A Tool for Measuring SOA Service Granularity

Geetha J.
Research Scholar
Bharathiyar University
Coimbatore, India

Karthikeyan T.
Associate Professor
P.S.G. College of Arts and Science
Coimbatore, India

ABSTRACT
Service granularity plays a major role in designing services in SOA. Service Granularity can be defined as the measurement to identify how broad the interaction happening between a Service consumer and Service provider so as to meet their requirements. Granularity also helps in determining the optimality of service. The improper service granularity may lead to service duplication and service maintenance problem. Thus the importance of measuring service granularity helps in better performance, reusability and efficiency of service. At the design stage itself measuring the granularity helps to improve the design as well as its performance. This paper proposes a tool for measuring the service granularity and to check the optimality of services. The metrics used in this paper to determine service granularity are Composite level of service, Functional Richness of service and Interface granularity. The proposed tool evaluates the granularity of the service at the design time.

General Terms
Service Oriented Architecture. SOA characteristics, Granularity of services in SOA

Keywords
Service Granularity, Evaluating granularity of service, SOA, UML, web services.

1. INTRODUCTION
The granularity of Web Services is an important design consideration. The different elements of a service influences its granularity. A service includes its contract, interface and implementation. The contract of a service contains the informal specification of the service, i.e. its purpose, functionality, constraints and usage [1]. The description of the interface is specified in the service contract. The implementation of the service physically provides the required business logic and appropriate data. These are the elements which influences granularity. Internal and external structural software attributes are influenced by Service granularity [2]. The concept of granularity is a relative measure of how broad the interaction between a Service consumer and provider must be in order to address the need at hand.

The term service granularity refers to the size and the scope of functionality of a service exposes in web services. It can be quantified as a combination of the number of components/services composed through a given operation on a service interface as well as the number of atomic services. In single interaction, to accomplish a business unit of work, the right granularity should be adopted. The aspects that influence the level of granularity are functionality, flexibility, reuse, complexity, context independence, performance, genericity and sourcing. Defining service granularity is very challenging task which requires considering not only service characteristics but also provisional service types [3].

Service granularity is the one which can be determined by quantity of functionality encapsulated by it. It can be coarse grained or fine grained. The larger the quantity of functionality will leads to coarse grained level of service granularity. The coarse grained level of service reduces the times of service interactive application. Because it encapsulates a lot of business and technology ability in an abstract interface. It is the one which interact with large volume of data and the flexibility is bad. For example, returning the whole catalog entries in a set of categories in online shopping system. Furthermore, when granularity is coarser, the composition cost is less because of the fewer number of components and interactions that are required [4]. The lesser the quantity of functionality will leads to fine grained level of service granularity. It is the one which interact with less volume of data and the flexibility is good. For example, an operation to browse to a catalog by item number or item wise. Building a Java program from scratch requires the creation of several fine-grained methods that are then composed into a coarse-grained service that is consumed by either a client or another service. Businesses and the interfaces that they expose should be coarse-grained. Coarse-grained components have high reuse efficiency but low reusability.

The quality of the service is directly affected by the size division of the service granularity. The service quality includes many aspects such as flexibility, efficiency [5]. To solve the Service granularity issue, we have to focus not on an individual Service, but rather on overall business processes and how Services might meet the needs of multiple processes in the business [6]. Erl et al[7] suggest to assign coarse grained interfaces to service designated as solution endpoints and allow fine grained interfaces for services confined to predefined boundaries so that interoperability is promoted in coarse grained services and reusability can be more promoted in fine grained services The various type of design granularity includes service granularity, capability granularity, constraint granularity and data granularity.

The rest of the paper is organized as follows. In section 2, we survey and analyze previously proposed metrics and research work related to evaluating granularity of services. Section 3 discuss about the phases involved in the tool. In section 4 the tool has been explained for evaluating granularity of services. Finally in section 5 concludes the paper.

2. RELATED WORK
A number of literatures have addressed the significance of service granularity in different aspects and their impact in designing the web services. Guidelines are available for measuring appropriate granularity and neither it gives

concrete solution and nor a method to decide the right granularity. The service granularity metrics proposed by [8] considers the number of operations within the system and the similarity between them. The author identifies service granularity as design property and also considers parameter granularity. The author of [9] describes granularity metrics as a measure of reusability, business value, context independency and complexity. The design issues are the value that is produced by WGLA model is not equivalent to the granularity level and it just measures the appropriateness level of composite service granularity. The value that is produced by this model is not a definite value. However, this approach neither covers important granularity attributes not recommends any criteria to evaluate appropriateness of service granularity Schmelzer [10] addresses that the granularity measurement applies in relation to the services available and number of interactions necessary for satisfying specific goal. Foody[11] suggests some guidelines such that combining small operations or breaking larger operations result in right granularity. This limits the size of the messages exchanged between the service provider and consumer. Feurerlichit [12] describes service granularity based on the data normalization. The author applies Boyce- Codd Noraml Form(BCNF) to determine functional dependencies which in turn select appropriate level of service granularity. It specifies that the excessive use of coarse-grained services results in poor reusability and high level of data coupling. This is contrary with most practitioners. They believe that coarse grained services minimize the number of SOAP messages and results in lower communication overheads and less possibility of failure. After the analysis of previous works we proposed set of metrics in [13] for evaluating service granularity. The tool described in this paper takes those metrics for evaluating service granularity.

3. GRANULARITY MEASURES

The tool proposed in this paper has taken the following measures [13] to evaluate the granularity of the given service.

3.1 Composite level of Service

A service is composed of many atomic and composite services. The granularity of the service increases as the number of composite service increases. Therefore the composite level should be taken to measure service granularity. A Composite service structure aggregates smaller and fine-grained services. The hierarchical service formation is characteristically known as coarse-grained entity that encompasses more business or technical processes. Composite services may aggregate atomic or other composite services [14].

3.2 Functional Richness of Service

The functionality of the service is a key for the granularity of the service. The functions of a service include both business logic function and CRUD function. While calculating functional richness, the main parameters to be calculated are function count and CRUD functions of service. Functionality can be measured from the function points in the given services. Function point is a measure of software size that uses logical functional terms business owners and users more readily understand [15]. The parameters used in calculation of function count are as follows.

The parameters are used in calculation of function count are :

(1) Data Functions - Internal Logical Files (ILFs)

(2) Data Functions - External Interface Files (EIFs)

(3) Transaction Functions - External Inputs (EI's)

(4) Transaction Functions - External Outputs (EO's)

(5) Transaction Functions - External Inquiries (EQ's)

The word CRUD denotes create, Read, Update and Delete. These are the operations which relate with database. This is for counting total number of operations which relate to CRUD function.

3.3 Interface granularity of Service

The interface of service exposes service’s granularity. The granularity concept refers to the number of service operations or their signatures which are related to service description that is interface granularity [9]. Interface granularity is the one which can be calculated by the number of input parameters and number of output parameters involved in a service.

The average of metrics such as composite level of service, functional richness of service and interface granularity will give the service granularity. It always lies between 0 to 1. Lower the value granularity of the service is less, higher the value says granularity is more.

4. DESIGN AND PHASES

The tool is developed using java and net beans. The various phases in our tool are :

1. Data retrieval

2. Analysis of service from Meta data

3. Evaluation of granularity

4. Report generation-granularity level of given service

Phase 1: Data Retrieval

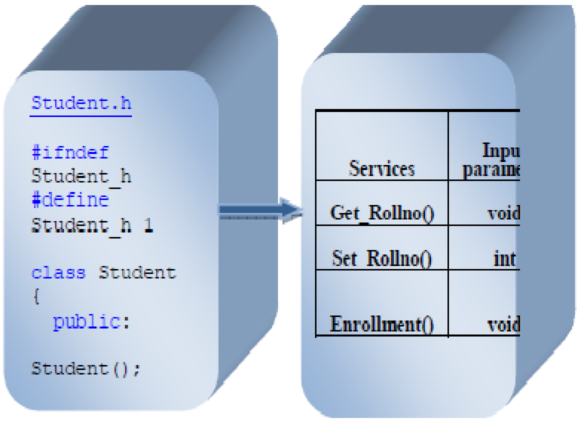
First, the pseudo code has been generated for the given service using the class diagram of Unified Modeling Language (UML). UML is an object oriented approach which will identify services from different prospective like application level. This provides a mechanism to align business and software based aspects using the functionality provided and allows service metrics to play a key role in identifying the optimal service granularity [3]. Data are retrieved from the pseudo code of the given service (Fig.-1). Then the retrieved data from the pseudo code which is generated from UML diagram are stored it in database. These stored values are used for calculating various metrics and generation of report.

Phase 2: Analysis of Service from Meta Data

The services that are obtained from the Meta data of the UML diagram for the given service are analyzed so as to evaluate the service.

Phase 3: Evaluation of Granularity

From the meta data the following data are grouped .Service Granularity and service optimality can be measured from various parameters: Number of atomic services in service, Composite level of service , Number of CRUD functions in a service, Function count of service, Functional richness of service, Input parameter granularity, Output parameter granularity, Interface granularity. Since the metrics are going to be applied on each individual service, granularity level of the service will be more appropriate. The evaluation of correct granularity level of service will lead to better performance, reusability and efficiency of service.



```
Student.h

#ifndef
Student_h
#define
Student_h 1

class Student
{
  public:

Student();
```

Services	Inpu parame
Get_Rollno()	void
Set_Rollno()	int
Enrollment()	void

Figure – 1 Extraction of data

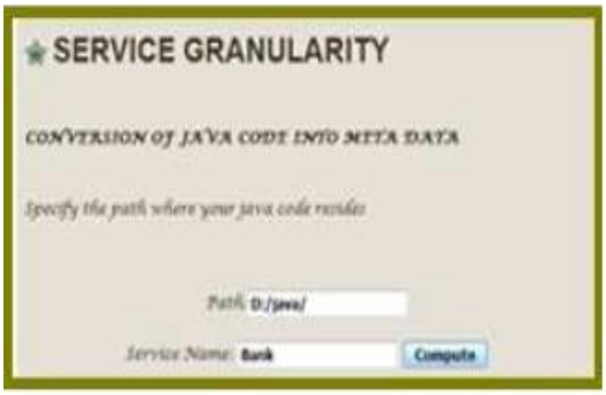
Phase 4: Report Generation

Granularity for different services are calculated and generated as a report. Comparison of different services can be indicated using a chart diagram.

5. EVALUATION TOOL FOR SERVICE GRANULARITY

Step1: Specify the Java Folder of UML diagram and the Service Name

The UML Class diagram can be converted to java code using any conversion tool. In our implementation, we have used Netbeans module to covert class diagram to java code. The code will be generated in any specific folder. That folder name is given as input for our proposed tool. The code generated contains some java1, java2 extensions temporary files. Thus our tool separates java files alone and sends to next stage for analyze (Fig.-2).



SERVICE GRANULARITY

CONVERSION OF JAVA CODE INTO META DATA

Specify the path where your java code resides

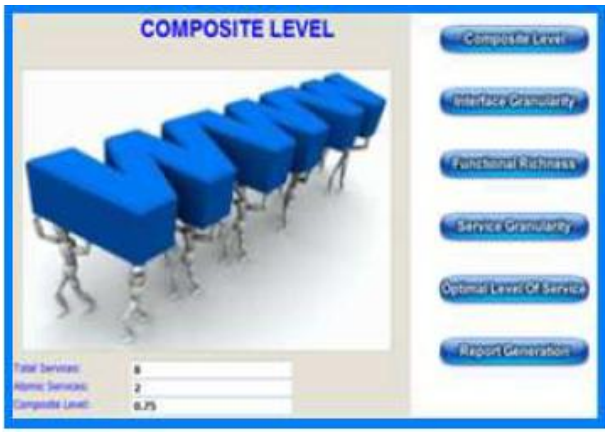
Path: D:/java/

Service Name: Bank

Figure – 2

Step 2: To get Composite Level of Service

The next stage computes composite of service. Number of atomic services can be determined using minimum number of operations in class diagram. Some class contains only member variables and thus they cannot be further divided. Those classes comprises atomic services. Since class diagram is drawn for various sub-services, they come under total number of services. Using these two parameters, composite level is determined (Fig.-3).



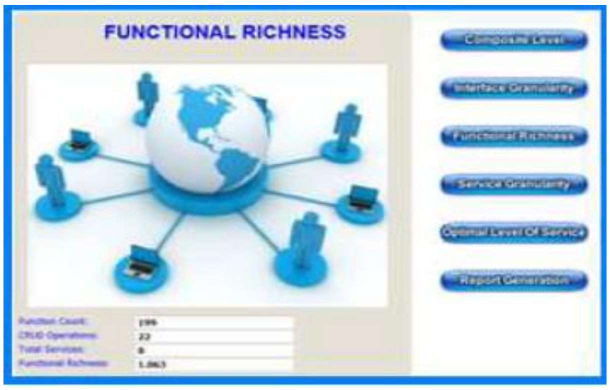
COMPOSITE LEVEL

Total Services:	8
Atomic Services:	2
Composite Level:	0.75

Figure - 3

Step 3: To get Functional Richness of Service

Function count can be determined by following parameters. External input specifies the number of input values given to each service. This can be calculated from the input parameters in operations of each sub-service. External output specifies the number of output values passed from service to user. This can be determined by the return type of each operations. External inquires can be determined by matching the operations names with set of predefined terms such as ‘search’, ‘browse’ etc., CRUD functions can be calculated by service’s database operations. Each input, output, inquiry operations has an effect on database. The screen shot of this is shown in Fig.-4.



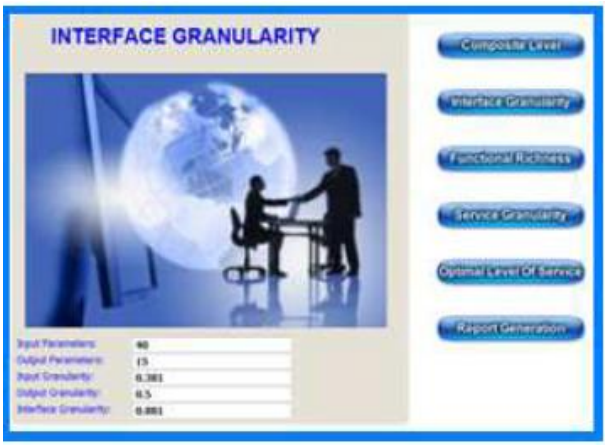
FUNCTIONAL RICHNESS

Function Count:	199
CRUD Operations:	22
Total Services:	8
Functional Richness:	1.063

Figure - 4

Step 4: To get Interface Granularity of Service

Interface Granularity can be calculated from input, output parameters of each operations in service. It also includes the weightage value for data type of parameters. For eg., we have used the value 0 for void, 0.25 for primitive data type, 0.5 for user-defined data types. The input and output parameter specifies the count of total input and output parameters of all the sub-services respectively. Input granularity (Fig.-5) is calculated using number of input arguments and weightage value for argument’s data type. Similarly Output granularity can be determined using return type and return value. Interface granularity is the sum of input and output granularity.

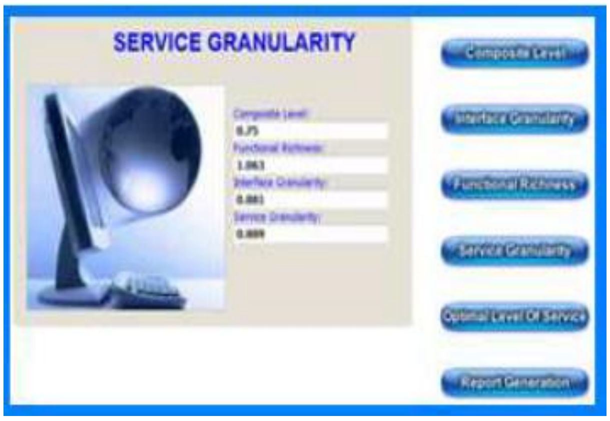


Input Parameters: 40
Output Parameters: 15
Input Granularity: 0.381
Output Granularity: 0.5
Interface Granularity: 0.881

Figure - 5

Step 5: To get Service Granularity of Service

Service Granularity can be determined by taking average of Composite Level, Functional Richness and Interface Granularity of services. Service Granularity helps in determining the optimality of services. For business service, service granularity value of 0.75 to 1 determines the optimal level. This is shown in Fig. – 6.

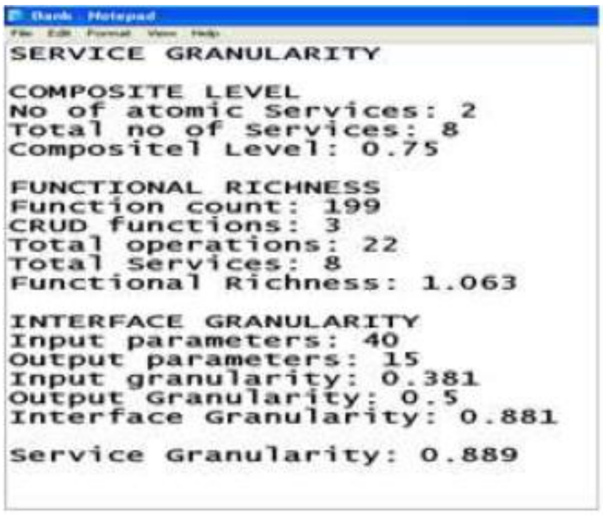


Composite Level:
0.75
Functional Richness:
1.063
Interface Granularity:
0.881
Service Granularity:
0.889

Figure -6

Step 6: Report Generation

The final reports are generated as below :



```
SERVICE GRANULARITY

COMPOSITE LEVEL
No of atomic Services: 2
Total no of services: 8
Compositel Level: 0.75

FUNCTIONAL RICHNESS
Function count: 199
CRUD functions: 3
Total operations: 22
Total Services: 8
Functional Richness: 1.063

INTERFACE GRANULARITY
Input parameters: 40
Output parameters: 15
Input granularity: 0.381
Output Granularity: 0.5
Interface Granularity: 0.881

Service Granularity: 0.889
```

7. CONCLUSION

While measuring the quality of service, granularity plays an important role. Thus in our proposed approach, service granularity can be measured from important attributes such as composite level, Function count, number of CRUD function, functional richness , Input granularity, output granularity and interface granularity. Different designs of single services can be taken and their granularity can be compared using this tool. From this a service with adequate granularity can be identified. The correct granular service is more reusable one. Therefore from this granularity measurement tool, we can evaluate the granularity of the given service as well as we are able to detect the services with more reusable.

6. REFERENCES

[1] Steghuis C., "Service Granularity in SOA Projects: A trade off Analysis", M.Sc. Thesis, Business Information Technology, University of Twente, 2006.

[2] Saad Alahmari, Ed Zaluska, David C De Roure, "A Metrics Framework for Evaluating SOA Service Granularity", on 2011 IEEE International Conference on Services Computing.

[3] Thomas Erl, "SOA principles of Service Design", Prentice Hall, 2009.

[4] Hong YING, Yu WU,Fuming LIU, "Research on The SOA – based Service Granularity Control", Second International Conference on Information Technology and Computer Science, IEEE, 2010.

[5] Xie Zhengyu, Dong Baotian, and Wang Li, "Research of Service Granularity Base on SOA in Railway Information Sharing Platform", in Proceedings of the 2009 International Symposium on Information Processing (ISIP '09) Huangshan, P. R. China, August 21-23, 2009, pp 391-395.

[6] Jason Boomberg "How to Define a Business Service the Art and Science of Service Granularity", Zap Think White paper, 2007 available at www.zapthink.com.

[7] Kulkarni N., Dwivedi, "The Role of Service granularity in a successful SOA Realization A Case Study", IEEE Congress on Services – Part-I, vol.no. pp 423-430, 6-11, 2008.

[8] B. Shim, S. Choue, S. Kim, and S. Park, "A Design Quality Model for Service-Oriented architecture", 15th Asia-Pacific Software Engineering Conference, 2008.

[9] A. Khoshkbarforoushhaa, R. Tabeinb, P.Jamshidia, F. Shamsa, "Towards a Metrics Suite for Measuring Composite Service Granularity Level Appropriateness", 2010 IEEE 6th World Congress on Services.

[10] R. Schmelzer, "The service granularity matrix", August 3, 2007 available at www.zapthink.com › Research › ZapFlash

[11] James Mc Govern, Sameer Tyagi, Michael E Stevens and Sunil Mathew, "Java Web Services Architecture", Morgan Kaufmann Publishers, An imprint of Elsevier, 2005.

[12] Feuerlichit, G., Service Granularity Considerations Based on Data Properties of Interface Parameters", International Journal of Computer Systems Science and Engineering, Special Issue: Engineering Design and

Composition of Service-Orientation Applications", ISSN 0267 6192.

[13] T. Karthikeyan, J.Geetha, "A Quantitative Measurement and validation of Granularity in Service Oriented Architecture", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 2, No 1, March 2012

[14] Spyridon Antakis, "Security Service Granularity", 2008, available at www.*spyros.loonydesk.com/whitepapers/ssg.pdf*.

[15] Takuya Uemura, Shinji Kusumoto, and Katsuro Inoue, "Function Point Measurement Tool for UML Design Specification", available at http://www.cs.unibo.it/~cianca/wwwpages/labspo/uemura.pdf