent variable which varies by modifying the CRUD matrix clustering or modifying services.

6.4.3.2 Selecting Experts

In total, 10 experts, including 4 software engineering and industry practitioners, 4 PhD students and 2 MSC students in software engineering, participated in the study. All participants were unpaid volunteers who had personal and/or professional interest in SOC. Selected PhD and MSC students had passed 50 hours of ULS (Ultra Large Scale System) course with focus on SOA and business processes. Also, their thesis is in the SOA field. Software engineering and industry practitioners, PhD students and MSC students have 8-20, 5-7 and 2-4 years experience in their research and skill area, respectively. Finally, all of the experts have good experience in software development, using SOA technology (web services), SOA methodology and software patterns including analysis, design, architecture and process patterns. Table 6 provides the profile of all experts who participated in the experiment.

6.4.3.3 Planning the Experiment

To perform the experiment, apart from the sales scenario described earlier as our case study, we used a typical library scenario. Due to the space limitation, we do not give the details of this scenario here. Moreover, in order to have more cases and also to regard a completely different business area, we considered a master business process of Iran Ports and Maritime Organization (PMO) [30]. This master process, called “maritime transport infrastructure development”, consists of two process groups:

Table 6 Experts' Profile.

Research-Skill Area	Affiliation	Major	Degree	Age
SOA Quality Attributes, SOA Metrics	Shahid Beheshti University	Information Technology Engineering	MSC student	24
Software and Process Metrics	Tarbiat Modares University	Information Technology Engineering	MSC student	26
Contributed to the development of an automatic service-oriented software architecture tool	Ferdowsi University of Mashhad	Software Engineering	PhD student	28
Software Product Line Engineering	Shahid Beheshti University	Software Engineering	PhD student	35
Software Architecture	Shahid Beheshti University	Software Engineering	PhD student	31
Semantic Web Services Issues in SOA	Shahid Beheshti University	Software Engineering	PhD student	31
Software Architecture	Shahid Beheshti University	Software Engineering	PhD	55
Programming	Fanavaran Tehran Co (A software development company)	Software Engineering	MSC	28
Designing, Implementing and Deploying Software Applications	Nama Samaneh Iranian – Maham Holding (A software development company)	Software Engineering	MSC	36
Designing, Implementing and Deploying Software Applications	Peykasa (A software development company)	Software Engineering	MSC	30

1. Management of maritime training centers which include the following process:
    a. License issue for training centers
    b. License issue for performing training courses
    c. Training centers auditing
    d. Confirming the start of a training course
    e. Inspection of the performance of a training course

f. Cancelling the license of a training center

2. maritime tests management

    a. Test registration

    b. Performing tests

    c. Test evaluation

    d. Revising test results

    e. Test calendar preparation

    f. Maintaining and updating questions bank

Two different CRUD matrices (each matrix incudes 4 clusters or identified services) from the sales scenario, two matrices (each matrix incudes 3 clusters) from the library scenario, and two other matrices (one matrix incudes 3 clusters and the other one involves 4 clusters) from the PMO scenario were given to the experts. These six CRUD matrices which totally include 21 identifies services are shown in Figure 6.

Service	BE / EBP	customer	Credit	Account receivable note	Order	Discounts	Invoice	Shipping schedule	Draft	Inventory	Warehouse voucher
1	Add Customer	C	C								
	Add an Account receivable note	R	U	C			R				
	Check Credit	R	R		R						
2	Receive order	R			C						
	Calculate discounts				R	R					
	Check inventory				R					R	
	Calculate price				R	R					
	Add discounts					C					
3	Issue invoice	R	R		R		C				
	Schedule shipping						R	C			
	Issue draft						R	R	C		
4	Add an Item									C	
	Add a warehouse voucher	R					R			U	C

Service	BE / EBP	customer	Credit	Account receivable note	Order	Discounts	Invoice	Shipping schedule	Draft	Inventory	Warehouse voucher
5	Add Customer	C	C								
	Add an Account receivable note	R	U	C			R				
	Check Credit	R	R		R						
	Receive order	R			C						
6	Calculate discounts				R	R					
	Check inventory				R					R	
	Calculate price				R	R					
	Add discounts					C					
7	Issue invoice	R	R		R		C				
	Schedule shipping						R	C			
8	Issue draft						R	R	C		
	Add an Item									C	
	Add a warehouse voucher	R					R			U	C

Service	BE / EBP	Book Title	Book Copy	Loan	Book Copy On Loan	Patron	Reservation
9	Search for book title	R	R				
	Print book title report	R					
	Update book information	U	U				
	Enter new book information	C	C				
	Remove book from library	D	D				
10	Check out books		U	C	C		U
	Delete book copy information		D				
	Check in book			U	U		
	Print loaned book reports			R	R		
	Remove old loan information			D	D		
11	Update Patron information					U	
	View overdue books				R		
	View reservations						R
	Correct reservation						U
	View Patron information					R	
	Reserve book						C
	Remove old reservation information						D

Service	BE / EBP	Reservation	Patron	Book Title	Book Copy	Loan	Book Copy On Loan
12	Reserve book	C					
	View reservations	R					
	Search for book title			R	R		
	Correct reservation	U					
13	Update Patron information		U				
	Enter new book information			C	C		
	View Patron information		R				
	Delete book copy information				D		
	Remove book from library			D	D		
	Update book information			U	U		
14	Remove old loan information					D	D
	Check out books	U			U	C	C
	Check in book					U	U
	Print loaned book reports					R	R
	Print book title report			R			
	View overdue books						R
	Remove old reservation information	D					

Service	BE / EBP	License issue information	License issue for performing courses information	Training centers auditing information	Test performance supervision information	Test	Test calendar	Questions bank
15	License issue for training centers	C						
	License issue for performing training courses	R	C					
16	Training centers auditing	R	R	C				
	Confirming the start of a training course		R	R	C			
	Inspection of the performance of a training course		R	R	U			
	Cancelling the license of a training center	U	R	R	R			
17	Test registration					C	R	
	Performing tests					U	R	R
	Test evaluation					U	R	R
	Revising test results					U	R	R
	Test calendar preparation						C	
18	Maintaining and updating questions bank							C

Service	BE / EBP	License issue information	License issue for performing courses information	Training centers auditing information	Test performance supervision information	Test	Test calendar	Questions bank
19	License issue for training centers	C						
	License issue for performing training courses	R	C					
	Training centers auditing	R	R	C				
	Confirming the start of a training course		R	R	C			
20	Inspection of the performance of a training course		R	R	U			
	Cancelling the license of a training center	U	R	R	R			
	Test registration					C	R	
	Performing tests					U	R	R
	Test evaluation					U	R	R
21	Revising test results					U	R	R
	Test calendar preparation						C	
	Maintaining and updating questions bank							C

Figure 6 Identified Services from the Sales, Library and PMO Scenarios.

Regarding different services which had been extracted from one scenario, experts could compare them from cohesion point view. First, we explained working procedures of processes and answered to related questions raised by experts. Then, experts were asked to examine services, their operations and input/output messages of each service operation. After studying services, each expert evaluated them from conceptual cohesion point of view and gave them a rank; finally, they assigned a number between 0 and 10 as conceptual cohesion to each service. Questionnaires were collected after 2 hours for further evaluations.

6.4.3.4 Statistical Analysis

In order to compare the results of the collected questionnaires and the results obtained by applying SCD, a statistical analysis was applied. The analysis is described below:

Correlation coefficient is a measure which describes direction and value of a relation between two variables. In other words, the correlation coefficient is a measure of the ability of one variable to predict the value of another variable. Since the distribution of data was not normal, we decided to use a nonparametric statistical test, namely the Spearman's rank-difference correlation coefficient (*rs*). Spearman's *rs* is a nonparametric statistics that is used to show the relationship between two variables which are expressed as ranks [27]. Using Spearman's correlation coefficient, we can show that SCD is correlated to each experts' provided ratings individually. Our null hypothesis is:

H0: "there is no correlation between SCD with the experts' provided rating"

The probability of the null hypothesis to be rejected is controlled by one confidence level: α_1=0.05. The decision rule used for rejecting the null hypothesis is:
α_1: reject H0 if *rs* ≥ 0.535.

Value “0.535” was selected based on the confidence level and the number of subjects [27]. Table 7 shows the correlation coefficient for 10 subjects (or experts). Each entry shows the correlation coefficient between SCD and subject *i* (i=1,2,…,10) referred as S_i (i=1,2,…,10) in a pairwise manner. As illustrated in Table 7, values resulted by SCD have high correlation coefficient with values provided by experts. The correlation coefficient is only low for the eighth participant. It is worth to mention that the correlation coefficients between other participants and this participant are relatively low, too.

Table 7 Correlations of the SCD Metric and Expert Values.

	SCD	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10
SCD	1.000	.851(**)	.839(**)	.809(**)	.626(**)	.854(**)	.679(**)	.834(**)	.421	.829(**)	.857(**)
S1	.851(**)	1.000	.819(**)	.889(**)	.763(**)	.904(**)	.732(**)	.874(**)	.258	.855(**)	.870(**)
S2	.839(**)	.819(**)	1.000	.779(**)	.603(**)	.764(**)	.745(**)	.823(**)	.276	.780(**)	.777(**)
S3	.809(**)	.889(**)	.779(**)	1.000	.674(**)	.841(**)	.798(**)	.842(**)	.197	.785(**)	.814(**)
S4	.626(**)	.763(**)	.603(**)	.674(**)	1.000	.795(**)	.579(**)	.754(**)	.174	.777(**)	.703(**)
S5	.854(**)	.904(**)	.764(**)	.841(**)	.795(**)	1.000	.742(**)	.880(**)	.297	.846(**)	.834(**)
S6	.679(**)	.732(**)	.745(**)	.798(**)	.579(**)	.742(**)	1.000	.655(**)	-.130	.645(**)	.689(**)
S7	.834(**)	.874(**)	.823(**)	.842(**)	.754(**)	.880(**)	.655(**)	1.000	.247	.845(**)	.852(**)
S8	.421	.258	.276	.197	.174	.297	-.130	.247	1.000	.169	.188
S9	.829(**)	.855(**)	.780(**)	.785(**)	.777(**)	.846(**)	.645(**)	.845(**)	.169	1.000	.831(**)
S10	.857(**)	.870(**)	.777(**)	.814(**)	.703(**)	.834(**)	.689(**)	.852(**)	.188	.831(**)	1.000

Table 8 Spearman’s Rank Correlations for Cohesion.

α_1	Rs	Subject
Reject H_0	.851	1
Reject H_0	.839	2
Reject H_0	.809	3
Reject H_0	.626	4
Reject H_0	.854	5
Reject H_0	.679	6
Reject H_0	.834	7
Accept H_0	.421	8
Reject H_0	.829	9
Reject H_0	.857	10

As it can be observed from Table 8, H0, i.e., not having correlation, was rejected for 90% of subjects with 95% of confidence level (since α_1=0.05). Thus, it can be claimed that there is a high correlation between values obtained by applying SCD and values given by experts. In other words, SCD has provided conceptual cohesion value much similar to those provided by experts.

6.5. Comparing SCD with some other metrics

In this subsection, SCD is compared with two metrics SIDC [6] and DM_IAUM [7] mentioned in subsection 2.2. The reason of comparison with these two metrics is that they are well known metrics in SOA which can be measured using the CRUD matrix data. In this way, we can calculate cohesion values of services shown in Figure 6 using SIDC and DM_IAUM and then compare resulting values with SCD values on the same services. Based on this comparison, we discuss and show that SCD is a better metric to measure how much a service concentrates on a single business functionality or semantically related business functionalities (or in other words, how much service operations are related in terms of a domain-level concept).

Table 9 DM_IAUM Values for the First Four Services

Normalized Value	DM_IAUM	Number of Messages	
0.5	0.66	6	S1
0.75	1.00	4	S2
0.60	0.80	5	S3
1.00	1.33	3	S4

DM_IAUM value of a service is the number of system services divided by the total number of related (either input or output) messages to that service. For example, considering the first service (S1) in our experiment, i.e., the blue cluster of the first CRUD matrix in Figure 6, the number of services is 4, the number of related messages is 6 (actions which are outside the cluster but are in rows and columns of that cluster are considered as the messages of the related service), and thus DM_IAUM value is 0.66. DM_IAUM values for all services in the first CRUD matrix are shown in Table 9.

SIDC measures the relationship degree between two service operations as the number of BEs processed by both of them divided by the number of BEs processed by either of them. Then, SIDC value for a service can be obtained by taking an average on relationship degrees of all possible pairs of service operations. As an example, the calculation of SIDC for S1 is as follows:

Table 10 SIDC Values for the First Service

OP_2OP_3	OP_1OP_3	OP_1OP_2	Operations Pair
$\frac{2}{3} = 0.66$	$\frac{2}{2} = 1.00$	$\frac{2}{3} = 0.66$	Operations Relationship Degree
$\frac{0.66 + 1.00 + 0.66}{3} = 0.77$			SIDC

Table 11 shows the cohesion values of all 21 under experiment services according to SCD, SIDC and DM_IAUM metrics.

Table 11 Cohesion Values of 21 Services

Servi ce	SCD	SIDC	DM_IAUM
S1	0.91	0.77	0.49

S2	0.74	0.5	0.75
S3	0.58	0.5	0.6
S4	0.93	0.5	1
S5	0.82	0.52	0.5
S6	1	1	1
S7	0.66	0.5	0.57
S8	0.52	0.17	0.8
S9	0.89	0.8	1
S10	0.49	0.6	0.5
S11	0.33	0.33	1
S12	1	0.5	1
S13	0.52	0.37	1
S14	0.39	0.38	1
S15	0.64	0.5	1
S16	0.63	0.75	1
S17	0.72	0.8	0.5
S18	1	1	0.5
S19	0.47	0.41	0.5
S20	0.40	0.40	1
S21	0.41	0.33	0.5

The analysis on the resulting values is as follows:

1. The viewpoint of SCD and SIDC metrics about cohesion is different from that of DM_IAUM. From DM_IAUM viewpoint, a service is more cohesive if it has less communication with other services (through input and output messages). The drawback of this metric is obvious; for example, suppose that whole of the first CRUD matrix in Figure 6 including all EBPs is considered as a single service. According to DM_IAUM, such a service does not communicate with any other services and is thus completely cohesive. But this service encapsulates various EBPs while there is not a strongly conceptual relationship between them. As another example, DM_IAUM results in the smallest cohesion value for S1 (see Table 11) while this service is among top services from cohesion point of view based on both SCD and SIDC metrics. Obviously, SCD and SIDC evaluate the cohesion degree of S1 better because its operations access and/or process three conceptually related BEs, i.e., “Customer”, “Credit”, and “Account Receivable Note”, such that one can consider and use this service as an “account management” service at the business level.
2. Now, we compare SCD and SIDC. SIDC measures the degree of relationship between two operations in terms of the number of BEs which are processed by both of those operations. SCD considers a special case of this calculation with the following two differences:

a. Unlike SIDC, SCD regards the type of actions performed on BEs.

b. In addition to direct dependencies, SCD considers transitive dependencies between BEs.

We use a simple example to demonstrate the importance of these differences: consider EBPs “Add customer” and “Add an account receivable note” from S1 and EBPs “Schedule shipping” and “Issue draft” from S3. According to SIDC, the relationship degree between operations in both of these pairs is 0.66 while SCD results in values 0.87 and 0.54 for the former and the latter, respectively. It is worth noting that SIDC yields value 0.66 for every pair of operations (in every context) that process three BEs two of which are common. However, in addition to considering common BEs, SCD regards the type of actions performed on the common BEs. Moreover, because of relying on the SVD algorithm, it considers transitive dependencies between BEs.

To be more precise, we first investigate the relationship between EBPs “Add customer” and “Add an account receivable note” from S1: “Add customer” performs “Create” action on both “Customer” and “Credit”. On the other hand, the action of creating a business entity has the highest affinity to the business process. Therefore, “Customer” and “Credit” has the strongest relationship with each other conceptually. In addition, “Add an account receivable” updates “Credit” and creates “Account Receivable Note”; hence, according to the priorities of action types given in section 4, we have another strongly conceptual relationship between “Credit” and “Account Receivable Note”. Relying on the above facts and also the importance of transitive dependencies between BEs, we can assert that the relationship between “Customer”, “Credit”, and “Account Receivable Note” and thus the relationship between two operations “Add customer” and “Add an account receivable note” which perform actions on these three entities are very strong conceptually.

Now, we probe on the relationship between EBPs “Schedule shipping” and “Issue draft” from S3. BEs “Invoice” and “Shipping Schedule” are processed by “Schedule shipping” using actions “Read” and “Create”, respectively. They are also processed by “Issue draft” using action “Read”. Finally, “Issue draft” performs “Create” on “Draft”. Since reading a business entity has the lowest affinity to the business process, we can assert that the relationship between “Draft”, “Invoice”, and “Shipping Schedule” and thus the relationship between two operations “Schedule shipping” and “Issue draft” which perform actions on these three entities are conceptually weaker in comparison to relationships investigated in the previous paragraph. Unlike SIDC, SCD shows this difference well.

7 Threats to Validity

This section deals with threats that might have an impact on the validity of our results and limit our ability to interpret or draw conclusions from the study data. The threats are defined according to the classification of Shull et al. [28].

7.1. Construct Validity

Construct validity refers to the degree to which the cohesion value measured by SCD (as the independent variable) actually represents the conceptual cohesion of a service (as the dependent variable) in the real world. To investigate this class of validity, first notice that SCD was defined in a formal manner and validated theoretically. On the other hand, SCD measures the conceptual cohesion of a service based on the LSI technique which its effectiveness in the text cohesion measurement has been proven in the IR context. Therefore, SCD can be considered as a constructively valid measure. Moreover, all of measurements are subjective and based on the perception of experts. Since experts in the experiment have medium experience in service orientation, their ratings can be considered significant.

7.2. Internal Validity

An experiment is internally valid to the extent that it shows a cause-effect relationship between independent and dependent internal validities. We have considered the different aspects that could threat the internal validity of the study, such as differences among subjects, precision of subject ratings, learning effects, and subject incentive. Experts were knowledgeable concerning the evaluation issues. Analysing the results of the experiment, we can empirically observe the existence of a correlation between SCD values and the service cohesion.

7.3. External Validity

External validity refers to the degree to which the findings of the study can be generalized to other populations or settings. External validity can often be a problem for controlled experiments in artificial environments where the same conditions may not hold in the real world. In section 6.4.3.2, we introduced participants in our experiment. Although they have industrial experience helping us to extract a final conclusion that can be generalized, it is necessary to repeat this experiment with a more diversified number of subjects, including practitioners and designers with more experience.

Apart from probing on threats to validity, we would rather state that "we have followed recommendations given in the papers related to empirical software engineering, such as [6, 24, 31, 32, 33, 34]". According to these recommendations:

1. It should be explicitly stated how subjects (or experts) have been selected: it was done; see subsection 6.4.3.2.
2. Different data should be used in the experiment: see subsection 6.4.3.3; we used services from three different business areas, i.e., sales, library and maritime transport infrastructure development.
3. Experts should be trained about the under experiment algorithm, method, tool, etc. In addition, if the experiment is done using an aid tool, that tool should be described for the experts, too: we did not use any aid tool during the experiment, but we explained working procedures of business processes and answered to related questions raised by experts.
4. To decrease threats of expert's representativeness, experts with various background, experience level and field (both academic and industry) should be selected: it was done; see subsection 6.4.3.2 and Table 6.
5. To increase data accuracy, expert's viewpoints and related values should be gathered in various sessions and via different methods (for example, entering data through an aid tool in a session and filling hard questionnaires in another session): unfortunately, because of low availability of experts, we gathered required data during one long session. Of course, to increase data accuracy, we asked experts to spend enough time in order to provide reliable data in that session.

8 Conclusions and Future Work

8.1. Conclusions

The proposed metric in this paper measures conceptual cohesion as one of the important service design principles in the service identification phase. Considering this fact that services should be inherently aligned with business, service cohesion must be measured based on the scope of functionality that individual services implement. As Erl pointed in [2], each service, based on its granularity level, can realize a couple of activities in the business process, from one activity to all activities. It is obvious that a service is at the right granularity level when its activities are strongly related from functionality point of view. Thus, the relationship between activities in a business process must be examined in order to determine cohesion conceptually. In our approach, the degree of this relationship is calculated based on the relationship degree between business entities processed by the business process.

A designer can discover relationships between business entities by analysing business processes in deep and by using his/her own intelligence and expertise; however, since the main application of our metric is in service identification methods, developing an approach to determine business entities relationships without dealing the designer's experience is a motivating and interesting work followed by the present paper.

The main advantages of our approach are as follows:

1. It proposes a metric which measures cohesion conceptually.
2. The proposed metric can be applied in the service identification phase, i.e., when services are not yet available, and the designer plans for developing services with high cohesion. It should be noticed that required information to measure service cohesion is acquired from enterprise business processes which are available in this phase.
3. It uses a LSI-based method to calculate cohesion. This method needs a statistical activity which could be performed automatically. On the other hand, LSI relies on a powerful mathematical model whose effectiveness has been proved in IR applications.
4. Our metric is defined formally based on the graph theory. Thus, the presented formulas are usable and understandable as well.
5. Scaling cohesion values by total edges of the graph causes that each service cohesion is comparable with cohesion of other services and helps the designer to make decision on different service schemas.

Every software metric which is proposed must be evaluated and validated as well. Measuring conceptual cohesion by SCD was described in detail and step by step using a simple case study. Thus, it is obvious that everyone can apply SCD to different contexts and business domains. On the other hand, the proposed metric was validated by both measurement theory and empirical methods. Obtained results show that SCD is a valid metric to measure conceptual cohesion of a service, and this metric leads to values which indicate semantic relationship between service operations well.

8.2. Limitations and Future Work

1. The first limitation of SCD is its dependency on the type of actions applied on business entities. Although business processes involve business entities, they do not specify which action type is done by an activity on a business entity. Therefore, if we do not have a structure like the CRUD matrix which specifies the type of actions on entities, or in other words, if business services are only identified based on business processes, SCD cannot be applied automatically. In summary, the organization under study should have documents of its enterprise architecture including CRUD matrices.

2. Another limitation is that SCD is only usable in the initial stages of the service oriented life cycle, i.e., the service modelling phase. Identifying services with proper level of cohesion results in the design and implementation of a cohesive service; however, SCD should be revised to be applicable on services designed before (for example, by the use of BPEL processes) as well.

3. We have extracted semantics based on the affinity between business entities and processes to measure the conceptual relationship between operations. More precisely, SCD is mainly based on the type of actions which EBPs perform on business entities. Although this method is completely based on the conceptual nature of cohesion, business processes contain other elements, such as gates, that are effective on the degree of relationship between business entities. For example, two business entities that are processed through an “and” gate are more operationally related than two business entities that are processed through a “xor” gate. Therefore, other information existing in processes, such as the available gates and the sequence of operations, which are effective on cohesion, shall be investigated in our future work.

4. Although the obtained results approve the applicability of the proposed metric, utilizing SCD in industrial organizations to evaluate service cohesion is considered as future work because having more EBPs likely leads to a better evaluation of the relationship degree between BEs. This assertion is based on the dependency of our approach to LSI; having more documents in LSI results in the better evaluation of the relationship degree between terms [9].

References

1. Perepletchikov M, Ryan C, Frampton K. Cohesion Metrics for Predicting Maintainability of Service-Oriented Software. In Proc. IEEE 17th Int. Conf. on Quality Software (QSIC 17), 2007.
2. Erl Tomas. Service-Oriented Architecture Concepts, Technology, and Design. 2005.
3. Papazoglou M, et al. Service-Oriented Computing: State of the Art and Research Challenges. IEEE Journal of Computer, 40(11), 2007, pp. 38-45.
4. Erl Tomas. SOA: Principles of Service Design. Prentice Hall, 2007.
5. Qian M, Zhou N, Zhu Y, Wang H. Evaluating Service Identification with Design Metrics on Business Process Decomposition. In Proc. IEEE Intl. Conference on Services Computing, 2009.
6. Perepletchikov M, Ryan C, Tari Z. The Impact of Service Cohesion on the Analyzability of Service-Oriented Software. IEEE Transactions on Services Computing, 3(2), 2010.
7. Shim B, Choue S, Kim S, Park S. A Design Quality Model for Service-Oriented Architecture. In Proc. IEEE 15th Asia-Pacific Software Engineering Conference, 2008, pp. 403-410.
8. Kazemi A, Rostampour A, Zamiri A, Jamshidi P, Shams F. An Information Retrieval Based Approach for Measuring Service Conceptual Cohesion. In Proc. IEEE 11th Int. Conference on Quality Software (QSIC11), Madrid, Spain, July 13 - 14, 2011.
9. Dominich S. The Modern Algebra of Information Retrieval Berlin. Springer, 2008.

10. Briand L C, Morasca S, Basili V R. Property-Based Software Engineering Measurement. IEEE Transactions on Software Engineering, 22(1), 1996.
11. Kim M, Kim S. Service Identification Using Goal and Scenario in Service Oriented Architecture. In Proc. IEEE 15th Int. Conf. on Asia-Pacific Software Engineering, 2008, pp. 419-426.
12. Chidamber S R, Kemerer C F. Towards a Metrics Suite for Object- Oriented Design, Object-Oriented Programming Systems, Languages and Applications (OOPSLA). Special Issue of SIGPLAN Notices, 26(10). 10, 1991, pp. 197-211.
13. Chidamber S R, Kemerer C F. A Metrics Suite for Object-Oriented Design. IEEE Transactions on Software Engineering, 20(6), 1994, pp. 476-493.
14. Li W, Henry S M. Maintenance Metrics for the Object-Oriented Paradigm. In Proc. 1th Int. Software Metrics Symposium, 1993, pp. 52-60.
15. Hitz M, Montazeri B. Measuring Coupling and Cohesion in Object-Oriented Systems. In Proc. International Symposium on Applied Corporate Computing, 1995, pp. 25-27.
16. Henderson-Sellers B. Software Metrics. UK: Prentice Hall, 1996.
17. Bieman J M, Kang B. Cohesion and Reuse in an Object-Oriented System. In Proc. Symposium on Software reusability, Seattle, Washington, 1995, pp. 259-262.
18. Gui G, Scott P D. New Coupling and Cohesion Metrics for Evaluation of Software Component Reusability. In Proc. IEEE 9th Int. Conference for Young Computer Scientists, 2008.
19. Marcus A, Poshyvanyk D, Ferenc R. Using the Conceptual Cohesion of Classes for Fault Prediction in Object-Oriented Systems. IEEE Transactions on Software Engineering, 34(2), 2008.
20. Dionysis A, Zarras A. Fine-Grained Metrics of Cohesion Lack for Service Interfaces. 2011 IEEE International Conference on Web Services, pp.588-595, 2011.
21. Daghaghzadeh M, Dastjerdi A, Daghaghzadeh H. A Metric for Measuring Degree of Service Cohesion in Service Oriented Designs. Intl. J. of Computer Science Issues. Vol. 8, 5, No 2, 2011.
22. Kumaran S, Liu R, Wu F. On the Duality of Information-Centric and Activity-Centric Models of Business Processes. In Proc. 20th International Conference on Advanced Information Systems Engineering (CAiSE'8), Springer, 2008, pp. 32–47.
23. Jamshidi P, Sharifi M, Mansour S. To Establish Enterprise Service Model from Enterprise Business Model. In Proc. The 8th Intl. Conf. on Services Computing (SCC'08), 2008, pp. 93-100.
24. Misra S. An Approach for the Empirical Validation of Software Complexity Measures. Acta Polytechnica Hungarica Journal, 8(2), 2011.
25. Khoshnevis S, Jamshidi P, Nikravesh A, Khoshkbarforoushha A, Teimourzadegan R, Shams F. ASMEM: A Method for Automating Model Evolution of Service-Oriented Systems. In Proc. 3th Int. Workshop on a Research Agenda for Maintenance and Evolution of Service-Oriented Systems (MESO), 2009.
26. ASER: Automated Software Engineering Research Group. Available at: http://aser.sbu.ac.ir/
27. Cardoso J. Process Control-Flow Structural Complexity Metric: An Empirical Validation. In Proc. IEEE International Conference on Services Computing (IEEE SCC 06), 2006, pp.167-173.
28. Shull F, Singer J. Guide to Advanced Empirical Software Engineering. London: Springer, 2008.
29. Available at: http://www.bpc.ir
30. Available at: http://www.pmo.ir/en/home
31. Richard W, Lanza M, Robbes R. Software Systems as Cities: A Controlled Experiment. In Proc. IEEE 33rd International Conference on Software Engineering (ICSE), 2011.
32. Juristo N, Moreno A M. An Adaptation of Experimental Design to the Empirical Validation of Software Engineering Theories. 23nd Annual NASA Software Engineering Workshop, 1998.
33. Cohen J. Statistical Power Analysis for the Behavioral Sciences, Technometrics, 31(4), 1989.
34. Briand L C, Bunse C, Daly J W. A Controlled Experiment for Evaluating Quality Guidelines on the Maintainability of Object-Oriented Designs. IEEE Transactions on Software Engineering, 27(6), pp. 513-530, June 2001.