University of Stavanger

Faculty of Science and Technology

MASTER'S THESIS

Study program/ Specialization: Masters of Science in Computer Engineering	Spring semester, 2013 Open Access
Writer: Mudassir Farooq	…… (Writer's signature)
Faculty supervisor: Erdal Cayirci External supervisor(s):	
Title of thesis: **TSCCM; A new model for Manual Service Composition**	
Credits (ECTS): 30 ECTS	
Key words: Service Composition Trustworthy Composition	Pages: 106 Stavanger, June 15,2013

DEDICATION

This work is dedicated to my parents, wife and my elder sister to whom I owe so much for my achievements.

Acknowledgment

I would like to thanks Professor Erdal Cayirci for his guidance, support and encouragement for this project. I feel so lucky to have him as my supervisor for this thesis. At times it looks to me that he is more involve than myself. I always find him humble and cooperative. I am here not hesitating to mention that without him this work will be much more difficult to complete.

Table of Contents

List of Tables

List of Figures

List of Symbols

ɣ = Gama; Time to create the SLA

α = Alpha; Accuracy in data content

β = Beta; Time to reach trust

μ = Mu; Time require to compose services

P = Probability of reaching trust

s = Service fan-out; shows number of services use in the composition process

c = Cloud fan-out; represents number of clouds involve in the composition process

n = Depth of nest; represents depth of services nest

t = Third party fan-out; shows number of third parties

R = Reusability factor; shows services reuse for user requirements

C = User credibility check; shows user credibility

M = Module implementation; shows TSCCM modules implementation

ω = Service complexity; Complexity of services

Abstract

Intercloud approach introduces new opportunities to improve the performance and to increase the utilization of distributed simulation systems in cloud computing. These improvements also imply significant reduction both in the initial investment and operations/maintenance costs of IT infrastructure. In last few years, computer resources are considered as services and the service oriented computing has become popular. Service providers offer resources as services. Collaboration among the services is necessary to fulfill user requirements effectively. Provision of different services has highlighted the use of manual service composition. Service composition creates new services which are used to resolve complex problems with the reduction in consumer cost. Introduction of trustworthy service composition in cloud computing brings up several challenges. Changing cloud environment is a big challenge to offer services as a resource and even it makes service composition difficult, other than this, Users have QoS requirements and trust is one of the most vital factor. User trust shouldn't be broken. Trust should be considered as an important aspect in service collaboration and emphasizes should be given on building trust among the service providers and with the subscribers.

Emergence of service usage has brought up several challenges which needs to deal in order to make service oriented computing successful. Now days, thousands of vendors are in market and it's not easy to present every resource as a service. We have discuss some service composition challenges which should be resolved. Service oriented computing has emphasize on service interaction because individual services are not always useful to provide complete solution of some problems. We need to compose services in order to solve more complex problems. A service composition requires good service interaction among the services within the same pool. Successful service composition depends on the trust factor. Bearing in mind the importance of Trust in cloud architecture, we have discussed the challenges and factors which can affect the trust. This master thesis addresses emerging challenges in service composition and trust in Intercloud and proposes a new model for trustworthy manual service composition. Trustworthy Service composition model can effectively be used to compose and build trust among the services involved in composition process and improves the interaction among the services.

1 Introduction

1.1 General Overview

The emergence of cloud services is being in discussion on all the forums. Researchers are focusing on cloud services and more mechanisms are being formed in order to solve challenges to implement cloud services. Worldwide popularity of cloud environment increases demand for high performance and QoS at low cost. Increase in demands results in building more and more services and improving their interaction between each other to solve complex tasks more efficiently and effectively. The concept of combining different clouds or improving their interaction is further extended to service composition. Individual services have been offered by cloud providers for a long time. They are very effective but they have many limitations. A single individual service may not fulfill all the requirements of users where as to solve complex problems, they are inefficient or they don't have all the features required to solve complex problems. Keeping in mind this challenge, concept of service composition was introduced. Composed services have grand pool of resources, which offer more flexible and richer set of services to subscribers. Everything can be put together to facilitate subscribers. Other than problem solving, service composition offers many benefits which can directly benefit consumers. It increases the performance, reduces the cost and time, required to solve a problem. Service composition has been under discussion for last 4 years but it is still a novel topic because inter-cloud itself, is a new concept and there are still several existing challenges, which need to be addressed. In our last year work on inter-clouds, we highlighted some of the emerging issues in inter-clouds and focused on their solutions. The proposed solutions were very effective and can be beneficial for constructing mechanisms, which can resolve the challenges in inter-clouds. Service composition is a useful idea and will be beneficial for the cloud environment itself and will improve provision of services, which will eventually benefit the subscribers of cloud, which is our Goal. There have been several discussions on service composition and useful techniques exist but still, there are number of issues and challenges, which need to be addressed. In this Master's thesis, we survey the existing service composition techniques. After survey we address the challenges in service composition. Improvement areas for service composition are also discussed.

Efforts are done to highlight emerging challenges in service composition and trust in intercloud. Firstly, we discuss challenges in service composition, which are necessary to highlight because for trustworthy service composition, composing services is the first important thing. After the composition, we move towards the trust. Challenges for trust development are also discussed and trustworthy service composition conceptual model (TSCCM) is designed keeping in mind the challenges. After discussing the service composition in detail, we address the trust relation among the service providers and consumers. A detailed research on trustworthy communication is done and weaknesses are discussed. A conceptual mode for trustworthy service communication among the services to be combined is proposed. The Model proves to be very effective in building trust among the services involved in service composition process. Trustworthy service composition has greater importance especially in dynamic service composition, where billing is charged as "pay-as-you-go". Subscribers or service providers will be willing to pay under better trust environment. In future, some parts of service composition will be shifted towards the cloud users and they will be able to decide service composition at their own. Despite of all the work done in Service composition and trustworthiness, the success is still far away. [15]

Trust is an emerging challenge in service computing and has great importance, where anonymous parties interact with each other consistently at higher speed; because of thousands of services and different platforms of services, discovering known services is difficult, so for better service composition anonymous services should be composed together and mechanisms should be designed to address trustworthy problem among them. Trustworthy communication among shared pool of services is a major factor, which is discussed in this Master thesis. We have used trust as a vital component for effective service composition. There is a need of trustable environment, in which all the participating parties can interact in a more secure and less vulnerable environment. During the discussion, Manual and Automatic service composition is taken into account and is discussed. We focus on manual service composition because it has been seen that in most of the written papers automatic service composition is discussed and less attention is given to Manual service composition. Manual Service composition is as important as is automatic. During the provision of IT solutions to clients, there are several cloud providers involve and each cloud provider provides several services. When we say IT solutions, it actually

means Infrastructure as a Service (IaaS) and Software as a Service (SaaS). During manual service composition to provide IaaS and SaaS, there is need to build trust among the compose services and trust should be taken into account right from the beginning. There are several discussions in the literature about service composition and trust among the services but most of them focus on the automatic service composition.

We have proposed a trustworthy model which takes into account the manual service composition and will serve as a very good model to build trust during the manual service composition. Model is designed after a vast literature survey on service composition and building trust in intercloud. A phase approach is adopted in the model and trust is considered as a key parameter right from the start of the communication between the provider and the user and similarly among the participating services in service composition. Implementation of model will lead to trustworthy environment in manual service composition, where as it will also reduce time to build trust and fulfills the user requirements with greater accuracy. We use a module approach and modules are implemented in all phases. Each module is responsible to do some task, which will eventually help to do trustworthy service composition. Trustworthy service composition conceptual model (TSCCM) model, proposed in the thesis actually defines levels of trust development, which are necessary for trustworthy communication among the shared pool of services and between the subscriber and the service provider.

1.2 Motivation & Goal

Service composition in inter-cloud has been around, for almost a decade. It is becoming popular among enterprises, many organizations are taking benefit from service composition resources, but still there are many challenges, which are needed to address and resolve in order to have full impact of service composition on computing world. The thesis is written after an extensive survey on service composition and building trust among the composed services. Future challenges in manual service composition are also examined in detail. A new conceptual model (TSCCP) is also proposed to provide trust among the services. Efforts are made to highlight and address challenges in service composition and Trustworthy service composition. Work is in headway to provide solutions to these. This thesis thoroughly addresses this issue; highlighting some vital challenges and solutions to these challenges. TSCCP provides a framework for

interaction among the entries and it will really help in building trustable environment for service cloud. (TSCCP) will improve QoS in inter-clouds, which is main area of focus now days. Rest of the paper is structured as follows: Section I contains the introduction to the paper. Section II discusses the Service composition in intercloud whereas Section III focuses on the Trust in service clouds. Section IV contains the proposed conceptual model for trustworthy communication and section V discusses the evaluation of conceptual Model. Section VI explains the conclusion withdrawn from the research work.

Keywords

Manual Service Composition, Composing SaaS, Trust, proposing new layered modules of procedures, Trustworthy service composition conceptual model (TSCCM), Service level agreement (SLA)

2 Cloud and Service Composition

Individual services in inter-cloud are not fully capable of performing complex tasks. We can call these services, not fully functional. These services can be made fully functional by composition. Service oriented computing enables the composition of individual services. The composed services are useful to resolve more complex problems. Composed services can also be used to solve the problems, which are handled by individual services but the main benefit is improved QoS. Service usage in cloud computing and inter-cloud is increasing day by day and service distribution is increasing across the networks. Due to distribution across the network, the services performance will include dependency on the network. It is important for the service users to consider the service performance independent of the network issues. Sometimes, the service delays are only because of the networks. The users should be aware of the fact that it is due to problems in the networks or service providers should communicate it to the users. This network factor directly affects the service composition process, which mainly includes service discovery, service selection and eventually the service composition.

Selecting optimal set of services is very crucial when there are number of services with equal functionality. Service composition directly affects the cost, time and performance. Service

composition has put focus on quality. Service providers are aware of the fact that by providing best service they will be able to include their services in services pool.

There exist, many techniques on the on-demand service composition and some of them are very affective but still we can say that despite of a lot of research on service composition, it is still a novel topic. One of the main reason is, majority of the techniques introduced are more abstract rather than practical. They are far behind from the practical implementation. All the service requests by users are online and services will be demanded online and user needs a quick response so service composition will be done online and in no time but due to lack of more practical approaches it is not very easy to come up with composite service, which can solve the user problem efficiently. In general, the idea is very good but it needs to be implemented by introducing more practical service composition approaches. During service composition, Trust development between the services is one of them, on which we will put our most of the focus and then introduce a conceptual model for trustworthy communication. One of the solution to this is to introduce service, whose functionality would only be to develop trust among the participating services we call services of such time as Trust Services. They should be available to service providers as a web service. Even, it can be viewed as a Trust service for a single cloud if every single cloud has its own trust service then it will be very easy for a service to develop a trust. The Trust service should make sure that all the models of TSCCM should be implemented during the service composition process.

Most of the existing service composition techniques do not consider cloud computing environments and work independently or according to local cloud hosting environments. There is strong need to focus on cloud computing environment of service composition. Service composition techniques should consider cloud computing environments and exploit the benefits of cloud environment. Cloud interaction is the growing area of research in computing and this will be addressed by composing services, which are interoperable and work together to solve more complex problems. Reliable hosting of services is most important of all. Cloud environments provide reliable hosting because of their vitality. Service composition should be aimed at benefitting from inter cloud environments.

As there are thousands of services available and future concept is, everything on cloud will be provided as service so services will increase and user will have multiple options to choose a service. With increase in offered services, there is a strong a possibility that identical services will exist to provide solution to the consumers. Now a question arises, how we will choose one service from those multiple identical services. There is a strong need of mechanisms, which can solve this issue. As our scope is related to service composition so we will look into it in this perspective. As we know complex problems can be solved by composing several services together. As we discussed before, there is question how we will choose service from identical services for service composition. Service should be chosen on the basis of some parameters and that approach will be more practical. Its open area of research and any useful parameter can be used as criteria to pick the service. In this chapter, we have discussed some general terms and challenges in service composition.

2.1 Services Categorization

Service categorization is a useful concept discussed by [10]. Categorization of service composition is easy to manage and will speed up the composition process. Effective categorization of the service is dependent upon the proper description of subscriber demand. A useful option for this is user interfaces, which we already discussed. On the basis of information in user interface we can decide that what type of service composition is required. [10] Divided services in three categories which are Static, Dynamic and Manual. Static actually means services are composed at design time. We get the user description of problem and compose the set of services. This method is actually very useful if we see this from static point of view. If the subscriber does not demand more services or features then this approach seems very good and very effective from Quality point of view. Static pool of services is very good for reusability. The composed services can be used several times when needed. Dynamic service composition is useful if the user requirements tend to be change with time. Services are composed when they are needed and after completion of the task they are decomposed. This is actually very challenging and is the biggest emerging challenge. Service composition at runtime is hard to achieve. This increases the cost and is time consuming. Service reusability is difficult to achieve and requires complex Database operations as different services are continuously discover and being utilized. It is complex to take the record for future use. There is need to develop more

techniques for dynamic service composition. Algorithms are also needed, which can fetch the services from the cloud, whenever it is needed. Manual service composition discussed in [10] is user driven and user is more involved for service composition. Subscribers choose service which is to be composed. The approach looks very easy to implement but QoS is greatly affected. Users are technically not very good and there is a large possibility that they might choose services which are not interoperable which eventually affects the QoS. The user interfaces discussed in the Thesis can be used to improve the Manual service composition. Manual service composition can be improved by improving the user interfaces. After reading all the details of service composition type I suggest to use a hybrid approach to build service composition. The idea is simple, the services which are required at the beginning or at the design time can be chosen while the services which are on demand can be added later on. Shared pools of services contain services, which are included at the beginning and they will decompose, when the actual problem is resolved whereas during the problem solving new services can be added and removed (Dynamic). Services chosen by user can also be added if they are beneficial for the service composition process.

2.2 Service Compos ability

In future, services will be offered compositely. Nowadays, individual services are available on the cloud and can be requested on demand. Now the trend is changing due to invent of service composition. Composite means shared pool of services. Instead of looking for services to be composed, composite service will be available on the cloud platforms and can be requested by the users as they do for the individual services. Currently the biggest challenge for service compatibility is lack of service composition implementation. Day by day the trend is changing and more services are being composed to solve the problem. Once we have thousands of composite services, they can be provided directly to user. Another big challenge is, there are no cloud platforms which offer composite services. Providing composite services involve many stake holders which we already are discussing in the thesis. Without fixing those issues service composition concept cannot be realized. It is possible for a single cloud to offer service compatibility but that cannot be very effective. Service composition can be more usable when it is possible to receive services from multiple clouds.

Better service composition can also be reached by combining services on the basis of their classification. This will eventually ease service discovery. If services are classified e.g. if we have data dictionary of the network services, gaming services, distributed computing services etc. It will be easy to identify services. During the service discovery process instead of going for all the services, the composition technique can easily go to the particular classification and then look into the desired services which will really improve the response time during the service composition process. This classification concept can be further enhanced to classify the services on the basis of standards they are using, interoperability support. Mechanisms are needed, which can classify the services and also trace them, when they are required.

2.3 User Interfaces

Use of user interfaces for service composition is in discussion now days. User templates are very helpful in forming useful service composition. Services are composed on the basis of the problem to be solved. It is better to have clear description of the problem and user requirements so that composed services would be affective. Diversity of user requirements is difficult to cope for the service composition techniques and is more time consuming. This will affect the three major parameters (Time, Cost & Performance) which eventually affect the QoS. There is a strong need for user templates. User templates will be very helpful in defining the user requirements. Service composition would be very affective and user problem would be resolved in less time with greater performance even it would be easier for user's to define their problems. User templates creation is a difficult and time consuming task, although there exit some user templates but they are ineffective and still there are many things needs to be done. The templates to be build should neither be technical nor non-technical. They should be easily understandable by the users (non-technical) and by the service composition techniques. A trustworthy third party (TTP) should intermediate the development of user Templates. The TTP should form standard operating procedures (SOPs) which are followed by both the users and service providers. TTP can also be involved in listing or filling user requirement in the templates and negotiating with the service providers. The user templates should be available at cloud providers and users should have direct access to them. After submission, cloud will start processing the templates and will come up with best solution for the user.

There is a future prediction that users will also be able to do the service composition. Interfaces can be introduced which will allow users to choose which ever services they want to compose but this area is a novel because there are a lot of things to be done in service composition and user interfaces. After the successful launch of state-of-the art service composition techniques and user interfaces, work on user end service composition will start. For example, if a user requires new server for accounting database. He will need a physical server, storage and network connectivity. If we see from service composition perspective, we will take infrastructure service for the server, storage service and network service. These all will work together to deploy server successfully. After the completion of installation phase, now there is need for Operating system installation. Suppose user want to run three OS on the machine i-e UNIX, OS and LINUX. We can compose three services and they will install the operating systems individually and after that run together to make sure that server is in production. By giving service composition access to the user, he will be able to choose these services at his/her own. This was a simple example. In the same way many complex problems can be solved. Benefit of involving user in service composition will remove the third party dependencies and problem resolution time will be decreased.

Development of standardized user interfaces has increased the demand to develop mechanisms or Languages which can understand the semantics of user requirement and services. If there is a common standard by which we can understand the user requirements this can help to choose more beneficial and interoperable services. User interfaces should be such type that it is easy to understand the semantic which will eventually help in discovering useful services for composition.

2.3.1 Knowledge Sharing in User Interfaces

Knowledge sharing through user interfaces should be defined properly. User interfaces doesn't mean exposing everything to the subscriber. This will not be helpful for the subscriber and neither will be accepted by the service provider. The information which is relevant for the subscribers should be shared through the user interface other than that user shouldn't be aware of any details. This should be taken from business point of view. Service providers do not want their users to know every aspect. They just want them to know what is relevant for them. For

example, if a user wants to resolve a complex problem, the only thing he/she should know about the service is; what is the functionality of the service and what QoS can be offered by the services. Subscribers don't want to know additional details for example to which cloud service belongs to, how they operate, what kind of security mechanism are implemented at the backend etc. The service owner also doesn't want to share such type of information with the user.

2.4 End user division

End users are divided into different categories. There are some, who are technical users and they already have knowledge of service composition or we can say they are from IT background. For example, the people who work on second and third line IT support, then there are some users who have some idea of IT for example the people who work on 1st line support. They don't have full technical knowledge but still they are capable to understand some technical things. Non-technical users are the ones, who are not from IT back ground. We actually classify end users in three basic categories which are following

1. Technical
2. Middleware
3. Non-Technical

End user templates and semantics can be described on the basis of above mentioned user categories.

2.5 Cloud user categories

The cloud users can be divided into two basic categories, private and corporate users. Private users should be divided in above mentioned categories where as corporate clients have more resources and they can provide their requirements in technical format. Mostly the companies are using Web service description language (WSDL) and Business process execution language (BPEL) to describe their requirements [21]. These languages are vastly used for service composition and interaction among the users and the service providers. Companies should be motivated to provide their requirements in a more technical way so that useful service composition can be achieved. The Dynamic CoS framework introduced in [21] is more technical and useful for the users who have good technical knowledge. It is not helpful for the middleware and non-technical users. There is need of good user templates for all categorizes of users.

2.6 Third party Agents in Service composition

Third party plays key role in service composition. It has been seen that in most of the issues a better third party technique can improve performance and choose more appropriate services. [26] Have discussed the use of software agents as a third party and emphasize on their properties is given. Efficient implementation of software will be very helpful in improving service discovery and later will be useful for service composition. Software agents can actually be implemented as softwares, which elaborate user requirements in a more machine readable way. Existence of good agents will also increase user satisfaction. Currently, the issue is not to provide services to cloud; the actually problem is how to transform user requirements and provide him with the services he/she wants. Choose user specific services from thousands is big issue that is why use of third party agents has been discussed a lot and their primary goal is to describe user requirements in a way that it would be easy to look for user related services. Brahim and Athman [26] enlist software agent's properties in their book and emphasize that an agent having such properties will be ideal however, it is not easy to implement all the properties. So far these things are in discussion and it is expected that they will be implemented soon. We discussed those properties and explained them in a service composition perspective. Following are the properties which are useful for software agents

a. Continuous [26]:

Third party agents should be a continuous running process and it shouldn't be stopped. Agent should be in search of services for the users. This continuous evolving will also help in discovering more useful services. Service discovery is a time consuming task so it better to have software agents who have continuous property which will help in locating more useful services for consumers.

b. Autonomous [26]:

This property is very useful, Consumer's complaints that they are not able to interfere in their agent role. If the agents are efficient, then there is no need to interfere in the working. Agent softwares are designed after complete analysis as soon as they get the update from the user about the requirement they start working according to the requirements. The process continuous until

the desired services is not located. During the service discovery process there is no need of user intervention because the agents are good enough to look for services and explain user requirement in a cloud. Hence should be able to work freely and they should perform tasks as described in the program.

c. Cooperative[26]:

Cooperative actually means how agents will act with the Service providers and their clients and in some case with the other agents. There is a need to define rules by which affective communication can be done. There is a need of common communication language which should be understandable by both the subscriber and the agent. Language can be defined in the form of ontology, which is understandable by the both the parties. While creating trustable agents this property should be taken into account similarly during communication with other trust agents and service providers there is a need of common language by which affective communication can be taken place. Language definition for software agents and cloud providers can be more technical because it doesn't involve any laymen who cannot understand technical terms.

d. Reactive[26]:

Agents should be capable enough to understand user requirements comprehensively and in a quick time so that requirement gathering shouldn't get long time. The cooperative property of agents would help in reaching the understanding. Similarly, agents should be good enough to communicate the gathered information to service providers. There is a chance that user requirements would change during the time. The agents should be capable to adapt to the changes so that trust among the user and agents and service providers shouldn't break.

e. Adaptive[26]:

Agents should be capable enough to review their previous interaction with the users and service providers and rectify the behavior accordingly. This property will be helpful in building good trustable trust agents. Trust review mechanism should be part of continuous evolving process so that better robust and trustable agents can be formed.

f. Pro-active [26]:

Agents should play active role during the service composition and requirements gathering process. Pro-activeness means negotiation should be done by agents to resolve the issue. There are sometimes scenarios, in which as it is services are not available. In such scenarios agents are required to convince user and change their requirements. Agents should be capable enough to convince users and service providers in case of any changes in requirements and agents should play active role to negotiate between user requirements and available resources.

2.7 Use of semantic web ontology for intercloud directories and exchanges

Interoperability among the composed services and clouds is still a novel topic. There have been many schemes available but interoperability is a big as the number of service scalability is increasing day by day and it is very difficult to come up with a scheme which can resolve the interoperability issue. QoS in Service oriented computing is also dependent on the interaction among the composed services. If the composed services are interoperable, then they will better mediate with each other, which will affect the overall performance and QoS will be improved. Finding out interoperable services is a difficult and time consuming task. Use of third party is highly encouraged to select services which are interoperable. [23] Proposed such a scheme which actually based on the idea of maintaining catalog of resources to provide better interoperability. A semantic web resource definition framework (RDF) is proposed to improve interaction among the clouds.

2.7.1 Semantic interaction in service composition

Semantics is being in discussion from long time. The ways interaction is being done, are of great importance. Good semantics are very useful to for good interaction. Latest web based semantic technologies like ontology's and XML standards helped a lot in improving service composition interaction. We will also over available languages and standards used to build ontology's. Ontology's transforms information in a machine readable format. The emergence of ontology actually brings revolution in Artificial Intelligence and then it's being used everywhere. The use of semantics is easily understandable by machines and humans; we can say it provides a common understandable language between the human and the system. Ontology actually gives the abstract overview of the system and defines the domain. Ontology actually is based on the

concepts similar in way object oriented programming is based on the object. The concepts are used to actually classify the domain and each concept depicts some properties. Relationships can also be formed. Ontology is much stronger concept than OOP. Defining ontology for a domain requires a language. Languages are used to define ontology's there have been many languages which are used to define the ontology, which are RDF/RDF Schema, Description Logic, DAML+OIL, and OWL. There have been many other languages which are used to define ontology but we have just listed some of the majorly used one's as those are not in our scope.

2.8 Service Composition Languages

Service composition languages play vital role for effective communication and increase interoperability among the service. Standardization of languages in inter-cloud is a big challenge. There is a lot work on the standardization of languages but still there aren't widely accepted standards which can overcome language interoperability challenge.

Composition languages have great significance to compose useful service composition. There have been many languages developed to compose services such as Business process execution language (BPEL), WSCL (Web services composition language), XML based languages etc [10]. There is a need to adopt some languages as common standard for service composition. This will ease the composition process and more useful techniques can be developed to increase the QoS in service composition.

2.9 Factors to improve Service Composition

2.9.1 Provision of services across the cloud

Emergence of intercloud emphasizes on the provision of services across the cloud. Service composition process should be able to get service from any of the clouds in intercloud environment instead of just able to get the service within a cloud. Interoperability in this scenario becomes more challenging. For provision of a service within cloud; the only interoperability issue involved is how both the services will be interoperable and are designed to work with each other independently. Interoperable services across the cloud are difficult to achieve and is a time taking task. It involves several other steps. Before proceeding for a service selection, platform interoperability should be considered. If platforms are not interoperable, then it would not be

possible to combine services. Semantic web (RDF) introduced in [23] can be used for service composition across the clouds. According to the framework, each cloud provider actually will have resource catalog which contains all the details of resources a cloud has. In a general term, resources are termed as services. Our discussion is related to services so we will use the term service instead of services. Cloud providers will have catalog of all the services they offer along with the information of interoperability details. This would make the service composition process easy, instead of searching whole cloud only the service catalog will be looked to get services which are interoperable.

2.9.2 Service Development

Better services can be designed by putting more focus on the service creation process. Better services can provide better service composition. [25] Describes adaptable service system life cycle functionalities. Among those functionalities the 1st one is service creation. We will discuss here only the service creation remaining functionalities are not of our concern in the thesis. Standardization of services at the point of creation will be a big success for service composition. We already discussed in detail the standardization in previous chapters and we agreed that consensus on common standard is difficult but necessary for the success of intercloud service composition. After studying the adaptable service system life cycle functionalities, I come to a conclusion that it is better to use service creation sub functionalities as a standard for service creation. The sub features look very comprehensive and meaningful. We have referred service creation and its sub features to be used as a standard for service creation. In the following paragraph, each sub feature is explained in detail. During the discussion, a question might come that why we are focusing on service creation. Our major concern is service composition. The answer is, service composition is actually a pool of resources or services, which are used together to solve complex problems. Now, if you don't have useful services then how a good pool of services will be formed. Due to distributed nature of cloud computing, heterogeneity is the emerging challenge, which is a big concern but this cannot be stopped because cloud concept is going as internet was growing in late 90s. It is better to put more focus on the foundation rather on the top. Standard for service creation can be formed and all the stake holders or service providers should be aware of the fact that you are free to develop services but during development at least you are required to maintain the service creation standard. The four features used by [25] can be further extended to standardize service creation. Following are the features,

which service should contain service specification, service integration, service validation and service repository [25].

2.9.3 Service specification

Service specification describes the specification which means; why service is designed, what are the functionalities and in which areas service can be useful. This specification elaborates and clearly defines the functionality of service, which will make it easier for the service composition program to pick up the useful services. The running platform information should also be mentioned in the service, which will clarify the interoperability. Standardization of service specification will be very helpful for service discovery. If the services are specified according to common standard, it is easy to develop useful soft wares, which can quickly discover required services. Furthermore, clarifying the concept, we will take a simple example of library. Suppose, books are categorized on the basis of subjects but within the shelves no proper sequence is followed and books are placed in different order in each shelf. If you have to search some books from different subjects guess how much time consuming it would. You will first go the shelves and locate the book but when you will go to another shelve the sequence is different so you will again try to understand in which sequence the books are engaged. Each time you require a book you need to understand the sequence, in which they are arranges. Now take another scenario, if the books sequence is described at library entrance and all the books are placed according to sequence specified. It would be very easy to locate the book. Similarly, if the service providers specify their services according to one common standard it will be easy for the manipulation software to locate useful services in a less time which eventually will reduce the cost and increase performance.

2.9.4 Service validation

Service validation will work as a check on service, which will make sure that the service actually performs according to the specifications. Service validation should also be included as standard. When a service is designed and its specifications are described, then it should be primary to responsibility of the service provider to check whether the service is performing the required functionality or not. This will improve the performance of the service and service pool in which it is added. During the service discovery process for service composition service validation will help to choose the right service. This actually means, if we have a service which has very good

service specification but do not have good performance. There might be chance the software can choose the service on the basis of specification but later it proves to be a failed service therefore, it is necessary to 1st look the service specification and then check the service validity feature if the service validation feedback is satisfactory; service will picked up for composition otherwise search for a better service will continue.

2.9.5 Service repository

Service repository is maintained by every service providers, which will provide information the number of services a service provider is offering. It is the responsibility of service provider that repository should be updated at earliest priority if any new version of service comes or any update. Maintaining a service repository will be useful especially for clients. It is easy to have a look on latest services available.

2.9.6 Service Negotiation

Service negotiation and composition is difficult to achieve without better semantic descriptions. There is need to describe semantics in detail so that service negotiation can be achieved which will eventually improves service composition. UDDI is semantic mechanism discussed in [23]. The idea is based on taxonomies and is called tModel [23]. TModel is not considered very useful to achieve negotiation and composition. The taxonomy does not provide any means to discover services. TModel actually provides namespace for taxonomy but because of service discovery limitation model is not accepted widely. Use of RDF/OWL with UDDI can very useful for service negotiation and composition. Ontology languages define the whole domain with the advantage of using query. Queries can be executed, which are very helpful to discover resources in the intercloud environment. Ontology based models are very useful for service composition. Service composition involves many entities in the process. Hence, it is necessary to clearly define the domain on which the entities will work. Good ontology model is helpful to create SLA and policies. Before proceeding for service composition it is better to define the domain completely with ontology model. The service providers are able to query the ontology databases whenever they want. SPARQL is a useful language to define ontology and provide good query interface.

2.9.7 SLA service

Services, which help in developing good SLAs among the entities in cloud environment, can be formed. SLA creation will itself be offered as a service to improve SLAs and this will reduce time required to create SLAs.

2.10 Service Composition Challenges

Service composition in inter-clouds is still a new topic. There are many challenges involved in it. Service composition for static applications is not a big problem and can be solved easily. As use of web services is increasing day by day, applications are mostly accessible online and user demands change on claim, this dynamicity is big problem in composing services.

2.10.1 Cost and performance efficient services

Cloud subscriber requirements should be matched with the resources provided by cloud before going for the composition. Many cloud approaches use third parties to implement this. There is strong need to provide trust between third parties, service providers and with the service providers in order to figure out which cloud source is more suitable for users. Better use of web services in cloud architecture increases the use of services, user interactions and the service providers, which look apparently good but there is a big problem to see which cloud resource is more suitable for the user. An agent based service is proposed in [5] which are good for dynamic contracting but due to involvement of several agents trust can be compromised and it will be difficult to reach trustworthy environment.

2.10.2 Service addition in Service Pool

Suppose, if another service is to be added on the composite service during the service composition, there should be a mechanism to add that service to the composite services. There is already a mechanism proposed by [14] which is called Compostable service Middleware (CSA-MW). This mechanism is actually very good and can be used to add an individual service to composite services pool. Trust can also be established by reviewing log of services. Log of service means the record of the services provided to the users. We have introduced a new service called ‘registry service’ whose purpose is to evaluate the log of each service which would be part of composite service. The registry service can be called as an application programming interface

(API) during the service composition process. It will go through log of the services and on the basis of the reputation it can decide whether a service is trustable or not. The evaluation mechanism can be of different types. Currently, we are considering three evaluation criteria. It can be on the basis of Feedbacks, registered complaints or profit in business. Nevertheless, feedback and complaint mechanism look more appropriate. So, during the service composition process, API is called and it will evaluate service logs. Registry services can be called differently if each cloud providers has own Registry services and it's only task is to keep tract of the services. All the registry services in service clouds should be managed by a central cloud which we call it registry cloud. During service composition, this service from this cloud can be used. This will raise another problem of privacy. Every cloud has own privacy policies, the registry service provider should agree with the privacy policy of the cloud whose service is to be proposed.

SLAs play vital role during the service composition. Good SLAs actually define Quality of service and security requirements between subscribers and service providers [14]. Several processes are involved in service composition and building a consensus on majority of processes is a big task, which is difficult to achieve without a good service level agreements. There should be an autonomous body whose sole purpose is to make good SLAs among the entities. When we talk about clouds; we actually talk about services so going for service which helps in signing SLA between the subscriber and service provider can be helpful.

2.10.3 Interoperable Services Discovery

Service discovery is in focus from the beginning of cloud computing concept and there have been several techniques and mechanisms available to tackle this. Service discovery in intercloud and during the service composition is still a novel topic. There is a need of techniques which can discover required services at earliest priority. Discovery of a service for composition is quite different from an individual service because of many participants. Services which are to be discovered should be interoperable. So, both requirements should be fulfilled at the same time. Demand of providing services (service composition) with less delay is increasing day by day due to the growing use of Intercloud infrastructure. Subscribers require earliest resolution of their problems with less delay. It's a challenge to discover inter operable services. Mostly, cloud providers are scattered and provide services independently. It's difficult to discover interpretable

services from other cloud providers and the biggest concern is the delay involved. Mechanisms are needed, which can discover interoperable services with minimum latency. Standardization of services can be very useful to resolve the problem. If services are standardized and they use same interoperability standards, service can be discovered as soon they are required.

2.10.4 Dynamic Service Composition

Dynamic service composition is increasing day by day due to large increase in user demands. There have been several frameworks available to solve this problem but still there is a need of more comprehensive platforms, which provide cost affective dynamic services. DynamicCoS Framework is designed to facilitate users in service composition. The frameworks allows user to discover, select and compose services [21]. Framework is one of the earlier proposed schemes for service composition at user end. The template is usable by only users who have technical knowledge of service. A layman cannot use the framework until and unless he has knowledge of semantics. Services are discovered in a quick time as the user's already had knowledge of semantics. System developer play vital role in service composition. System developer provides information of the semantics via DynamicCoS framework. In return, the users specify their requirements according to the semantics provide by the system developers and then user requirements are processed by the system developers. The provision of information by system developers makes their job easy because the user requirements they get, are already in a form which are easy to understand and this makes the service discovery and composition task easy. Framework actually emphasizes on the more active role of end users. Users should be given more active role because they are the one who can better describe their requirements. There is strong need to develop frameworks for every category of user. Service composition in intercloud is still a novel topic but its use is increasing day by day. In future, users will increase and they should be given active role in service composition. There is a need of development of different semantics which should be understandable for every type of user. No matter if a user is technical or non-technical. If we can describe the semantics in which user should enlist his/her requirements, service composition process will be speed up. Comprehensive user templates should be designed and their use should be encouraged. A requirement varies from user to user as each have own preferences and way of describing what they need. Semantic languages can be very helpful for this. User can use any of the language to describe his/her requirements which

eventually will better the service providers understanding and speed up the service composition process.

2.10.5 Services management

Service management includes how to initiate the service composition process, how the services will coordinate with each other and how services will be decomposed. Service coordination with each other is already discussed in the thesis. Erdal SLMS scheme [11] can be very useful to initiate the service composition. Although, the scheme focuses on locating and migrating of federations, Service composition scenario and hierarchy is related to federations. Proposed SLMS scheme can be used to locate already composed services and to migrate composed services from one cloud to another or generally we say from one location to another. Composed services locating and migrating issue can be resolved by the scheme. For service composition initialization, the two way Algorithm is proposed in [11] which can be used and will be very effective. Similarly algorithm can be used to reconfigure the composed services. By reconfiguring, we actually mean to re-use the existing combination of services. Service management is another emerging challenge, which we have highlighted in the paper and used the SLMS scheme to solve initialization and reconfiguration problem. Idea for effective service management is already proposed in the paper, where building consensus on coordinator is proposed.

After composition of the services, a question arises that who will manage the services since several services participate at the same the time it's difficult manage them at a time. Look for techniques to handle services after the composition. Composed services act like computer networks and there is a need of proper protocols, techniques which can manage them. Several procedures are involved in this like which service will perform the task 1st and what will be order of theory execution, all services can work at a same time and then single service can combine the result. It is better to choose coordinator node (service) among the composed services and that service will coordinate with other nodes in accomplishing the task. But this will increase the overhead of the services and their might be a responsibility that functionality of that particular node cab be effected. Coordinator role can also be provided as a service and after the service composition the coordinator service can be called to manage all the modes. Schemes are needed

which can actually manage the service during the execution. If the same coordinator from the service needs to be selected a question arises, how to elect coordinator from the pool of services. A distributed Flooding consensus [29] algorithm can be used to build consensus on a particularly service to choose the coordinator.

Coordinator selection can also be done by adopting a tree approach for service composition. We will start with a root services and then the services, which are chosen next to the root services can be represented as leafs of the root services. In this way, we will keep on adding services until the required service composition is done. Every node will be held accountable for tasks performed by its leaf. The service response time will be quicker by adopting tree approach for service composition.

2.10.6 Cloud federates for service composition

Federate clouds can also be implemented for service composition but this is still a novel topic and hasn't been discussed so much. Implementation of federated clouds creates new challenges of load acceptance. Cloud providers need mechanisms to accept load for resource renewal. Service usage by multiple clouds requires obligatory monitoring, which assures best performance among cloud providers. Monitoring in federated clouds is easy to tackle when there is single execution going on but arouses many challenges for Virtual Execution Environment (VEE) transferring to remote clouds .There is no existing proper standard among cloud providers by which they can share interfaces. Standardization of interfaces is a big hurdle in federated clouds implementation. Standards need to be generalized, which can address federation according to need of consumers. Standardization of hybrid cloud is required, which will provide common interaction interface to cloud providers. Implementation of federated clouds introduces service clouds, which are used to handle multiple services. Handling multiple services require rigorous monitoring of services handled by service clouds. Emergence of service clouds arouses new challenges. Stuart Clayman introduces idea of service catalog in service clouds. Service catalog is proposed to store updated status of consumers, offered services and workload migration procedures [9]. Provisioning of services by cloud providers should be located in federation of clouds. Utilization of load among service providers is another big challenge. This challenge can be resolved by creating mechanisms, which will find location of consumer. By knowing, from

where request is generated, most suitable cloud providers from that region are assigned to consumer, which will improve QoS and services can be assigned more efficiently. This idea is very useful to choose services, which are nearer. Implementation requires development of more and more Virtual Execution Environments (VEE). Federate implementation among heterogeneous clouds also requires authentication mechanisms (discussed earlier in interoperability) which will improve coordination among clouds. Single sign in scheme can be best suitable for authentication in intercloud architecture [6]. Addressing the discussed issue, it will help in improving collaboration and multiservice deployments among clouds.

2.10.7 Measuring QoS (Services and networks)

Need of new techniques, which can measure the QoS of services and network separately. [22] Proposed a scheme for measuring QoS of services and networks separately. The scheme is divided into three tasks. It starts by locating service, which are near to the user in order reduce network latency. QoS measurement mechanism is proposed, which calculates the latency and transfer rate of the data. The calculation helps in identifying the network delay factor involved in service composition. The third step includes an algorithm, which actually calculates the latency for service composition. The calculation of algorithm is very accurate and is considered as the best latency algorithm proposed so for. Network latency is calculated by adopting a network coordinate system approach, which actually calculates the latency between any two networks. On the basis of that initial value latency between other networks is calculated. After that a hash table scheme is adopted, which actually helps in locating nearby services so that the latency factor should not become a key in communication. Latency, cost and service availability are considered as QoS attributes which should be fulfilled to provide good QoS to users.

Network impact on QoS increases with increase distribution of services. User should mention the network latency factor in SLAs at the time of contract. Latency directly depends on the number hops involved from source to destination. To reduce the network impact, it is better to improve the service discovery and nearest available services should be offered to accomplish the task. Shortest path algorithms can be used to discover services, which at shorter hops from the source and destination. Shortest path can also be determined on the basis of latency like 100ms, 200ms etc. scalability of services on cloud is increased a lot and it’s increasing at a rapid speed. There

are thousands of services for the same functionality or in some scenarios a single service providers offers several same type of services. The benefit of having several options is each service providers has own SLA and the services actually differ in QoS which different categories of user. User can easily choose the service whose SLA is according to their requirements. Service providers SLAs for a single service also varies from user to user on the basis of user location. The number of resources Service provider are using directly affects the terms in SLA. There might be a chance that the resources used for one client can be more than the resources required for another user.

2.10.8 Service Decomposing

We have discussed service composition a lot but service decomposing can be another big challenge. Like service composition, service decomposition also involves many procedures and there is a strong need to create mechanisms and services for service decomposing. There are no such techniques, which will solve the problem of releasing resources, maintaining trust during the process, removing data of other entities without any data loss, how to coordinate with user and third parties etc. All these issues need to be addressed. We discussed some of these issues briefly and tried to give some general understanding how they will be addressed. Following are the major decomposing challenges.

1. **Data removal**: During the problem solving process, services use each other's data freely. After the solution of the problem that data needs to be deleted from the databases of all the services. A better approach can be to make SLAs, which should include deletion of data (which didn't belong to the service) from all the resources and sending an acknowledgement to the service whose data is deleted. The acknowledgement mechanism will also create trust between the two services and will be helpful in strengthen trust between the services.
2. **Releasing Resources**: Services use each other's resources to accomplish the specified tasks, upon completion of tasks resources needs to be released at earliest priority so that the owner of resources can utilize them on some other problem. Resource scheduling algorithms (Operating Systems) can be used to schedule resource. Resources have three states which is running, ready and blocked. During the running state, essentially the

resources are occupied and some job is running on them. During the ready state, a resource is actually waiting to be used by some service, while the block state shows that resource is waiting for an event.

Note: (Further details will be written later)

3. **Dependencies**: Dependencies can delay the release of resources. For example, if a particular resource is occupied by service and that service is still busy in doing the job. The former service has to wait for the resource until the other party can release the source.

2.10.9 Service Recomposing

Service composition reusability is another important benefit, which we can take from service composition. We will name this as service recomposing. Let's consider a scenario that we have solved a complex problem successfully. After that a new came with similar details. The better approach will be to reuse the composed services instead of again going for the whole process which will be time consuming and costly. We need to perform all the procedures required for service composition. The most difficult one is the trust, which we need to develop it from scratch. Regardless of this, we can recompose all those services which have been used previously to solve the problem. We will introduce a new service called decomposition service, whose only task will be to recompose the services which were used previously. This service will take all the relevant information from the registry services, which we will discuss later in the Trust development. Registry services will contain information of all the previous correspondences a service had. Service recomposing will affect three parameters directly.

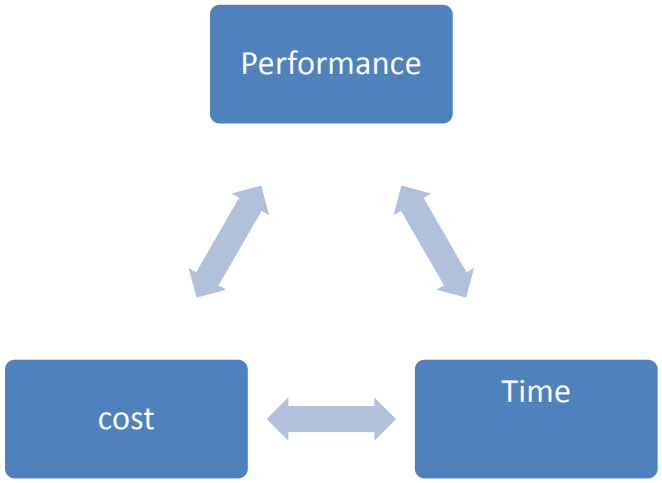

Figure 1: Performance, cost and Time constraints

Right now service recomposing is a new concept and it's being in IT market from last 4 years. In future, service recomposing will become popular because of its performance, cost and time relationship. Services will be composed only once and after that they will be recomposed when required.

2.10.10 Backup and Recovery

Service composition process involves several services and they work together to accomplish the task. Client on the other hand has nothing to do with technical issues involved in combining services. His primary concern is the on- time solution of the problem. We already have discussed many issues involved in composition process. Backup plan has not been discussed yet. We will consider a service crash scenario. What if a service fails to perform the task or service is crashed? There is a need of backup plan, by which new service with same features should be added to pool of services or if not all the tasks performed by service should be backed up somewhere and upon crash the new service can be proved from there or the existing service upon recovery can resume from there. Backup techniques for service composition in intercloud are needed, which can better handle the crash situation on the cloud. This is considered as an emerging challenge. At the time of SLA finalization, clients also ask about any back up plan because without a useful back up technique, it is risk to continue with problem solution.

2.10.11 Capability adaption in service systems

Designed services should be adaptable. The services participating in service composition should posse's adaptable property. Adaptable behavior actually means the service should be interoperable with services from different platforms and can adopt according to requirements of clouds. There is need of open source services and they should not be bound to specific clods rather they should possess characteristics by which they can be adjustable with other services. Service composition is considered in a broader scenario, where there is no bound on the domain. Service can be taken from anywhere in the cloud and the service should be able to adapt according to composition requirements. Standardization of common standards can solve this problem. If there is a common generalized standard on the basis of which services should be designed. Each service providers should follow that common standard during the development can be very useful to resolve this problem. Existence of common standards will speed up the service composition and it would be easy to reach the consensus. This would also resolve many interoperability issues but reaching a consensus on a standard is not an easy task. Now a days, hundreds of vendors are in the market and every vendor have their own standards. It is very difficult to reach a consensus. A consensus can be reached, if emphasize should be given on some common standard for service composition and that would be more acceptable. If consensus is built on such a common standard, then discovering a service for composition will be a lot easier as compared to current scenario. Adaptable services should possess functionalities discussed in [25].

2.10.12 Cloud Heterogeneity

Day by day, more and more clouds are coming in cloud environment. Due to this heterogeneity of clouds, it is becoming a big challenge to compose services. Every cloud possesses its own specific properties and it is difficult to come with a common standard for inter-clouds. Clouds have their own firewall settings, privacy policies, procedures etc. That makes it complicated to compose the services. By creating good composition techniques this problem can be overcome.

2.10.13 Cloud data format

Data format is another big hurdle for service composition. There exists different data formats implemented by clouds; it is hard to communicate in presence of different data formats. If common data formats are introduced they can be very helpful in reducing the complexity of data format. Same data format for several inter-clouds can be used, which would be quite practical to compose services. Cloud, libcloud and deltacloud APIs can be used to bound several inter-clouds to use same data format [7].

2.10.14 Untrustworthy services

In an inter-cloud environment, there are thousands of services running and all the services are not trustworthy neither standardized. The growing trend of cloud computing and the inter-cloud concept has encouraged the small business organizations to put their services on the clouds to get business. This trend has benefitted the subscribers but has raised a serious concern for the service composition. If any untrustworthy service is included in the composed services, it will affect the overall problem resolution and will finally affect the business. Subscriber now days require services, which are authenticated and trustable. Distinguishing untrustworthy services from the trusted services is complicated. A simple question is how you would separate services from pool of services? There is not too much work done on this issue yet. Distinguishing an untrustworthy service before service composition is another emerging challenge in trustworthy service composition. Challenge can be tackled, if cloud implements more strict policies for including services.

2.10.15 Protocols development

Nowadays, many protocols exist for service composition and most of the protocols are good enough to compose service in a centralized environment. As dynamic service composition is gaining popularity on cloud environments, there is a need to develop trustworthy protocols. Trustworthy service composition protocols will allow fast composition of services due to better rules and trust environment for service composition. There is a need to develop more and more protocols to benefit trustworthy service composition in a distributed environment. When we discuss trustworthy service composition protocols, we need to consider two types of networks; which are client-server networks and second one is the Ad-Hoc networks. In client-server

networks there is already a lot of work done but the real challenge is making protocols for ad Hoc networks. In Ad Hoc networks nodes act separately. There is no hierarchy; each node is on the same level. Protocol for trustworthy service composition in Ad Hoc networks is a challenge and resolution of this will promote dynamic service composition in Ad Hoc Networks. Broker-based composition architecture is proposed in [19] and Ad Hoc Network scenario is also discussed in detail but the protocol discussed will only work to discover services. Discovery of trust worthy service composition is still a challenge in Ad Hoc network. In broker based approach the author has discussed the role of coordinator in Ad Hoc networks. One of the Nodes from Network is selected as a coordinator and then that node is used to communicate with all the other nodes in the network. All the requests for new service discovery and composition are handled by the coordinator [19]. Another protocol for composite service making is proposed in [20]. The consensus approach is used in the protocol. During the composition process, a consensus among the participating services is tried to reach and on the basis of consensus, it is decided services should be composed or not [20]. Study some other protocols on the topic. Have a look on ref [8] content distribution protocol.

2.10.16 Robust Security Techniques

Existing security techniques are not up to the level of security needed for service composition, which eventually affects the user confidence on a service. There is a need of more robust security techniques, which allow service composition in more confidential environment. The whole concept in service composition is that everything can be offered as service. During the service provision, anything, which is required can be requested and get it on demand. Security can be offered as a service.

2.10.17 Lack of interoperable mechanisms

There is strong need to build frameworks, which can provide better interoperability among the services. There are circumstances, in which services are composed because they fulfill the criteria required for problem resolution but due to lack of interoperable we can’t choose the service. This problem has made it difficult to find the useful services, which are interoperable and service discovery response time is increased to a greater extent. The cloud resources are wasted due to non-interoperability issue and will also affect the business badly. Just imagine a

scenario, that you have guts to do the work but you are not allowed to do because you are not good in communicating with other. Frameworks need to be developed, which can address this issue.

2.10.18 End user's templates development

End user templates are in discussion from the beginning of service composition. Service users are divided in several categories. It is a difficult task to design user templates, which can fulfill the requirements of all types of users. By having a single general template, it will be quicker to communicate with the user. Templates can be divided on the basis of services, a user needs, rather than classifying on the basis of users. User classification makes development of templates difficult. As each user has own specific requirements and understanding. Educating a user is another tedious task and is time consuming. There is necessity of user templates, which can be divided on the basis of services like security service, e-commerce service, gaming service etc.

2.10.19 Service composition in Peer-2-Peer Networks

Currently, most of the discussion on the service composition is focused on the client-server architecture. The existing techniques mostly address the service composition in client-server network. Service composition in P2P networks is still a novel topic. In P2P networks the total approach is decentralized and all the nodes are at the same level. Coordination among nodes becomes difficult, which hence increase the complexity to compose services. There is a need of mechanisms to do service composition in P2P Networks.

2.11 Service composition beneficial areas

Service composition is highly beneficial to solve complex problems. Its major applications are in space research, Robotics and e-science. All these fields require high computation and the problems are very difficult to solve even it becomes hard for a single individual service to compute. Combining several services together can increase performance and solve complex problems. By using good Cost effective algorithms, service composition can also be a cheap solution. Greedy Algorithm for fractional Knapsack (Algorithms) can be used.

Service composition concept has increased the business opportunities everywhere and especially in small business organizations. Before the emergence of service composition small businesses were struggling to step into inter-cloud. Inter-cloud is a new paradigm and the existing market leaders like Microsoft, Amazon, Google etc are leading in this because of enough resources to invest in cloud environment. Intercloud is a new concept and due to lack of confidence in the technology the subscribers or cloud users are not willing to invest in new organization even though if they provide better services. So far, for small business companies, putting an individual service in a cloud is not only costly but it ends up a loss in business. Service composition emerged due to increase in demand. Choosing the best services from the cloud or web services, is the main goal for service composition, which eventually improves the QoS. This idea gives opportunity to small business providers. Best services can be picked for service composition, which will eventually gain consumer confidence and will provide them the opportunity to offer service at low expenses.

2.12 Web based Service Composition

	Major Modules	*Communication Standards*	*Content and Business Process Standards*	*Key Technologies*
WebSphere	Application Server, MQSeries, Business Components, WebSphere Commerce	MQSeries, JMS, IIOP, SOAP, HTTP	WSDL, XML, RosettaNet-PIP, cXML, EDI	Components (J2EE), XML, Web Services
ONE	Forte tools and	JMS,	SOAP,	EDI, XML, WSDL	Components

	iPlanet	LDAP, WAP, IIOP, HTTP		(J2EE), XML, Web services and Work°ow
e ration Server	Integration Modeler, System Monitoring and Administration, Business Process Monitor, Business Intelligence	Oracle Queue, JMS, SOAP, IIOP, MQSeries, TIBCO/rendezvous, HTTP	XML, WSDL, EDI, RosettaNet-PIP, ebXML	Components (J2EE), work-°ow, XML, data mining, Web services
etAction	HP Opencall, HP Chat, HP NetAction Internet Operating Environment	SOAP, JMS, IIOP, HTTP	XML, WSDL	Components (J2EE), XML, work°ow (ChangeEngine), Web services
soft .NET	.NET Framework and Tools, .NET Enterprise Servers, .NET Service Building Blocks	MSMQ, SOAP, Microsoft Host Integration Server, HTTP	XML, WSDL, RosettaNet-PIP, XLANG from BizTalk Server	DCOM, MSMQ, Web services, XML, BizTalk Orchestration Engine
WebLogic rator	Application Server, Application Integration, Business Process Management, B2B	SOAP, JMS, IIOP, HTTP	WSDL, XML, RosettaNet-PIP, BEA-XOCP	Components (J2EE), XML, work°ow, Web services

	Integration			
Methods	Enterprise Server, Enterprise Adaptor, and Enterprise Rule Agent	SOAP, IIOP, JMS, HTTP	WSDL, XML, EDI, RosettaNet-PIP, ebXML, cXML, OBI	Components, work°ow, Web services and Agents
a Business	Business Process Management, B2B Communications, Enterprise Application Integration and Real-Time Analysis	SOAP, IIOP, JMS, HTTP	XML, EDI, RosettaNet-PIP, ebXML, xCBL, cXML	Components, XML, work-°ow, process model, process analysis
O *e* *prise*	InConcert, IntegrationManager, MessageBroker,	SOAP, JMS, IIOP, MQSeries, HTTP	WSDL, XML, HL7, EDI, RosettaNet-PIP,	Messaging software, XML, work-

Table 1: Deployment Platforms-1[26]

Web service composition has gained popularity and is being used vastly to compose services. Due to vast use of web services, web service composition is the best way to facilitate users with the features of service composition. Web service itself is offered as a service and user, who needs services, they can generate their requests through the deployment platforms. Deployment platforms are needed for web based service composition. Without a deployment platforms web based service composition is difficult to implement. Several platforms are available to deploy web based service composition. Above table contains some of the major available platforms.

3 Cloud and Trust

Trustworthy communication in intercloud is the growing demand of the users. With the emergence of service composition, trust has become vital factor to do the communication. In the next chapter we have proposed a trustworthy conceptual model for manual service composition, which will help in building the trust in less time and increase the accuracy of the work. In this chapter, we have discussed some of the emerging challenges to reach trust in cloud environment specifically during the user and service provider interaction and among the composed services. Other than that, some factors which help trust and some general terms related to trsut are discussed. Data is the most important for the organization or we can say; it is very important for every entity involved in service composition. Every entity should trust on all other entities involved in the composition. There is a need for mechanisms that can develop trust so that every entity trust on the other entity while sharing the data. It's not only the mechanism, which can address this problem. When data is evolved every organization has its own rules. For instance, most of the companies have a rule that they do not share their storage media to any one and normal procedure is dispossing. We need to take into account all these rules and regulation before going for trust development among the services. Chapter discusses emerging challenges to build trust.

3.1 Trust

We will start with some conventional definitions of the trust. Following are the definitions

Trust defined as in the Webster dictionary, is:

1. "An assumed reliance on some person or thing. A confident dependence on the character, ability, strength or truth of someone or something
2. A charge or duty imposed in faith or confidence or as a condition of a relationship.
3. To place confidence (in an entity)." [31]

"We define *trust* as "the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context" (assuming dependability covers reliability and timeliness).'' [31]

"Gambetta defines trust as a particular level of the subjective probability with which an agent performs a particular action, before it can monitor the action and in a context in which it affects others actions." [27]

"Trust is the extent to which one party is willing to depend on
Somebody or something, in a given situation with a feeling of relative security, even though negative consequences are possible." [28]
"Mui *et al.*[27] refer to past encounters and state that trust is a subjective expectation an agent has about another agent`s future behavior based on history of their encounters. " [38]

3.2 Cloud from Trust point of view

IaaS and SaaS require several services to work together to facilitate the user. The involvement of several parties and mechanisms requires high degree of trust, which can be established by adopted processes, which can help in reaching trust. However, trust level also depends on the category of the cloud service belongs too. That means service is either from public or private cloud. There is one more category, which is actually formed by combining two clouds. Cloud environment is classified into three major categories which are public, private and Hybrid cloud. Each category has its own benefits and cloud depends on type of services and collaboration required.

3.2.1 Public Cloud

Public clouds are used, where a lot of collaboration is required and resources share across the enterprises and users from all the parties can freely use resources. From trust point of view, this category is not trustworthy and there are security vulnerabilities. In a shared cloud, the users normally use resources, which are allocated to them or might use which they are authorized. There might be a chance of a hacker, who can act as an alias user and use resources, which are not actually required and later breach the data privacy. Encryption techniques can be very useful to overcome this issue. If services pool contains services from public clouds, then there might be a chance of security breach due to a lot of data sharing. So, from reliable communication point of view public clouds are not recommended. They can be more helpful in scenarios, in which individual services are required to fulfill the user requirements or where user privacy is not a primary concern.

3.2.2 Private Cloud

Private clouds are more secure and difficult to breach security since only one enterprise handles all the data. So, it is not possible for intruder to get the data as no one outside the organization

can access the data. Stronger firewall implementation can resolve the issue of accessing private data but it is very costly and still there is risk to data. Uses of service, which belong to a private cloud are highly recommended and help in establishing trustworthy communication among the entities. During the service composition process, services should be taken from private clouds so the data privacy should be maintained, which will make the user feel better. Service discovery mechanisms should be made more efficient so that they can mostly discover services from private clouds. There is a general perception that private clouds are more secure than the public cloud so for non-technical user this would be enough to trust the provider.

3.2.3 Hybrid Cloud

Now, we move to Hybrid clouds, they actually have both (public and private) cloud features. The future focus is on hybrid cloud. Mostly, we need some services, which require high privacy and there are some services which can be general purpose and should be put on the public clouds. This strategy will save the cost. Service can be composed by choosing them from hybrid cloud. Service which requires high privacy and data security should be taken from private cloud where as the service which does not require high data security and privacy standards having low budget, should be taken from Hybrid cloud. There is greater need to distinguish among the public, private and hybrid clouds and their importance should be realized. The cost effective solution can only be chosen, if we consider the cloud category factor and choose service according to the requirement and their utilization.

Access control mechanism is required, which can limit the user access in public and hybrid clouds. Access control mechanisms are also required in private clouds but not of that extent as used in public and Hybrid. During the composition process, services are taken from different enterprises (clouds) and then they are used to access the data of the other clouds. There is a need of access control mechanism with in a pool of services, which can limit the services access. Limiting services will be helpful in getting enterprises confidence. Resources in service composition are allocated to services on temporary basis. So, strong access control mechanisms are required, which can actually restrain service from accessing other resources. [35] Proposed idea of credentials and is very useful. In order to develop trust between the two unknown parties, both the parties should have credentials. Here another question arises what if we the issuing authority (service) is not trustable or the other parties don’t trust the authority? Question is then

who will act as issuing authority? We have proposed the use of directory service and as we have discussed that each service provider should register the service with the directory and the directory service should make sure that the service is fulfilling all the requirements and trusted. Implementation of directory service will resolve this issue. This approach will really help in gaining trust among the services in composition and enterprises whose service will be frequently shared by the services. [35]

3.3 Trust Development (Manual vs. Automatic)

We have discussed in previous communications that service composition can be done in two ways, which are either manually or automatically. Manual service composition is a time taking process and cannot be done on demand basis. It is normally used, when immediate time constraint is not required and project is big enough. We follow all the procedures, which also involve formal meetings among the parties involved in the composition process. Trust is easy to reach in such a scenario because all the parties have time and they can share their concerns and will better elaborate what measures are taken place to build trust with other parties. So, trustworthy service composition is not a big concern in manual service composition. However, due to the growing trend on demand provision of resources and implementation of "pay-as-you-go" model automatic service, composition is a need that's the main reason cloud computing become so popular and enterprises are moving on clouds. Reaching trust in automatic service composition is more challenging and is still a open area of research. In automatic service composition there is a strong time constraint and resources are provided on demand basis. Trust development process should be very fast and trustworthy. Automatic service composition is dependent on service discovery mechanisms, the faster the service discovery process faster is the service selection process. In our proposed conceptual model, we have suggested layered approach for trust development and implementation of mode will help in building trust among the services and clients.

Data flow is considered as an emerging issue in service composition. When different services are put together to work on a problem, it is very important that all the services should be able to finish their tasks in time and develop a good coordination with other services. One might be confused of word coordination. By coordination, we actually mean the manner, with which interaction among the services will take place and the overall flow will be maintained. Data flow

means the sequence, in which services will perform their tasks. Flow charts will be a better idea to control the overall flow of data among the services. The flow control mechanism will act as a central body, which will assemble all the services and is considered as trustable mechanism to reach trust among the participating entities. Flow control management is a key factor to develop trust among the parties because if the providers trust the flow control mechanism, they will automatically trust the participating entities with the composition process.

Non deterministic behavior is another factor, which affects the trustworthy communication. This issue has already been discussed several times but in a different perspective. We emphasized on the use of registered service which is a measure to prevent services that exhibits non deterministic behavior. Non-deterministic actually means the service behavior or performance changes according to the scenario. There might be a chance that you chose a service due to its history but it didn't perform according to the history. This random behavior needs to deal. There are several other factors, which affect the behavior of service involving the interoperability issues, security breach problem, data privacy checks etc. Due to growing usage of cloud environment, this factor is a considerable threat in a cloud environment. Day by day, enterprises are moving to cloud environment and it is difficult to restrict enterprises to develop non deterministic services. This is because there are thousands of users, who just use services free of cost and use services for fun or things which are not important for them. So, this is not a big issue. The best suitable option is to make sure the service which is to be chosen for the composition process and registered with some directory so that non deterministic factor should not count. Trust is badly affected due to non-deterministic behavior of services. Suppose, services are composed and each service is performing the tasks according to the functionalities. But one of the services is showing non deterministic behavior that leads to the failure of overall problem. This will cause lack of trust among the services and the other services will not continue with the service and in future the providers will not be willing work with the non-deterministic service. In order to avoid such consequences, it is better to choose registered services which don't exhibit non deterministic behavior.

3.4 Trust development with Third parties

Third parties play vital role in service composition and are consider as most authentic resource to describe user requirements. Trustworthy service composition ensures authenticity and

authorization of every entity involved in the composition process. We talk about trust between the clients and service providers a lot but we cannot ignore the trust development between the client and Third party and similarly among the third part and service provider. As we have already discussed in our previous conversations that the technical user trust requirements differ from the non-technical users. Third parties can be seen as technical users. In the same manner, same techniques can be used to develop the trust between the Third party and the service provider. Here, we have to bear in mind that the third parties are not one's who can make decisions by their selves. They are dependent on their clients but it's the responsibility of third party to interact with the client. Provider can share information of technical measures adopted to secure the user data on cloud to build trust. Likewise, the above mentioned mechanisms can be used to develop trust. The real point of highlighting this third party factor here is, one should realize the existence of third party. In a client and third party scenario, we are required to consider the whole scenario during which a client chooses a third party. In these days, there are several third party agents, software's are available which help client choosing services according to his/her requirements. In this scenario, trust development will be helpful in a quite similar way in which service providers and clients trust is developed. Third parties should convince the clients that their data will be secured and they are the best to cope user requirements and helping in discovering new services. There is a need of Models with which the agents can satisfy the clients and build their trust. From users perspective the best point for trust development is the agent should be competent enough to cope the user requirements. They should describe in a more technical way to the service provider so that user will get what he/she wants. Reputation is another factor, which can be used to locate trustable agents. This factor is very authentic way of checking an entity track record but this will not help in choosing new agents. As intercloud is a new environment so, this will help up to some extent but cannot be used as a standard to choose the agents. Trustworthy composition between the client third party and similarly between the third party and the service provider is vital for success of trustworthy service composition.

As we have discussed in the previous paragraph, reputation can be used to develop trust between the clients and third part agents. Nevertheless, this cannot be the only option because intercloud domain is not far spread. Most of the third party and clients are new. That is why; it is difficult to decide on the basis of the reputation, keeping in mind several that other techniques can be used. There can be several sources by which trustworthy communication can be established; we

already discussed some of them in our thesis. There are two more sources on which we can decide which part is the most trustable. First, one is the record of previous interaction between the client and the third party. There can be a possibility that client is already using third party to use services. The previous interaction history between the two parties can be analyzed and based on them it can be decided that the entity is trustable or not. However, this scenario is not common and we need some alternative to this. Second, source is to take opinion from other clients in the market. This seems very useful. Suppose if a client do not have any existing third party then it is better to interact with some other client and ask for the opinion about third agents. This technique can be very useful and resembles the real life scenario. In our social system, if we do not know about the person whom which we intend to do the business we ask our friends or any reputable person to give opinion and then that can be used as a basis for trust between the two parties similarly client can go for opinion to opt third parties so that to develop better trustable environment.

3.5 Trust (Subscriber perspective)

We have discussed many scenarios in trustworthy communication between the clients and service provider similarly among the service providers themselves. There is no doubt trust is the concern for all the parties participating in the composition process. In our previous discussion, we mostly emphasize that client is usually more concerned about the trust but the service provider shouldn't be ignored. Service provider has own concerns and trust requirements by clients should be ensured for successful communication. There might be a chance that client will misuse the resources of the provider, which will affect the provider business. This scenario will create untrustworthy environment between them. To avoid this scenario, clients also needs to ensure the service provider that the steps needed to build trust will be adopted and provider privacy and security requirement will not be avoided. This scenario actually is applicable on corporate clients or the service providers by itself who takes services from other service providers. Most of the individual service users do not include in this category. Let's take a scenario, a user who is using the free services or using services which are just meant for entertainment doesn't lie in this category. This category includes users who choose service composition for business use. Mostly the organizations are considered as client or small business organizations. The organizations, which are in cloud or planning to include in the intercloud

domain needs to adopt measures and mechanisms which can satisfy the service providers that security measures are adopted. Following figure shows a service composition scenario in a trustworthy environment

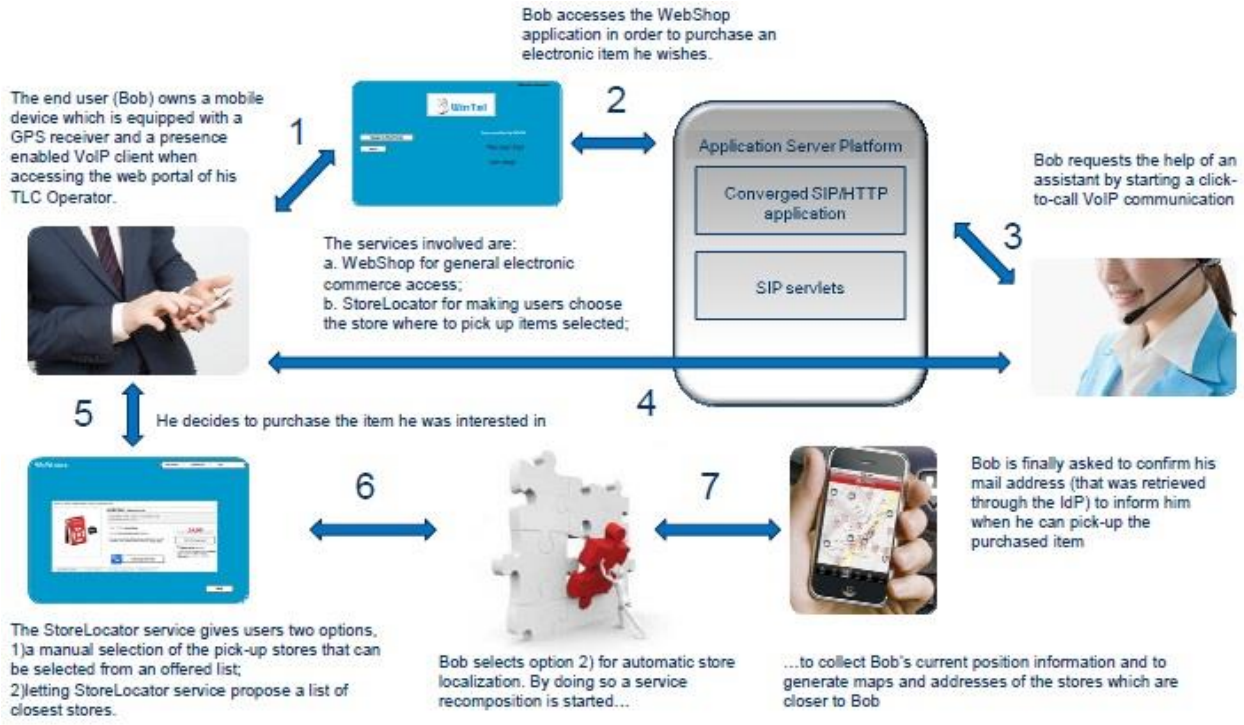

Figure 2: Service composition from user perspective [37]

3.6 Trust development (Social Aspect)

Trust doesn't only include the technical mechanisms but social factor are also involved. Trust is created due to mutual understanding among the entities involved in a composition process so its social aspect cannot be ignored. We will call social aspect of trust as behavioral trust. This term is used in several papers but discussed in a different sense. Behavioral trust actually includes all the mechanisms, which are implemented to improve trust among the entities. The feedback mechanisms and the reputation techniques lie in a behavioral trust and are used rapidly to know about a service provider and the user. Belief can be another factor, which is part of behavioral trust. Belief is directly proportional to the trust. Increase in belief will definitely increase in level of trust between the entities. A user having strong belief on service provider will result into taking service from the provider similarly during the inter services communication in a pool of

services. If the services have belief that the all other services are good this will result in a good trustworthy environment. According to [38] belief can be defined mathematically. Honesty is another factor, which is included in behavioral trust. Honesty of a service can be judged from the track record and can be used to develop trust among the entities.

Trustworthy service composition cannot be reached without considering assumption. This actually means trust has to start from some point otherwise it can't be reached. If every entity will wait for the other then trust cannot be reached. We have to take an assumption and initially consider one party as trustworthy only then further processes will be initiated to achieve trustworthy service composition. Suppose, if there is a central body which is responsible for the composition. Then initially, it should be assumed as trustworthy only then further communication will start. Third parties also need to be assumed trustable at some stages. Suppose, a user chooses a third party and this third party will be used to reach trust among the user and the service provider. Before starting communication between the third party and service provider, the third party should be assumed trustworthy. Assumption factor must be considered during the communication and will be used as starting point for trustworthy service composition.

3.7 Trust policies

Trust policies are another factor which affects the trust. Ontology is vastly used to describe the domain. These are very helpful in defining the semantics as we already discussed in our previous discussions. They can be used to define the policies. It will benefit to define the policies in a way, which is acceptable by both the user and the provider. By acceptable means, policies can actually be defined in a way that a layman can understand. As ontology is a semantic language, it defines the domain in technical way which is understandable by machine. Creating a policy, which is understandable by layman and machine helps in overall system performance and increase in Trust level otherwise separate policies format for user and provider will make translation problems and make job of machine readable software's difficult. So, trust policies should be designed by using any ontology language to benefit from the semantic languages. We have spoken about the importance of feedbacks to gain trust but accuracy factor shouldn't be ignorable. There is no surety that feedbacks are always reliable. There might be chance that a user can intentionally give wrong feedbacks. Keeping in mind this accuracy factor, there is a

need to create strong trust policies, which can overcome this issue. Just because of accuracy, we can't ignore the importance of feedbacks. Strong trust policies should be made and cloud user should be liable to the policies mentioned. Reputation mechanisms should be in place to see if a user is reputable or not. A simple mechanism to judge about the reputation of a user is to see he/she following the trust policies if not then the user is not reputable. A user, who is not reputable his/her feedback would be ignored and is considered as unauthentic. Reputation mechanism will overcome the problem of accuracy, which will help in gaining trust. Reputation can be used in another way. Reputation can be judged from user track record. If the user is reliable as per the previous record history, this can be used as a basis from feedback. However, this mechanism has two major issues. Firstly, there might be a chance that user is reputable but this time he/she can give wrong feedback. There is no mechanism in place, which actually overcome this problem. Secondly, this mechanism cannot be used for the new users. As new users have no track history, there is no mechanism by which we can judge the authenticity of the user. RATEWEB [38] model is very helpful to elaborate the reputation and opinion approach in the trustworthy service composition.

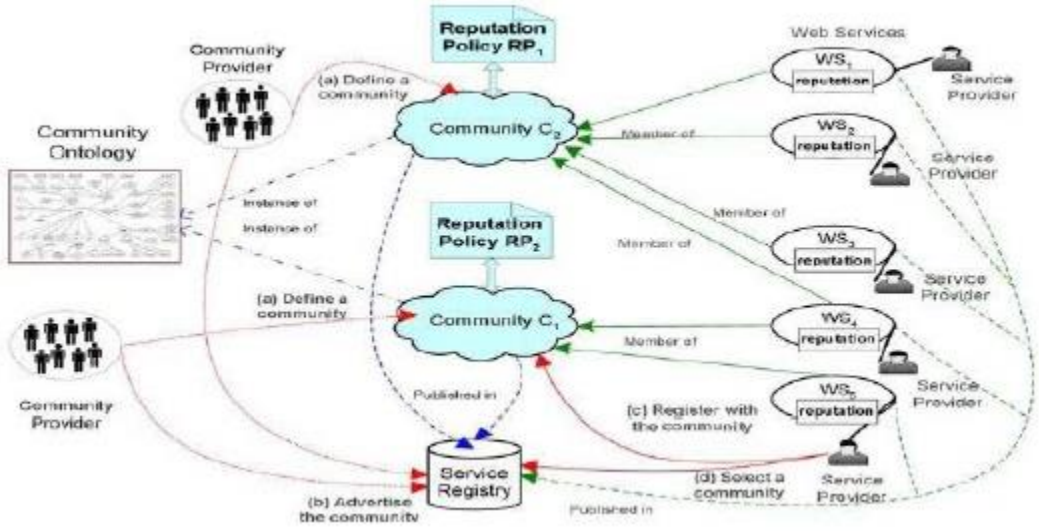

Figure 3: Trust Policy overview [38]

3.8 Factors affecting Trust

We have discussed some factors, which can affect the trust between users and service providers similarly among the service providers. Following are the details

3.8.1 Security

Trust has very strong relationship with the security. Implementations of robust security mechanisms are very helpful in gaining trust. The participating entities in cloud environment can be users and can be other service providers. As discussed in earlier chapters, user trust requirements are not technical and users trust can be gained by explaining the overall features of services and things done to secure the system whereas, the trust development among the service provider is something which is difficult to achieve. Service providers are the ones, who actually provide trust environments to clients. Security plays very important role in such a scenario. Service providers are aware of security techniques available in the markets and they know all the positive and negative aspects of the technology. So, if a service provider is using a service from another service provider the other party should implement robust security mechanisms in order to get service provider interest. When it comes to trust, the only thing in mind of user (subscribers and service providers) is whether his/her data is secure. Network security is the key thing, which makes data secure. So, in order to develop good trustworthy environment security factor is a key. Tokens and digital signatures [38] should be used to strengthen the security requirements among the entities.

3.8.2 User Privacy

Privacy of user data is another important factor, which influence trust development between the customer and seller. The seller should make sure that customer information should be kept private and customer should be taken in to confidence. Written agreements should be created to build customer confidence. If a customer knows that the seller will respect the privacy, it will be helpful in achieving trust at certain level. On the other hand, the seller should implement procedures which can make sure user privacy. User accesses are the most important factor in managing the privacy. Within the organization user privileges should be given appropriately and no user should be given privileges which are irrelevant to his/her job scope. Similarly, in a cloud

perspective, the cloud provider should make sure that privileges given to the users and the access and security level of mechanisms used for implementing trust.

3.8.3 Deliver what you advertise

Deliver what you advertise is another key factor for trust building and is very helpful for successful trustworthy service composition. The service discovery process is so quick that it is not possible to check. The features offered are actually offered or not. After all, we have to trust the providers what is advertised. Cloud providers make sure that only those services should be offered for composition, which provides exactly the same features as advertised. This would be very important factor in getting client confidence because it will be easy for them to choose the service they need, instead of thinking whether the service provider will provide all the features or not. Similarly, there are some requirements at clients end and the most important of all is they should pay what they get. Service providers trust the clients only when they have surety that they will get their money.

A directory service is very useful to deliver what you advertise. There should be directory service that contains pool of services. Any service provider, who wants to offer services to a user, should register the service along with the specification. It would be the responsibility of directory service to make sure that service should provide what they adversities. If not, the service would be taken out of the directory and it wouldn't be added again. This will also improve the performance of the service providers and they will be more conscious of providing what they are advertising. During the service composition process, services will be picked up from the directory and the clients will be informed that the services we are using are registered and picked up from the directory. This will help in gaining clients trust. Service directory will act as central body and contains pool of services, which will fasten the service discovery process. Time consumption during service composition will highly reduce due to use of directory services. Each cloud has all the services at one location. It will be easy t discover services from one central location [33].

3.8.4 Dependency (Nested services)

Dependency is another vital factor for trustable services. The composed services often have dependencies on each other or on some other services to accomplish the task. The dependencies

in a composition should be defined properly so that each service should know about its responsibilities, having such an environment. It will build confidence of other parties involved in the composition process which helps in reaching a trustable service composition process. Secondly, the services which are dependent on the other services to accomplish the task needs to be define the dependencies clearly so that in case of any delays the other parties should know that the delay is because of dependencies. A service reliability and QoS can measure and estimated if we know from where the service is composed. [40] Discussed Bayesian Approach, which define the service dependencies as a parent child relationship. For example, if we have a service named as A and it is composed of a service named B similarly B is composed of service C. The reliability and QoS service offered by the service A can be measured by estimating the same of parameters of its parent likewise going back up to the root. So, we have concluded that a service performance can be measured by estimating the reliability and QoS of all the parents. Bayesian Approach can define the service dependencies clearly and eventually will help in trust build up because all the participating entities in the composition process know that what type of dependencies services obligate, and what can be their performance.

3.9 Trust Development Challenges in intercloud

Trustworthy communication in intercloud is a new concept and there is need to do a lot of work in this domain. There are several challenges needs to addresses in order to build good trustworthy environment within the cloud environment. We have discussed some of the emerging challenges to develop trust. Following are the details

3.9.1 Dynamicity of Services

Dynamicity actually raises many issues and they needs to be addressed. They include sudden increase in number of users using the services, sudden increase on the work load, sudden increase in the required resources, user needs high system performance, user needs more functionalities. This dynamicity can be a serious threat for the trustworthy service composition because all the participants, who agreed on certain parameters to do trustable communication, will not give room to any new service or feature to come in. Adding a new service at this stage, to increase a performance, will add a valuable threat to trust of existing entities on the newly add entity. During the trust development process, this needs to be handled by some useful

mechanism, which can provide facility to add or remove service or features during the composition process.

3.9.2 Authentication of services

Trust doesn't simply mean, to develop trust among the services involved in composition. It actually aims to check the services, we are dealing with are authorized and authenticated. Authorized can be taken as a standardized service. Only those services should be authorized to go for service composition processes, which are standardized. Standardizing can be done by an autonomous body or the service provider itself. Similarly, authentication of a service is also vital for trust development. Authorization and authentication will go parallel in trustworthy communication. Authentication contains identity related information about the services, which means what's the functionality of service, which cloud it belongs to, does the service is already involved in service composition process, what is the reputation of the service (feedbacks from the clients) etc. During the trust development process, authentication of the service will be done. Authentication can be achieved through some existing cryptographic authentication algorithms. In our trust worthy layer approach, we have included authentication step and it is necessary to achieve a certain degree of trust. After the authentication, authorization of the service is the next step to look into it. Authorization means, the features, service has the access or what will be its role among the services. The service will perform only those functionalities, which are entitled to do any work. Other than that, it will disrupt the trust development process. A service is said to be ready for the service composition process once it is Authenticated and authorized.

3.9.3 Reusability factor

The reusability factor in trustworthy communication needs to be dealt carefully. We cannot use the same trust principles between the entities again and again. Similarly, same trust rules cannot be easily implemented with other entities. This is because trust changes from the user to user and every user have different requirements of trust. Correspondingly, as we have already discussed trust between the same entities changes with the time so it is a challenging task to reuse the same trust policies again and again. There is a need of mechanisms, which can actually extract the difference in requirement so that new policies among the entities should be created with minor changes. Similarly, for the service providers, there is a need to implement several mechanisms,

which can ensure trustable communication. There might be a chance that different user needs different level of trust development.

3.9.4 Distinguishing Identical Services

As we know, there are thousands of services in cloud environment and day by day they are increasing. Due to increase in services, there are multiple services with same criteria and functionality. During the service discovery and selection choosing a single service from identical services is another emerging challenge which needs to be addressed. We have suggested use of two parameters to choose among identical services. Following are the details about the parameters

1. **Trust**

   Trust itself can be chosen as a parameter. We have used trust as a parameter to choose the services. Identical services have different trust levels. Service trust level can be judged by feedbacks and interaction history. Among the identical services, the service, which is more trustable, will be chosen. The service selection on the basis of trust will offer two benefits. One would be like; it will help us in choosing more appropriate service from identical services and the second one, would be easy to build trust among the services during the service composition. So, trust will be used as a parameter to select single services from identical. Following figure will elaborate it more [7].

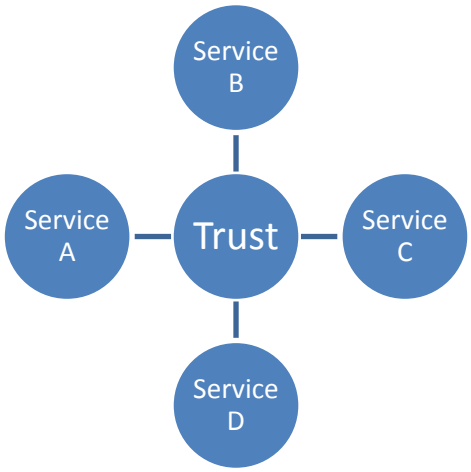

Figure 4: Trust as a parameter among identical services

2. **Compatibility**

There can be one or more parameters, which would not be skipped. Along with the trust, compatibility can also be used a parameter to distinguish among the identical services. Platform dependency still exists at cloud level and compatibility can be an issue in combining services. For example, Apple cloud services and Samsung cloud services have compatibility issues. So, we come to a conclusion that only that service should be picked which is compatible and the most trustworthy among all other services. There can be a deadlock situation arises here. If there exists a most trustable service but it is not compatible on the other hand. A situation arises, in which a service is compatible but not trustable. In that scenario the most suitable option would be to go for the service, which is compatible and then trust will be built later during service composition. If a trustable service is needed to be picked up, then the compatibility of a service should be negotiable. We can negotiate with the cloud owners to resolve the compatibility problem.

Organization or consumers using clouds for services have their own organization polices for data sharing and they should abide by the rules imposed by the organization standards. Trust development in such cases become more difficult. Even for the third parties, it is very hard to know which data client wants to share and which not or to what extent it can be shared. This approach will be time consuming so better approach can be to let the user choose what data he wants to share and what not. It should be up to the user to decide what data to be shared. After user selection it will be easy to built trust among the user and composed services. The user interface concept discussed before will be very useful to implement this. Features in user interfaces can be added, by which user can list the information about sharing of the data. We will also discuss another scenario here, which is our primary focus here. Data share among the services will also be assumed as same as it is for the users. At the time of composition let the service choose what it wants to share and what not. After data sharing description by the service work Trust development will be continued. Trust development among services is a hot topic now days. As user demands are becoming versatile day by day and it is hard for single services to fulfill the requirements or if we see from business perspective no business man wants to lose his/her clients so the concept is if user demands any particular service,

even though the service does not have all the features required by the user. The service provider can contact other services at the backend to get those features and then will offer to the client as single solution. This seems quite practical, as you see this trend in normal business now days. The business men are providing complete solution to their clients. Even though, they themselves do not have all the features. They actually contact other vendors in the markets and will get services from in order to facilitate their clients. Now, the question arises how the service will contact other services and develop trust so that because of delay from any other service its user is affected and what would be the plenty or how they will continue further. The proposed conceptual model in next chapter discusses this problem in detail and will help in resolving the trust among the services. Now we are talking about two scenarios here.

a. The 1st one is service composition at the beginning. In this scenario the client knows that he is getting services from multiple entities. All the services will equally be involved in providing services and development of trust among each other and to the user.
b. In 2nd scenario, client only know that he is taking service from only one entity and he will held responsible to only that entity in case service level agreement is not fulfilled. Whereas, on the other hand the service is taking services from other SPs providers and it should have own SLAs with them.

3.9.5 Trust development (Among services)

When we talk about Trust, the direct question comes to the mind is trust at what levels. Trustworthy service composition involves many services. So, it is obvious that data will be transferred on the Network also. Trust development at the Network end is not an issue as there are already enough techniques, which securely transfer data on the Network. Nowadays, networks are everywhere and after the emergence of internet the network trustworthy communication was in focus for a longtime. These days, we enjoy trustworthy communication on the networks. The subscribers are not worried about anything on the Network end as they are already using computer networks. Our main discussion of trust will focus on the trust development at the service end. How services are using subscriber data, how services interact, what kind of security mechanisms implemented by services etc.

3.9.6 Contents, Data Privacy and Data Reliability

Trustworthy communication includes several issues, which need to be addressed. Some of them are contents, Data privacy and data reliability. There should be a clear understanding among the service providers, which contents it contains and the contents are beneficial for what. The content distribution as a service can be used to develop trust on the contents [8] .During the Trust development process agreements related to data privacy should be taken into account and every entity should have trust that its data is fully secure and there will be not violation of data privacy. Reliability here actually means the correction of data to be shared. Each participating entity should make sure that the shared data is reliable.

3.9.7 Achieving Subscriber confidence

There are many hurdles to achieve trustworthy communication. There are two important reasons, which have made trustworthy communication difficult. They are subscriber confidence on the technology and the acceptance of service as a standard [15]. New technologies are being introduced day by day and it is very difficult to gain confidence of the subscribers. Subscribers normally opt for a technology, which is popular and recommended by other subscribers. They rely heavily on the services, which are in operations from a longtime. Getting services or subscribers confidence is big hurdle for trust development. Performance mechanisms can resolve this issue. Performance mechanisms can be implemented in inter-clouds, which can analyze the performance of the new technologies (services) and on the basis of evaluation, subscribers or services confidence can be achieved. But this is still a novel topic. Another scenario can be service providers and subscribers can be educated so that they can understand that going for the newer technology is not risk. There are thousands of services exist on the cloud but many of them are not properly standardized so service provider normally do not go for the service, which is not standardize. Clouds should include only those services which are properly standardized.

3.9.8 In complete Information

In complete information about the service and within services it raises a considerable to trustworthy service composition. There is a need for a common directory database, which should include all the services. It should be the responsibility of the service platforms to put their

services in a common database so that services can be utilized in time of need. There is also incomplete information about participants in intercloud. Mechanisms are required to put all participants in a single platform. A cloud regulatory authority can be one solution. The central authority should make sure all the participants should register with it. However, it is not easy to bring consensus on the regulatory authority due to diversity of the distributed services. Implementation of directory database and cloud regulatory authority can resolve the problem of incomplete information in intercloud.

3.9.9 Cheapest service Best solution (CSBS)

Cheapest solution with best performance is the top priority of the users and with growing number of services, consumers want cheapest services. There may be a possibility that single service is more costly than a multiple services. Graph algorithms can be used for this. Algorithms are necessary which will compare price of single service with multiple services and calculate the cheapest price with best solution for the user. Contract Net Protocol (CNP) can also be used for this [5].

3.9.10 Trust development among unknown services

Service overlay networks (SON) are built on other networks. We can take example of cloud networks, peer-to-peer networks and client-servers networks. All those networks are SON because they build on internet. In a more meaningful way, we can say that a network, which is dependent on another network is called overlay network. The root network is always the main network because if that network fails, all other networks dependent on it, fail. Subscriber requirements can actually be divided as functional and QoS requirements. We have already discussed functional requirements in our previous discussion on service composition, which actually includes the semantic languages and other procedural requirements. QoS requirements are related to business and affect the business. There is need to do a lot of work in user QoS requirements, which will actually benefit the business because user is the one who gives business. Subscribers and Service providers has QoS requirements when they opt to take some service from the service provider and they are prioritized means which one is at top and which one below that etc. Among those Trust is considered as the most crucial QoS by subscriber. The

other QoS requirements are not our concern here we will only discuss Trust here. Achieving a good trust relationship among the entities in cloud environment is a difficult task. [30]

In a services composition environment different services are put together to solve a complex problem. If we look into the scenario, it actually means different services are combined together on temporary basis to complete a project. As the services are different and in most of the scenarios they belong from different cloud providers, it is quite challenging to build up trust among them. If we take a real time scenario, we can see if new people are hired for project, it takes some time to develop trust and they need to trust on each other blindly. In our case scenario is pretty much same. There will be unknown services and they will be put together to accomplish a task. We are required to build a trustworthy environment among them. In such as environment Swift Trust [30] is very useful. It can be an ideal approach for trust development. Each participating entity should exhibit a behavior, which indicates it is trustable and other can trust on it. Truly, the participating entities behave like trust is present but really it is no there. The goal of trustworthy communication among the services is not to increase a better communication. But it aims at accomplishing the task. The pool of services should cooperate and trust each other only to complete the task successfully other than that there shouldn't be any common interfaces. So, during the trust development process, it should the actual goal of trust that should be kept in mind rather than spending time on things which are not of primary concern [30]. Pool of services, cooperate with each other in a trustworthy way to accomplish the task assigned to them. There is another major difference in the term discussed above. There can be a possibility those entities (services or subscribers) trust each other very well but reference to the task they do not trust. This factor should considered at the time of trust development so there is need of more goal oriented trustable communicate rather than simple communication.

[35] Proposed automatic composition synthesis technique. The technique is used to build trust among the services. The scheme mainly focuses on the access control mechanisms and preferred use of credentials. The scheme provides effective trust environment among the pool of services used to resolve complex problems. [35] Discussed the two different scenarios in the service composition. The 1st scenario is how the composed services will communicate with each other either manually or automatically. Similarly, second scenario is how the flow will be maintained among the services. [35]

Aniketos project started in august, 2010 by Aniketos team to ensure trustworthiness and security in service composition and it will finish in February, 2014. Aniketos will be very useful to ensure trustworthiness in service environment. A team of leading professionals is working on the project and is considered as the largest project on service composition by Europe. [37]

3.9.11 Standardized Trust Protocols

There is a need of standardized trust protocols, which can be easily integrated to any cloud platform and benefit in developing trust among the service providers and users [31]. Applications on cloud environment require some form of formal trust specification to gain trust and by having trust specifications for application. It would be easy to gain user trust [31]. Emerging use of cloud computing has brought up the importance of trust worthy service composition. As more and more organizations are moving to cloud and services are increasing in number. An organization joining cloud environment belongs from different domains, having different trust requirements services also belonging from different domains. It is not possible for existing trust mechanisms to cope with such kind of diversity. This diversity in cloud environment highlights the need for flexible and trustworthy service composition model, which can benefit in developing trust among services from different domains. Development of trustworthy service composition protocols will help in developing trustworthy service composition.

Although, there are several protocols available in the market but still there is a need of more strong and comprehensive trust protocols, which can be really helpful to achieve trustworthy service composition among the composed services. However, developing a trust protocol for service composition in intercloud is a tough task and all the major factors for trust development should be considered to develop a robust trust protocol.

3.9.12 Trust Management

Trust between the entities changes with the time. There might be a possibility, it will be changed or decreased with time. It depends upon the requirements from the client and new features offered by the service provider. There is a need of mechanisms, which can actually manage the trust between the entities with the time like if the trust between the entities is decreasing the trust management mechanism should figure out what are the factors by which trust is decreasing. It

will also help both the parties in increasing the trust level. Change in trust, needs to be handling on priority because there are several services involved in the composition and problem with one of the service can affect the overall performance of the system. Trust management protocols can be very helpful in building trust among the entities.

The interaction layer is responsible for interaction among the users and the service providers. The interaction is based on service patterns. We will discuss the service patterns involved in interaction among the seller and the buyer. [33] Describe service patterns, which will include message passing, request-response, subscribe-notify and publish-subscribe. We will discuss all of them from trustworthy service composition perspective. All the entities (services) involved in the composition process will communicate with each other through interaction patterns. We have used here three interaction patterns remaining one is not of our use in service composition. [33]

1. **Message passing** [33]: This is a one way communication, which does not have any acknowledgement. It is used by a user to send the data to the service provider.

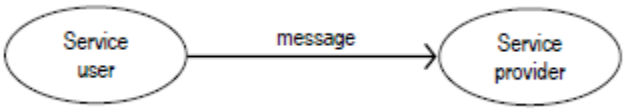

Figure 5: Message Passing[33]

2. **Request-response** [33]: This is a two way communication but not simultaneously. 1st the service user sends the message to the service provider. In response, the service provider replies with the requested functionality.

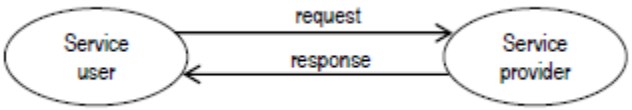

Figure 6: Request-response[33]

3. **Subscribe-Notify** [33]: This is also a two way communication between the entities involved in service compositions and the users. At first, the sender (can be a user or

service provider) sends a request to the service provider. The service provider manipulates the request and then responds with the sequence of replies. The series of replies are sent on the basis of following two approaches:

a. **Time based** [33]: The messages are sent after a specific interval. The number of messages to be sent depends upon the functionality.
b. **Event based** [33]: The messages are sent, when any event occurs. The occurrence of event depends on the functionality.

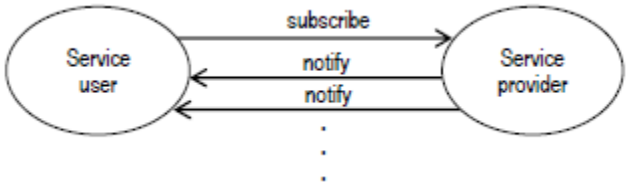

Figure 7: Subscribe-Notify[33]

Trust varies from user to user. Further elaborating, we can say trust changes by type of required services. The user requiring mathematical service requires different level of trust than a user, who requires IT services. Each classification of a service has own trust requirement, which needs to be met. Trust management system, we discussed earlier should also address this problem. We have already discussed that trust with the time changes. Both the concepts are different. The trust, which changes with time means the user and provider will be the same. Their trust requirement will change with the time whereas; here we are actually discussing the classification of services. Trust requirement for each service classification are different. There are some services, which are used to handle highly secret data and privacy is main requirement. Scenarios like those have very high trust requirements and difficult to achieve where if security is not very important, it is easy to reach trust between the user and the provider.

Policy maker and Key Note [41] can be very useful for trust management. The key functionalities of both approaches are they handle the authorization directly instead of dividing it

into further tasks. Key Note compliance checker [41] is very useful in authentication the signatures and security mechanisms like encryption, hashing etc if they defined in a proper format, which is supported by the Key Note. There is a need to implement security mechanism according to the Key Note assertions so that key Note can better authenticate the security mechanisms. Compliance checkers are very helpful in ensuring that the services are working according to the requirement and the security mechanisms implemented are compliance. [41]

3.9.13 Free-services (Threat)

There might be a possibility that during the service composition some of the services involved in the composition process are free services and will breach the user security, which will eventually breach trust between the entities. There can be two solutions possible to reduce this breach and gain trustable environment. Firstly, during the composition process, it should be make sure that the services chosen are registered services. If the service is a registered one and it confirms that all the required security measures are taken to for data protection. But this first solution doesn't suit always because it is not useful to take registered services every time there might be a possibility the required functionality can be performed by a free service more effectively and going for a cost efficient solution is more appropriate. But now a question arises, how we will make sure that use of the service will not breach the security. There is a need of mechanism, which can only share the required information with the service and then immediately initiate the service decomposing process to remove data from the service end after accomplishment of the task. This will help in gaining user trust level and communication will be done in a better trustworthy environment. There are several services, which are free to use and everybody wants to use the free services. The frequent use of free services on intercloud raises new question of security vulnerabilities in enterprises. The confidential data of enterprises is being shared over intercloud and are held responsible of security breach. There is a need to manage these services properly either the uses of these services should be prohibited or they should be registered and managed by cloud provider more adequately. The reason, why I discussed this point in trustworthy communication is: it has great impact and can damage the good trustable environment. This issue needs to be delt because users are aware of these security breaches and they will immediately ask you about the use of free services. One may raise a question here that if the free or small services are destroying trusts so much why not going for a registration. The

problem is the cost. It is not possible to register every service. Registration is costly and if the service is not doing so much business it is useless to go for the registration. In a new survey by [34], it is indicated that a huge number of services on cloud environment are not detected and neither monitored. Users share their information frequently with the services where as on the service end there are no security mechanisms, which can make sure data security hence breaching the user security. [34]. Due to a lot collaboration in intercloud. Security provision in collaborative environment is nightmare for the enterprises and is difficult to implement. As we already mentioned in our previous discussion trust is directly associated with the security so tackling this nightmare should be the highest priority. Mechanisms are needed, which can cope with the unmanaged services on cloud platform and make sure the enterprise data security because sometime due to some low cost cheap service companies face a considerable threat to expensive firewalls. Implementation of strong firewalls will also help in developing trustworthy environment. This feature will help the end users in maintaining their data security but convincing them to establish trust is more difficult because they just know the overall details and they didn't interested in the technical details. However, the scenarios in which one service provider is taking service from another provider would be very helpful. In this scenario, both the users know the technical details and implementation of good security mechanism and strong firewall, which can help in keeping data privacy will increase the trust level among the entities.

3.9.14 Trust Monitoring

Trust monitoring is also very important for trust development and trustworthy service composition cannot be reached without a good trust monitoring. Trust monitoring is actually an additional check on the participating entities so that to ensure proper trustable environment. The parameter used for trust monitoring is the properties or we can say the features specified but the entities and the contract between the entities. It should be the responsibility of monitoring mechanism that the service is performing according to the properties and there is no deviation. Similarly, the contract among the participating parties should be monitored, which will be used as basis for the performance evaluation and future trustworthy communication among the entities. While implementing the trust monitoring the challenging task is where the trust monitoring mechanism should be placed so that it can be affective and who will manage it. The workflow mentioned in Aniketos project [37] shows that it should be implemented between the

service provider and client, it shouldn't be implemented at service composition end because actual communication will take place between the client and provider the other technical details and service should be kept away from the monitoring. During the coordination, the monitoring mechanism will monitor the correspondence and service offered by the provider. Effective monitoring of trust will improve the trustworthy service composition and will help in future trust development among the entities. Secondly, monitoring of trust needs to be managed by some authority. Any participating party can monitor the trust but there might be a problem in reaching a consensus on the monitoring authority. So, it is better to choose an independent party. Trust monitoring itself can be offered as a service and this can be most suitable option. First of all, it would be easy to call the service and then it will work to monitor the communication among the entities. Building a trustworthy monitoring service is a challenging task and is not an easy thing. The service should be good enough to understand the level of trust between the entities, which is quite difficult task because trust is a more general term although due to growing efforts it is becoming specific, If good trust monitoring services are developed than the trust monitoring tools will be very helpful to maintain a trustworthy communication.

3.9.15 Service selection

There have been several discussions on how to select the services. There is a need to understand the difference between locating a service and to select a service. Locating a service according to user requirements doesn't mean that you have the service but it requires selection process, which needs to use to pick up the services. Cloud providers or mechanisms are used to select services use some parameters to opt for a particular service. Each parameter has own plus and minus like if a user focuses on QoS then selecting a service, which offers more QoS have to be used. Parameters are very important let's take a practical example. Suppose, ten services are located during the discovery process than how one service will be chosen is based on the parameter, as we discussed before like if Quality is required then the service which offers more Quality will be chosen. Similarly, keeping in mind the importance of Trust in trustworthy communication Trust itself can be used as a parameter for trustworthy communication [40]. The paper [40] discussed the Trust aware service composition, which helps to choose services on the basis of Trust. Selecting services on the basis of trust will serve two tasks. Firstly, we come up with a useful parameter for service selection and secondly, complexity of trust development process among

the services will be reduced due to trust ability of the composed services and this will help in achieving trustworthy service composition. After the selection process, there is need to counter check the service parameter on which it is chosen throughout the process so that it should be known that service is not deviating from the parameter. There is need of mechanism, which can punish service if there is any violation. By punishment, we mean that service should be taken out of the composition or shouldn't be part of future composition or the particular cloud provider should be informed so that the service should be taken out of the pool of services which are in service composition candidate pool. Trustworthy service composition also depends on other parameters at the time of service selection, which is reliability factor. There is need to opt for services which are reliable. Reliable services will help in achieving the confidence among the participating entities which will eventually help improve the trust.

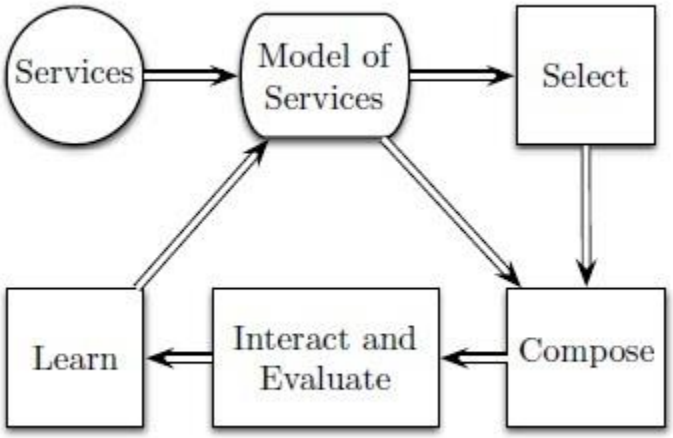

Figure 8: Service selection and composition[40]

3.9.16 Threat prevention in Trust

Threats are always a hurdle to build good trustworthy environment. There are some threats, which can be predicted by the provider and user should be told that due to these threats service quality can be affected. There is a need to understand this issue. Threats, which are known, can be handled before they occur so that they didn't affect the trust. A repository can be formed, which should include all the possible threats which can affect the system performance and

mechanism should be implemented to avoid such threats and mean while the participating entities should reach a consensus that measures are taken to avoid these threats. Policies should be discussed if these threats affect the system. This approach will help in coping with the threats and will be helpful for trust improvement. On the other hand, there are several threats or worst time scenarios, which cannot be predicted and cannot be known. During the policy making process, these factors should be considered and proper documentation exists so that in time of such scenarios the trust level shouldn't break. During the dynamic provision of services, the worst time scenario can be listed and should be sent to all the participating entities so that every entity should be aware of this.

4 TSCCM; A new model for Manual Service Composition

TSCM model defines levels of communication required to build trustworthy environment among the Cloud providers (Service providers) and between cloud providers and subscribers (clients). Trustable environment among the participating entities is vital to produce useful work and it will improve performance and QoS. Cloud environment has several services, which are used for sharing resources to provide better services to the user. This resource sharing work is done by a trustable service composition. Service composition involves high resource sharing, which requires good level of trust. Trust can only be reached by adopting good security mechanisms and policies which can help in countering the threats to trust. Excitingly, security mechanisms are not a big issue because internet is around from more than two decades and a lot research is done in the computer networks. There are several existing mechanisms, which are implemented and can be implemented to build a secure network. The real issue is to develop trustworthy environment. Although security can be very helpful in building trust, it does not suffice. There are several other issues, which needs to be addressed. Due to the growing trend of cloud computing and intercloud concept, there are several challenges in security also but those are not our primary concern here. By procedures, we actually mean a mechanism, which should be helpful in reaching a trustable environment among the composed services. We propose a model, which can be very helpful in reaching the trust among the participating entities and pursue the

trustworthy communication in service composition in intercloud. Overall the model is divided into phases and each phase has some modules, which are used to implement and adopt procedures to achieve trustworthy service composition. Implementation of TSCCM will resolve current issues in trustable service composition, and it also defines the sequence how procedures should be implemented. In the following section, we will explain all the phases and modules of our model in detail.

This model is purely designed for Manual service composition. Usually manual service composition is done for larger projects, which requires detailed analysis of the requirements. Model will help in reaching trust in less time with higher accuracy. The modules proposed in the model will help in reaching trust among the parties involved in service composition. Adoption of the modules will help in reaching timely consensus among the parties. We haven't discussed automatic service composition in detail but in future some of the modules can be used to reach trust in automatic service composition. Model is applicable for both Software as a service (SaaS) and Infrastructure as a service (IaaS) in cloud environment. Services participating in the composition process and the subscribers will reach trust in less time and the quality of work will be improved highly. Model will serve as role model for reaching trust. During the design, we have kept in mind the two important areas of cloud computing which is Software as a service (SaaS) and Infrastructure as a Service (IaaS). Model is applicable to both areas and will be very useful. However, the modules discussed in our layered approach can be useful for building trust also in Platform as a Service (PaaS). When we say SaaS, it actually means the customized solutions required by the clients. If a client needs a customized solution from cloud providers, model will be affectively used to build trust and reaching an agreement among the involved parties. Currently, there are several issues during the trustworthy process and it is difficult to reach trust by following existing techniques and models. Our phase approach will be very helpful for all the parties involved in the composition process and will improve the Quality of work required by the client.

IaaS as a service is much broader concept and time consuming as well. Implementation of infrastructure requires heavy investment and involvement of several parties. Currently, most of the companies are implementing and managing their own infrastructures, which results in high cost and employee's liability. For example, every company has big IT infrastructure. The

companies not having IT background and IT is not their core business, it is very difficult to manage such a big infrastructure. Trend is changing nowadays and in future most of the companies are expected using IaaS. The idea is, cloud providers will provide Infrastructure services globally. A single cloud provider will provider Infrastructure services to multiple clients. We can say that a single cloud provider will manage multiple Data centers with all the services and Quality of work required by the client. Client requirements in IaaS actually means backup of the Data, handling worst time scenarios, resource management, less use of hardware etc. Our Trustworthy model is useful to implement in IaaS. Implementation of model will help in reaching SLA between the parties. Use of model will help in defining accurate time, cost and performance constraints, which will help in reaching trust among the parties. It will also be helpful in doing management between the computing resources, storage resources and network resources.

We use the concept of nested services in our model. As we have discussed earlier in the paper during the service composition. There can be a chance that service providers or cloud providers are receiving services from other providers. We call this as nesting of service (Service within a service). This is general market practice, every provider wants that they should facilitate their customer with all the features so the customer doesn't need to go to another provider. This scenario looks good but when we talk about trustworthy communication, there are several dependencies, which need to be considered. By dependencies, we actually mean the involving factor, which can affect the trustworthy service composition. While taking service from another service provider, the provider should consider the fact that the delay in service from other provider can affect the trust between the user and the provider. During the composition the other provider should be aware of the fact.

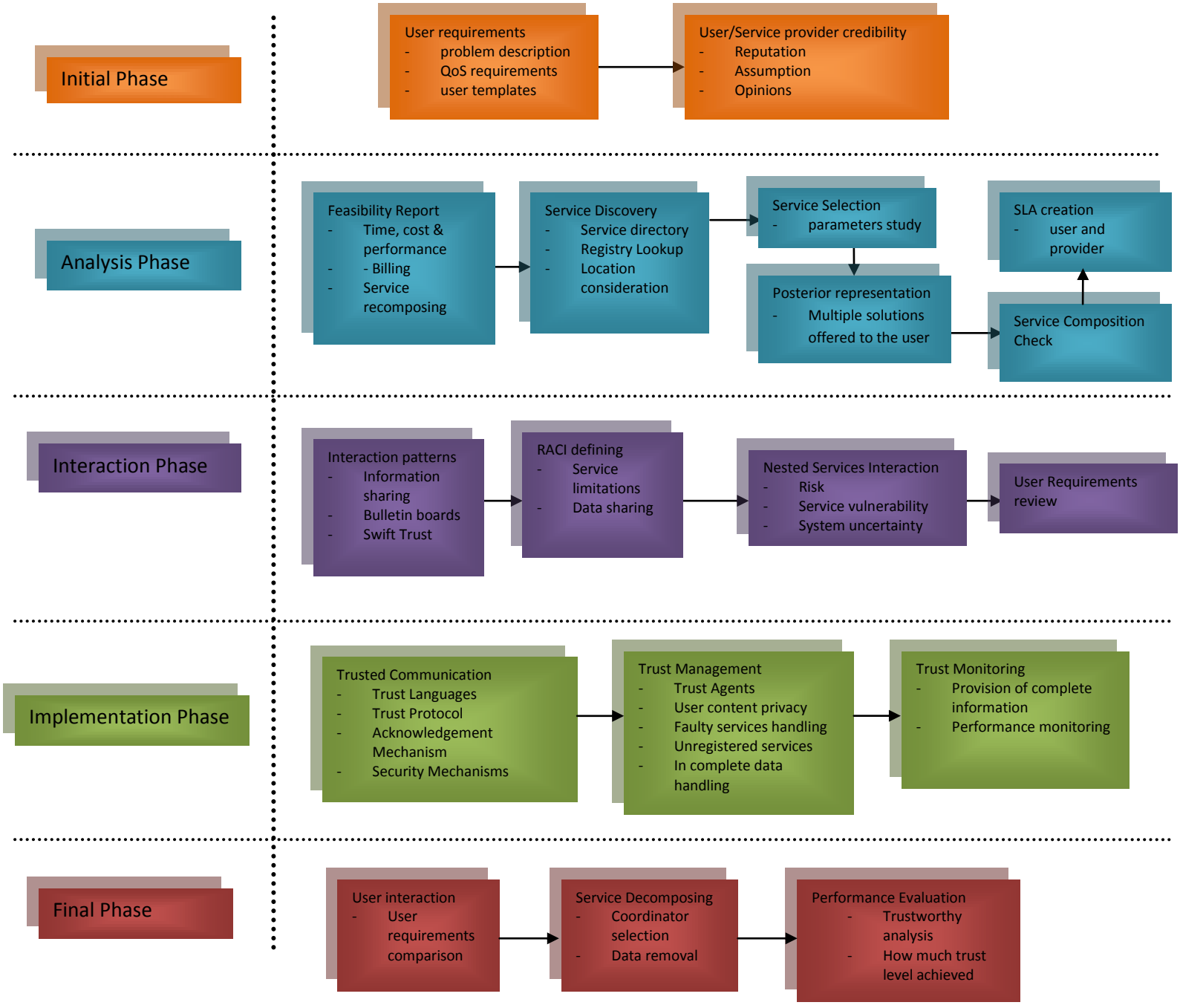

Figure 9: Trustworthy Service Composition Conceptual Model (TSCCM)

4.1 Trustworthy Service Composition Conceptual Model (TSCCM)

4.1.1 Initial phase

Initial phase is the starting point of our trustworthy service composition Model. Phase is used to gather user initial requirements and some user credibility checks are performed. This phase contains two important modules, which we will explain in our next discussion. Following are the modules.

4.1.1.1 User requirements: Module is used for collecting the user requirements and information. User information involves collection of user personal details, which includes name, address, date of birth, contact details etc other than this, information that user is private or business and is existing or new user. Non-technical users are not competent to explain their requirements. There is a need of third party agents, who can explain the user requirements so module take requirements either directly from a user or a third party agent representing the user. User requirements should always be the basis for any system design and therefore it is important to collect and analyze them. As we discuss in Service Composition Chapter, user templates are good to represent user requirements in an effective manner. User requirements will be represented in user templates and our module will be used to take information from the user templates. There are two important parameters, which need to be taken in consideration during the information collection, which are problem description and QoS parameters. It should be made sure that all the necessary information is received from user and QoS parameters are clarified. During the user requirement gathering, we assume that user is not aware that there may be a need for service composition. It is the responsibility of the service provider to decide whether the service composition is required or not. User shouldn't know about the technical details at the backend.

4.1.1.2 User/Service provider credibility: It is necessary to check the user and provider credibility in order to make sure to what extent trust will be reached between the user and the service providers. This module will be used to check the user and service provider credibility. Various techniques and strategies can be used to check the

credibility. We will use two mechanisms, which are reputation and assumption. These mechanisms will be used to assess the credibility of users and service providers. We have considered two categories of users and providers which are; existing and new. Reputation mechanism is used to check the credibility of the existing user's. Reputation involves opinions of other service providers and users, feedbacks and old interaction history. Assumption is a critical notion for trust development and there are setups, in which we have to do the assumption at the start. We have adopted assumption mechanism for new users or providers. Reputation mechanism is not fully applicable on new entities. However, it can be implemented partially which includes opinions and feedbacks from other entities (users and service providers). Assumption mechanism can also be adopted in case no information from other parties is available, then we can assume trust on the basis available user information. For existing users (i.e. who are already the users of the services) their track record can be checked as an indicator for their credibility. On the other hand, the assumption approach will be adopted for users, who are new and whose reputation cannot be judged based on experience. User credibility check will allow us to see how much a user is trustable or not and will help in reaching future trustworthy service composition. Interaction of user with other service providers is also checked and their opinion will be taken into account. Opinions are very important and can be very helpful to develop trust on a user.

4.1.2 Analysis Phase

After the initial phase, we move to the analysis part. User requirements will be analyzed in this phase and possibility of the trustworthy service composition will be taken into consideration. Analysis phase is the key for service composition and involves feasibility reports, service discovery, service selection, posterior representation and SLA creation. A comprehensive analysis of the problem is necessary for successful trustworthy service composition. All the involved parties should accept the feasibility study and SLA before proceeding to the interaction phase. As we already discussed in the initial phase, there is a need to distinguish between the user's that they are existing users or new. Developing trust with old users is easy where as among new users it is difficult. For old users, already existing SLA and feasibility reports will be

taken into account, this will reduce the time to build trust and sign SLAs. In the following section, we will explain all the modules involved in analysis phase.

4.1.2.1 Feasibility Report: Module is used to perform the feasibility study for the service composition. A complete feasibility should be formed, which should include the details of service composition process. By details, we actually mean time, cost and performance constraints will be discussed in the report. The cost constraint will actually cover the billing area of service composting and it should consider all the expenses. Time constraint needs to be defined accurately because in some situations, where several parties interact with each other, time is a difficult thing to manage. Time analysis should also include the time spent on composing and decomposing the services. Performance should be the key for QoS, which is the user basic requirement. Feasibility reports should be accurate because all the future agreements will be based on the feasibility report. This module will be used to see, whether service composition is possible or not. There are scenarios, in which the service composition cannot be possible or the user requirements can be resolved by individual services. We need to consider all those options. After the user requirements are gathered and the credibility among all the involved parties is assessed, we see whether composition is possible or not and if possible what are the available resources we have. Service recomposing is also considered in this module. Service recomposing concept is very useful if we get a request from existing user and user requirements match to some existing service composition or a different user with same requirements come then we will consider service recomposing. Service recomposing is highly beneficial for the trustworthy service composition and will greatly improve the time required to do the composition and trust reaching process. All the remaining phases and modules of the model will be implemented but the time taken on each phase will be reduced considerably.

4.1.2.2 Service Discovery: We have discussed this term several times in our thesis. For trustworthy service composition process, trustworthy service discovery is very important. In this module, we will look for the services, which are trustworthy and interoperable. Service directory and registry look up is performed to discover interoperable and

trustable services. Service directory and service registry concept is discussed earlier in the thesis. Registry service is used to register services. The register services are assumed as more reliable as compare to unregistered services. We have emphasized on maintaining a service directory, which should contain all the services useful for service composition and are registered in registry service. Service directory should be implemented in a tree hierarchy. It will be easy to maintain the service directory with tree structure and searching can be done quickly which will speed up the discovery process. BST (Binary search Tree) can be used to discover the services in a tree. After the discovery of the service, we will perform the service registry look up. Registry look up is performed to check whether service is registered or not. Unregistered services can be a hurdle to reach trustworthy service composition. Location factor should also be considered during the service discovery process. Priority will be given to the services, which are located near the user and service providers.

4.1.2.3 Service selection: After the service discovery, we go for service selection. In our selection process, we will use trust as a primary parameter to choose the services. This will help us in choosing trustable services which will later provide trustworthy service composition. We use following parameters to select services; Service credibility, Belief, Opinion, Reputation, Assumption and Service Registry information. Service credibility parameter is useful to check the credibility of the service, which is to be selected. Belief refers to the trust on the selection that means how much a service is confident to select a service. Opinion, Reputation and Assumption are the same parameters which we discuss earlier in user/provider credibility module. Service registry look up is used to check service is registered or not. Base on the above mention parameters we will assume that service is trustable and it is good to select the service. By having in detail analysis of the above parameters, we can choose trustable services.

4.1.2.4 Posterior representation: This module is used to gather all the information collected through service discovery and selection phase and represents in a way that provides multiple options to the customer. On the basis of service discovery and selection, we will get multiple solutions that what we can offer to the customer. Multiple

packages will be formed for the user. In this phase, customer interaction will be done and customer would be provided with multiple packages. Packages will differ on the basis of features and cost. Customer will be given a choice to choose anyone among the multiple solutions. After the customer selection, we will move to next module in which negotiations will be done about the contract so that trustworthy interaction among the parties will be done. Subset of services can also be offered to the user.

4.1.2.5 Service composition (Check): Before the SLA creation between the user and service providers or among the service providers we are good to go but here in this module we will perform one additional check which will make sure that all the services are selected and they are authentic. This will help in maintaining future trustworthy service composition. If during this service checkup we found that service is not up to the requirements, we will go for another service so that SLA creation between the user and provider is not affected which will help in reaching trust.

4.1.2.6 Service Level Agreement (SLA): After the feasibility module, we will move to build SLA between the client and the service providers or coordinators responsible for the negotiation. Feasibility report will form basis for the SLA and all the scenarios of service composition and trust development should be considered in the SLA. All the threats to trustable service environments and mechanisms to avoid the trust should be documented in the SLA. SLA should be signed by all the participants so that in case of future confusion it will be used as a reference, SLA is vital to build trust.

4.1.3 Interaction Phase

After the analysis phase, we move towards the interaction phase. This phase work as a primary phase in developing trust and if the participating entities reach a trustworthy relation then in next phase the actual work starts. In this phase; interaction patterns and responsible accountable consulted and informed (RACI) among the services and service providers is defined. This module contains several modules which actually help to start the trustworthy composition process. Following are the modules in the interaction phase.

4.1.3.1 Interaction patterns: After the service selection, we will move to decide the interaction pattern for the communication among the services. This module is very important to improve interaction among the services and good interaction patterns will provide ground for good coordination among the services. All the information sharing mechanisms are defined in this module. We will use swift trust to develop trust among the composed services, which will help in reaching better trustable environment. Use of bulletin boards is also recommended to increase better interaction among the services, which will help in reaching trust. Abstraction layer discussed in [7] can be very useful to build good bulletin boards. Bulletin boards are used as abstraction layer for communication among the services and all the participating services will have central access to the information available in service composition. Interaction patterns discussed in [33] can be used to for coordination among the services.

4.1.3.2 RACI: A complete RACI will be formed in the interaction phase and roles of all the services involved in composition process are defined. Previously, we have discussed SLAs between the Provider and the client but not between the participating services. SLA among the services is very important and we will define service limitation so that no service can interfere with other services in terms of performance and every service can play its part independently. Complete RACI should be defined and shared among all the providers so that they can know about their limitations. In this phase business rules will be shared which may include how much data a service is allowed to share and what will be the privacy policies after the data sharing. SLA should also contain the details if the particular service user is affected because of another service. Service recomposing and decomposing details should also be written in the SLA. It should also state the dependencies on other parties in a service. Dependencies on other parties (services or service providers) can affect the interaction among the composed services. Service and Service provider dependencies are define in this Module. Bayesian approach [40] is used to define dependencies among the services. Bayesian approach is useful to define dependencies in a parent-child relation form. Dependencies of all services are defined in

a parent-child relationship form; this will help to define nested services relationship. RACI will provide ideal platform for the trustworthy service composition.

4.1.3.4 Service Interaction (Nested services): We already discussed a module, which does the user interaction but here module is responsible to do service interaction with the nested services. While doing interaction with other services user trust requirements should be considered. Service interaction with nested services will aim at maintaining trustable communication environment so that user trust level shouldn't be affected. We have used some parameters, which will help in gaining trust between the nested services. During the communication the provider who takes service from another service provider should explain all below mentioned parameters to the service provider so that user shouldn't be affected. Following are the parameters which need to be considered [30].

4.1.3.4.1Risk: During the service interaction with nested services this factor should be considered and service provider should consider how much risk is involved to take service from another service provider and the risks for the trustworthy communication should also be considered. Service provider should be told how much risk level is involved if there is any service delay from your end. If the risk level suits to the provider, it will become part of nested services. Provider understanding of risk is very important for good trustworthy environment. Risk factor directly affects the trust. If provider thinks that risk factor is high in going for the service he will not trust the service. If the trust level is very high between the user and provider then in some scenarios user can be told that some services are taken from another service provider their might be a possibility of some delays from the service provider.

4.1.3.4.2 Service Vulnerability: Service vulnerability factor should also be vital to consider. Service provider should know that the other service provider shouldn't be vulnerable. Service providers know about the robustly system and the security measures, taken to overcome cyber attacks. Implementation of strong firewalls will gain service provider trust on the service. Less vulnerability of service will gain service provider trust on the service. Other

than that, there are some threats which are already known to the system, measures should be adopted to overcome and counter such threats and a description should be given to the provider which explains the mechanisms implemented to avoid the threats. Threat avoiding mechanisms are also very useful to manage the trust. There are several threats which are already known most of the time knowing the fact if the service provider didn't implement mechanism to counter the threat. This will be considered as the violation of the trust rules which may affect trust.

4.1.3.4.3 System uncertainty: Uncertainty is also the key factor to consider while opting for nested services. It should be explained to the user how much certain the composition process is. If the service is the existing one then from the history of service, its behavior can be seen but explanation factor becomes more challenging if a service is newer and belongs to a small organization in a cloud. This can be the case in several scenarios because service composition in intercloud is a new concept and day by day new services are created. User can be given a simulation prior to the start of actual working, which will help in gaining user trust on the service.

4.1.3.5 User Requirements review: During the initial phases user expectations are considered but we need to revisit them later, because user requirements may change. User expectations should be fulfilled as per the SLA. The provider should have clear understanding what user wants and should explain to user accordingly. The features offered by a service provider should actually fulfill user expectations. This module reviews the user requirements according to the SLA between the user and service provider. If user changes the requirements, he should be referred to the SLA and price for changes should be negotiated accordingly.

4.1.4 Implementation phase

Before the start of the implementation phase, everything is set to start the trustworthy service composition. All the services are composed and RACI is defined among the services. User trust

level is also built. Now we will start the real working in trustworthy environment. As the name shows this phase involves the implementation, which includes the use of trust languages used to communicate among the services, trust protocols, Trust management and trust monitoring mechanisms. Following are the modules in implementation phase

4.1.4.1 Trusted communications: There is a strong need to come up with a common trust protocol or language on behalf of which service can communicate with each other effectively. This will increase the interoperability and performance of the services. This module is used to do the trustable communication among the services. Trustable communication requires use of common trust languages and protocols. Trust languages will be used to communicate among the services, whereas protocols will define the rules for the communication. Complete trust languages and protocols are vital to make this module successful. However, currently there aren't so many languages and protocols, which can actually provide sound platform for services communication during the problem solving. This module also implements the acknowledgement mechanism. The services should acknowledge after completing the task. When acknowledgements from all the services are received, it is considered that the problem resolution is completed. Acknowledgement is another factor that cannot be ignored. The entity, which is sending the data, will wait for the acknowledgement from the receiver. Once the sender receives the acknowledgement, it is assumed that now it is good to start sending the data. Similarly, once all the data is sent, the receiver will again send the acknowledgement to the sender which indicates that data is received successfully. There is already a lot of work done on how acknowledgements are handled on the networks so we can use any of the existing network protocols for handling the scenario. However, this might add some delays in the problem resolution process because of several services involved. Suppose, if we have a pool of services working on problem. On backend, almost all services are using some features from some other services. Therefore, during the task execution, they will wait for the acknowledgement from those services so we have to wait until all the services receive acknowledgement from their providers. But these issues are common problems whereas a number of services involved and can be ignorable until and unless if some services are taking too much time to respond. The importance of discussing the

problem here is to highlight technical difficulties involved. The acknowledgements from the entities will increase the reliability, which will eventually help in gaining user trust and also help in gaining trust among the pool of services. This process is actually an extra check on the performance of the service. This module also implements the security mechanisms. Security mechanisms also include encryption mechanisms, Authentication, Authorization and firewall implementation.

Access control mechanisms should be implemented which makes sure that the client has only access to the required features and no unknown user can get access to the features. Access control mechanisms will help in increasing level of trust among the entities. By having good access control mechanisms, data privacy will improve. It should be the responsibility of the service provider to implement access control mechanisms and will explain to the clients so that they can better trust the provider. Access control mechanisms shouldn't be static and they are supposed to change with the time. As we discussed in the Trust management, the trust relationship among the entities are supposed to change with the time. So, there is a need to keep check on the change in trust so that the access mechanisms will be altered according to the changes. There is another major reason which shows why access control mechanisms are important. There might be a chance any intruder can limit the features offered to the client by provider, which will affect the seller and buyer relationship hence eventually breaks the trust between the entities. A better access control mechanism will help to protect the cloud providers from intruders attack.

4.1.4.2 Trust Management: Trust management is as important as trust build up. Managing the trustworthy environment is a time consuming and tedious task. We have familiarized a new role to manage the trust. Trust agents can be used to manage the trust. Any service provider among the providers can act as a trust agent. Trust agent should be the one, on which there is a consensus. Its role is to manage the trust among the services and service providers and keep check on faulty entities. If an entity is faulty there is a need to identify that particular node and it should be removed from the composition process if it remains faulty. Distributed algorithms are used to identify the faulty nodes. Currently, this idea is more conceptual but in future there is a need to develop trust protocols to manage the trust. New rules can be defined in future. Trust agents are

responsible if there is anything happening, which affects the trust among the participants and users. Trust Agents can work as witnesses in the interaction and can be used to provide direct interaction information [30]. Following are the other issues which are considered in the trust management module. Content privacy of the user data and the data of other services need to be managed in a way that data of all users should be secure and no one violates the privacy constraints. Access control mechanisms are implemented which will make sure data privacy at all levels. Faulty services also need to be handled. By faulty we actually mean services, which do not perform functionalities which they intend to do for example wrong data sharing, acting on behalf of another node without informing service etc. Trust management includes mechanisms, which should handle the faulty services. Consensus algorithms [29] of distributed computing are very useful to handle faulty nodes. Mechanism to handle unregistered services should also be implemented. Incomplete data sharing should also be managed by the trust management module.

4.1.4.3 Trust Monitoring: This module monitors the trustworthy service composition. There is a need to understand the difference between the management and monitoring of the trust. Monitoring is more related to check all the services and trust mechanisms are they working properly? If anything wrong happens, monitoring module informs the trust management module which will then work on the correction. Monitoring feature is very useful for future correspondence with the services because it checks the performance of the services which will help in future trust development. Incomplete information sharing should also be monitored. Efforts should be done for provision of complete information from the cloud providers.

4.1.5 Final Phase

Final phase is also called as termination phase. It is the last phase and service composition ends in this phase and problem resolution is handed over to the user. It contains three modules which are user interaction, service decomposing and performance evaluation. Following are the details

4.1.5.1 User interaction: In this module user is contacted and he/she is provided with the problem resolution. This module not only handed over the final solution to the user but also performs the comparison between the user requirements and the final solution. If in any case user is not satisfied then a review is performed to check where the problem is. This will help in building user confidence and future trustworthy communication.

4.1.5.2 Service decomposing: After the problem solution and user interaction, we move towards service decomposing. In this module, the composed services are decomposed in a trustworthy manner. Service decomposing is a tedious task and involves many processes. Mechanisms should be implemented which make sure that data of other parties is completely removed from the services and user private data is completely deleted from the system. All this decomposing should be done in trustworthy manner and one service will act as a coordinator to deal all this decomposing process.

4.1.5.3 Performance evaluation: The last module is the performance evaluation. This module will be a key for future service composition of such problems and trustworthy communication. The adopted trustworthy service composition is analyzed and future improvements should be taken in consideration. Performance evaluation includes services individual performance and their collaborative correspondences secondly, how much user and service trust level is achieved by following the trustworthy approaches.

5 Performance Evaluation

The proposed TSCCM is useful for Manual service composition and provides trustworthy service composition between the user and service providers and among the service providers. Use of our TSCCM will minimize the time require to create SLA and Trust. Model is also helpful to improve accuracy in service composition and data content. We have used some metrics and parameters to judge the performance of our model. Values of parameters are randomly selected and graphs are generated on the basis of the values. The performance of the model is discussed on the basis of generated graphs.

5.1 Metrics

Metrics represents benefits of our TSCCM. Below mention factors will improve when TSCCM will be implemented.

1. Time to create the SLA; Gama (ɣ)

   Model is useful to reduce the SLA creation time among the parties involved in service composition process. We have used ɣ to calculate the time to create SLA.

2. Accuracy in data content; Alpha (α)

   As discussed earlier in the thesis, implementation of our TSCCM improves the accuracy in data content. Data is frequently shared among the parties involved in the service composition process and it is very important that data sharing should be bug free. Use of model improves the data accuracy considerably.

3. Time to reach trust; Beta (β)

   We have used β to calculate, time to reach Trust. Implementation of model reduces time, required to reach trust.

4. Time require to compose services; Mu (μ)

   Use of TSCCM reduces the time require to compose services considerably.

5. Probability of reaching trust (P)

We have used this parameter to see on what factors probability of reaching trust depends.

5.2 Parameters

Parameters represent factors on the basis of which we can judge performance of our TSCCM. Below mention parameters are used to judge performance of our TSCCM.

1. Service fan-out (s): shows number of services use in the composition process
2. Cloud fan-out (c): represents number of service providers involve in the composition process
3. Depth of nest (n): represents depth of services nest
4. Third party fan-out (t): shows number of third parties
5. Reusability factor (R): shows reuse services
6. User credibility check (C): shows user credibility
7. Module implementation (M): shows TSCCM modules implementation
8. Service complexity (ω): Complexity of services

5.3 Relationship between Metrics and Parameters

5.3.1 Time require to reach SLA

$$\gamma = \frac{c * n}{s * R}$$

Equation 1: Time to create the SLA

Equation 1 shows the relationship between time to create SLA and cloud providers, services, and depth of service nest and reusability factor. Reusability means if the services need by user are already available or composed then it would be easy to create the SLA. In another scenario, if we have an old user and he/she wants the same solution or with some minor changes then it would take less time to create the SLA. Gamma increases if the cloud fan-out increases and number of services and depth of nest decrease, Hence, it takes more time to create SLA if there is more

cloud providers involve. Gamma decreases if the cloud fan-out decreases and number of services and depth of nest increase, Hence, it takes less time to create SLA if there is less cloud providers involve. If the ratio of increase or decrease in cloud fan-out and service fan-out and depth of nest remains same then gamma also increases or decreases accordingly.

We have used some random numbers to show cloud fan-out, service fan-out and depth of nest in the form of graph. Following table and graph shows the relationship between gamma and the parameters.

Cloud Providers (c)	Number of services (s)	Depth of Nest (n)	Reusability factor (R)	SLA Time (ɣ)
11	36	11	8	0.420138889
38	10	16	9	6.755555556
38	7	10	1	54.28571429
48	46	13	2	6.782608696
12	24	6	6	0.5
12	33	10	1	3.636363636
11	39	25	8	0.881410256
31	40	8	5	1.24
32	37	17	8	1.837837838

Table 2: Time to create SLA

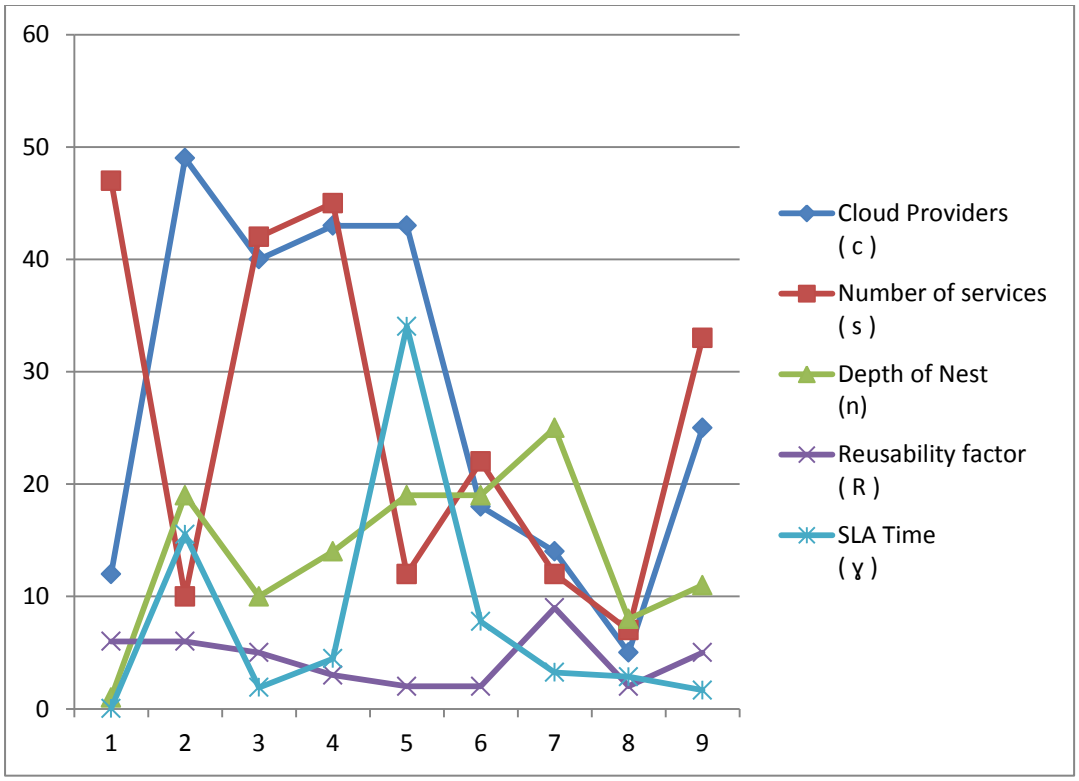

Figure 10: Time to create SLA

As shown in the graph; if c and n increases where as s and R decreases then gamma increases which means more time is require to create SLA similarly if s and R increases where as c and n decreases then gamma decreases which means less time is require to create SLA. If the parameters increase or decrease gradually then gamma also increases or decreases according to the parameters

5.3.2 Alpha (α); Accuracy in data content

$$\alpha = \frac{s * n}{t * c} \quad ; \quad \alpha \text{ proportional } C$$

Equation 2: Accuracy in Data Content

The above relationship between **α** and parameters show; Data accuracy directly depends on the service fan-out and depth of nest where as inversely on third parties and cloud providers. If there

are more third parties and cloud providers involve then data accuracy is less otherwise we have less cloud providers and third parties and greater services and nested services then we get higher data accuracy. Use of third parties and cloud providers includes different data formats and there is greater complexity as compare to using more services. Our TSCCM model recommends use of less cloud providers and third parties so that data accuracy can be improve. Maintaining data accuracy among the services from same cloud provider are much easier than from different cloud providers similarly data sharing with one or two third parties is easier and accurate as compare to several third parties.

Other than relationship discuss above, α also depends on credibility check. Credibility check shows reliability of a service. Data accuracy is directly proportional to the credibility check. Higher the service credibility higher would be the data accuracy. Graph shows that increase in service credibility increases data accuracy. Following table shows parameters, we have used to analyze data accuracy.

Depth of Nest (n)	Third Party (t)	Cloud Providers (c)	Number of services (s)	Data Accuracy (α)
35	18	29	28	1.877394636
16	5	12	29	7.733333333
31	8	22	39	6.869318182
28	30	28	25	0.833333333
26	7	17	8	1.74789916
21	20	4	32	8.4
30	8	22	38	6.477272727
14	17	6	7	0.960784314
38	23	35	16	0.755279503
14	26	26	10	0.207100592

Table 3: Accuracy in Data content

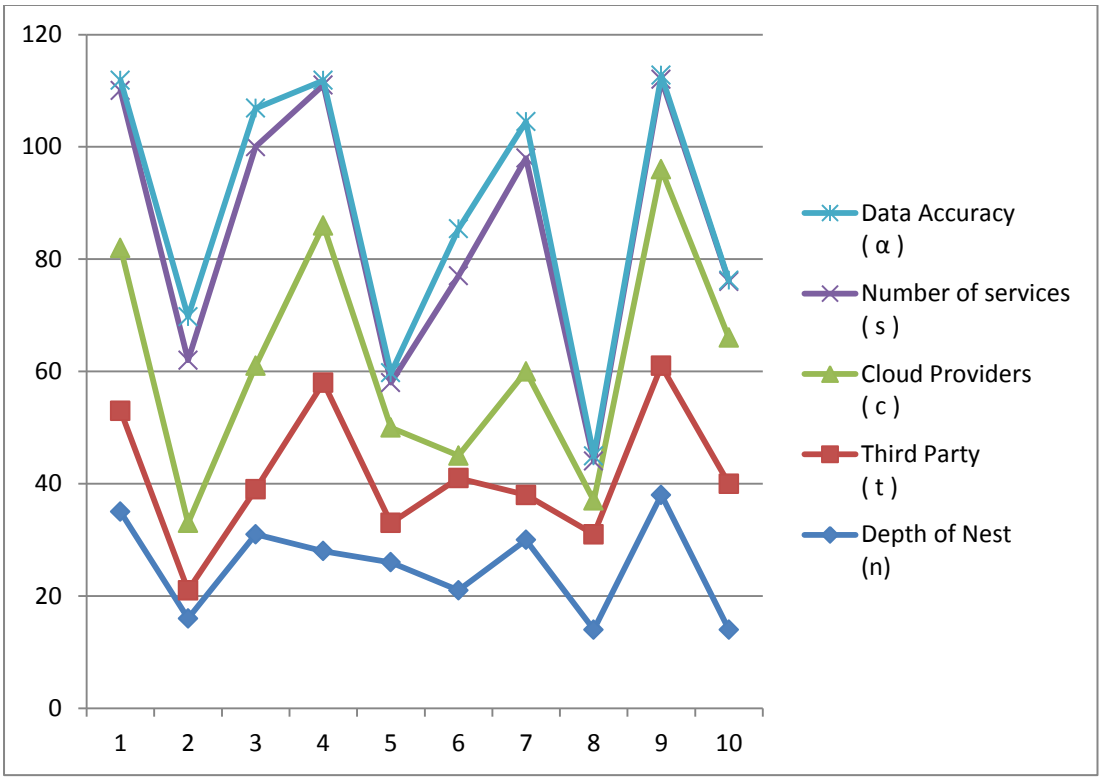

Figure 11: Accuracy in Data content

Above graph shows the relationship between Accuracy in data content and the parameters. We can see from the graph if we have higher number of services and depth of Nest we get higher data accuracy and vice versa similarly if the changes in parameters are constant than data accuracy changes accordingly. Following graph shows relationship between Accuracy and credibility check. Use of credible services increases the data accuracy.

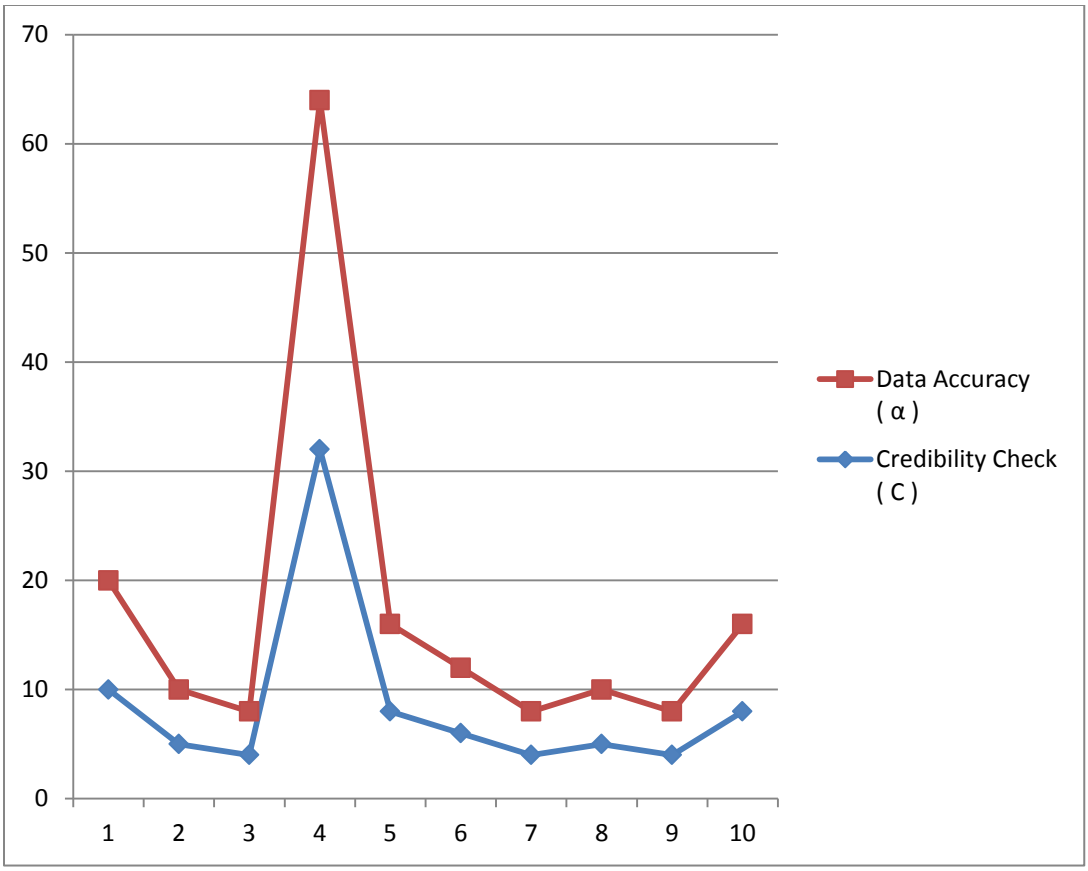

Figure 12: Accuracy in data content

5.3.3 Beta (β); Time to reach trust

$$\beta = \frac{s * n}{c * R}$$

Equation 3: Time to reach Trust

Above relationship shows that time requires to build trust depends on the number of services, depth of services nest and number of cloud providers. If we have more services and services nest than the number of cloud providers, it takes more to reach the Trust. During the service composition process, we need trust among the services. If the number of services are higher than it takes more time to reach trust similarly if the services nest is higher it takes more time to reach

trust. Time to reach trust is considerably decreased if we have a higher reusability factor. Service reusability requires less time to build trust. We have used some random values to see how the parameters affects the time requires to build trust.

Cloud Providers (c)	Number of services (s)	Depth of Nest (n)	Reusability Factor (R)	Time to reach Trust (β)
26	15	4	5	0.461538462
25	6	1	4	0.06
16	76	10	3	15.83333333
8	39	10	2	24.375
27	81	10	3	10
30	74	3	3	2.466666667
25	47	8	3	5.013333333
28	68	8	4	4.857142857
10	53	2	2	5.3
18	65	8	2	14.44444444

Table 4: Time to reach Trust

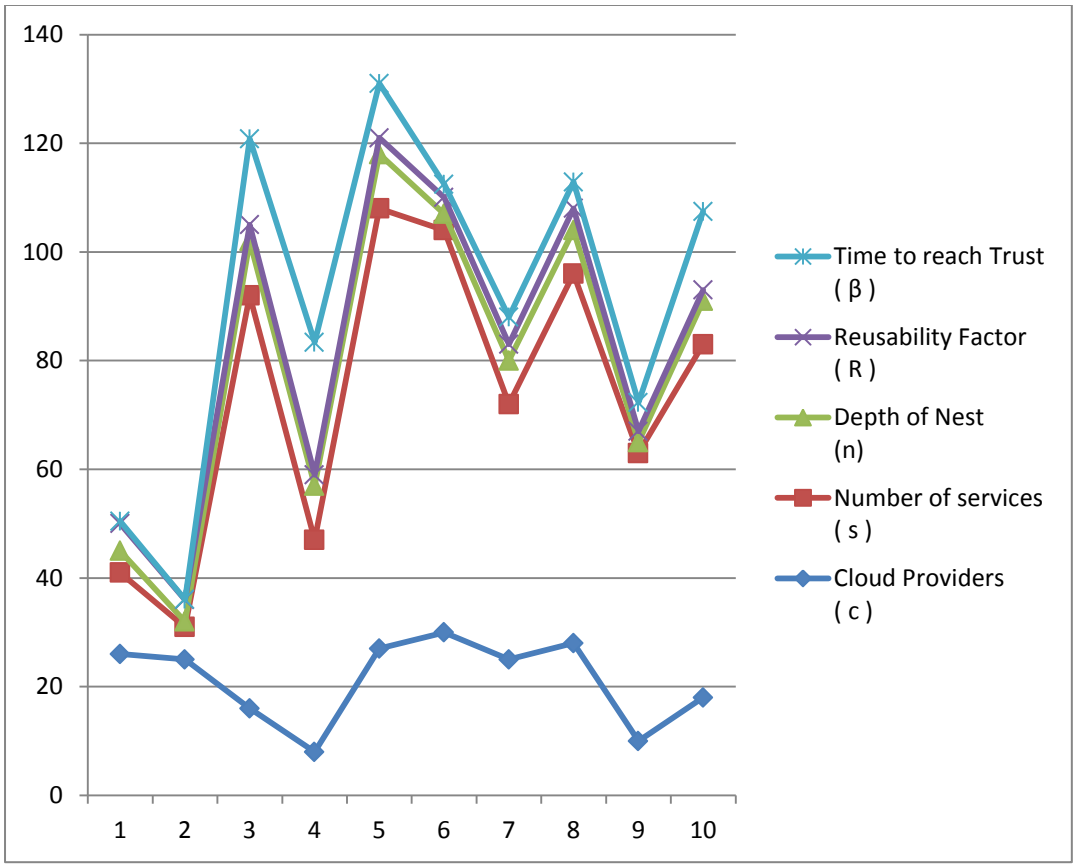

Figure 13: Time to reach Trust

Above graph shows relationship among Beta and other parameters. We can see from the graph that if we have higher number of cloud providers and Reusability factor is high as compare to number of services and services nest, it takes less time to reach trust. During the service composition, services interact with each other frequently and trustworthy environment is required. If we have more services then it takes more time to reach the trust similarly if the depth of services nest is high then it takes more time to reach the trust. Trust agreement among services is more continuous as compare to Cloud providers. Reusability factor plays vital role in trust development and time require to develop trust reduces considerably, if we are using any existing combination of services. Higher the reusability factor, lesser time require to build Trust. The above graph also shows that if we have higher reusability factor and Cloud providers as compare to services and depth of nest it takes less time to build trust and vice versa. Similarly if the increase or decrease is gradual then the Trust time also increases or decreases accordingly.

5.3.4 Mu (μ); Time to compose services

$$\mu = \frac{\omega}{R}$$

Equation 4: Time to compose services

Our model is useful to gather user requirements in less time. Time efficient gathering of user requirements speed up the service composition process. Use of model improves the time to gather user requirements which eventually reduces the time to compose services. However, there are some parameters on which service composition is dependant; which are ω and R. μ **,** depends on service complexity and the reusability factor. Following table shows the random values we have use as a sample to check the relationship between the metric and the parameters.

Service complexity(ω)	Reusability factor (R)	Time to compose services (μ)
39	20	1.95
98	28	3.5
34	48	0.708333333
88	35	2.514285714
5	19	0.263157895
58	17	3.411764706
29	19	1.526315789
2	4	0.5
11	42	0.261904762
59	3	19.66666667

Table 5: Time to compose services

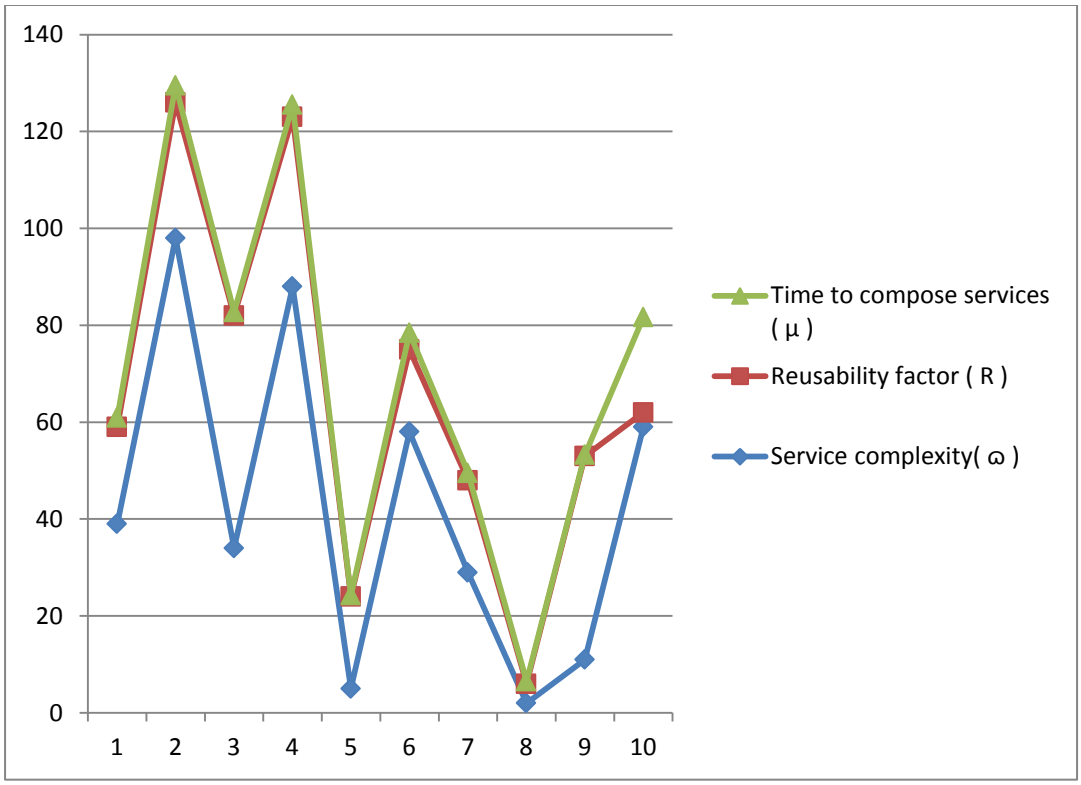

Figure 14: Time to compose services

We can see from the above graph that complexity of service directly depends on the time require to compose services. Services pool, which has high complexity requires more time to compose services. Higher the services complexity, it takes more time to compose the services. Reusability factor is inversely proportional to the time require to compose services. As we discussed earlier, it plays vital role and reduces the time require to compose services considerably. Higher the reusability factor, it takes less time to compose services.

5.3.5 P; probability of reaching trust

$$P \text{ depends on } C \text{ and } M$$

Equation 5: Probability of reaching trust

Probability of reaching trust depends on the service credibility and implementation of TSCCM modules. If the user is credible and all the modules are implemented then there is greater chance to build trust. Estimating probability of trust is very important metric because we can judge

performance of our model by this. If the use services are credible and all modules of trustworthy model are implemented properly than there would be greater probability of reaching. Following table shows random values we have used to analyze probability of reaching trust.

Credibility Check (C)	Module Implementation (M)	Probability of Trust (P)
10	10	10
20	20	20
5	5	5
30	30	30
25	25	25
35	35	35
40	40	40
45	45	45
50	50	50
55	55	55

Table 6: Probability to reach trust

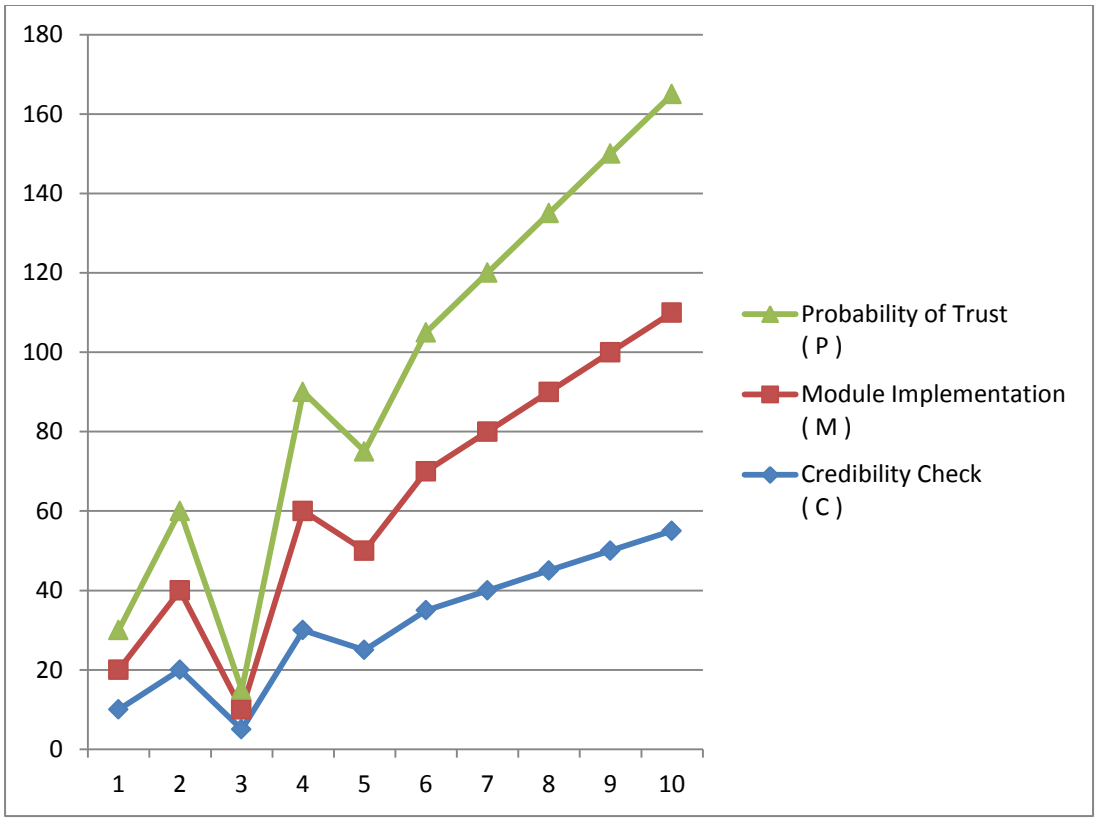

Figure 15: Probability to reach trust

As we can see from the above graph that if the service are credible and all modules of TSCCM are implemented than we have a greater probability of reaching trust.

6 Conclusion

Trustworthy service composition has emerged as an important factor in cloud computing. This feature has made problem resolution easier but there are several challenges which are needed to be addressed. We have discussed in this thesis that trustworthy service composition has several challenges and to successfully implement it in cloud environment, there is a need of robust trustworthy manual service composition models. Evolution of Service composition brought composition and trust development challenges, discussed in paper. These initial challenges should be addressed before full deployment of Service composition in intercloud.

After having in detail study of the literature and knowing the emerging challenges, we aimed at providing a solution for trustworthy manual service composition. Our main objective was to bring up the service composition and trust development challenges and then to propose a model which could be helpful in composing services and doing trustable communication among the services.

TSCCP provides a framework for interaction among the entities and it will really help in building trustable environment for service cloud. (TSCCP) will improve QoS in inter-clouds, which is main area of focus now days. Experimental results proves that implementation of model improves time require to create the SLA, accuracy in data content, reach trust and compose services considerably. Implementation of model will lead to trustworthy environment in manual service composition.

Our propose model provides platform for trustworthy manual service composition and is very useful to create SLA and build trust among the services in intercloud and will serve as a base point for future manual service composition models. It is expected that the implementation of model will help in composing services and building relationships among the services more affectively.

References:

1. WENGUANG WANG NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY, CHANGSHA 410073, CHINA WGWANGNUDT@GMAIL.COM, ANDREAS TOLK+ OLD DOMINION UNIVERSITY, NORFOLK, VA 23529, UNITED STATES, ATOLK@ODU.EDU , WEIPING WANG NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY, CHANGSHA 410073, CHINA WANGWP@NUDT.EDU.CN : THE LEVELS OF CONCEPTUAL INTEROPERABILITY MODEL: APPLYING SYSTEMS ENGINEERING PRINCIPLES TO M&S, YEAR 2009.
2. MARGARET ROUSE, FEDERATED CLOUD DEFINITION: HTTP://WHATIS.TECHTARGET.COM/DEFINITION/FEDERATED-CLOUD-CLOUD-FEDERATION: YEAR JULY, 2011
3. MARCIA SAVAGE: FEDERAL CLOUD COMPUTING STRATEGY FACES CHALLENGES, GAO FINDS : HTTP://ITKNOWLEDGEEXCHANGE.TECHTARGET.COM/SECURITY-BYTES/FEDERAL-CLOUD-COMPUTING-STRATEGY-FACES-CHALLENGES-GAO-FINDS/: YEAR, JULY 2012.
4. ABRHA, ABRHA AND HEGGEN SKOGEN, ENDRE (NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY, FACULTY OF INFORMATION TECHNOLOGY, MATHEMATICS AND ELECTRICAL ENGINEERING, DEPARTMENT OF TELEMATICS) TITLE: SERVICE COMPOSITION IN A CLOUD COMPUTING ENVIRONMENT : YEAR, 2011
5. AUTHOR(S): GUTIERREZ-GARCIA, J.O. DEPT. OF INF. & COMMUN., GWANGJU INST. OF SCI. & TECHNOL., GWANGJU, SOUTH KOREA KWANG-MONG SIM: SELF-ORGANIZING AGENTS FOR SERVICE COMPOSITION IN CLOUD COMPUTING CLOUD COMPUTING TECHNOLOGY AND SCIENCE (CLOUDCOM), 2010 IEEE SECOND INTERNATIONAL CONFERENCE ON; DATE OF CONFERENCE: NOV. 30 2010-DEC. 3 2010
6. YURI DEMCHENKO, JEROEN VAN DER HAM, RUDOLF STRIJKERS, MATTIJS GHIJSEN, CANH NGO, MIHAI CRISTEA: GENERIC ARCHITECTURE FOR CLOUD INFRASTRUCTURE AS A SERVICE (IAAS) PROVISIONING MODEL RELEASE 1: 15 APRIL 2011
7. JIE TAO, DANIEL FRANZ, HOLGER MARTEN, AND ACHIM STREIT STEINBUCH CENTER FOR COMPUTING KARLSRUHE INSTITUTE OF TECHNOLOGY, GERMANY {JIE.TAO, HOLGER.MARTEN, ACHIM.STREIT}@KIT.EDU, DANIEL2712@GMX.DE8:HTTP://DL.ACM.ORG/CITATION.CFM?ID=2002420: AN

IMPLEMENTATION APPROACH FOR INTER-CLOUD SERVICE COMBINATION INTERNATIONAL JOURNAL ON ADVANCES IN SOFTWARE, VOL 5 NO 1 & 2, YEAR 2012.

8. AUTHORS:YONGGANG WEN NANYANG TECHNOLOGICAL UNIVERSITY, SINGAPOREGUANGYU SHI INNOVATION CENTER, HUAWEI, SANTA CLARA, CAGUOQIANG WANGINNOVATION CENTER, HUAWEI, SANTA CLARA, CA
: DESIGNING AN INTER-CLOUD MESSAGING PROTOCOL FOR CONTENT DISTRIBUTION AS A SERVICE (CODAAS) OVER FUTURE INTERNET: PUBLISHED IN: CFI '11 PROCEEDINGS OF THE 6TH INTERNATIONAL CONFERENCE ON FUTURE INTERNET TECHNOLOGIES, ACM NEW YORK, NY, USA ©2011
9. AUTHOR: MUNINDAR P. SINGH DEPARTMENT OF COMPUTER SCIENCE NORTH CAROLINA STATE UNIVERSITY RALEIGH, NC 27695-7535, USA SINGH@NCSU.EDU: TRUSTWORTHY SERVICE COMPOSITION: CHALLENGES AND RESEARCH QUESTIONS PUBLICATION: YEAR, 2002.
10. AUTHORS: G. KAPITSAKI, D.A. KATEROS, I.E. FOUKARAKIS, G. N. PREZERAKOS, D.I. KAKLAMANI AND I.S. VENIERIS: SERVICE COMPOSITION: STATE OF THE ART AND FUTURE CHALLENGES: YEAR, 2007.
11. ERDAL CAYIRCI ELECTRICAL ENGINEERING & COMPUTER SCIENCE DEPARTMENT UNIVERSITY OF STAVANGER, NORWAY E-MAIL: ERDAL.CAYIRCI@UIS.NO : A SCHEME FOR LOCATING AND MIGRATING FEDERATION COMPONENTS IN A DISTRIBUTED SIMULATION INTERCLOUD: YEAR, 2012.
12. ERDAL CAYIRCI, ELEC. ENG. & COMP. SCI. DEP. UNIVERSITY OF STAVANGER, NORWAY CHUNMING RONG, ELEC. ENG. & COMP. SCI. DEP. UNIVERSITY OF STAVANGER, NORWAY MACIEJ KOCZUR, SMC4 DIVISION NATO JWC STAVANZER, NORWAY, KAI HWANG, ELEC. ENG. & COMP. SCI. DEP. UNIVERSITY OF SOUTHERN CALIFORNIA LOS ANGELES, CALIFORNIA:A MULTI-CRITERIA DESIGN SCHEME FOR SERVICE FEDERATING INTER-CLOUD APPLICATIONS: YEAR, 2012.
13. HTTP://EN.WIKIPEDIA.ORG/WIKI/CONCEPTUAL_INTEROPERABILITY
14. YURI DEMCHENKO, JEROEN VAN DER HAM, RUDOLF STRIJKERS, MATTIJS GHIJSEN, CANH NGO, MIHAI CRISTEA: GENERIC ARCHITECTURE FOR CLOUD INFRASTRUCTURE AS A SERVICE (IAAS) PROVISIONING MODEL RELEASE 1: 15 APRIL 2011

15. KHALIL, ISMAIL; MANTORO, TEDDY (EDS.): TRUSTWORTHY UBIQUITOUS COMPUTING SERIES: ATLANTIS AMBIENT AND PERVASIVE INTELLIGENCE, VOL. 6: YEAR, 2012.
16. Y. WEI AND M. B. BLAKE: "SERVICE-ORIENTED COMPUTING AND CLOUD COMPUTING: CHALLENGES AND OPPORTUNITIES," IEEE INTERNET COMPUTING, VOL. 14, NO. 6, PP. 72–75, 2010.
17. R. AGGARWAL, K. VERMA, J. MILLER, AND W. MILNOR, "CONSTRAINT DRIVEN WEB SERVICE COMPOSITION IN METEOR-S," IN PROCEEDINGS OF THE IEEE INTERNATIONAL CONFERENCE ON SERVICE COMPUTING, 2004, PP. 23–30.
18. C.-H. HSUAND AND H. JIN: SERVICES COMPOSITION AND VIRTUALIZATION TECHNOLOGIES," IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 4, NO. 3, PP. 181–182, 2011.
19. DIPANJAN CHAKRABORTY PH.D. STUDENT UMBC DCHAKR1@CS.UMBC.EDU YELENA YESHA PROFESSOR UMBC YEYESHA@CS.UMBC.EDU ANUPAM JOSHI ASSOCIATE PROFESSOR UMBC JOSHI@CS.UMBC.EDU: A DISTRIBUTED SERVICE COMPOSITION PROTOCOL FOR PERVASIVE ENVIRONMENTS: YEAR, 2004.
20. ANDREAS WOMBACHER UNIVERSITY OF TWENTE, ENSCHEDE, THE NETHERLANDS A.WOMBACHER@UTWENTE.NL: DECENTRALIZED DECISION MAKING PROTOCOL FOR SERVICE COMPOSITION: YEAR, 2005.
21. EDUARDO SILVA, LU´IS FERREIRA PIRES, MARTEN VAN SINDEREN CENTRE FOR TELEMATICS AND INFORMATION TECHNOLOGY UNIVERSITY OF TWENTE, THE NETHERLANDS P.O. BOX 217, 7500 AE ENSCHEDE {E.M.G.SILVA, L.FERREIRAPIRES, M.J.VANSINDEREN}@CS.UTWENTE.NL: SUPPORTING DYNAMIC SERVICE COMPOSITION AT RUNTIME BASED ON END-USER REQUIREMENTS: YEAR, 2009.
22. ADRIAN KLEIN THE UNIVERSITY OF TOKYO JAPAN ADRIAN@NII.AC.JP FUYUKI ISHIKAWA NATIONAL INST. OF INFORMATICS JAPAN F-ISHIKAWA@NII.AC.JP , SHINICHI HONIDEN THE UNIVERSITY OF TOKYO NATIONAL INST. OF INFORMATICS JAPAN HONIDEN@NII.AC.JP: TOWARDS NETWORK-AWARE SERVICE COMPOSITION IN THE CLOUD: APRIL 16–20, 2012.
23. DAVID BERNSTEIN DEEPAK VIJ HUAWEI TECHNOLOGIES, USA 2330 CENTRAL EXPRESSWAY SANTA CLARA, CA 95050 CLOUD STRATEGY PARTNERS, LLC 3260 NIPOMA COURT SAN JOSE, CA 95135 DBERNSTEIN@HUAWEI.COM DEEPAK@CLOUDSTRATEGYPARTNERS.COM: USING SEMANTIC WEB ONTOLOGY FOR

INTERCLOUD DIRECTORIES AND EXCHANGES: YEAR, 2010: FOR SUBMISSION TO ICOMP'10 - THE 2010 INTERNATIONAL CONFERENCE ON INTERNET COMPUTING, LAS VEGAS, "V JUL 12-15 2010 AS PART OF WORLDCOMP'10 - THE 2010 WORLD CONGRESS IN COMPUTER SCIENCE, COMPUTER ENGINEERING, AND APPLIED COMPUTING.

24. YURI DEMCHENKO, MARC X. MAKKES, RUDOLF STRIJKERS, CANH NGO, CEES DE LAAT, JOAN A. GARCIA-ESPIN CONTACT: YURI DEMCHENKO <Y.DEMCHENKO@UVA.NL> CONTRIBUTING PROJECTS GEYSERS – GENERALISED ARCHITECTURE FOR INFRASTRUCTURE SERVICES - HTTP://WWW.GEYSERS.EU/ GEANT3 JRA3 TASK 3 – COMPOSABLE SERVICES (GEMBUS) - HTTP://WWW.GEANT.NET/ YURI DEMCHENKO, MARC X. MAKKES, RUDOLF STRIJKERS, CANH NGO, CEES DE LAAT (UVA), JOAN A. GARCIA-ESPIN (I2CAT): DEFINING INTER-CLOUD ARCHITECTURE FOR INTEROPERABILITY AND INTEGRATION: YEAR, 2012.
25. PATCHAREE THONGTRA, THESIS FOR THE DEGREE OF PHILOSOPHIAE DOCTOR TRONDHEIM NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY, MATHEMATICS AND ELECTRICAL ENGINEERING DEPARTMENT OF TELEMATICS: A SERVICE FRAMEWORK FOR CAPABILITY-BASED ADAPTATION IN ADAPTABLE SERVICE SYSTEMS: YEAR, SEPTEMBER, 2012.
26. MEDJAHED BRAHIM, BOUGUETTAYA, ATHMAN: CHAPTER 2: ENABLING INTERACTIONS ON THE WEB: A TAXONOMIC PERSPECTIVE: YEAR: 2011 PRINTED: NEW YORK, NY: SPRINGER NEW YORK.
27. D. GAMBETTA, "CAN WE TRUST TRUST?" IN D. GAMBETTA (ED.), TRUST: MAKING AND BREAKING COOPERATIVE RELATIONS, NEW YORK, NY: BASIL BLACKWELL, 1988, P. 217.
28. AUDUN JØSANG1 AND ST´EPHANE LO PRESTI DSTC _, QUEENSLAND UNIVERSITY OF TECHNOLOGY, GPO BOX 2434, BRISBANE QLD 4001, AUSTRALIA. AJOSANG@DSTC.EDU.AU; UNIVERSITY OF SOUTHAMPTON __, SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE, SOUTHAMPTON SO17 1BJ, UNITED KINGDOM SPLP@ECS.SOTON.AC.UK: ANALYSING THE RELATIONSHIP BETWEEN RISK AND TRUST: YEAR, 2004.
29. CHRISTIAN CACHIN, RACHID GUERRAOUI, AND LUIS RODRIGUES: INTRODUCTION TO RELIABLE AND SECURE DISTRIBUTED PROGRAMMING, SECOND EDITION: YEAR, 2011.

30. XIAONING MA† SCHOOL OF COMPUTER SCIENCE AND TECHNOLOGY, CIVIL AVIATION UNIVERSITY OF CHINA, TIANJIN 300300, CHINA: TRUSTWORTHY SERVICE COMPOSITION USING SWIFT TRUST IN SON: JOURNAL OF COMPUTATIONAL INFORMATION SYSTEMS 6:14 (2010) 4761-4766 AVAILABLE AT HTTP://WWW.JOFCIS.COM 1553-9105/ COPYRIGHT © 2010 BINARY INFORMATION PRESS DECEMBER, 2010.
31. TYRONE GRANDISON, MORRIS SLOMAN IMPERIAL COLLEGE, DEPARTMENT OF COMPUTING 180 QUEEN'S GATE, LONDON SW7 2BZ, UK {TGRAND, M.SLOMAN} @ DOC.IC.AC.UK: A SURVEY OF TRUST IN INTERNET APPLICATIONS: IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, FOURTH QUARTER 2000, HTTP://WWW.COMSOC.ORG/PUBS/SURVEYS/, 24 JANUARY 2001.
32. MUNINDAR P. SINGH DEPARTMENT OF COMPUTER SCIENCE NORTH CAROLINA STATE UNIVERSITY RALEIGH, NC 27695-8206, USA SINGH@NCSU.EDU : TRUST AS DEPENDENCE: A LOGICAL APPROACH: THE 10TH INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS - VOLUME 2 PAGES 863-870, 2011.
33. PATRÍCIA DOCKHORN COSTA ENSCHEDE, THE NETHERLANDS: ARCHITECTURAL SUPPORT FOR CONTEXT-AWARE APPLICATIONS: FROM CONTEXT MODELS TO SERVICES PLATFORMS: CTIT PH.D.-THESIS SERIES, NO. 07-108 TELEMATICS INSTITUTE FUNDAMENTAL RESEARCH SERIES, NO. 021 (TI/FRS/021),2007.
34. SECURE FILE SHARING AND COLLABORATION IN THE CLOUD: MAXIMIZING THE BENEFITS WHILE MINIMIZING THE RISKS: ACCELLION, INC. TEL +1 650 485-4300 1804 EMBARCADERO ROAD FAX +1 650 485-4308 SUITE 200 WWW.ACCELLION.COM PALO ALTO, CA 94303 INFO@ACCELLION.COM, WHITE PAPER, SEPTEMBER 2012.
35. FAHIMA CHEIKH INSTITUT DE RECHERCHE EN INFORMATIQUE DE TOULOUSE, 118, ROUTE DE NARBONNE 31062 TOULOUSE, FRANCE CHEIKH@IRIT.FR GIUSEPPE DE GIACOMO MASSIMO MECELLA UNIV. ROMA LA SAPIENZA DIPARTIMENTO DI INFORMATICA E SISTEMISTICA VIA SALARIA 113 – 00198 ROMA, ITALY FDEGIACOMO,MECELLAG@DIS.UNIROMA1.IT: AUTOMATIC WEB SERVICES COMPOSITION IN TRUSTAWARE COMMUNITIES, NOVEMBER, 2006.
36. ELISABETTA ERRIQUEZ DEPARTMENT OF COMPUTER SCIENCE UNIVERSITY OF LIVERPOOL LIVERPOOL L69 3BX, UK E.ERRIQUEZ@LIVERPOOL.AC.UK: DEALING WITH TRUST AND DISTRUST IN AGENTS SOCIETIES (EXTENDED ABSTRACT), 2011.

37. David Llewellyn-Jones Zeta Dooly Marina Egea González http://aniketos.eu: Aniketos: Ensuring Trustworthiness and Security in Service Composition Exploitation and Impact 6th September 2012 Effects plus Clustering Event, Padua, Italy.
38. Sharon M. Paradesi (Under the direction of Prashant Doshi): Integrating Behavioral Trust and Reputation in Web Service Compositions, 2009.
39. Sharon Paradesi, Prashant Doshi and Sonu Swaika LSDIS Lab, Dept. of Computer Science University of Georgia Athens, GA, 30602 paradesi,pdoshi,swaika}@cs.uga.edu: Integrating Behavioral Trust in Web Service Compositions, 2009.
40. CHUNG-WEI HANG and MUNINDAR P. SINGH North Carolina State University: Trustworthy Service Selection and Composition, February 2011.
41. Matt Blaze1, Joan Feigenbaum1, John Ioannidis1, and Angelos D. Keromytis2 1 AT&T Labs – Research 180 Park Avenue Florham Park, NJ 07932USA {mab,jf,ji}@research.att.com 2 Distributed Systems Lab CIS Department, University of Pennsylvania 200 S. 33rd Str., Philadelphia, PA 19104 USA angelos@dsl.cis.upenn.edu:The Role of Trust Management in Distributed Systems Security, 2000.
42. Shahab Mokarizadeh1, Nima Dokoohaki1, Mihhail Matskin, Peep Küngas3: ICT School, Royal Institute Of Technology (KTH), Stockholm Sweden, Norwegian University of Science and Technology (NTNU), Trondheim, Norway University of Tartu, Tartu, Estonia {shahabm, nimad, misha}@kth.se, peep.kungas@ut.ee: Trust and Privacy Enabled Service Composition using Social Experience , 2010.