Faculty of Engineering and Technology

Master Program of Computing

Master Thesis

Optimizing security for software deployed on cloud environment using evolutionary search

تحسين أمن البرمجيات التي تعمل في البيئة السحابية باستخدام الخوارزميات الجينية

Author: Wafaa Radwan

Supervisor: Dr. Yousef Hassouneh

This Thesis is submitted in fulfillment of the requirements for the Master Degree in Computing from the Faculty of Engineering and Technology at Birzeit University, Palestine

June 2016

Optimizing security for software deployed on cloud environment using evolutionary search

By Wafaa Radwan

Approved by the thesis committee:

Dr. Yousef Hassouneh, Birzeit University

....

Dr. Abdel Salam Sayyad, Birzeit University

....

Dr. Nariman Ammar, Birzeit University

....

Date Approved:

Birzeit University

Faculty of Engineering and Technology

Master Program of Computing

Master Thesis

“Optimizing security for software deployed on cloud environment using evolutionary search”

By Wafaa Radwan

Abstract

This research studies heuristic search-based optimization of service compositions. We have investigated applying Genetic Algorithms (GA) to optimize service-oriented architectures in terms of security goals and cost.

Service composition security risk is measured by implementing the aggregation rules from the local security risk values of the aggregated services in the composition. We adapt the DREAD model for Security risk assessment by suggesting new categorizations for calculating DREAD factors based on a proposed service structure and service attributes.

We implemented the YAFA-SOA Optimizer as an extension of an existing implementation of the GA to solve multi-objective optimization problems for varying number of objectives in the context of service oriented architectures.

We conducted an experiment to investigate our Research Questions. The experiment results showed that applying multi-objective GA is feasible to find the optimized security and cost in Service oriented architectures. We were able to approve that adding security services to the generated composition reduces the risk severity of the generated composition and enhances its security in terms of confidentiality, integrity and availability (CIA).

المقدمة

تحسين أمن البرمجيات التي تعمل في البيئة السحابية باستخدام الخوارزميات الجينية

وفاء رضوان

قدمت هذه الاطروحة دراسة لايجاد النموذج الامثل للتركيبة الخدماتية (Service Composition) باستخدام الخوارزميات الجينية بالنسبة لمتطلبات الأمن و التكلفة.

اعتمد النظام المقترح في قياس جودة التركيبة الخدماتية على المعادلات التجميعة (aggregation rule). و تم قياس جودة الخدمة (Service) عن طريق تبني نظام دريد (Dread Model) و ذلك عن طريق اضافة تصنيفات جديدة عليه تتناسب مع مجموعة من الخصائص المقترحة للخدمة (Service) و التي تمكن من قياس جودتها من ناحية أمن المعلومات و التكلفة.

تم بناء نظام (YAFA-SOA Optimizer) و الذي بني باستخدام (Jmetal framework) لحل المشاكل متعددة الأهداف بالنسبة لنظام التركيبة الخدماتية(Service Composition).

بعد اجراء التجربة تبين امكانية استخدام البحث التطويري في ايجاد التركيبة الخدماتية(Service Composition) الأمثل لمتطلبات الأمن و التكلفة.

كما و تمكنا من تحسين الأمن في التركيبة الخدماتية (Service Composition) عن طريق اضافة خدمات أمن (Security Service) للتركيبة من ناحية السرية و النزاهة و التوافر.

Keywords:

Service oriented architecture, Meta-heuristic Search, Multi-Objective Optimization, Search-Based Software Engineering, Service Composition, Security, Cost Effectiveness, J-metal, Genetic Algorithm, NSGA-II, DREAD, and Security Model.

Acknowledgements and Dedication

My thanks and appreciation to my Supervisor Dr. Yousef Hassouneh, I truly appreciate his help and supportive encouragement throughout the research time. I would like to thank him for his effort and time; it was a pleasure to work with him.

I would like to thank the members of my dissertation committee, Dr. Abdel Salam Sayyad and Dr. Nariman Ammar, thanks for your time and valuable comments.

My special thanks to my Husband, for his support and encouragement.
I am grateful for my family; Mom, Dad, my sisters and brothers to their encouragements and support.

This dissertation is dedicated to the friendship and memory of Dalal Ibraghith. She was my colleague in master program, but she left us early, may her soul rest in peace

Table of Contents

Abstract III

المقدمة IV

Keywords: V

Acknowledgements and Dedication VI

List of Figures IX

List of Tables XI

Abbreviations XIII

Chapter 1..1

Introduction..1

1.1 Introduction and motivation...1
1.1 Research Methodology 4
1.2 Contribution 5
1.3 Organization of this thesis 6

Chapter 2..7

Background 7

2.1 Introduction...7
2.2 Cloud computing architecture..7
2.3 Service oriented architecture..11
2.4 Service composition aggregation patterns 14
2.5 Cloud computing security concerns...18
2.6 Search based software engineering using Genetic algorithm 20
2.6 Multi-objective Genetic algorithm “NSGA-II” 24

Chapter 3..26

Literature Review...26

3.1 Search Based optimization..26
3.2 Security for SaaS...29
3.3 Security Analysis and risk assessment: Models and Approaches........32
3.4 QoS of the Service composition architecture...................................44
3.5 Service oriented architecture as multiple objectives problem 48

Chapter 4..61

Research Methodology..61

4.1 Introduction 61
4.2 Literature Review: 62
4.3 Problem Definition: 63
4.4 Implemented Genetic Algorithm: 64
4.5 Running the experiment: 64

Chapter 5 66

Conceptual Model 66

5.1 Introduction 66
5.2 Business process model 67
5.3 Case Study: airline reservation system (ARS) 67

Chapter 6 81

Multiple objective GA implementation to optimize SOA 81

6.1 Introduction 81
6.3 GA Representation 82
6.4 GA objectives 84
6.5 GA operators 85

Chapter 7 98

Experiment 98

7.1 Introduction 98
7.2 YAFA SOA Optimizer architecture 98
7.3 Running the experiment: 103
7.4 Example solutions 104
7.5 Experiment Results 106

Chapter 8 111

Threats to validity 111

8.1 Threats to construct validity 111
8.2 Threats to Internal Validity 112
8.3 Threats to External and conclusion Validity 112

Chapter 9 113

Validation 113

Chapter 10 118

Conclusions and Future Works 118

References 120

List of Figures

Figure 1: The Conceptual Reference Model [5]..8

Figure 2:web Service architecture [17]..13

Figure 3: Service composition using sequence pattern...14

Figure 4: Service composition using parallel and sequence pattern.....................15

Figure 5: Service composition using Synchronization, parallel and sequence pattern...15

Figure 6: Service composition using XOR, Synchronization, and Parallel and sequence pattern. 16

Figure 7: NSGA-II procedure [32] 24

Figure 8: Research methodology 61

Figure 9: ARS-Whole business process..69

Figure 10: Customer identification Sub process...71

Figure 11 :Booking/ reserving tickets sub process..72

Figure 12: Edit flight sub process...73

Figure 13:Edit flight sub process..74

Figure 14: Proposed Service Structure...76

Figure 15: Proposed Service Structure Example. 76

Figure 16: chromosome and supergenes architecture using UML diagram.......83

Figure 17: Service Composition architecture..84

Figure 18: cross over operator **..86**

Figure **19***: Mutation operator***..87**

*Figure 20: Decomposition of security Objectives***..91**

Figure 21: Requirement Tree for ARS **..91**

Figure 22: YAFA SOA optimizer architecture **...99**

*Figure 23: :Jmetal Class diagram [59]***..100**

*Figure 24: YAFA SOA Optimizer Class diagram***...101**

Figure 25: BP4 initial SC example. **...105**

*Figure 26: BP4- optimized Service Composition example.***..................................106**

Figure 27:SC1 **...115**

Figure 28:SC2 **...115**

Figure 29:SC3 **...115**

List of Tables

Table 1:Aggregation rule for SC security[24] 16

Table 2:Aggregation of Upper and Lower Bounds of cost [24]............................17

Table 3:DREAD model estimaion range[40]..38

Table 4:Ranking DREAD Model values..39

Table 5: Security Literature Review Studies...44

Table 6: related works in optimizing SOA as Search-based problem...................60

Table 7: service attributes and security objectives..79

Table 8:service attributes, source and value ranges [57]......................................80

Table 9: Security Services(SS) List..89

Table 10: Deriving Security objective from the security Requirements...............90

Table 11: Threat and its effects on CIA[58]..92

Table 12: GA attributes..104

Table 13:BP4 Running example attributes...104

Table 14:Initial SC with 4 objectives example....105

Table 15 BP4 initial SC with 2 objectives example. 105

Table 16: BP4 optimized SC with 4 objectives example.....................................106

Table 17: BP4 optimized SC with 2 objectives example.....................................106

Table 18: HV values for running NSGA-II with 25,000 evaluations.................107

Table 19: HV values for running NSGA-II with100,000 evaluations................107

*Table 20: objectives average values for running NSGA-II with two objectives.*108

Table 21-a: objectives average values for running NSGA-II with four objectives and 10 concrete services. 108

Table 22-b: objectives average values for running NSGA-II with four objectives100 concrete services. 108

Table 23: median of time values for running NSGA-II with25, 000 evaluations. 110

Table 24: median of time values for running NSGA-II with 100, 000 evaluations. 110

Table 25: the security requirements...114

Table 26: controls table (Added by security expert) 115

Table 27: SCs QoS reslts and validation..116

Abbreviations

ARS	**Airline Reservation System**
ATAM	**Architecture Trade-off Analysis Method**
BP	**Business Process**
BPEL	**Business Process Execution Language**
BPMN	**Business Process Management Notation**
CIA	**Confidentiality, Integrity and Availability**
DREAD	**Damage Reproducibility Exploitability Affected users Discoverability**
ENISA	**According to Information Security Agency**
FIPS	**According to Federal information Processing standard**
GA	**Genetic Algorithm**
IaaS	**Infrastructure as a service**
IBEA	**Indicator Based Evolutionary Algorithm**
NSGA-II	**Non-Dominated Sorting Genetic Algorithm**
PaaS	**Platform as a service**
QoS	**Quality of Service**
QUIRC	**Quantitative Impact and Risk Assessment Framework for Cloud Security**
SaaS	**Software as a Service**
SAML	**Security Assertion Mark-up Language**
SBSE	**Search-Based Software Engineering**
SC	**Service composition**
SLA	**Service Level agreement**
SLO	**Service Level Objective**

SO	**Security Objective**
SOA	**Service Oriented Architecture**
SOAP	**Simple Object Access Protocol,**
SOC	**Service Oriented Computing**
SPEA2	**Strength Pareto Evolutionary Algorithm 2**
UDDI	**Universal Description Discovery and Integration**
WS	**Web Service**
WSDL	**Web service definition language**
XaaS	**Everything as a Service**

Chapter 1

Introduction

This chapter introduces the thesis. It describes the problem statement, motivations, goal and objectives, contributions and organization of the thesis.

1.1 Introduction and motivation

Cloud computing provides a new paradigm which delivers IT resources and services on demand. It provides scalable and real time services and it grows rapidly. Deploying an application to the cloud is fast and cheap; it depends on pay-for- use principle. [1]

Cloud computing comes to support the new vision of software engineering, optimization of resource usage can be achieved by consolidating hardware and software infrastructure into massive datacenters, from which these resources are rented by consumers' on-demand [2].So many software deployments are moving from non-cloud environment to cloud environment.

Service Oriented architecture (SOA) is an example of SaaS cloud deployment model; in SOA services are aggregated in patterns to form the Service Composition (SC). Services are available through internet from different service providers in the service registry.

Software engineers have to choose a best group of services that fulfill system functional and non-functional requirements. This process is infeasible manually, because of the huge number of available services that provide the same functionality with different Quality of service (QoS). An effective way is needed to automat SOA optimization problem.

Search based software engineering (SBSE) is an approach that applies meta-heuristic search-based optimization techniques, Genetic algorithm (GA) is use to solve optimization problems based on heuristic search. [2]

Optimizing SOA can be considered as SBSE problem and can be benefited from the heuristic search approach.

This research is concerned in solving SOA optimization problem, enhancing SOA security in a cost effective manner.

We use empirical study concept, Airline reservation system (ARS) is used as a case study, it is modeled using BPMN and aggregation patterns [52], and the whole ARS problem is divided in to 4 sub process based on the functionality and the aggregated patterns. Each sub process contains a group of abstract services; each abstract service is mapped to a concrete service from service registry. These concrete services are aggregated to form the service composition.

This study is divided in 2 scenarios based on the optimization objectives; the first scenario is two objectives which are security risk severity and cost, and the second scenario is four objectives that are confidentiality risk severity, integrity risk severity, and availability risk severity and cost.

A GA called NSGA-II is implemented to solve the problem; NSGA-II is used to search for optimized SC, throughout the search process, security services are added to the SC randomly to enhance the generated SC security by reducing the risk severities.

There are many risk assessment approaches that are used in the literature, in this research we adapt DREAD model and modify it to assess the service security risk. We added new categorization to measure service security risk based on a group of service attributes.

The generated SCs are evaluated using aggregation rules based on the QoS values of the aggregated services in the SC.

Research Objectives and problem Statement

The purpose of this study is to find the service composition that gives maximum security, and minimum cost using evolutionary search. By selecting the best group of services that achieves the research objectives.

Research Questions

Based on the assumption that software is considered as composition of services, and the fact that these services are provided by many vendors, we assumed that the services composition is a search problem that can be benefited from the evolutionary methods.

My research questions are:

- **RQ1:** Will the GA help in finding an optimal set of security services that satisfy multiple business objectives, mainly minimizing cost and data security risk?
- **RQ2:** Can we use DREAD model to assess service risk severity?
- **RQ3:** What is the effect of using aggregation patterns for dividing the Whole Business process into sub process on security and cost of the generated service compositions?

The Research objectives are:

1. Develop a search based tool called YAFA SOA Optimizer that is a method to optimize SOA based on security and cost.
2. Implementing search based meta-heuristics software engineering methods for optimizing SOA to find the optimized Solutions. The solutions are group of Service compositions called Pareto Fronts.

3. Measuring Service Risk severity by adapting DREAD risk assessment model.
4. Enhance the Service composition security by adding security services to the Service composition.
5. Monitor the performance of the Genetic Algorithm NSGA-II in different number of optimization objectives (2 and 4 objectives).

1.1 Research Methodology

The Research is an empirical study; an experiment has been developed and built to validate the research questions, a group of steps are followed in order to develop and run the experiment:

- We collect data for the experiment and specify the research factors, we perform a literature review for related works in the research area and find the following:
    - Service oriented Architecture (SOA) patterns that are used in the research.
    - Evaluate the risk assessment models and find the DREAD model to be adapted and modified for measuring the service security.
    - List a group of well-known threats to eliminate their risk severity.
    - List a group of security solutions for the listed threats.
    - Specify the case study.

- Selecting the research case study: Airline reservation system (ARS) has been selected and used in the experiment.
- Modeling the problem using Business process modeling notation (BPMN)
- ARS is divided into 4 business process based on functionality.

- Each business process consists of a group of abstract services, each abstract service is mapped to one concrete service from service pool, and these concrete services are aggregated to form a service composition (SC).
- Implementing the GA to investigate the research objectives, encode the problem into GA, adapt GA operators, use DREAD model to estimate service risk and aggregation rules to estimate the SC risk severity.
- Develop an optimizer called YAFA SOA optimizer to run the experiment in multi-objective optimization problem and different configurations to investigate the result.
- Use different Quality indicators to analyses the results.

1.2 Contribution

The main contributions of this work are in three main fields listed below:

- **In SBSE:**
    - Using GA to optimize SOA in term of cost and security.
    - We found that applying multi-objective GA is feasible to find the optimized security and cost in SOA.
- **In GA**:
    - Extending NSGA-II in new experimental context which is optimize SOA in term of cost and security by Developing YAFA SOA optimizer.
- **In security risk assessment:**
    - Adapting Dread model to assess service security and add new categorization depending on selected service features

    - Enhance SOA security by reducing Risk Severity through adding new security services to the SC.
    - Dividing security into 3 objectives (CIA) gives the Software Engineer an accurate view for the security measurements and enhances the achievement of security objectives.
- **In SOA**
    - Provide a proposed structure for Services in the composition using a group of important attributes.

1.3 Organization of this thesis

The rest of this document is organized as follows: Chapter 02 presents the research background, cloud computing architecture, Service oriented architecture (SOA), service composition aggregation patterns, cloud computing security and Search based software engineering using genetic algorithms (GA).

Chapter 03 summarizes literature review of search based software engineering, software risk assessment and security metrics. Chapter 04 introduces the research methodology.Chapter05 describes the research conceptual model. Chapter06 presents multiple objective Genetic Algorithm implementation to optimize SOA. Chapter 07 describes the experimental configurations and results. Chapter 08 illustrates the threats to validity, Chapter 09 presents validation. And finally, Chapter presents conclusion and future work.

Chapter 2

Background

2.1 Introduction

This chapter covers the research area background, it Explains Cloud computing architecture, Service oriented architecture (SOA), Service composition aggregation patterns and aggregation rules, Cloud computing security issues, And finally, Search based software engineering using Genetic algorithm.

2.2 Cloud computing architecture

Cloud computing infrastructure provides a new challenge in the IT industry, since it leads a huge growth. Migrating service from physical environments to the cloud computing improves the availability of the service, saves energy, gives better utilization of the resources and reduce service hosting cost, enables pay per use and on demand service request.

Moving services to the cloud emerged important advantages and challenges:

Most important Advantages of cloud computing: [3]

- Reduce IT cost
- Increase flexibility and scalability of it services.
- Ability to pay per use
- Gives cloud user's ability to put their concerns in considerations.

Most Important Challenges of cloud computing:[4]

- Security
- Integration
- Performance
- Service cost.

The conceptual model of cloud architecture in Figure1 is proposed by the National Institute of Standards and Technology NIST [5].

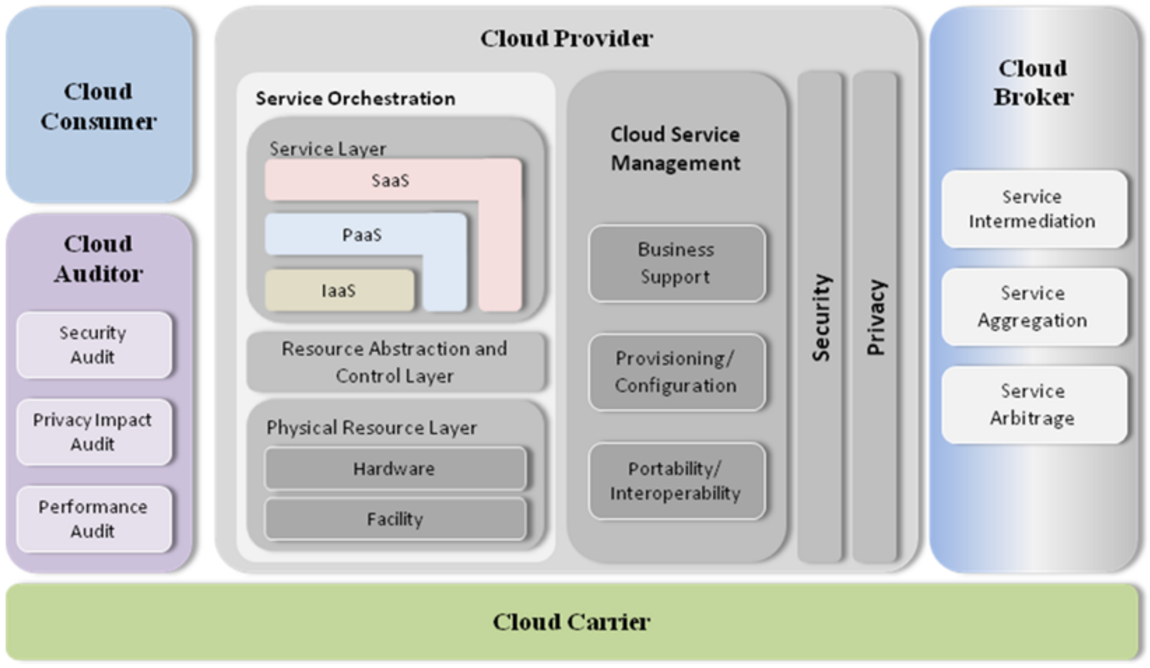


Figure 1: The Conceptual Reference Model [5]

According to NIST , cloud architecture has five actors[5]:

1. Cloud consumer: requests the service from cloud provider.
2. Cloud provider: makes the service available to use and, manage the cloud infrastructure. It provides service deployment, service orchestration, cloud service management and, security.

3. Cloud carrier: a transporter between provider and consumer that helps in service delivery.
4. Cloud auditor: audits the cloud system performance, security, and implementation.
5. Cloud broker: manages the SLA between the provider and consumer. It may combine more than one service from different providers to provide customer with requested service.

Cloud services are provided in four different deployment models [6]:

1. Private Cloud: this approach used for the private data and services. Each company has its own cloud data centers, for example VMware private cloud [7].
2. Public Cloud: a single provider makes the service available to public use through internet such as Amazon Elastic Compute Cloud (EC2),[8] Windows Azure Services Platform[9].
3. Hybrid Cloud: this model is a combination of other models, the same company can have both private and public cloud according to the business needs; the private services can be deployed to private cloud and the public services to the public cloud. For example, a business company can deploy an on-premises private cloud to host sensitive or critical data and operations and use a third-party public cloud provider, such as Google Compute Engine[10] for public data and normal operations,
4. Community Cloud: cloud service is provided by more than one cloud provider, they may be external or internal providers. One example of community cloud is mentioned by Kleyman in [11]Several organizations need to access same application from same provider, the application will be hosted on provider environment and each customer request an access to the same application and make a community cloud.

Cloud computing provide everything as a service XaaS, and so three delivery models are provided [1]:

SaaS is a model provides an application to the customer as a service to use on demand and it is accessible anytime from anywhere. It is in the top layer of cloud conceptual model that is shown in Figure 1. It provides a web based applications that are provided as a service to the end users. So It is a service oriented architecture , which is a composition of services [12].

IaaS is a model that deliver infrastructure (visualization environment) as a service, instead of owning a local data centres, it is a hardware virtualization used as a pool of resources consists of a collection of virtual machines (VM), and it gives the ability to partition the hardware resources such as CPU, memory and storage. It is a composition of cloud hosting and resources [12].

PaaS is a model that delivers a platform as a service to use it in application deployment in a cheap way instead of buying and managing the underlying hardware and software layers. It provide application development platform includes web based interfaces, command line tools, and frameworks for programming. PaaS includes infrastructure it is considered a part of the provided services to the customers. [12]

Service Orchestration gives three layers of components that are provided by cloud provider and used by cloud consumer:

1. Service layer: provide the access interface for SaaS, PaaS and, IaaS.

2. Resource abstraction and control layer: system component that cloud providers use. The resource abstraction is used to manage the hypervisor, virtual machine ensure the security of the hardware layer and other software components it enable dynamic allocation of resources from resource pool.
3. Physical resource layer: responsible for the physical resources such as hard disks, CPU, networks, storage, power and all other physical component.

This research studies Service oriented architecture (SOA) which is an example of SaaS deployment model and it is discussed in the next section.

2.3 Service oriented architecture

Service oriented computing(SOC) is a new computing paradigm that utilizes services as fundamental elements for developing applications [13]

Service oriented architecture (SOA), is a computing approach that consists of web services. Services in SOA are aggregated in certain patterns and using certain communications protocols to form the service composition (SC).[14]

The service is the atomic unit of the SOA[15]; service is defined by Thomas, E in [16] as "A service is a software system designed to support interoperable machine-to-machine interaction over a network".

In addition, W3C in [17] defined the service similar definition as "A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-process able format (specifically WSDL)

Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards."

Service is defined in [13] as: Software constructs that support and deliver business functionalities over networks

Another service definition is Software constructs that support and deliver business functionalities over networks [13]

Services are defined using a description language (WSDL) and have invoking interface that is called to perform business processes. [18]

Furthermore, Beal in[19]define the web service as follows "The term Web service describes a standardized way of integrating Web-based applications using

the XML, SOAP, WSDL and UDDIopen standards over an Internet protocol backbone. XML is used to tag the data, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI is used for listing what services are available. "

SOA may depend on Business Process Execution Language (BPEL) which is used to describe and execute the business processes and SOAP is used to exchange messages.

WSDL is used to define the web service interfaces Business process is a set of workflows that are executed in an ordered manner. The orchestrations layer contains the largest change provided by SOA and it describes the business process interactions in SOA [20].

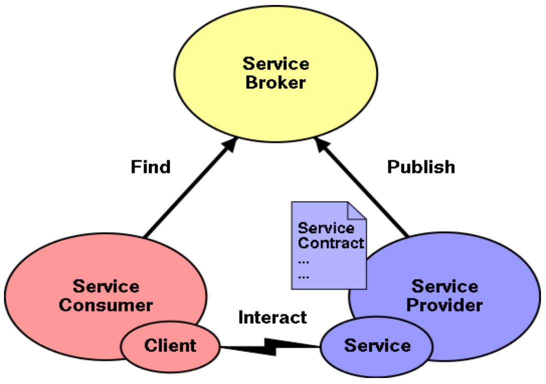


Figure 0002:web Service architecture [17]

From the service architecture that is shown in figure 2 we can export the following roles and operations:

Web service Roles: [17]

- Service provider **publishes** the service to a service registry (broker).
- Service requestor **finds** services from service registry.
- After finding the service information the service requester then **bind** to the service and use it.

Web services operations:[17]

- Publishing : making a service available through internet
- Finding – locating web services
- Binding – using web services

Web services features:[17]

1. Loosely-coupled: it can run independently from other services in different platforms and run-time environments.
2. Encapsulated: The only visible part of a Web service is the public interface, e.g., WSDL and SOAP.

3. Standard Protocols and Data Formats: The interfaces are based on a set of standards, e.g., XML, WSDL, SOAP, UDDI and etc.
4. Invoked over Intranet or Internet: Web services can be executed within or outside the firewall.
5. Components: The composition of Web services can enable business-to-business transactions.

Services are interacted to form the service composition in a certain execution pattern, the service composition patterns is discussed in the next section.

2.4 Service composition aggregation patterns

Service is the atomic unit of the service composition[19]; services are connected using the transition or call. There are 5 type of service composition basic control Patterns[21],[22], patterns are illustrated bellow for care rental system example[i1]

Pattern 1: sequence or serial pattern, the service is called after the execution of another service is completed, the service execution is depending on the execution of the previous service as it is shown in figure3.

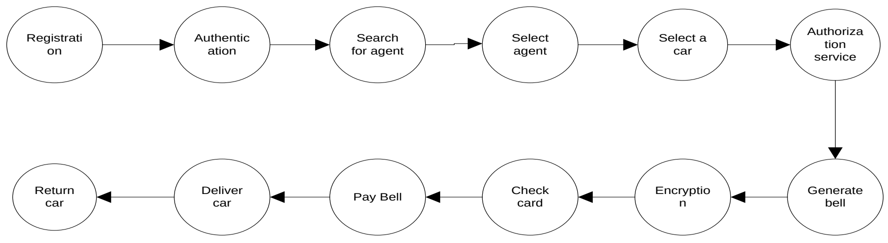


Figure 3: Service composition using sequence pattern

[1] Car rental system case study is used to illustrate SOA patterns only, and it is not used in the experiment.

Pattern2: Parallel, and split or fork; the service is calling multiple services execution and the called services are executing simultaneously in any order. Figure4

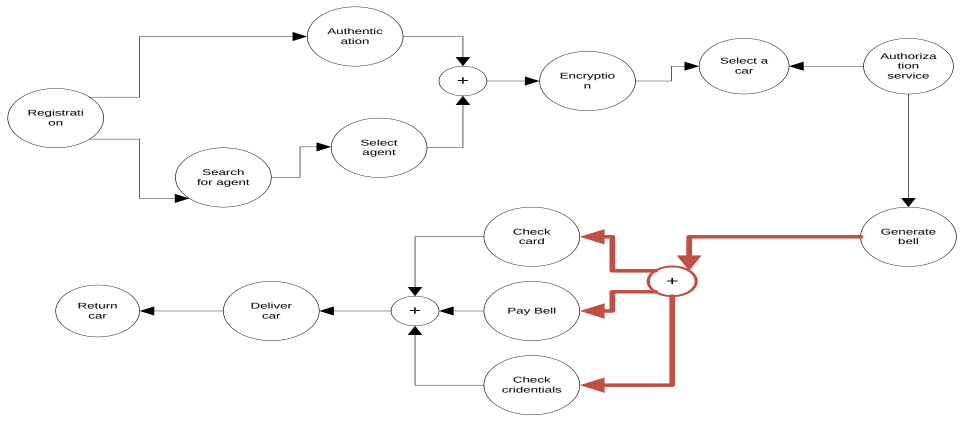


Figure 4: Service composition using parallel and sequence pattern.

Pattern 3: Synchronization pattern or and join pattern; service multiple service call one service, each caller is executed once, and this pattern can be used with other patterns such as parallel execution patterns it is shown in figure 5.

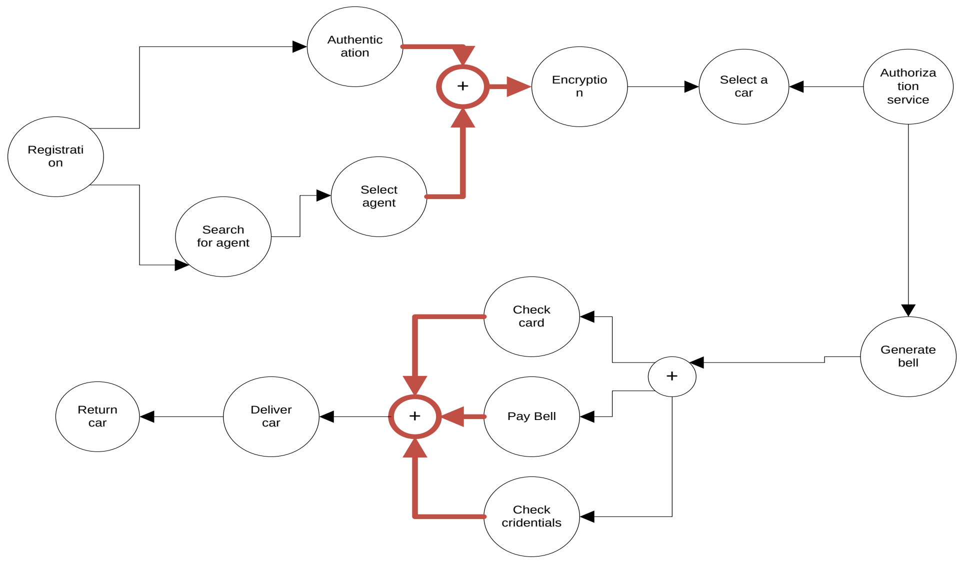


Figure 5: Service composition using Synchronization, parallel and sequence pattern.

Pattern 4: Exclusive choice or XOR split the point that the service execution depends on a decision by execution several services from a choice. The service is called by more than one service depending on a condition as it is shown in figure 6.

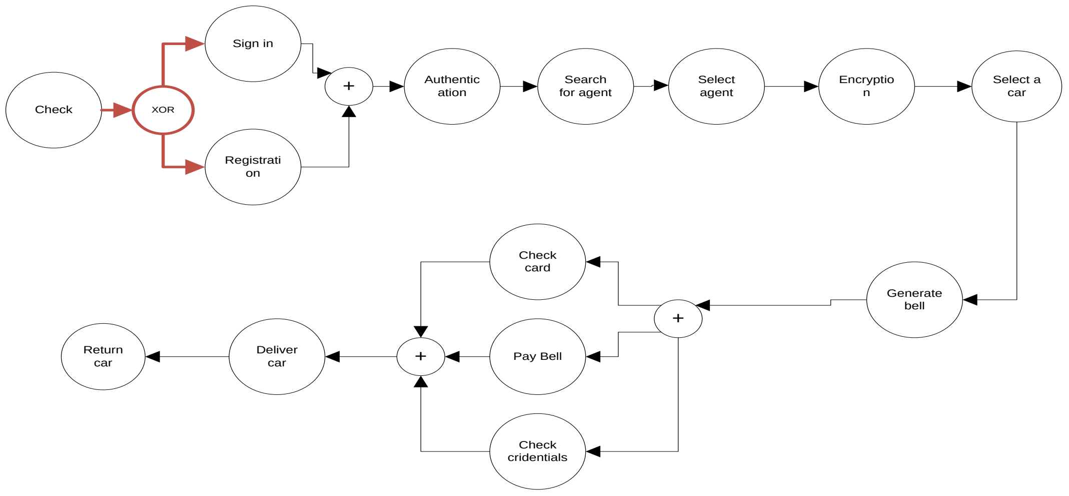


Figure 6: Service composition using XOR, Synchronization, and Parallel and sequence pattern.

Services aggregation Rules

Services are aggregated in a service composition (SC) using service composition patterns, the QoS for the SC is called global QoS and it is calculated using aggregation rules. [23] The aggregation rules that are used to find the global security valued of SC is shown in table 1 below:[24]

Composition pattern	**Security**
Sequential execution patterns	
Sequence	$\mathbf{Xa} = \mathbf{min}\{x, \ldots, xn\}$
Loop	$\mathbf{Xa} = \mathbf{x}$
Parallel composition patterns	
XOR-XOR	$\mathbf{Xa} = \mathbf{min}\{x, \ldots, xn\}$
AND- AND	$\mathbf{Xa} = \mathbf{min}\{x, \ldots, xn\}$
AND-DISC	$\mathbf{Xa} = \mathbf{min}\{x, \ldots, xn\}$
OR-OR	$\mathbf{Xa} = \mathbf{min}(\mathcal{P}(\mathbb{T}))$
OR-DISC	$\mathbf{Xa} = \mathbf{min}(\mathcal{P}(\mathbb{T}))$

Table 1:Aggregation rule for SC security[24]

The cost of the services execution is calculated by aggregation rule as shown in table 2 below:

Composition pattern	**Maximum cost**	**Minimum cost**
Sequential execution patterns		
Sequence	$\mathbf{Xa} = \sum_{i=1}^{n} \mathbf{Xi}$	$\mathbf{Xa} = \sum_{i=1}^{n} \mathbf{Xi}$
Loop	$\mathbf{Xa} = \mathbf{kx}$	$\mathbf{Xa} = \mathbf{kx}$
Parallel composition patterns		
XOR-XOR	$\mathbf{Xa} = \mathbf{max}\{\mathrm{x}, \dots, \mathrm{xn}\}$	$\mathbf{Xa} = \boldsymbol{min}\{\mathrm{x}, \dots, \mathrm{xn}\}$
AND- AND	$\mathbf{Xa} = \sum_{i=1}^{n} \mathbf{Xi}$	$\mathbf{Xa} = \sum_{i=1}^{n} \mathbf{Xi}$
AND-DISC	$\mathbf{Xa} = \sum_{i=1}^{n} \mathbf{Xi}$	$\mathbf{Xa} = \sum_{i=1}^{n} \mathbf{Xi}$
OR-OR	$\mathbf{Xa} = \mathbf{max}\{\mathbf{z}: \mathbf{z} = \sum_{y \in S}^{n} \mathbf{y}, \forall \mathbf{S} \in \boldsymbol{\mathcal{P}}(\mathbb{T})$	$\mathbf{Xa} = \mathbf{min}\{\mathbf{z}: \mathbf{z} = \sum_{y \in S}^{n} \mathbf{y}, \forall \mathbf{S} \in \boldsymbol{\mathcal{P}}(\mathbb{T})$
OR-DISC	$\mathbf{Xa} = \mathbf{max}\{\mathbf{z}: \mathbf{z} = \sum_{y \in S}^{n} \mathbf{y}, \forall \mathbf{S} \in \boldsymbol{\mathcal{P}}(\mathbb{T})$	$\mathbf{Xa} = \mathbf{min}\{\mathbf{z}: \mathbf{z} = \sum_{y \in S}^{n} \mathbf{y}, \forall \mathbf{S} \in \boldsymbol{\mathcal{P}}(\mathbb{T})$

Table 2:Aggregation of Upper and Lower Bounds of cost [24]

Where:

n: number of services.

k: number of repetition in the loop

xn: values of a service

N: set that contains all xn.

For the OR split the power set $\boldsymbol{\mathcal{P}}(\mathbb{T})$

F ($\boldsymbol{\mathcal{P}}(\mathbb{T})$):= F ({x: x=F(S, for all S € $\boldsymbol{\mathcal{P}}(\mathbb{T})$)})

Zeng et al. handled the loops by estimate the number of iterations and then multiply it with the attribute value. [25]

2.5 Cloud computing security concerns

Security is one of the major factors that affect the growing of cloud computing, so migrating software to the cloud may require changes on software architecture; which will affect other features and capabilities[1].

According to Lowis [26], many factors must be taken in consideration when services are provided in cloud architecture, such as high availability, safety, integrity, maintainability, confidentiality, and reliability.

The wide spread of cloud services adds new vulnerabilities because it adds new entries.[26]

The general security concerns in service models as described in [12] and are listed below:

- SaaS: deploying a software for the customer by the service provider and make it available on demand, in an efficient way and by reducing the cost. Customers in SaaS can`t see how their data is stored and secured this gives uncomfortably to moving an application to the cloud in SaaS model.

  In SaaS Provider has to separate each costumer data from others, and make sure that user data is available when it is requested. By using SaaS there is a great level of discomfort regarding to the lack of control.

  The main concerns related to the SaaS are data security, network security, data locality, data integrity, data segregation, data access, authentication and authorization, data confidentiality, web application security, data breaches, visualization vulnerability, availability, backup and, identity management and sign-on process.

- PaaS: is a model that delivers a platform as a service to use it in application deployment in a cheap way instead of buying and managing the underlying hardware and software layers.

  The developers have to control their application life cycle. Provider gives access to the developers to deploy applications, but provider still have to make assurances for the host, network, and separation of the user data from others.

  PaaS is more extendable than SaaS and this extends security capabilities and additional security can be applied. A security metrics should be applied to evaluate the security levels such as quality of application coding, how attackers can hack text code through attacking the infrastructure and running a black box testing.

- IaaS: it provides virtual resources to the developers to upload their applications instead of having their own data centers.

  IaaS provides limited security for firewall and load balancing; other security issues need higher level of security provided at the host. It provides a complete control for developer to control their software without any hidden action from the provider side. The developers have a full control to protect their own data and applications if there are no security holes at visualization level.

  The problem is still at the location of the stored data, as it stored in the provider hardware and the customer actually has no idea about who have access to this storage. Each cloud deployment candidate (Microsoft Azure, Amazon EC2, etc..) offer its own rules for security controls.

The historical work in SaaS security will be presented in chapter 03and the security measurement of SaaS cloud model as it is the target of this research.

2.6 Search based software engineering using Genetic algorithm

Search Based Software Engineering (SBSE) considers software engineering problem as search problem, and it looks for an optimized solutions that satisfy single and multiple objectives; such as performance, security, cost, and service level agreement violations. According to Harman[27], deploying a software in cloud computing environment is considers SBSE problem and can use heuristic search to solve it. They identify four areas for future research:

1- Provider resource efficiency (cloud stack configuration): the challenge is to reduce resources consumption for both client and provider by subject to a certain configuration to optimize performance characteristics such as network bandwidth and energy efficiency.

2- Client resource efficiency (image specialization): instead of using only a fraction of software within an image using specialization will help to reduce image side, execution time, and migration time, which will benefit both client and provider by cutting down the cost.

3- Virtual machine assignment and consolidation by providers: the goal is to reduce the power consumption by efficiency use of VMs allocated to an application according to demand. The SBSE methods provide sophistication and efficiency managements.

4- Scale management for cloud clients: cloud application must minimize the resources usage while giving a certain output by well-constructed rules for usage profile.

5- Spot-price management: cloud provider want to maximize the usage of the running server to maximize his profit, they divide the instances in to spot that are provided to the customers who pay according his usage of these spots. This will save resources from loss, and customers pay for usage only. Researchers are carried in the efficiency of spot pricing.

From Harman[27] five research area my work will be in the fourth and fifth areas

The major step in search based software engineering is to formulate the software engineering problem as a search space problem and then evolve a set of candidate solutions known as Pareto Front solution. [28]

SBSE problem can be solved by group of steps, first of them convert the problem in “Search problem” by present the solution candidate, present how these candidate can be obtained and how to measure the quality of solution. [2]

In this research we the GA is used to optimize SOA as a SBSE problem, an algorithm called NSGA-II is implemented in an experimental environment, the next section gives a background about GA.

Genetic algorithms GA:

Genetic was invented on 1960 by John Holland, the purpose of this invention was to import the evolution of nature in to computer science and use it in solving computation problem [29].

Genetic algorithms (GA) depends basically in biological terminology; any living body consists of cells and each cell contains a group of chromosomes that are a string of DNA, each chromosome contains a group of genes, these genes specifies the living body specific features, and each gene has a specific allocation in the chromosome structure.

Reproducing process “crossover” occurs when tow parents chromosomes exchange their genes, and produce new chromosomes called offspring’s.[29]

The reproduced offspring’s maybe exposed to a mutation which change a single bit of the DNA structure, and changes one of the individual features.[29]

The fitness of the new generated individuals reflects the ability of this chromosome to survive in next generations. The term population is used to call a set of generated chromosomes in a period of time.[29]

The GA uses the evolution ideas in computer science to solve some exponentially growth problems, hence it helps to select the optimized solution from a very large solution set.

Applying GA requires: encode the solution into chromosomes, create the initial population, run cross over operator, mutation operator, and evaluate the generated chromosomes and finally select the next generation chromosomes.

The basic algorithm for GA is shown in the box below[30]:

```
Initialize (population)
 Evaluate (population)
While (stopping condition not satisfied)
 do {
        Selection (population)
        Crossover (population)
        Mutate (population)
        Evaluate (population)
       }
```

Encoding:

In GA the chromosome represents the solution, encoding the solution into chromosome means format it in a certain way to represent all solution feature in a computational valid way, it cloud be formatted as a series of Boolean values and each bit represent a feature of the representative solution, or it could be formatted as a string of natural number, which is considered the pure genetic algorithm. [29] regarding to Räihä in [29] there are many several ways to represent the chromosome and the GA developer can decide the suitable way to encode it ether numerical or non-numerical way.

Crossover

Crossover operator is also called recombination, cross over merges and exchanges the genes of the selected chromosomes (parents).

Cross over starts in a randomly selected point within a pre defines cross over probability, cross over generates new generation individuals with genes from both parents, the new created chromosomes have different features from the parents and it is also called offspring[31]**.**

Mutation

Mutation operator is used to create a new unexpected change on the created offspring, by changing one or more genes based on the mutation probability in a random selected point, mutation operator is run to avoid repetition of the generated candidates and to add diversity in the generated individuals, and mutation produces new good traits. [29]

Selection

Choosing candidates to the next generation, selection process depends on the fitness value, a fitness function is run on each created offspring for each objective and the chromosomes are ranked regarding to the fitness value. Fitness value is the capability of the candidate to accomplish the search objectives. Selection operator selects the individuals for the next generation.

In the next section we describe the NSGA-II algorithm and its importance.

2.6 Multi-objective Genetic algorithm “NSGA-II”

Non-dominated Sorting Genetic Algorithm II (NSGA-II) is implemented in this research to optimize multiple objective problem.

NSGA-II is selected because of many reasons; it is an elitist algorithm, it provides lower computational complexity, and it replaces the sharing function with a crowd-comparison approach[32], on the other hand it has been used widely in literature for optimizing multiple objectives problem.

Deb, et al in their work on [32] presents the procedures that is followed by NSGA-II, Figure 7 illustrate it, the procedures start with creation a random population P_0 then population is sorted based on the non-dominated approach, a rank value is given to each solution, then the binary tournament selection is performed with replacement on P_0, the cross over and mutation operator are run and are used to create the new offspring Q of size N.

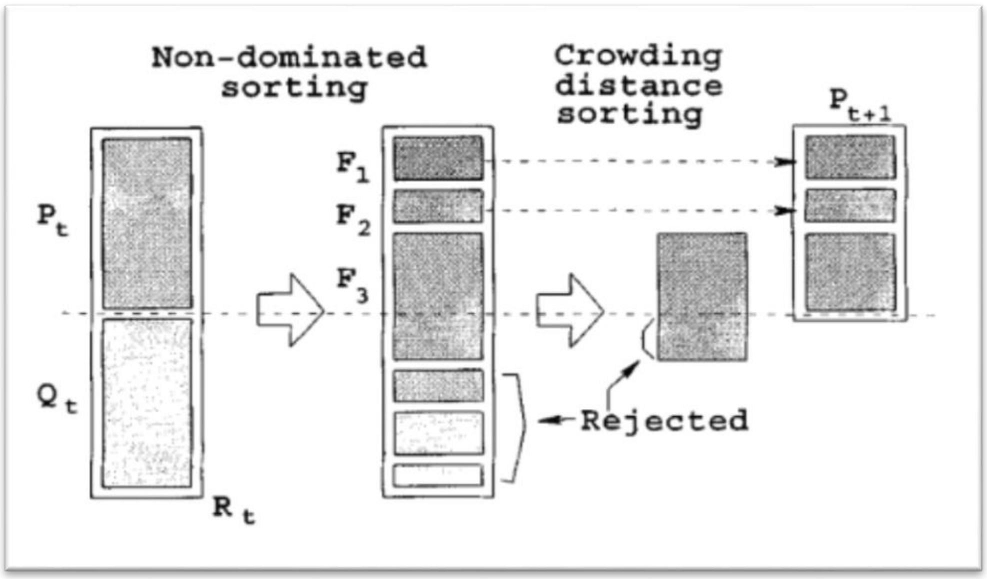


Figure 7: NSGA-II procedure [32]

NSGA-II is elitist approach, it is compare the new offspring with the best non-dominated chromosomes; in each generation t, it combines the new offspring Q_t with the non-dominated population P_t, (R_t= P_tU Q_t). Where the size of R_t equals 2 N, and the population Rt is sorted regarding to non-dominated approach. The best solutions are classified in to group F_1 and it is moved to the new population P_{t+1}.

The size of F_1 is smaller than N then the remaining solutions of the population P_{t+1} are chosen from the F2 and F3 and so on. And the other solutions are rejected.

If the group F_1 is larger than N then the N number of chromosome are selected based on crowd distance rank in descending order.

The new population P_{t+1} is used to follow selection cross over and mutation operators to find the new generation Q_{t+1}.

The selection process in NSGA-II depends on the crowded distance operator <n and it depends on two factors ranking and crowded distance, it is calculated through generation the new off spring.

The overall complexity of the NSGA-II algorithm is O (MN^2) where is M the population size and N the number of objectives. Which is better than non-elitist algorithms such as NSGA that provide a high complexity equals O (MN^3).

NSGA-II is a computationally elitist and fast algorithm, it gives better spread of solutions and more convergence of the non-dominated fronts.

Chapter 3

Literature Review

This chapter reviews and summarizes the previous works related to our research area.

The literature aim to introduce the research field, give a clear vision of the other relevant work contributions, help to justify the research contributions, and connect the research with other related works in the field.

The rest of this chapter include Search based software engineering (SBSE) for multiple objective optimization problem, SaaS cloud model security concerns, software security risk assessments, Service oriented architecture quality of services (QoS), SOA as a multiple objective problem.

3.1 Search Based optimization

SBSE based is based on a heuristic search to optimize a set of conflicted objectives. One of heuristic search techniques is known as evolutionary approach, in which the solutions are iteratively evolved.[33]

Genetic Algorithms is a kind of evolutionary approaches and it was used in the literature to solve multiple objective optimization problems[27]. It consists of a population space that contains individuals, each individual known as chromosome that conceptually represents the problem in context, the chromosome consists of genes and each gene represents an attribute or a feature of the candidate solution[34]. In general GA follows four main steps which were mentioned by Konak in[28]:

- Selecting the pair of chromosome form the population, i.e. search space, from which the next generations will be produced.

- Crossover operation is performed by swapping the parent genes to create the new offspring,
- Applying a mutation operator on the chromosomes edge in order to avoid local optimized solution and to omit unwanted repeated generations.
- Evaluating the created individuals by applying a fitness function. Fitness function is important to direct the random search for the best solution.

Many studies have addressed cloud optimization at software architecture level, using genetic algorithm. One of the common used GA is called NSGA-II.

In [34] Frey has used Search-based Genetic Optimization for Deployment and Reconfiguration of Software in the Cloud for Infrastructure as a service (IaaS) model. Fery work aims to migrate none cloud software architecture to cloud environment. They defined three objectives for their study cost, SLA and response time.

They develop a genetic algorithm called CDOXplores based on the NSGA-II. To find suited deployment model and reconfiguration rules. They have defined four inputs to the CDOXplore:

1. The Architecture model: that is generated by their tool (CloudMIG Xpress) which satisfied ISO/IEC Standard Knowledge Discovery Meta Model (KDM) that was developed by the Object Management Group (OMG).

2. A status-quo deployment model, which is used to measure and specify the performance capabilities of the computing nodes, it measured by the benchmark which is the Mega Integer plus Instructions per Second (MIPIPS).

3. Work load profile: read monitoring logs file and find the service code response time.
4. Cloud profile: it describes VM instance type (performance capabilities and benchmarks).

The architecture is represented based on the Cloud Deployment Option (CDO) model. The CDO models captures the following information about the software architecture, cloud environment, VM instance type, initial start configuration, service composition and shrink/grow rules for adding resources to the VM.

Each CDO is represented by a chromosome; which consists of   genes that represent the CDO model elements. The gene sequence is called Genotype, it specifies a certain CDO.

GA introduces four steps for looking for solution in[34] they are as follows:

- The cross over operation is used for producing an offspring, in this case a new CDO model, from two parents, two CDO models, by mixing their information.
- They defined five mutation operators that are applied to the chromosome boundaries for the diversity of individual and to avoid convergence to local optimum solution. It makes a random change on the genes pool of the parents during the evaluation process.
- They used the fitness function to direct CDOs generation, for the performance they used the response times and the MIPIPS value as a measurement metrics. They depend on a simulator called CDOSim to which the cloud profile is loaded and run. The number of services violate the SLA is used as a metrics and the threshold was set to 2s. The cost were measured in monetary based on the prices for VMs and their features settings, such as CPU times, Memory size, data storage, etc.

They run the experiment using an open source ERP system. Their results show that GA helps in finding a design solution.

GA does not provide single solution, but determines Pareto optimal solutions (set of non-dominated solution). These solutions satisfy the objectives in an acceptable level, but any of these solutions is non-dominated by others. Pareto solution can`t be improved with respect to one objective without affecting the other objectives.

If a solution is better in one objective than the other solution there will be a sacrifice in the other objectives, it is better in one objective but not all, this is why it is called a non-dominated solutions.

One of the drawbacks of Fery et al, (2013) work is that they depend on the simulation to compute the fitness function which limited the search space.

3.2 Security for SaaS

The security issues related to the service interactions are studied in this research, security interactions according to Erl, T [35] are listed below:

1- Data confidentiality:

When the messages are moved through different intermediaries it may expose to different threats that is not supported by point to point security solutions. Unencrypted messages may be accessed by another agent or services. And when the sensitive data leaves the secure area such as secure memory it may be vulnerable in a message queue, data base or a file, it may be revealed by an eavesdropper application in a network.

The solution is using XML encryption technology for web application; the data which is called plaintext are passed in encryption mode called cipher text to the recipient who has a decryption key to decrypt it again to the plain text.

There are two types of xml encryption: asymmetric and symmetric encryption, the difference between them in the encryption key, symmetric use the same key for encryption and decryption but asymmetric uses public private key in which the key used for encryption is different from that used in decryption.

Impact of the encryption: it will affect the resource usage and add overhead to performance. Also add a key management infrastructure to the organization.

2- Original data authentication:

Attacker may edit the message content while it is transformed between different layers and this may edit the service behavior maliciously.

The solution has used XML signature technology that provides a digital signature that enables receiver to recognize that it is a modified massage and is not the original one from the expected sender.

The digital signature is also Asymmetric or symmetric signature. Symmetric signature use Message Authentication code MAC which is created using a check sum and a shared secret. Asymmetric signature is processed in another way. It uses a public private key for authentication. Private Key used to create the signature and public for sharing it.

This pattern also shared the same impact as Data confidentiality it has performance impact, governance related consequence.

3- Direct Authentication

Some services must be only accessed by certain type of customers. How can the service verify the costumer credential to access a certain resources?

The solution adding additional controls such as authentication and auditing, a unique identity and a shared secrete are required to check authentication. If the authentication success the request is sent to the customer otherwise a fault message is sent.

There are many ways to apply authentication: username and password, key Hash massage in which a check sum is used by the hash function to generate the secret key after that the consumer is directed to the requested services with a suitable permission.

This can affect in centralized and decentralized data stores, for centralized data stores it is gives latency and behavioral fluctuations. For decentralized data stores each service can has its own data store with keeping all data stores synchronized.

The impact of this way it does not support single sign on so, customer is forced to keep inserting the credentials. Attackers may try to detect the transition of the shared secret.

4- Broker Authentication

When the service first designed, it needs a direct authentication with users this may need a communication between users and service to build a trust, or if the user request more than one service and these services don't trust each other, this required users to save a credential for each service.

The solution is authentication broker which enables consumer and services to trust each other without a direct trust relation between them by trusting the authentication broker. This is a centralized trust authentication that reduces the burden governance by identity management.

The security technologies that are associated with the service broker are:

1. X.509 PKI which help to secure the authority for X.509 certificates.
2. Kerberos protocol which uses authentication for Kerberos tickets.
3. WS- trust.

If a user request a certain service he will submit his credential to the broker and it will give authentication for this service, a trust relation can also be built between different brokers so broker's authentications can go cross different organizations.

Broker impact is summarized in providing a single point of failure, it may cause a delay for Validation and some brokers takes a long time before it return result. Also if the attacker gains credentials he may gain access to authorized services.

Assessing security attribute has to discover all hidden properties of security that may cause risk, not only measure if the security meets the business requirements. [36]

3.3 Security Analysis and risk assessment: Models and Approaches

Many researches were performed in IT computing security analysis, which included studying threats, vulnerabilities, attacks and risk measurements, this section presents some previous works from literature in security analysis approaches and risk assessment models.

According to Federal information Processing standard (FIPS) the confidentially, integrity, availability, authenticity and accountability are the information security key principles[37]. The security level must be appropriate with the risk range for a certain data or service.

FIPS defines categories that it considers the security objectives to accomplish organization day to day work without enclosure of the organization data to risks. Threat represents the violation of security that is potential and has negative effect and it impose the security to an attack. By modeling a threat and analyzing the risk in cloud the attack surface could be specified.

In [38] Wolter provides an extension modeling for security configurations goals confidentiality, integrity, and authorization in a business process. It discusses the difficulties to manage security mechanisms and integrate it into process system, they analyze security goals and concerns then generate a general security policy, they find the dependencies of the security policy and constraint models for business. Then they specify the security configurations for business processes.
Microsoft STRIDE is a threat modeling approach that was used by [38]to give a weight to every Security objective (SO) used.

3.3.1 Evaluate the security of software architecture

There are many approaches to evaluate security of the software architecture; this section illustrates the most important risk assessment approaches that were introduced in literature.
One of these approaches is Scenario based that was presented in [36]
The profile that is used to evaluate the architecture is generated by assigning a weight to the assessed attributes. It uses the architectural review for measuring the quality of design and meets the requirements. It depends on the Architecture Trade-off Analysis Method (ATAM) which gives a boarder framework and the risk analysis model, and the security patterns that help to increase the quality of software security components.
They use risk analysis model that is called OWASP`S Risk Rating Methodology.

The evaluation process consists of:

1- Define the evaluation goal: the goal could be qualitative, quantitative or a trade-off assessment.

2- Generate security scenarios which consist of security requirement, threat, precondition system behavior under a certain scenario, and the patterns that are used to safeguard behavior, and it is not easy to select the needed patterns. Some patterns can be assigned to more than one scenario and some scenarios need more than one pattern to protect.

3- Create a security profile that depends selecting the criteria and scenario prioritization:

Since threats landscape is changes daily it’s difficult to use complete scenario of security. Using representative method is better depending on the risk;

Risk = likelihood*impact (OWASP Risk Rating Method).

Where:

- Likelihood: depends on vulnerabilities.
- Impact: depends on technical factor.
- Both factors are represented using simple numerical values for simplicity:
- 0 to < 3 => Low,
- 3 to < 6 => Medium,
- 6 to 9 => High.

The scenarios are extracted from a scenario profile using a table called Full Scenario Table (FST).Scenarios often generated with patterns, in case that anew threat without known patterns, and if the threat is fixed by a known best practice.

4- Describe the software architecture:

The UML is used to represent the patterns that are used to generate the scenario and security profile so it is the best way to represent the software architecture.

5- Evaluate the Scenario:

This work presents three ways to evaluate the security scenarios:

a. Pattern-based Evaluation:

To evaluate scenarios I need to study all scenarios from FST table profile and the historical evaluation of the structure using security pattern of each profile.

The patterns are derived from the architecture and the evaluation is performed for each profile in the scenario.

b. Risk-based Evaluation:

This way estimates the impact and the likelihood of each scenario to estimate the risk. Using the history of the organization platform security is helpful in estimating the risk.

c. Design decision based evaluation:

Every design decision is discussed and reversed to analyze its impact on the security risk.

6- Interpreting the results:

The results are discussed to detect the security level and some transformation may occur to fix the lack in security level.

Another approach was introduced by Saripalli in [4]; a Quantitative Impact and Risk Assessment Framework for Cloud Security (QUIRC).

QUIRC presented the advantage of the quantitative framework for assess the security impact and risks for deployment an application in to cloud.

QUIRC contained two approaches:

1- The cloud security defense must be strong, scalable and effective from cost point of view. According to Information Security Agency (ENISA) [4] if the probability of the event is high and has a high impact the risk will be high. And it is a semi-quantitative it is a range.
2- FIPS model: the Potential impact is consider low if the loss of one of the Security Objective (SO) has low effect on the organization assets, operations or resources. In this case the effect could be in the minor operations of the organization or has a low effect in assets or cost.

   The Potential impact is consider moderate if the loss of one of the SO my has a Serious affect such as reduces the effectiveness of the organization main operations or causes a significant damage on assets.

Quantitative impact and risk QUIRC methodology for the cloud security is used to define security risks and their negative impact by defining six SO, the SME is defined using Delphi method, to find the probability of the threat.

The importance of QUIRC it enables companies to compare between cloud service vendors security.

The limitation of this approach is in finding Subject Matter Experts (SME) for the probability of the threat. And SME depends on an expert prediction for the threat occurrence

Cloud platform must be designed in a way to assure non-duplication, pedigree tracking in order to helps in detecting and preventing security attacks. [4]

In [26] Lowis et al, presented ATLIST analysis method for vulnerabilities in the SOA Based business process scenario, which combined the notation of the attack tree and the Fault/attack trees[7], [8] with Failure Mode and Effects Analysis (FMEA) approach. Furthermore, Zevin in [6] assumed that vulnerabilities have been sorted in order of their estimated risk level. By looking at the triggering properties each path to the root node represented a vulnerability type. ATLIST trees in [26] helped in the derivation of the vulnerability patterns by narrowing down the search space.

DREAD model is a risk assessment model that was introduced by Microsoft in 2008 [39]. The model consists of the following steps, as described in [39-41]:

- Identifying the valuable system assets
- Identifying system boundaries, including subsystems and data flow,
- Decomposing application (network, infrastructure design, data, etc.),
- Defining and documented threats,
- Rate the threats.

DREAD model depend on the following factors to evaluate risk the threat risk [39]:

- Potential Damage (D) shows the damage degree when the threat occurs.
- Reproducibility (R) is the probability to reproduce the risk.
- Exploitability (E) shows how it easy for the risk to happen.
- Affected (A) indicates the number of affected users.
- Discoverability (D) is the system availability to detect the vulnerability

There are alternatives methods to evaluate the DREAD values, which are qualitative values.

Table 3 below shows a guideline to assign the values to of DREAD factors proposed by the Open Web Application Security Project (OWASP)1[40]

	0	5	10
D	Leaking Trivial Info	Sensitive,	Admin level
R	Very difficult to reproduce	three steps	web Browse er
E	very skilled	can be automated	novice programmer
A	few users	some users	all users
D	Unlikely	accessible only to few users	published

Table 3:DREAD model estimaion range[40]

Table 4 shows another guideline that is proposed by Microsoft[39].

	Rating	**High (3)**	**Medium (2)**	**Low (1)**
D	Damage potential	The attacker can subvert the security system; get full trust authorization; run as administrator; upload content.	Leaking sensitive information	Leaking trivial information
R	Reproducibility	The attack can be reproduced every time and does not require a timing window.	The attack can be reproduced, but only with a timing window and a particular race situation.	The attack is very difficult to reproduce, even with knowledge of the security hole.
E	Exploitability	A novice programmer could make the attack in a short time.	A skilled programmer could make the attack, then repeat the steps.	The attack requires an extremely skilled person and in-depth knowledge every time to exploit.
A	Affected users	All users, default configuration, key customers	Some users, non-default configuration	Very small percentage of users, obscure feature; affects anonymous users

D	Discoverability	Published information explains the attack. The vulnerability is found in the most commonly used feature and is very noticeable.	The vulnerability is in a seldom-used part of the product, and only a few users should come across it. It would take some thinking to see malicious use.	The bug is obscure, and it is unlikely that users will work out damage potential.

Table 4:Ranking DREAD Model values

Risk severity in DREAD Model is evaluated for every identified threat that might threaten the system. The security threats are identified by a domain experts based on the system security requirements, and historical data about security vulnerabilities and breaches, and the security objectives of the organization[42].The experts also use the guidelines descried in Table 3 and 4 above[39]. The formula for calculating the overall risk is given in equation 3.1

$$\boldsymbol{Rating = (D + R + E + A + D)/5} \quad \textbf{…………………….. } \textbf{\textit{Equation 3.1}}$$ [2]

Yautsiukhin in his work on [42] provide a new quantitative approach to assess the patterns used in the software architecture, they predicted if the software architecture is protected against a certain threats, they created a link between security between security objectives and domain requirement in order to specify the needed security pattern by establishing the security requirement tree.

They used a combination of Microsoft STRIDE threat list and Microsoft DREAD method to assess threat severity.

2 The nominator: D+R+E+A+D the algebraic sum of DREAD measures

And the denominator: is the count of Dread measures

A semi qualitative method was used to estimate effect of each security pattern on threats at end they calculated the total effect of the patterns through tree branches by using a weighted aggregation metric.

Another important Risk Assessment and Optimization Model (RAOM) was presented in [41]. Which assessed the software risk and selected the required countermeasures in cost-effective manner, it provided effective risk assessment procedure and minimized the cost by security countermeasures.

ROAM introduced the following steps to assess the risk and determine the countermeasures:

1- Identify the critical functions
2- Identify the critical systems
3- Assess the vulnerabilities of the critical systems
4- List identified vulnerabilities, identify the threats through the vulnerability properties using visualization techniques, after that define if there is a chance for an attack to exist
5- Analyze the real impact on a defined vulnerabilities
6- Threat-vulnerability analysis step: risk happens if there is a threat which affects the vulnerabilities
7- Determine the likelihood that a potential vulnerability will be exploited
8- Estimate the initial risk before any countermeasures are implemented
9- Security control recommendation.

RAOM model used heuristic search to solve a tradeoff problem between cost and security.

RAOM seeks to assess risk considering an impact on confidentiality, integrity and availability (CIA); it considered likelihood that possible threats will exploit identified vulnerabilities

In their work DeGramatica, et al in [43] they studied catalogues of threats and security control effectiveness on security risk assessments. They assessed the effectiveness of using domain specific versus, domain general catalogues by non- experts and compare it with running the same assessment by security experts without using catalogue.

They found that in quantitative terms there was no different in effectiveness of using catalogues in risk assessments by non-experts or run risk assessment by security experts without catalogues, on the other hand in the qualitative term the security experts used catalogues as a check list to make sure that nothing was forgotten but non- experts were worried how to use the catalogues. As a result they found that catalogues can be used as a common language in security risk assessment. [43]

In addition Vitti, et al in [3]provided a cloud security monitoring tool, that gathered data from VMs and network and them gives the administrator a clear data about their systems through cloud. The system defined the cloud entities, cloud user, administrator and security application. Then it defined cloud components that gather data from different needed sources.

They used, individuals security metric, data security metric, access control metric and server security metric, the provided system have a pre-defined action that it can perform to protect the VM depending on sec- SLA between service requester and provider.

From the above summarization, it can be concluded from studying the related works in risk assessments that many researchers provide different risk assessment approaches in assessing software security risk, the shared factors between these approaches are studying the system elements, security objectives, and the available weakness in order to select security countermeasures and configurations, additionally all these approaches depend on historical data and security expert opinion.

We found that DREAD model can be adapted and modified in our work because it was used before by many researchers and it selects a group of threats to be avoided, on the other hand DREAD model is clear and effective we will modify it to assess SOA security risk. Table 5 shows a summary for Security Literature Review Studies.

Author	**Work**	**Approach**	**Description**
Wolter, 2008 [38]	Modeling Security Goals in Business Processes	model-driven transformation	The model driven transformation enhances the security of business process by the following steps: - Provides an extension modeling for security configurations goals in a business process. - Discusses the difficulties to manage security mechanisms and integrate it into process system. - Analyzes security goals and concerns to generate a general security policy. - Finds the dependencies of the security policy and constraint models for business. - Specifies the security configurations for business processes.
Microsoft [39]		Microsoft STRIDE	Threat modeling approach that is used to give weight to every Security objective
Alkussayer, 2010 [36]	Scenario based	A scenario-based framework for the security evaluation	- A Systematic scenario-based framework to evaluate the security of software architected. - Construct Security profile by using security template scenarios.

		of software architecture.	- Security profile used to identify a set of scenarios to be used during the evaluation process by using a Full Scenario Table (FST).
Saripalliet al. , 2010 [4]	QUIRC: A Quantitative Impact and Risk Assessment Framework for Cloud Security	QUIRC	Presents a quantitative framework for assess the security impact and risks for deployment an application in to cloud by introducing two approaches: 3- The cloud security defense must be strong, scalable and effective from cost point of view. 4- FIPS model: the Potential impact is consider low if the loss of one of the Security objective has low effect on the organization assets, operations or resources. The importance of QUIRC it enables companies to compare between cloud service vendors security. The limitation of this approach is in finding Subject Matter Experts (SME) for the probability of the threat. It depends on an expert prediction for the threat occurrence
Lowiset al., 2011 [26]	Vulnerability analysis in SOA-based business processes	ATLIST analysis method	Provides analysis method for vulnerabilities in the SOA Based business process scenario which combines the notation of the attack tree and the Fault/attack trees and Failure Mode and Effects Analysis (FMEA) approach.
Zevin, 2009 [37]	Standards for security categorization of federal information and information systems	ATLIST	Assumes that vulnerabilities have been sorted in order of their estimated risk level. By looking at the triggering properties each path to the root node represents a vulnerability type. ATLIST trees help in the derivation of the vulnerability patterns by narrowing down the search space.
Viduto, V., et al. , 2012 [41]	A novel risk assessment and optimization model for a multi-objective network security countermeasure selection problem. Decision Support Systems	ROAM model	- Presents a model that Satisfies security needs in cost-effective manner. - It gives effective risk assessment procedure and minimizes the cost by security countermeasures. ROAM consists of nine steps: 1- Identify the critical system functions. 2- Identify the critical systems. 3- Assess the vulnerabilities of the critical systems

			4- List identified vulnerabilities. 5- Identify the threats. 6- Define if there is a chance for an attack to exist, analyze the real impact on a defined vulnerabilities. 7- Threat-vulnerability analysis (risk happens if there is a threat which effects the vulnerabilities). 8- Determine the likelihood that a potential vulnerability will be exploited, and Estimate the initial risk before any countermeasures are implemented 9- Security control recommendation
Microsoft, 2008 [44]	Risk assessment model (DREAD)	DREAD	Risk assessment model is used to estimate vulnerability's risk level which represents Damage potential, Reproducibility, Exploitability, Affected users, and Discoverability.
Yautsiukhin, A., et al., 2008 [42]	Towards a quantitative assessment of security in software architectures.	DREAD	Using DREAD to estimate software architecture security to evaluate software risk severity for a group of threats, by specifying the software special security requirements and estimating DREAD five factors values. And then uses security added patterns to enhance software architecture security.

Table 5: Security Literature Review Studies.

3.4 QoS of the Service composition architecture

In SOA Services are combined to get reliable service compositions (SC), the preferred SC is the one that provides functional requirements with a good level of Quality of Service (QoS).[45].

Many researchers studied the SOA QoS, and there are many researches focused on finding the best SC based on the QoS, in this section some of previous works surveyed.

Ivanovic in [45]provides an approach to presented a model to predict the QoS for service composition based on the Service Level Agreement (SLA) between the service provider and Consumer, the SLA was used to decide the acceptable level of QoS.

QoS Analysis: The QoS aggregation approaches in general focus on the control structure and ignoring the data operations, their results were not sufficient to do reasoning for the probability of SLA violations.

Other approaches focus on the upper and the lower limits of the component and composition QoS, and without taking the distribution of values in consideration which were not accurate. [45]

The QoS was defined on SLA, it contained service-level objectives (SLOs), concrete numerical QoS objectives that was the requirements that services must meet, any violation in SLA may cause a monetary compensation. [46]

Sun and Ping in [47]discussed finding the best SC based on the individual service QoS, to minimize the failure; the services were selected based on QoS

Composing a low-quality service was supposed to inherit the draw back to the SC. The reliability of the composite service was the product of the aggregated services, so the unreliable service will affect the overall reliability of the composite service.

Four quality of service were defined in [47]:

1- Time (T): The time taken by a service to process its sequence of activities.
2- Price (P): The amount of money for a single service execution.
3- Reliability (R): The probability that a service to perform its functions correctly under stated conditions.
4- Fidelity (F): The average reputation rate of the service reported by clients.

The service composition was defined in the following scenario:[47]

1- A composition planer generates abstract composition plans to fulfill user requests, each plan contains a set of abstract tasks and the control flow (and data flow) among them. In general.
2- A service matchmaker semantically discovers candidate services that fulfill the functional description for each abstract task.
3- If there are more than one candidate the optimal services will be selected based on the QoS and the balance between QoS and risk.
4- The execution manager executes the selected composite service and returns the result to requester.

There optimal selection of composite service was as follows:[47]

1- Individual service selection according to the plan:
2- Optimal composite service selection

Predicting quality attributes of software product lines using software and network measures and sampling model was presented in[48]. In this work they don`t use run time parameters to predict the quality attributes such as workload, input data, execution path and loop boundaries. Instead they use static data that make calculation faster and avoid exponential calculations. This gives a non-accurate prediction. They depend on a smart sampling algorithm.

They defined quality category of the quality attributes and find the predictor of each attribute. And then performed the sample a representative feature set for the quality category using the predicators.

Categories are either a high risk categories or low risk categories. High risk categories are measures using node rank that contains high severity of bugs.

Predicators are determined using a measure or a combination of several measures and corresponding threshold. To measure the prediction they use information collected from software and network measures numerically. And by comparing between source code components and memory consumption

The internal attributes can be measured by analyzing the products itself. But the extern al attributes measured by the product behavior in the environment. And now other attributes such as workload, platform characteristics, etc.to find a good predictor they use a binary classifier to determine a threshold.

Another approach for predicting the Quality of service (QoS) was presented in [45]. They use uses the approximation of ranges of actual QoS values for the service in acceptable values, instead of using the average or median of the service properties that are not usually useful in predicting the quality of the service,

They mentioned that the acceptable range of values often need to be enlarged, especially that because services exhibit a “long-tail” behavior. And it considered that they can get a higher level of accuracy by representing QoS variables directly with full-fledged probability distributions and operations on them. They provide a discrete probabilistic model to find the uncertainty of QoS for the service composition. The QoS attributes that they work on are execution time, amount of data sent/received, required bandwidth, number of general or specific operations or component invocations executed monetary cost of operation, availability, and any other measured attributes.

Other QoS attributes have been studied in [16] were: execution time, the required network bandwidth, cost, availability, and other relevant performance and user experience factors.

3.5 Service oriented architecture as multiple objectives problem

This section summarizes some important researches that studied SOA as a multiple objectives problem.

Leitner in [46] provided a model for service composition optimization model depending on reducing the cost and SLA violations. This work presented the tradeoff between the reducing service cost and reducing the SLA violation as optimization problem. A manufacturing case study was used to evaluate their approach. The manufacturer's business was based on a service-based notion, and the process was mapped to one or more services, and the whole business process was implemented as service composition. They listed some Service Level Objectives (SLO) and gave each SLO a target value and then, the tradeoff was taken between duration, costs, and quality, and the services must be optimized between these three factors.

They presented PREVENT framework which was a closed-loop system used for self-optimizing service compositions. It depended on VRESCO the existing SOA runtime environment. The seminal steps of PREVENT were:

- Monitor
- Analyze
- Plan
- Execute

They focused on implementation of the Cost-Based Optimizer; the component received an estimation of concrete SLO values from the Violation Predictor component, and decided the suit adaptations to the composition instance.

There were three types of data

1- Facts: measured data at optimization time.

2- Unknowns: the data that is entirely unknown at optimization time.

3- Estimates: Not facts and not unknowns, the data that are not yet available, but it can be estimated.

Facts and estimates can be used in Machine learning process to use it for violation predictor. The optimization problem was choosing the cost effective adaptation to avoid the predicted SLA violations.

I: the set of all possible composition instances of a client.

$i \in I$: a concrete instances which can be monitored by PREVENT tooling.

$S= \{s_1, s_2, \dots, s_k\}$: a set of SLOs defined in SLA.

$P= \{P_{s1}, P_{s2}, \dots, P_{sk}\}$: a Penalty functions that present a cost depends on SLO measures.

$m_s: I \to [0,1]$, where ms is a SLO measured.

$P^i_{s} = P_s(m_s(i))$: penalty function

$A= \{a_1, a_2, \dots, a_l\}$: *A all possible adaptations.*

$A^* \in \wp(A)$: $\wp$ *power set of A*

They considered it a one-dimensional discrete optimization problem, and presented it in deterministic and heuristic solution algorithms.

In a research article by Ye, et al [49] the services have been classified into application services and utility services, application service provided the functionality but utility service provided the virtual machine, database, and network services.

GA was used to optimize service composition. The optimization objectives were response time, price, availability and, reputations.

The service composition (SC) was generated by mapping the abstract service into functionality services and utility service, SC had two parts; the matching string and scheduling string, the matching string included the representation of the service, the virtual machine, database and network that were used to fulfill the functionality, and the schedule string was a topological sort of the data flow in the composition.

The problem formulation started with driving the initial population and the chromosomes were selected randomly for the next generation.

Crossover operator was run with probability equaled 0.4. two pairs of matching string were selected randomly then a cut-off point was selected to cut the chromosome into top and bottom parts, the bottom parts were then exchanged, then the abstract service in each bottom was reordered, the new order of the services in one bottom part was relative position of these services in the original scheduling string in the pair.

The mutation operator was applied with probability equaled 0.1 by selecting a target service from the scheduling string randomly and replacing it with another service from valid range without violating constraints.

The chromosomes were classified according to their fitness values; the fitness values were calculated by finding the distance from constraints satisfaction, the below equations show how the distance values were calculated and how the distance is used in the fitness function:

$$\boldsymbol{D(c)} = \boldsymbol{D(c)} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{l}} \boldsymbol{QC^i(c) * ei * weight^i}, \quad \boldsymbol{ei} = \begin{cases} \mathbf{0} & \boldsymbol{QC^i(c)} \leq \mathbf{0} \\ \mathbf{1} & \boldsymbol{QC^i(c)} > \mathbf{0} \end{cases}$$

$$\mathbf{F(c)} = \sum_{\mathbf{i=1}}^{\mathbf{4}} \mathbf{w^i * Q^i(c) + weight\ p * D(c)}$$

Where wi : weight for each QoS attribute.

Weight p: penalty factor.

The process was iterated until the distance from constraint equaled 0 or reached the maximum number of generation.

The experiment was run on two scenarios:

1- Large scale scenario that contains 100 abstract services, each abstract service can be mapped in to 30 concrete services, and can run in 20 different VM, db or net.
2- Small scale scenario was run with 5 abstract services each were run by 2 concrete services and handled by 3 VM, db and net.

Zo, et al, in [13] presented an approach for optimizing service composition. The business process was divided in to tasks and then to sub tasks, each task was supported by one service or more. The tasks were aggregated in predefined patterns; in sequence, parallel or loop. The optimization problem was to maximize security and minimize the execution time.

The work flow patterns was used to compute the QoS values for the generated service composition from the QoS values of the services aggregated in the composition, the values provided by service provider for both security and processing time for each single service. Measuring performance requires more than one metric. They took the number of processed transactions per time. (Processing time, communication time, delay, and onsite processing), the total processing time was the appreciate performance metric to assess deferent services that accomplish specific task.

Security was measured by the ratio between the number of transaction that were not compromised by security breaches and the total number transaction processed by service provider. The disadvantages of using such metric are that service provider may not provide real information, additionally it is not an accurate metric. On the other hand it is easy to be assembled and a simple metric that can be grasped by decision makers.

Security of processing task i with service j is given by: $S(t_{ij})=S_{in}+S_{en}+S(S_j)$

Where:

- S_{in}: security within internal network
- S_{en}: communication
- $S(S_j)$: security associated with the service

Assuming that there was available number of services accomplishing each task, the GA was use to handle the selection problem, although the functional requirement must be satisfied before considering the service as a candidate the nonfunctional requirements are a part of the selection in addition to other technical issues such as orchestration, choreography to check the effective interfaces between the services.

Chromosomes were created to each sub process with length of the number of tasks and number of services per task, they run the experiment in a drop ship case study, taking different number of sub task in each scenario, and the evaluation function was used as a combination between security and performance with tradeoff approach.

One randomly selected crossover point was used with probability equaled 0.8 and random mutation operator with probability equaled 0.5, the population size was 1000. Finally the weights of security and performance were between 0 and 1with incremental step equaled 0.1. [13]

On the other hand Canfora, et al in their work in [50] presented a heuristic search model for optimizing service composition using genetic algorithm. Service composition was built by bounding the abstract services of the problem to one or more concrete services that were functionality equivalent; the difference between services was in nonfunctional quality of service response time, reliability, availability and cost.

The QoS attributes values of the service composition were calculated from QoS values of concrete services that formed the composition using aggregation rules depending on the work flow patterns that followed by service execution. They didn’t introduce a measurement function of the QoS values they just compute it for the composition.

The chromosome was represented by an integer array with size equal the number of abstract services, each item is a pointer to another array of concrete services that were bounded to the abstract service.

The crossover operator was performed with standard two points with probability 0.7 while the mutation operator was randomly select an abstract service and randomly replace the concrete services that were mapped to it with probability 0.01.

The generated chromosomes were evaluated by both static and dynamic penalty function. They calculated the distance from pre-defined constraints values; the stopping condition was reaching a distance 0 from the constraints or reached maximum number of iterations equaled 100.

An approach for service composition optimization was presented by Leitner, et al. in [51]. The optimization objective was to minimizing the cost of caused by Service Level Objectives (SLO) violations in addition to minimize the service cost.

Service level objectives SLO were defined, these SLOs were optimized based on cost, the model depended on a PREVENT model.

PREVENT model worked as SLO predictor and it was a neural network model it consists of a set of estimator functions and it depended in human knowledge, previous historical data and domain specific metrics. The Prevent model predicted the values of service on functional QoS attributes, and then the aggregation rules were used to calculate the total cost of the service composition.

Moreover a set of penalty functions that were defined to find the fittest solution; there were stage penalty, static penalty, linear penalty and linear penalty with cap.

The dependency tree was generated to explain the SLO and helped analysts to explain the lack of performance vs. high cost.

The solution of this model was to select the best adaptation transactions from predefined set of transaction that included changes on the services and how they were aggregated in the composition to full fill the business process functionality in lowest cost.

The model was GA based, the chromosome was a binary vector of a services call, they randomly generated populations, selected a set of solutions for the next generation, the fittest solution was the lowest cost.

The cross over operator was performed to generate a new chromosome based on the selected solution from last generation. The cross over point was randomly selected. And the mutation operator was done by flip one bit of the generated solution randomly.

The experiment was run on manufacturing process example they defined a range from 1 to 15 to evaluate the generated adaptation process. [51]

The genetic algorithm was implemented by Karatas in [23] to optimize the interdependency between security objectives in SOA. The study introduced three relationships between security objectives; Security objective may strengthen, weaken or imply other objective, hereunder the explanation of these relations:

1- Strengthen: when enabling a security for one objective another objective may ensure, such as confidentiality and anonymity.
2- Weaken: when enabling the security solution for a security objective this may contradict with another enabled solution such as accountability and anonymity.
3- Implication: enabling a security solution ensure the use of another solution such as

They classify tasks in to classes according to their functionality, tasks have to be executed in a certain order with certain constrain. Each abstract task was mapped to multiple alternative tasks to fulfill the user requirement. The Semantic of the QoS attributes were depended on attribute type and measurement scale where the attribute type can be binary, continues or discrete, and the measurement scale are ration, nominal or interval

The problem was solved by GA using multiple algorithms, Random, IBEA, NSGA-II and SPEA2. The chromosome was an integer array with number of items equal number of tasks in the corresponding business model. The cross over operator was run by selecting one cut point of two chromosomes and exchange the array content to create the new individual after that they run the mutation operator. In order to find the fittest individual, the global QoS attributes are calculated from the QoS of the alternative services using aggregation rules. Each security objective was measured by a utility values and the interdependency between security attributes was measured by a single value. Each security objective was modeled by a utility function. And simple adaptive weighting rules were used to score the objectives values. The protection function was used to assets the utility service composition toward fulfilling certain security objective, for example the confidentiality security objective required measure the used encryption algorithm, the key size and the TSL message encryption. This study aimed to classify the used components in the composition rank it according to the feasibility the importance for each security objectives.

Each security objective is expressed by a protection function, it takes aggregation QoS vector as input and the utility value that fulfill the security objective as output. The weight interval were [-1-1] where -1 means negative impact, 0 no effect and 1 positive impact.

The experiment was run in Java environment that extended the Jmetal framework 4.0, with population size equaled 100 and the work flow varied from 10 to 50 with stepped 10, for each task there were 20 alternative tasks. 2 protection functions were added and they continued to ass one or more to maximum 6 protection functions.

Each protection functions contained at least two factors with uniformed weighting. So the fitness value was determined according to the protection value and the constraint violation of the user requirement. Where the user requirements were normalized and aggregated to a QOS vector

A tool called QoS model editor was used to define a nominal attributes of QoS values and assign it to single utility value. [23]

After studying SOA as an optimization problem in literature, we found that many researches solve SOA as a Search Based Software Engineering (SBSE) problem, and GA has been used in optimizing SOA with different objectives, such as security, cost, performance, reliability, SLA violations and availability.

We found that the approaches which studied security as an optimization objective based on information from service provider to assess the security risk, this information may be inaccurate and they did not provide an effective enhancement on SOA security. To cover this gap we will use a general service attributes in our model to assess the service security and add a security solution to the SOA architecture randomly through the search operation. The summary of SOA as multi- objective problem is shown in table 6

Author	Work	Objectives	Input	Encoding	Crossover	Mutation	Fitness	Algorithm
Ye , 2011 [49]	Genetic algorithm based QoS-aware service compositions in cloud computing. In Database Systems for Advanced Applications.	Response time, Price, Availability, and Reputations.	Application services and Utility services.	it consists of two parts the matching string and scheduling string	- Crossover probability =0.4 - Two matching string were selected randomly. - a cut-off point was selected to cut the chromosome into top and bottom parts. - The bottom parts were then exchanged, then the abstract service in each bottom was reordered.	- The mutation probability = 0.1 - selecting a target service from the scheduling string randomly and replacing it with another service from valid range without violating constraint	finding the distance from constraints satisfaction n	NSGA-II
Zo, 2010 [13]	Security and performance in service-oriented application	Security and performance	The business process was divided in to tasks and	Array of string	Single point cross over in randomly selected crossover point	Random mutation operator with probability equaled 0.5. replace one of the services by	Security: The number of transaction that	NSGA-II

	s: Trading off competing objectives. Decision Support Systems		then to sub tasks, each task was supported by one service or more		with probability equaled 0.8.	another service in the pool	were not compromised by security breaches/ total number transaction Performance: execution time.	
Canfora, 2005 [50]	An approach for QoS-aware service composition based on genetic algorithms. in	response time, reliability, availability and cost	concrete services and abstract services	Array of integers with size equal the number of abstract services	The crossover operator was performed with standard two points with probability 0.7	The mutation operator was randomly select an abstract service and randomly replace the concrete services that were mapped to it with probability 0.01.	Aggregation rules depending on the work flow pattern. And using static and dynamic penalty functions	NSGA-II

Leitner, 2013 [51]	Cost-based optimization of service compositions. Services Computing,	Service Level Objectives violations. And Cost	List of service level objective	Prevent model	Single point cross over	_	choosing the cost effective adaptation to avoid the predicted SLA violations	NSGA-II
Karatas, 2015 [23]	Service composition with consideration of interdependent security objectives. Science of Computer Programming	Dependences between services security objectives	3 relations between security objectives: Strengthen Weaken and Implication	integer array with number of items equal number of tasks in the corresponding business model	Selecting one cut point of two chromosomes and exchange the array content.		Aggregation rules. And protection function assets the utility service composition security.	IBEA, NSGA-II and SPEA2

Table 6: related works in optimizing SOA as Search-based problem

Chapter 4

Research Methodology

4.1 Introduction

This chapter aims to present the methodology that is followed to investigate the research objectives. The main objective of this research is to study the optimization of the service composition using heuristic search.

We have investigated applying Genetic Algorithm (GA) to optimize service oriented architecture in terms of security goals and cost.

This research is an empirical research; we develop an experiment to implement GA in multi-objective optimization problem in SOA. Figure 7 below illustrates the followed methodology throughout this research,

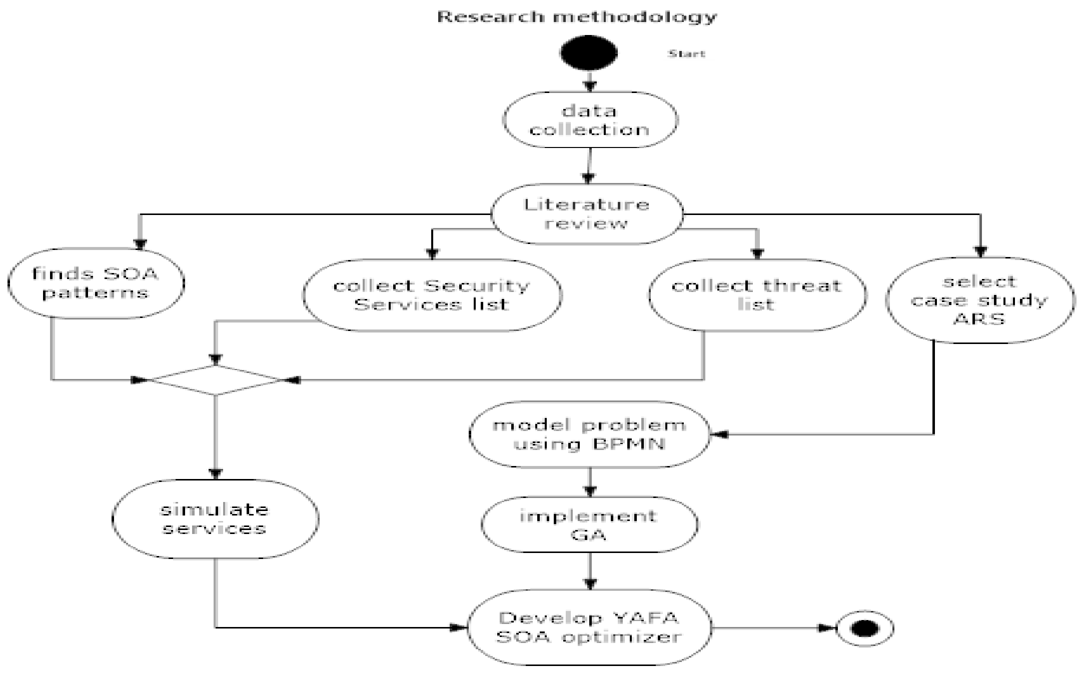


Figure 8: Research methodology

We develop an optimizer called YAFA SOA Optimizer to investigate our Research Questions. In order to run the experiment; data collection is required. We collect data and information about the research problem, we perform a literature review for related works in the research area, and then we select the research case study. We Modell the research problem using Business process modelling notation (BPMN) and implement the GA to investigate the research objectives, finally we run the experiment and analyses the results. Data collection process includes collecting the threats, security services, and simulate functional services.

The next sections illustrate the used research methodology steps; clarify the objectives and the input and output of each step.

4.2 Literature Review:

Surveying the related work aims to collect the needed that to build the model, collecting data from literature is important to find the gap that will be filled in our research.

We study Service Composition (SC) security, service composition, Search Based Software Engineering, and Security Risk Analysis; we determined the input derived data from literature review are:

1- Studying service oriented architecture (SOA) security issues, helps in selecting the security requirements, security objectives.

2- Studying service composition and service composition patterns, helps in modeling the problem into Business process modeling notation (BPMN) and selecting the aggregation rules.

3- Reviewing Search Based Software Engineering helps in selecting the Genetic algorithm (NSGA-II) and understanding the concept of heuristic search.
4- Surveying SOA risk assessment models; helps to select DREAD model to adapt it in this research.
5- Specifying a list of well-known threats, and work to eliminate their risk.
6- Prepare a list of well-known security services to add it to SOA in order to enhance the SC security.

The findings of Literature Review are summarized in Chapter03.

4.3 Problem Definition:

We search for a case study that contains different type of aggregation patterns, we chose Airline Reservation system (ARS); ARS is divided into 4 sub process Identification, Booking, Payment sub process and edit flight sub process, then each sub process is modelled using Business Process modelling Notation(BPMN)[52].

ARS is selected because it has been used before in literature, it contains different type of services and patterns, and it can be divided in to sub process.

ARS then is presented using BPMN [52]and service composition patterns (Chapter02)[52] in which the whole system was divided in to 4 sub processes based on the functionality.

Each sub process contains a grouped of abstract services, each abstract service is mapped to a concrete service and the aggregated concrete services forms the Service composition (SC).

We study each sub process separately and then study the whole system in order to evaluate the effectiveness of divide and concrete concept in service selection process. The conceptual model is described in Chapter 05.

4.4 Implemented Genetic Algorithm:

Optimizing SOA is considered as multi-objectives optimization problem, for which genetic algorithm called NSGA-II [2]is used to search for Pareto Front solutions, each solution represents SOA in GA representation.

The problem is encoded in to GA, each Service composition (SC) is represented a chromosome, the chrmomosome is an array of concrete services, and the service is represented by a gene in the chrmomsome.

GA operators are adapted to fit the problem architicture. The representation of the problem in GA will be discussed in details in chapter 06.

4.5 Running the experiment:

In order to run the experiment an optimizer called YAFA-SOA optimizer has been developed based on JMetal Framework [53]. Chapter 07 illustrates YAFA-SOA optimizer architecture in details.

The experiment has been run to optimize service composition in multi-optimization problem for 2 and 4 optimization objectives.

The input data of the experiment is synthesis data, it is collected depending on domain and security expert opinin, services registry information, and service definition language (WSDL).

The input data of the experiment is listed below:

- Services information, with included general information, cost, and security information.
- List of threats.
- List of security solutions.

The Security information for each service is calculated by adapting security model called DREAD model. The adaptation of security model is disscussed in Chapter 06.

The security for service composition is calculated using the aggregation rules.

Finally, the results generated by the experiment have been discussed and the following indicators were used to interpret the results HV, time, and median. The results are presented at Chapter07.

Chapter 5

Conceptual Model

5.1 Introduction

Business process is an abstract description of the process functionality, which could be modelled as a group of related tasks that perform specific functionalities such as providing services or product for certain customer [33].

Business Process (BP) has been modelled using Business Process Model and notation (BPMN)[54], which is a flowchart that describes a sequence of related tasks that interact to accomplish the business process functionality.

SOA will be mapped from the BPMN to service composition that fulfils both functional and non-functional requirements.

Business process is divided into sub process for calculation simplicity and diversity of solutions. Each sub process contains a group of tasks, each task of those can be mapped in to one or more services from a service registry, there for each sub process will form a service composition at run time.

In order to build the research conceptual model, we followed the steps below:

1- Selecting the case study.

2- Model the case study using Business Process Model and Notation (BPMN)[54]

3- Dividing the whole problem into sub process.

4- Determine the service structure.

5- Encoding the problem into GA in order to use heuristic search methodology to investigate the research Question.

The next sections illustrate modelling the problem in to Business Process Model and Notation (BPMN)[54] (Steps 1-3). Step 4 and 5 are presented in Chapter 06.

5.2 Business process model

This section illustrate the Airline Reservation System (ARS) case study, modelling the problem into Business Process Model and Notation (BPMN)[54], and dividing the problem into sub process using different Service Composition Patterns that was illustrated in Chapter 02 before.

5.3 Case Study: airline reservation system (ARS)

Airline reservation system (ARS) has been used as a case study to validate my research questions. ARS is widely used all-around the word nowadays; it enables users to manage their flights online through internet. The main transactions of the ARS is user authentication, reserving tickets, cancelling reservation, paying for travel and rescheduling tickets. In the rest of this section we describe the benefits of the ARS, the major functions and then present how it is used in the research.

5.3.1 Advantages of Airline reservation system (ARS):

According to [55, 56] ARS has a good advantages that are listed below:

- ARS is used to minimize the system administrator and reservation clerks work.
- Helps users and system administrator to maintain the consistency between different physical locations and different access types.
- Helps to manage emergency events such as flight cancelation through bad weather and keep customer updated though their profiles.

- Helps in improving the revenue of Airline Company through reducing transactions cost, increase the effectiveness of special offers and discounts, increase the flights utilization.
- Gives a dynamic pricing model based on the reservation time..
- Save customers effort and time.
- Gives customers privacy.
- Enables customers to edit reservations remotely.

5.3.2 The major functions of ARS system include: [55, 56]

ARS major functionalities that are used in modelling ARS are:

1- Manage user accounts.

2- Describes flights scheduling, directions, levels, etc.

3- Booking flight online through internet without physical attendance to the airline company office.

4- Pay through internet for the flights.

5- Edit flights, cancel flight, or reschedule flights online through a reservation code from user profile.

5.3.3 ARS system Business processes

ARS is modelled using BPMN and Service composition aggregation patterns, as it is shown in figure 8. Business process model and the service composition patterns is used to model SOA architecture problems [35], for example Zo used a drop ship case study for manufacturing business process in his work on[13] and he implements BPMN and SC patterns in modelling the problem.

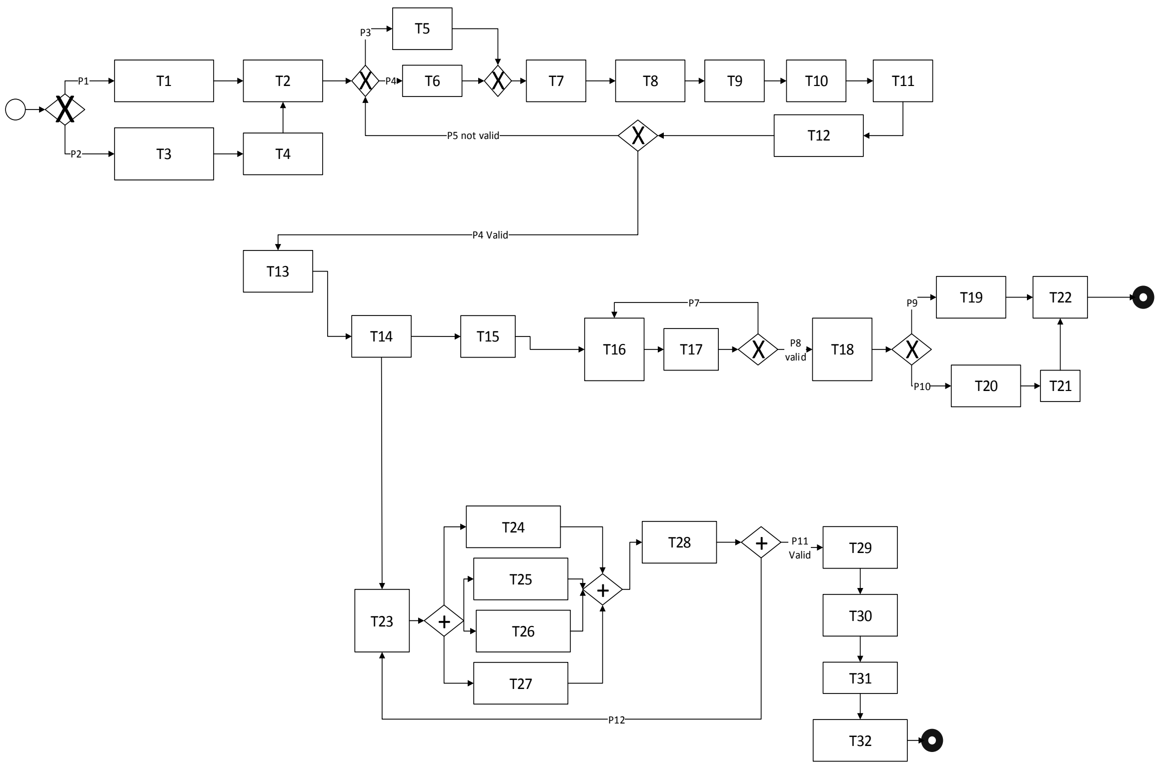


Figure 9: ARS-Whole business process

Where the abstract services (Task) are:

T1 is Validate Authentication

T2: Authenticate user

T3: Validate registration data

T4: Create user

T5: Search

T6: Brows

T7: Select Date

T8: Select Direction

T9: Select Class

T10: Select Seat

T11: Add to with list

T12: Validate reservation

T13: confirm

T14: Book

T15: create invoice.

T16: Provide credit card info

T17: Validate the card.

T18: chose payment option

T19: cash payment

T20: credit card payment

T21: Pay

T22: Send confirmation email

T23: select flight to change.

T24: change flight date.

T25: change flight schedule.

T26: change flight Seat.

T28: check flight availability.

T29: confirm change.

T30: get difference invoice.

T31: Pay Invoice.

T32: Send confirmation email.

ARS Business process has been divided into sub-processes. Each sub-process contains a group of tasks, which is labelled as abstract services, and each abstract service is mapped to one or more concrete services from the Services Registry that fulfils the functionality specified by the abstract services.

The concrete services are aggregated to form a service composition (SC). There will be a large number of concrete SCs that can be selected; each one could be a candidate for an optimized solution.

ARS has been divided into four-sub-processes based on the major functionalities of the ARS in section 5.3.2. The sub processes are Customer Identification, Booking, Payment sub process and edit flight sub process. Each sub-process consists of various abstract services which are aggregated together using the SOA aggregation patterns that was illustrated before in chapter 02, the sub-processes are illustrated below:

1- Customer identification Sub process contains 4 tasks as shown in figure 8, the process contains all tasks related to the customer registration authentication and accessing his profile. User either have an account or need to register a new user account, the probability P1 means that the user have a registered account and P2 means that he need to register a new one. Both probabilities P1 and P2 are given by the system administrator from the historical logs data.

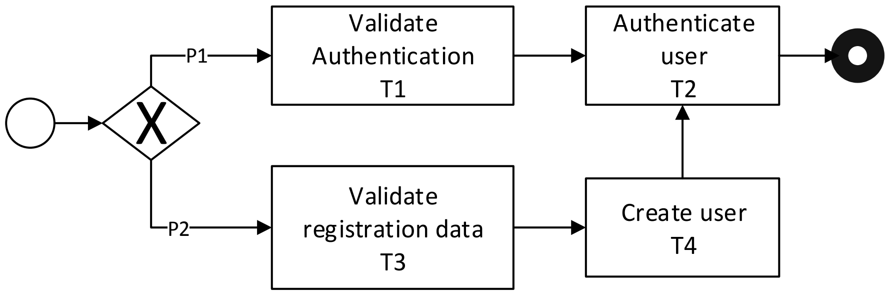


Figure 10: Customer identification Sub process

Where

P1: The probability that the user have an account

P2: The probability that the user has not an account and will register

P1+ P2 = 1

Task T1: Validate Authentication.

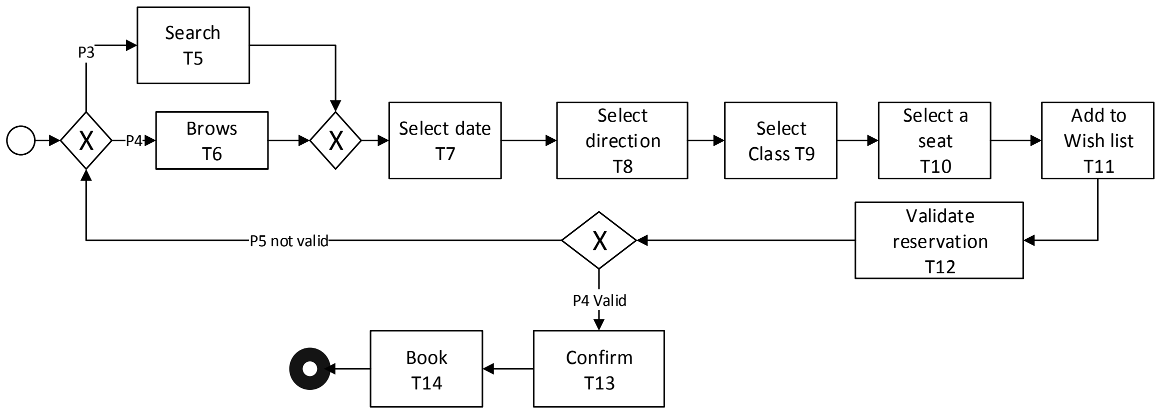
Task T2: Authenticate user.

Task T3: Validate registration data.

Task T4: Create user.

The used patterns are XOR, and Sequence patterns (Chapter02)

2- Booking or reserving sub process contains 10 sub process as shown in Figure 9, and this process contains all tasks that user perform to book a flight, user may search to a certain flight or Browse available flights in a certain detail, the probability P3 and P4 are provided by the system admin from the system historical logs data. The selected flight may not be available then the user has to search again for another flight, the probability of availability also is provided by the system administrator.



00:Booking/ reserving tickets sub process

Where

P3: the probability the user search for a certain flight

P4: the probability that the user browse all the flights

P3+ P4 = 1

P5: the probability that the flight is full

P6: the probability that there is still a seats

T5: Search

T6: Browse

T7: Select Date

T8: Select Direction

T9: Select Class

T10: Select Seat

T11: Add to with list

T12: Validate reservation

T13: confirm

T14: Book

The used patterns are XOR, Sequence patterns, loop and OR patterns (Chapter02)

3- Payment sub process contains 8 tasks as it was shown in figure 10, it contains all tasks that are related to the payment process, user may provide a valid credit card or invalid card and he needs to retry with another card. Additionally users may select to bay directly through credit card or later by cash payments. The probabilities P7, P8, P9 and P10 are given by the system administrator through historical log data.

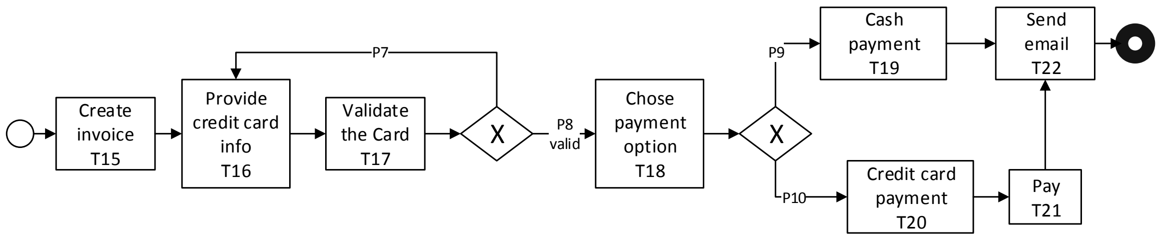


Figure 11: Edit flight sub process

Where

P7: the probability the user enters invalid credit card.

P8: the probability that the user enters a valid card number.

P3+ P4 = 1

P9: the probability that the flight is not valid, full or cancelled.

P10: the probability that the flight is still valid.

T15: create invoice.

T16: Provide credit card info

T17: Validate the card.

T18: chose payment option

T19: cash payment

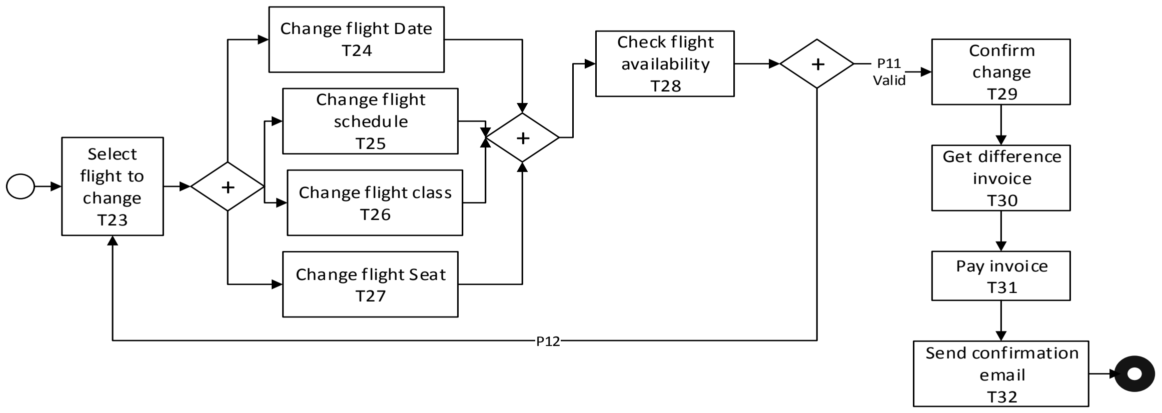
T20: credit card payment

T21: Pay

T22: Send confirmation email

The used patterns are XOR, and Sequence and loop patterns (Chapter02)

4- Edit flight sub process contains 10 tasks, as it is shown in figure 11, user edit one more flight details, he may change flight date, time class or, seat if the selected flight is not valid user have to select another flight , the probability is that the flight is valid is given by the system administrator from the historical data.



Select flight to change T23
+
Change flight Date T24
Change flight schedule T25
Change flight class T26
Change flight Seat T27
+
Check flight availability T28
+
P11 Valid
Confirm change T29
Get difference invoice T30
Pay invoice T31
Send confirmation email T32
P12

0: Edit flight sub process

Where

P11: the probability the user selects a valid flight.

P12: the probability that the user selects invalid flight details.

P11+ P12 = 1

T23: select flight to change.

T24: change flight date.

T25: change flight schedule.

T26: change flight Seat.

T28: check flight availability.

T29: confirm change.

T30: get difference invoice.

T31: Pay Invoice.

T32: Send confirmation email.

The used patterns are Sequence, parallel, loop and XOR patterns (Chapter02)

The above mentioned 4 sub processes are following the Service composition patterns. The QoS of the generated Service composition can be calculated using the aggregation rules that have been discussed in Chapter 02 before.

SC is a group of aggregated concrete services, to understand the model we have to discuss the Service structure, next section illustrates the service structure, and service attributes.

5.3.4 Service Structure

In SOA; services are aggregated to form the service composition, determining services structure is an important step in modeling process. Each concrete Service provides certain functionality, and has QoS values, services that provide the same functionality defers in QoS values.

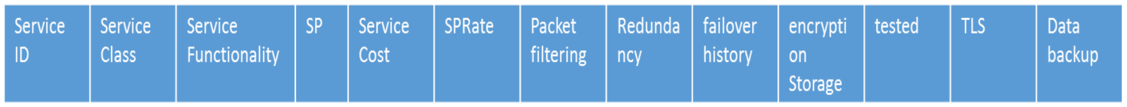
This research is concerned in service security and cost; and it is required to calculate security and cost for each service in the service registry.

We select a group of service attributes and include it in our research to estimate service security risk.

We avoid using the security information that is provided from service provider, because we assume that it may be inaccurate. and in literature it was one of the drawback of risk estimation probes provided in [13]. And instead we select a group of attributes that can reflect how secure the service is.

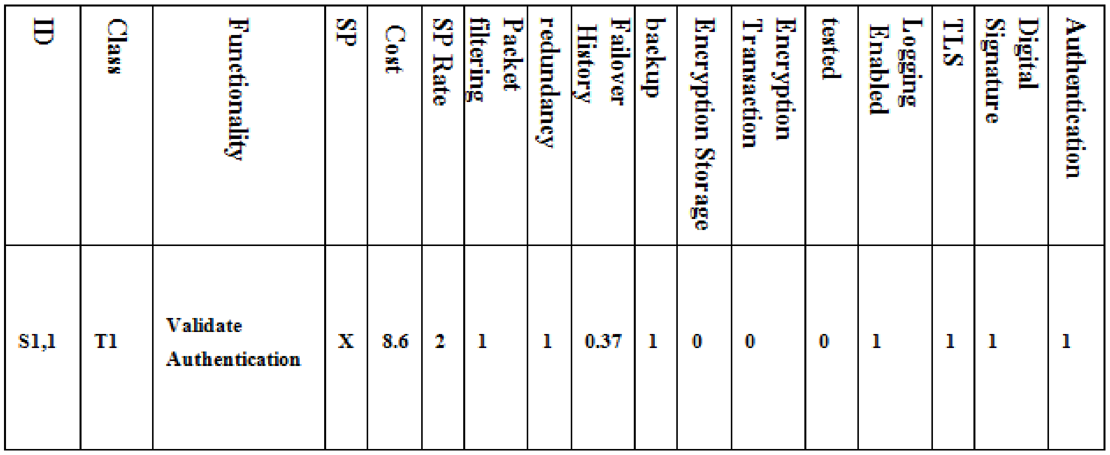
We looks at the well know protocols in service security and used Data that can be easily collected from WSDL file or service registry

The proposed service structure is shown in figure 12 and 13



Service ID	Service Class	Service Functionality	SP	Service Cost	SPRate	Packet filtering	Redundancy	failover history	encryption on Storage	tested	TLS	Data backup

Figure 12: Proposed Service Structure



ID	Class	Functionality	SP	Cost	SP Rate	Packet filtering	redundancy	Failover History	backup	Encryption Storage	Encryption Transaction	tested	Logging Enabled	TLS	Digital Signature	Authentication
S1,1	T1	Validate Authentication	X	8.6	2	1	1	0.37	1	0	0	0	1	1	1	1

Figure 13: Proposed Service Structure Example.

The selected Services attributes is a well-known attributes, and its values can be gathered from the service information registry and service definition Language (WSDL).

Proposed Service attributes are agreed with well-known security protocols applied on SOA such WS security. For example oracle presented a group of security attributes for SOA in [57], as listed below:

- TLS
- Authentication
- Logging integrity
- Authorization.

On the other hand the following protocols are used in securing web-services:[17]

- WS-Security
- XML Signature
- XML Encryption
- XML Key Management (XKMS)
- WS-Secure Conversation
- WS-Security Policy
- WS-Trust
- WS-Federation
- WS-Federation Active Requestor Profile
- WS-Federation Passive Requestor Profile
- Web Services Security Kerberos Binding
- Web Single Sign-On Interoperability Profile
- Web Single Sign-On Metadata Exchange Protocol
- Security Assertion Mark-up Language (SAML).

Proposed Service attributes the attributes are derived from all mentioned above protocols and are listed below:

1. Service Id: a unique attribute that used to identify the service in our model.
2. Service Class: classifying classes regarding to its functionalities
3. Service Functionality: the business function of each service.
4. Service Provider: business name of the company that provide the service.
5. Service Cost: monetary value in $

6. Service Provider Rate: each service provider has been evaluated by service users and this present a general indicator service QoS, this attribute is continually changed regarding to the users feedback. And it is considered a general QoS attributes
7. Packet filtering: this attribute is represented by a Boolean value and it indicates whether buckets are filtered or not through transaction step.
8. Redundancy: a Boolean attribute that indicates whether service has a redundant service or not.
9. Failover History: this variable indicates the historical data for the service previous failover.
10. Backup: whether service data are backed up or not.
11. Encryption Storage: Boolean value that indicates whether data are encrypted in storage level or not.
12. Encryption Transaction: Boolean attribute that shows if data are encrypted in transaction level.
13. Tested: if service is tested or not. This attribute can be found on Service registry site and it is easy to decide this Boolean value.
14. Logging Enabled: Boolean that shows whether service are saving access logs or not. This attribute can be determined from service definition.
15. TLS: a Boolean value that shows if TLS is enabled or not this attributes can be determined from service definition and the registry information about the service.
16. Digital Signature: Boolean value that shows whether it enabled or not. It can be determined from service definition and the registry information about the service.

17. Authentication: Boolean value that shows I the authentication is enabled for the service, and it can be determined from service definition and the registry information about the service
18. Confidentiality Severity: Calculated by DREAD model.
19. Integrity Severity: Calculated by DREAD model.
20. Availability Severity: Calculated by DREAD model.

In table 7 the security attributes are classified based on security objectives; confidentiality, integrity, availability and general attributes.

General Attributes	**Confidentiality Attributes**	**Integrity Attributes**	**Availability Attributes**
Service Provide Rate		Packet filtering	
Tested	Authentication	Digital Signature	Redundancy
Failover History	Encryption Storage	TLS	Backup
		Logging Enabled	
		Encryption Transaction	

0Table 7: service attributes and security objectives

Service attributes value ranges and the source of each service attributes are clarified in table 8.

Furthermore these values are clarified with an example in figure 13

Service attribute	**Affected goal**	**Source**	**Value range**
Service Id	_	Service registry	Text
Service Class	_	Service registry	Text

Service Functionality	Functional	Service registry	Text
Service Provider	_	Service registry	Text
Service Cost	Monetary	Service registry	(1-10) $
Service Provider Rate	General	Service registry	1-5 (higher is better)
Packet filtering	Integrity	Service registry and WSDL Web service specifications	Boolean value
Redundancy	Availability	Service registry and WSDL Web service specifications	Boolean value
Failover History	General	Service registry and WSDL Web service specifications	0-1 (Lower better)
Backup	Availability	Service registry and WSDL Web service specifications	Boolean value
Encryption Storage	Confidentiality	Service registry and WSDL Web service specifications	Boolean value
Encryption Transaction	Integrity	Service registry and WSDL Web service specifications	Boolean value
Tested	General	Service registry and WSDL Web service specifications	Boolean value
Logging Enabled	Integrity	Service registry and WSDL Web service specifications	Boolean value
TLS	Integrity	Service registry and WSDL Web service specifications	Boolean value
Digital Signature	Integrity	Service registry and WSDL Web service specifications	Boolean value
Authentication	Confidentiality	Service registry and WSDL Web service specifications	Boolean value

Table 8:service attributes, source and value ranges [57]

The next step after modelling the problem with PBMN is implementing the Genetic algorithm procedures and this is discussed in chapter 06**.**

Chapter 6

Multiple objective GA implementation to optimize SOA

6.1 Introduction

Many services provide the same service functionality with different quality of Service (QoS) attributes, which create large search space, additionally the search space increases exponentially by the moving from small business to enterprise solutions, which makes it infeasible for Software engineer to manually search the best group of services.

As a result, it is concluded that, finding the optimized group of services required an effective and professional automated way.

This research implements the GA to search for the best group of services that have the highest security and lowest cost.

This chapter shows the GA implementation in applying SBSE to optimize the Service Oriented Architecture (SOA) in term of security and cost, Presents Meta-heuristic search algorithms, illustrates GA Representation, discuss GA objectives and finally describes GA operators.

6.2 Meta Heuristic Search algorithm

Solving problem in GA starts with encoding the problem in to GA, creating the initial population, performing mutation and crossover operations, Evaluation process and finally selecting next generation chromosomes.

These steps are discussed in the below sections. As shown in listing 6.1 below:

```
❖ While termination condition is false
    ➢ Encode the problem in to GA
    ➢ Create the initial population
    ➢ Perform mutation and crossover operations
    ➢ Evaluate generated candidates
    ➢ Select next generation chromosomes
```

Listing 6.1: Pseudo code of GA.

An example of GA that is used by this research is NSGA-II, the NSGA-II pseudo code is shown listing 6.2 below:

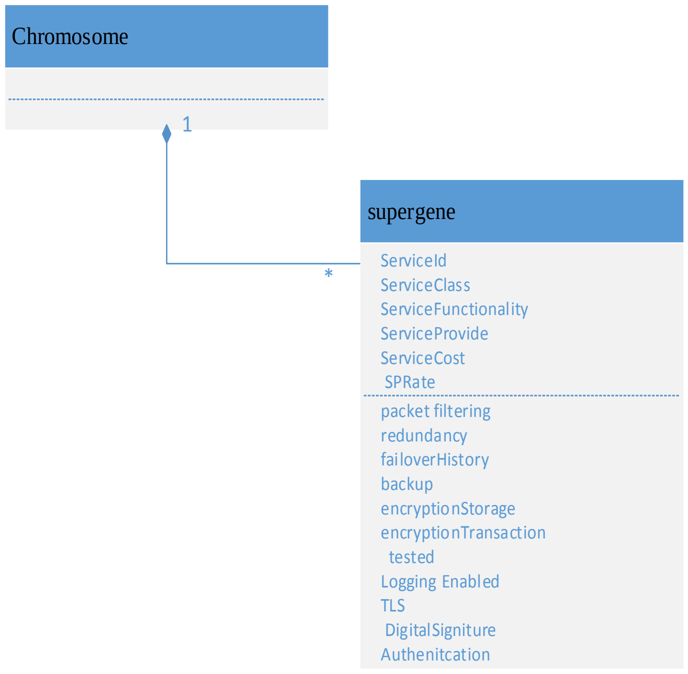
```
    ➢ population ← Initialize Population ()
    ➢ Q←Ø// Q = auxiliary population
    ➢ while not Termination Condition()
Do
        ❖ for i← 1 to (population size / 2) do
            ✓ parents←Selection(P)
            ✓ offspring←Recombination(parents)
            ✓ offspringMutation(offspring)
            ✓ Evaluate Fitness(offspring)
            ✓ Insert(offspring)
End for
        ❖ Ranking← population ∪Q
        ❖ Ranking And Crowding
        ❖ population ←Select Best Individuals
End while
```

Listing 6.2: Pseudo code of NSGA-II [29].

6.3 GA Representation

Representing the problem in to GA starts with encoding the solution in chromosome, chromosome consists of supergenes [49], each supergene represent a concrete service. From the Service structure which is discussed in Chapter02; service attributes are: service name, id, class, and service provider, service cost and service security attributes that are used to measure service security (Chapter 05).

The chromosome and supergenes architecture is presented in figure 14 using UML class diagram.



0Figure 14: chromosome and supergenes architecture using UML diagram.

Service composition consists of a group of services as it is shown in Figure 15, Services are aggregated in a pre-defined patterns each service composition are represented by a chromosome in GA, the chromosome consists of services and each service is represented by a supergene.

Chromosome is represented in our Research as an array of services, each service represent a supergene. The chromosome has a global quality of services values that are calculated by the local QoS of the concrete services aggregated in the composition[23], farther more local QoS values are calculated by DREAD model and the global values are calculating using the aggregation rules.

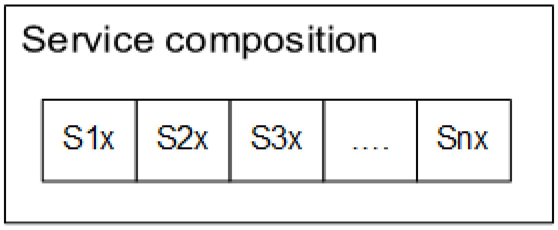


Figure 15: Service Composition architecture

6.4 GA objectives

This study consists of two scenarios based on the optimization objectives, we worked on four objectives two objectives, as it is discuss hereunder:

a- Multiple objectives optimization problem- Four objectives

1. Confidentiality severity
2. Integrity severity
3. Availability severity
4. Service cost

b- Multiple objective optimization problem- Two objectives

1. Security Risk severity
2. Service cost

Dividing security objective into three objectives confidentiality severity, integrity severity and availability severity provides the software engineer with a clear and accurate security view of the selected service based on the domain security requirements.

Hence each security objective has a value that reflects the risk severity of this objective. On the other hand these three values were grouped in the second part of this research on one value "Security Risk Severity" to study the behaviour of GA in optimizing Multi-objective problem for four and tow objectives.

6.5 GA operators

After encoding the problem with GA, search process starts with creating initial population, applying cross over and mutation operators are performed, and finally evaluation and selecting chromosomes for the next generations.

Initial population

The initial population is created randomly with pre-defined population size, a group of service compositions are created by selecting a random service from each service class.

The following procedure in Listing 6.3 shows how the initial population is created:

```
➢ For each service composition in the population
    ❖ For each class of services
        - Select a functional service randomly
        - Add the selected service  to the composition
```

Listing 6.3: Procedure of creating initial population.

Initial population contains a service composition that has random risk severity and cost, and by using the remaining GA process we work to find the composition of services that have lowest risk and lowest cost

Cross Over

Cross over generates new chromosomes from tow selected parents, these parents are selected randomly from the population, then a random cross over point is selected between two functional services to avoid a third party threat that can be occurred between functional and security service from different service provider.

After selecting the cross over point, the genes of the parents are exchanged on the cross over point, then the first offspring has same genes as the first parent before cross over point and same as the second parent after the cross over point.

On the other hand the second offspring contains same genes as the second parent before cross over point and same as the first parent after the cross over point. Listing 6.4 shows the Pseudo code of cross over operator.

```
Input: chromosomes parent one and parent two
Output: chromosomes offspring one and offspring two
i ← random Integer
where parent two. geneAt[i] and parent one. geneAt[j] is functional
        service  // select random cross over point between functional
        services.
for j ← 0 to i do
offspring one .gene At[j] ← parent one. geneAt[j]
offspring two. geneAt[j] ← parent two. geneAt[j]
end for

for k ← i +1 to two .length -1 do
offspring one. Gene At[k] ← parent two. Gene At[k]
end for
for m ← i +1 to one. length -1 do
offspring two. gene At[m] ← parent one. Gene At[m]
end for
```

Listing 6.3- procedure Pseudo code of cross over [29]

Figure 16 below illustrate cross over operator.

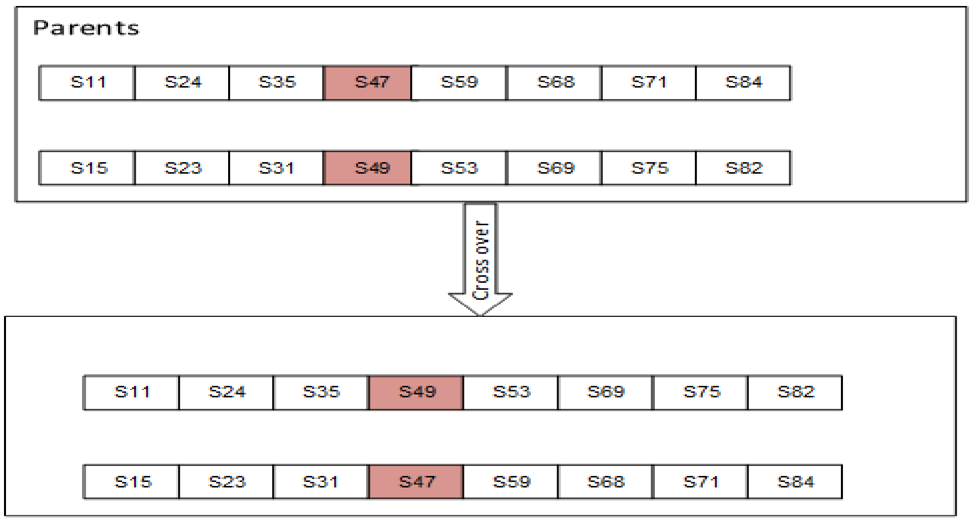


Figure 16: cross over operator

Mutation

Mutation operator aims to create a change in the generated service composition (SC) by adding a security solution service on a random selected point in the SC, the security service are from the same service provider of the selected service in order to eliminate adding a third party risk.

Figure 17 illustrates mutation operator on point 7 by adding security Service SS1 and SS2 Consecutively. Listing 6.4 shows the Pseudo code of mutation operator.

```
Input: offspring and mutation probability, security services list
Output: offspring

    ➢ Mutation point ← Select Mutation point (mutation probability)
    ➢ Security service ← select (security service list)
    ➢ Add security service (offspring, security service, mutation point)
```

Listing 6.4: Procedure Pseudo code of mutation operator.

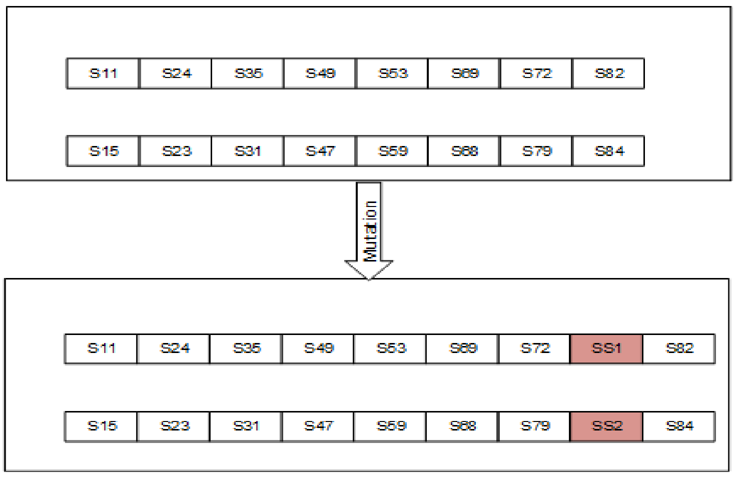


Figure 17*: Mutation operator*

Table 9 listed a group of security services that are used in mutation operator. These services are collected through our research based on security experts, and literature review. Security services (SS) is a countermeasure service for SOA. The table contains the SS id, name, the affect on confidentiality, integrity and availability in addition to the cost and the SS provider name.

ID	SS Name	Confidentiality	Integrity	Availability	cost	provider
SS1	Using Strong passwords	1	1	0	5	X
SS2	Encryption	1	1	0	4	Y
SS3	end idle session lifetime	1	0	-1	7	Z
SS4	Prevent data supplied by the attacker from being executed	0	1	1	8	R
SS5	Prevent return addresses from being overwritten	0	0	1	5	C
SS6	Prevent use of dangerous functions	0	0	1	7	F
SS7	Validate input	0	1	-1	4	R
SS8	Encode output	0	1	-1	2	C
SS9	Restrict the size length and depth of parsed XML messages	0	1	-1	3	X
SS10	deny saving sessions	1	1	0	1	R
SS11	Use least privileged accounts	1	0	0	5	C
SS12	Tie authentication to authorization on the same tier	1	0	0	6	Y
SS13	Consider granularity of access	1	0	0	9	F
SS14	Enforce separation of privileges	1	0	0	1	X
SS15	Use multiple gatekeepers	1	0	0	2	Y
SS16	Secure system resources against system Identities	1	0	0	5	Z
SS17	Identify malicious behaviour	0	0	1	5	R
SS18	Know your baseline Know what good traffic looks like	0	1	0	8	C
SS19	Use application Instrumentation to expose behaviour that Can be monitored	0	1	0	9	X
SS20	Throttle logging	0	1	0	5	Y
SS21	Strip sensitive data before logging	0	1	0	2	X
SS22	Periodically change your keys	1	1	0	2	R
SS23	Use proven platform-provided cryptography	1	1	0	1	F
SS24	Use message security or transport security to encrypt your messages	1	1	0	4	C
SS25	Partition the site by anonymous identified and authenticated users	1	1	0	1	X
SS26	Reduce session timeouts	1	1	0	2	Z
SS27	Avoid storing sensitive data in session stores	1	0	0	5	R
SS28	Secure the channel to the session store	1	1	0	3	C
SS29	Authenticate and authorize access to the session store	1	0	0	6	X
SS30	IPsec	0	1	-1	7	Y

SS31	access control list to deny private IP addresses	1	1	0	8	Z
SS32	Packet filtering	0	1	0	2	R

Table 9: Security Services (SS) List

The Mutation constraint is the SS must be provided from the same provider of the functional service that is allocated in the same location in the SC pattern.

Evaluation and Selection

Evaluation process is applied after performing cross over and mutation operator for each generated chromosome by implementing the fitness function. In GA there is a fitness function for each objective, the generated value from fitness function represent the global value and it indicates the degree of security or cost for the generated composition.

Security is measured using DREAD model and the cost by monetary value.

Using DREAD Model for measuring Service local security values and applying aggregation rule for each business process for calculating SC global QoS for both security and cost are ill-starred as follows:

6.6 Assessing Service security Risk using DREAD model:

DREAD model has been adapted and modified to measure the security risk of each service in the Service composition, DREAD starts with identifying the system assets, decompose the security requirements and specify the security objectives, defining threats and rate it.[41]

Hereunder, adapting the DREAD model is illustrated:

Security objectives and system requirements

In order to evaluate system security, the security objectives of the system should be determined[41]. The following security requirements are specified for the ARS case study and from the requirements the Security objectives are derived.

- R1: User Data is confidential and must be saved and transmitted in a secure channel.
- R2: ARS system data must be saved without any corruption.
- R3: ARS service must be available 24/7.

Requirement	Derived Security objective
R1	Confidentiality
R2	Integrity
R3	Availability

Table 10: Deriving Security objective from the security Requirements

The above requirements are mapped with three security objectives: Confidentiality, Integrity and Availability.

As it is cleared in table 10; the Security objectives areCIA:

- Confidentiality
- Integrity
- Availability

Yautsiukh[42] classify security requirements into three categories based on security objectives as shown in Figure 18 below:

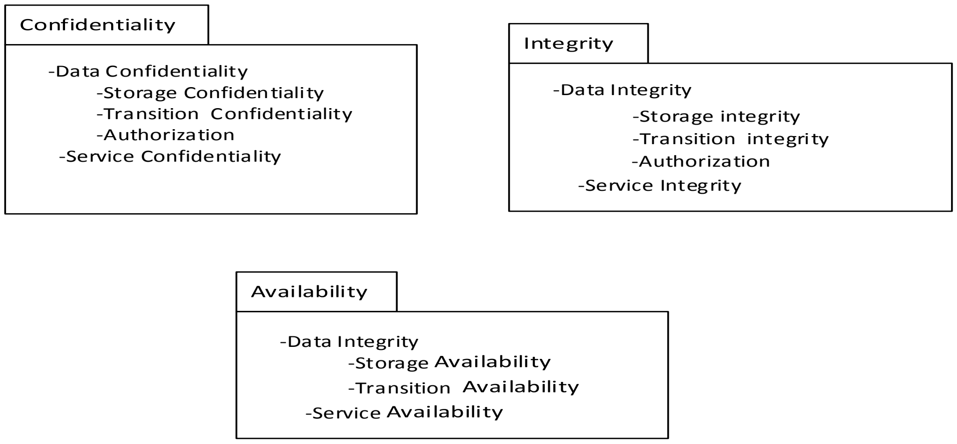


Figure 18: Decomposition of security Objectives

Yautsiukh[42] presents Another way to represent the security requirement, which is drawing the system requirements tree and it has been adapted for ARS system it is shown in figure 19 below:

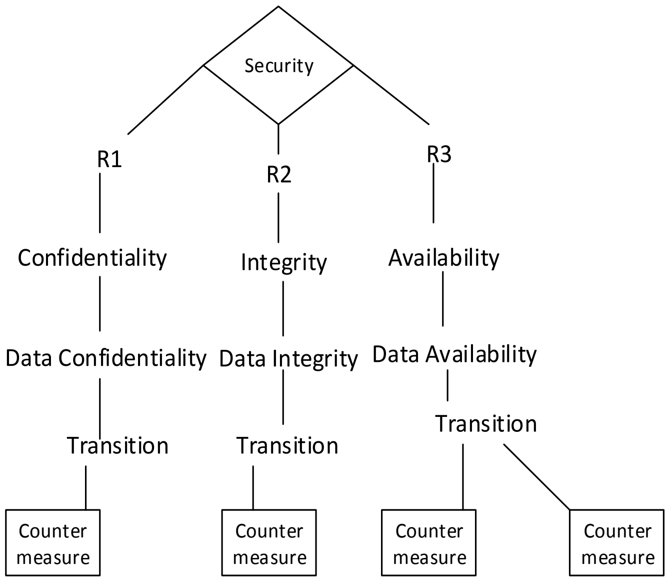


Figure 19: Requirement Tree for ARS

A. Defining the threat List:

Security experts listed well-known threats that affect SOA. These threats are used in our model to eliminate the risk of it; each run of the experiment the user can select a group of threats to reduce their occurrence.

The threats are identified based on the table 11 below which is adapted from[58]:

Threat Id	**Threat Name**	**C effect**	**I effect**	**A effect**
T1	Brute force attacks	10	10	5
T2	Buffer overflows	0	0	10
T3	Canonicalization attacks	10	5	5
T4	Cookie manipulation	10	5	5
T5	Cookie replay attacks	10	5	5
T6	Credential theft	10	5	0
T7	Cross-site scripting	5	5	5
T8	Connection pooling	5	5	5
T9	Data tampering	5	10	5
T10	Denial of service	0	0	10
T11	Dictionary attack	10	5	5
T12	Disclosure of sensitive/confidential data	10	5	0
T13	Elevation of privilege	10	5	5
T14	Encryption	10	0	0
T15	Information disclosure	5	5	5
T16	Luring attacks.	5	10	5
T17	Man-in-the-middle attacks	10	10	0
T18	Network eavesdropping	10	0	0
T19	Password cracking	10	10	5
T20	Repudiation	0	5	10
T21	Session hijacking	10	10	0
T22	Session replay	5	10	0
T23	Session fixation	10	5	0
T24	Spoofing	10	5	5
T25	SQL injection.	10	10	5
T26	Throttling	0	0	10

Table 11: Threat and its effects on CIA[58]

B. Rate Threat

Finding DREAD values for each threat, required taking security requirements in consideration, reading service security attributes values and reading selected threat effect on CIA.

In order to estimate threats values, DREAD model was adapted and modified, by adding new categorizations based on the service attribute values and the threat effect on each security objective.

- **DREAD values are high (9-6) for a certain security goal if:**
    - If the selected threat has 10 value on this security goal.
    - If the service attributes related to this goal are not enabled.
    - The service general attributes score is low.
    - If this goal is strongly required in this service

- **DREAD values are medium (6-3)**
    - If the selected threat has 5 value on this security goal.
    - If the service attributes related to this goal not all enabled.
    - The service general attributes score is medium.
    - If this goal has not strongly required in this service
- **DREAD values are low (3-1)**
    - If the selected threat has 0 value on this security goal.
    - If the service attributes related to this goal are not enabled.
    - The service general attributes score is high.
    - If this goal is not required in this service

After generating DREAD values for each CIA objectives we applied equation 3.1 to find the three severity values

$$\boldsymbol{Rating = D + R + E + A + D/5}$$

Then the Severity of each security objective is calculated for are services in the composition, to calculate the global severity, Aggregation rule is applied in the next section.

Calculating SC Global QoS attributes using Aggregation Rules:

Aggregation rule are applied in each case for both security and cost as follows:

I. Customer identification Sub process (BP1): see figure 8

- ***Confidentiality Severity:***

$\boldsymbol{CS1 = Max(P1 * CS1, P2\ (Max(CS, CS4), CS2)}$ *Equation 6.1-a*

- ***Integrity Severity:***

$\boldsymbol{IS1 = Max(P1 * IS1, P2\ (Max(IS3, IS4), IS2)}$ *Equation 6.1-b*

- ***Availability Severity:***

$\boldsymbol{AS1 = Max(P1 * AS1, P2\ (Max(AS3, AS4), AS2)}$

Equation 6.1-c

- ***Cost:***

$\boldsymbol{COST1 = P1 * COST1 + P2\ (COST3 + COST4) + COST2}$

Equation 6.1-d

Where CS: Confidentiality Severity, IS: Integrity Severity, AS: availability Severity

II. Booking or reserving sub process (BP2): see figure 9

- ***Confidentiality Severity:***

$$\boldsymbol{CS2 = Max(max((P3 * CS5 + P4 * CS6), CS7, CS8, CS9, CS10, CS11, CS12), CS13, CS14))}$$

Equation 6.2-a

- ***Integrity Severity:***

$$IS2 = Max(max((P3 * IS5 + P4 * IS6), IS7, IS8, IS9, IS10, IS11, IS12), IS13, IS14))$$

Equation 6.2-b

- ***Availability Severity:***

$$\boldsymbol{AS2 = Max(max((P3 * AS5 + P4 * AS6), AS7, AS8, AS9, AS10, AS11, AS12), AS13, AS14))}$$

Equation 6.2-c

- ***Cost:***

$$\boldsymbol{COST2 = P3 * COST5 + P4 * COST6 + COST7 + COST8 + COST9 + COST10 + COST11 + COST12)/(1 - P5) + COST13 + COST14}$$

Equation 6.2-d

Where CS: Confidentiality Severity, IS: Integrity Severity, AS: availability Severity

III. Payment sub process (BP3): see figure 10

- ***Confidentiality Severity:***

$$\boldsymbol{CS3 = max(CS15, max(CS16, CS17), CS18, (P10 * max(CS20, CS21)), (p9 * CS19), CS22)}$$

Equation 6.3-a

- ***Integrity Severity:***

$$IS3 = max(IS15, max(IS16, IS17), IS18, (P10 * max(IS20, IS21)), (p9 * IS19), IS22)$$

Equation 6.3-b

- ***Availability Severity:***

$$AS3 = max(AS15, max(AS16, AS17), AS18, (P10 * max(AS20, AS21)), (p9 * AS19), AS22)$$

Equation 6.3-c

- ***Cost:***

$$\boldsymbol{COST3 = COST15 + \frac{COST16+COST17}{1-P7} + COST18 + P9 * COST19 + P10(COST20 + COST21) + COST22}$$

Equation 6.3-d

Where CS: Confidentiality Severity, IS: Integrity Severity, AS: availability Severity

IV. Edit flight sub process (BP4): see figure 11

- ***Confidentiality Severity:***

$$\boldsymbol{CS4 = max(CS23, max(CS24, CS25, CS26, CS27), CS28, CS29, CS30, CS3, CS32)}$$

Equation 6.4-a

- ***Integrity Severity:***

$$IS4 = max(IS23, max(IS24, IS25, IS26, IS27), IS28, IS29, IS30, IS3, IS32)$$

Equation 6.4-b

- ***Availability Severity:***

$$AS4 = max(AS23, max(AS24, AS25, AS26, AS27), AS28, AS29, AS30, AS3, AS32)$$

Equation 6.4-c

- ***Cost:***

$$\boldsymbol{COST4 = (COST29 + max \llbracket (COST24, COST25, COST26, COST27) + COST28 \rrbracket)/(1 - P12) + COST29 + COST30 + COST31 + COST32}$$

Equation 6.4-d

Where CS: Confidentiality Severity, IS: Integrity Severity, AS: availability Severity

V. Whole ARS system (WB): see figure 7

- ***Confidentiality Severity:***

$$\boldsymbol{CSwb = max(CS1, CS2, CS3, CS4)} \qquad \textit{Equation 6.5-a}$$

- ***Integrity Severity:***

$$ISwb = max(IS1, IS2, IS3, IS4) \qquad \textit{Equation 6.5-b}$$

- ***Availability Severity:***

$$ASwb = max(AS1, AS2, AS3, AS3) \qquad \textit{Equation 6.5-c}$$

- ***Cost:***

$$\boldsymbol{COSTwb = COST1 + COST2 + COST3 + COST4} \qquad \textit{Equation 6.5-d}$$

Where CS: Confidentiality Severity, IS: Integrity Severity, AS: availability Severity

For two objectives part we take the rate of Confidentiality Severity, Integrity Severity and availability Severity to calculate Security Risk severity as shown in equation 5.6

$$\boldsymbol{SRS = ((CS + IS + AS)/3\ )} \qquad \textit{Equation 6.6}$$

Where CS: Confidentiality Severity, IS: Integrity Severity, AS: availability Severity

Furthermore the fitness function in our work aims to reduce risk severity and reduce the monetary cost.

Chapter 7

Experiment

7.1 Introduction

This Chapter presents YAFA SOA optimizer tool, which is an application that is developed through this research to run the experiment. YAFA SOA optimizer implements the model that was explained in Chapter 05 and it extends Jmetal framework [59], the incoming sections includes YAFA SOA Optimizer architecture, Running experiment, solution example and experiment results.

7.2 YAFA SOA Optimizer architecture

YAFA SOA Optimizer extends Jmetal framework[59]; which is an object oriented Java based framework, it implements GA and provides a large group of classes that can be used in solving multiple objective optimization problems. Jmetal is built based on code reuse and shared resources principles. [59]

Jmetal is selected because it is an open source library, it has been installed from the site in [53] for free. Furthermore Jmetal is flexible to use, user can edit the code and customize it regarding to the problem requirements. Additionally Jmetal supports multi-objective optimization problem and it has been used before in literature.

YAFA SOA optimizer is built to optimize security and cost in multi-optimization problems for four and two objectives.

YAFA SOA Optimizer extends a GA called NSGA-II to find the optimized SOA candidates for the case study ARS based on cost and security Risk.

YAFA SOA optimizer supports the following features:

- Read input; YAFA SOA optimizer reads Services file, threats file, and security services file.
- Implements NSGA-II algorithm for multi-objectives optimization problem for (4 and 2 objectives) for each business process in the case study.
- YAFA SOA optimizer finds Pareto front files and the related generated service compositions.

YAFA SOA optimizer architecture is shown in figure 22 below:

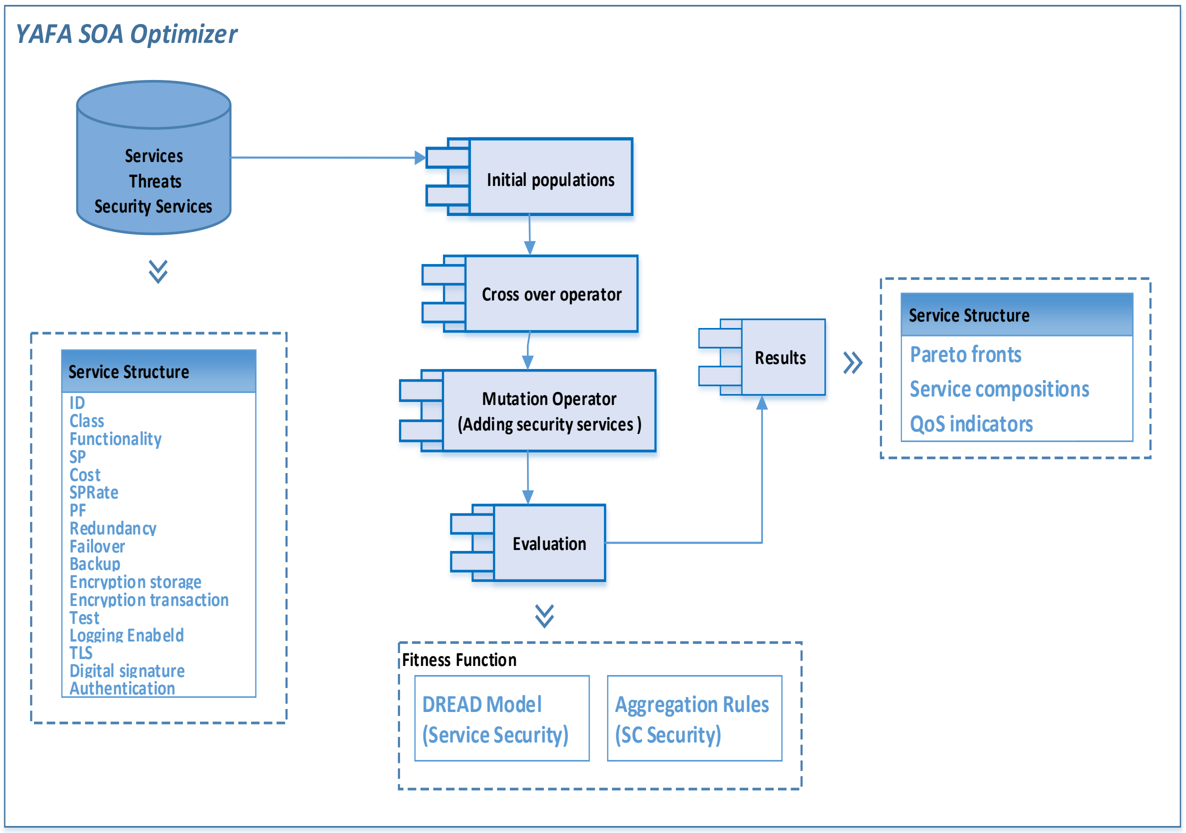


Figure 20: YAFA SOA optimizer architecture

YAFA SOA Optimizer implements Jmetal framework, Jmetal Framework class diagram is shown in figure23, and YAFA SOA Optimizer class diagram is shown in figure 24, in the below section the extending Jmetal Framework by YAFA SOA optimizer is explained:

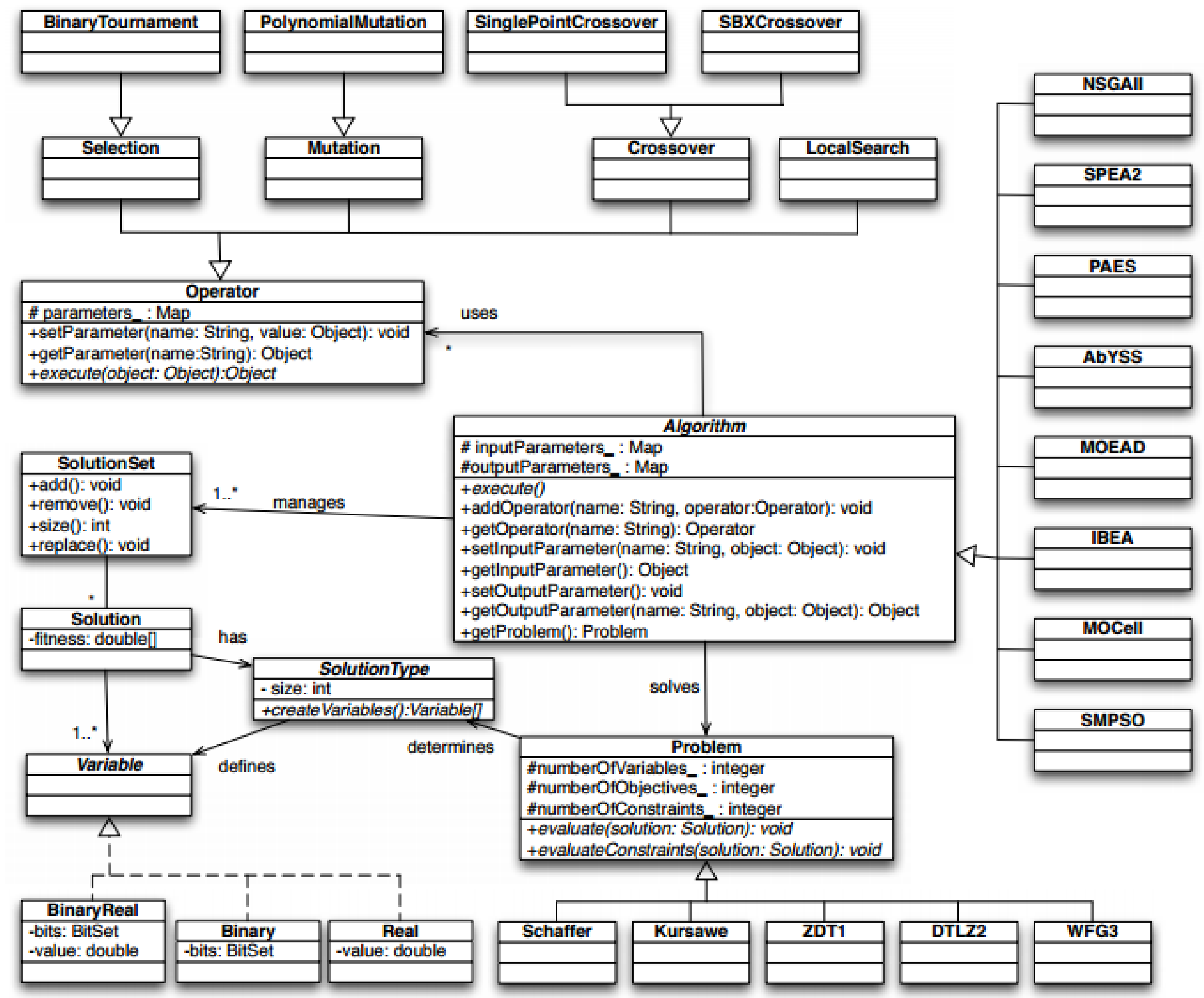

BinaryTournament
PolynomialMutation
SinglePointCrossover
SBXCrossover
Selection
Mutation
Crossover
LocalSearch
Operator
# parameters_ : Map
+setParameter(name: String, value: Object): void
+getParameter(name:String): Object
+execute(object: Object):Object
uses
*
NSGAII
SPEA2
PAES
AbYSS
MOEAD
IBEA
MOCell
SMPSO
Algorithm
# inputParameters_ : Map
#outputParameters_ : Map
+execute()
+addOperator(name: String, operator:Operator): void
+getOperator(name: String): Operator
+setInputParameter(name: String, object: Object): void
+getInputParameter(): Object
+setOutputParameter(): void
+getOutputParameter(name: String, object: Object): Object
+getProblem(): Problem
SolutionSet
+add(): void
+remove(): void
+size(): int
+replace(): void
1..*
manages
*
Solution
-fitness: double[]
has
SolutionType
- size: int
+createVariables():Variable[]
solves
1..*
Variable
defines
determines
Problem
#numberOfVariables_ : integer
#numberOfObjectives_ : integer
#numberOfConstraints_ : integer
+evaluate(solution: Solution): void
+evaluateConstraints(solution: Solution): void
BinaryReal
-bits: BitSet
-value: double
Binary
-bits: BitSet
Real
-value: double
Schaffer
Kursawe
ZDT1
DTLZ2
WFG3

Figure 21: :Jmetal Class diagram [59]

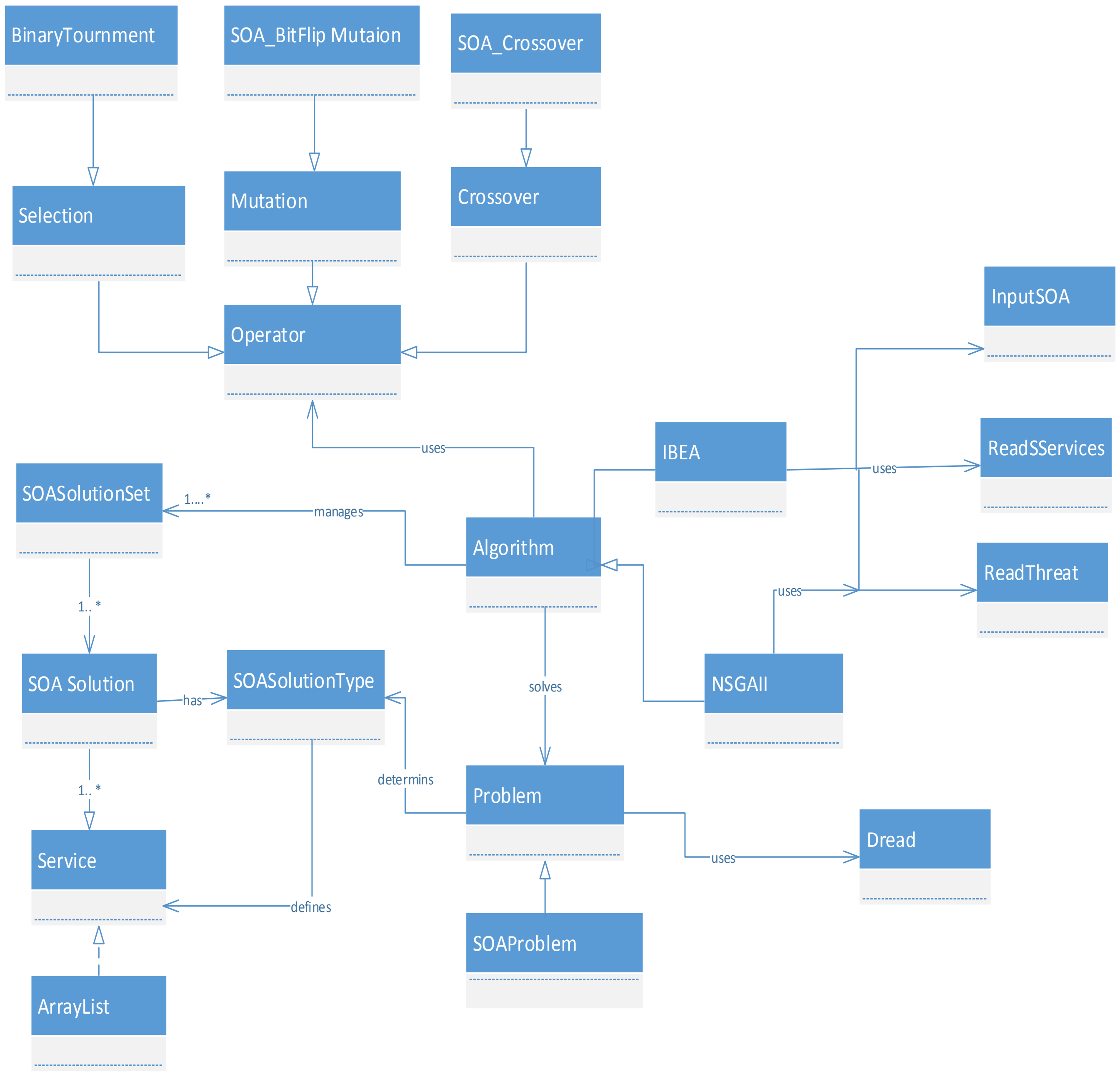


Figure 22: YAFA SOA Optimizer Class diagram

- Problem class from Jmetal framework is extended by "SOA Problem" class and it is edited to evaluate the generated service composition security risk and cost.
- Solution class from JMetal framework is extended by "SOA solution" class and it was customized to define SOA solution, and Solution Type classis extended by SOA Solution Type.

- Crossover class is extended by “SOACrossover” class and it is customized to generate new offspring’s by exchanging 2 parents genes on a randomly selected cross over point.
- Bit Flip Mutation class is extended by “SOA_BitFlipMutation” class in YAFA SOA Optimizer and it is edited to add a security solution from the same service provider at the mutation point in order to reduce risk severity.
- “NSGA-II” class is extended and customized to perform NSGA-II operations.
- “NSGA-II main” class is extended and customized to perform call NSGA-II algorithm and pass variables to it.
- Helper Package include classes that read Services “InputSOA”, “Read Threat”, and “Read Security Services” Classes
- DREAD Class extends DREAD model and calculate services security regarding to its attributes.

YAFA SOA Optimizer workflow

YAFA SOA Optimizer is running to optimize SOA in by the following steps

1- “NSGA-II_Main” class is run and called “INPUTSOA” class.

2- Input SOA Class creates the following:

   a. Service pool; services are classified according to their functionality into classes.

   b. List of security.

   c. List of threats.

3- Basic chromosome is generated randomly by selecting one service from each class. And the initial population is created by “NSGA-II” algorithm.

4- “SOACrossover” class is called to create two new offspring’s.

5- “SOA_BitFlipMutation” class is called to add security services to the generated service composition then the fitness values are stored and the selection process starts
6- The loop keeps running till the termination condition is reached.
7- “NSGA-II_Main” calculates hyper volume quality indicator and the simulation time.
8- And at the end the Pareto front is saved and the feasible service compositions too.

7.3 Running the experiment:

YAFA SOA optimizer implements NSGA-II GA to solve multiple objectives optimization problem. NSGA-II runs ARS case study that was illustrated in Chapter05 before, the experiments is run for each sub process separately and then for the whole system, with different experiment configuration. The purpose of changing the experiment is studying the effectiveness and performance of NSGA-II in each case.

YAFA SOA Optimizer starts simulation by reading the services file, threat file and security services file to create Service pool, threat pool and security service pool, then the initial population is created in size of 100 chromosome, after that the cross over operator is called with probability equals 0.9, and the mutation has probability equals 0.9, both mutation and cross over points are selected randomly.

The number of evaluations equals 25,000 and then it is increased to 100, 1000 evaluations. The experiment has been run 20 times for each sub process using NSGA-II algorithm for both 25,000 and 100,000 evaluations.

The number of concrete services also has been changed, first it starts with 10 concrete services for each abstract service and then using 100 concrete services for each abstract services.

These different runs is performed for different number of objectives; two objectives (Security Risk Severity and cost) and 4 objectives (“CIA” Risk Severity and cost) problems.

The quality indicators are calculated in the experiment for each run the hyper volume, the execution time, the median and the average of the optimization objectives are founded. Table 12 shows GA parameters

Parameter	**Value**
Population Size	100
Max Evaluations	25,000 , 100,000
Crossover Probability	0.9
Mutation Probability	0.9
Number of alternatives	10, 100

Table 12: GA attributes

7.4 Example solutions

This section presents an example of the initial chromosome for BP4[3] and monitors the change in the QoS for the generated Service Composition SC through the experiment run; the example parameter is shown in table 13.

Parameter	**Value**
GA	NSGA-II
Population Size	100
Max Evaluations	25,000
Crossover Probability	0.9
Mutation Probability	0.9
Number of concrete services	10
Business process (BP)	BP4

Table 13:BP4 Running example attributes

3The reason for selecting BP4 for the example is that it contains different type of patterns. (Chapter05)

The initial chromosome is shown in figure 23; and the QoS of the initial SC is shown in table 14 and 15.

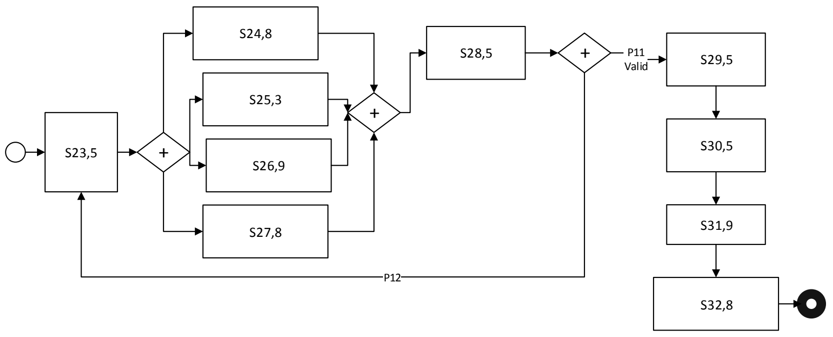


Figure 23: BP4 initial SC example.

QoS	**Value**
Cost	30
Confidentiality severity (CS)	0.454929
Integrity Severity (IS)	0.438723
Availability (AS)	0.428692

Table 14: Initial SC with 4 objectives example.

QoS	**Value**
Cost	30
Security Severity(SS)	0.440781

Table 15 BP4 initial SC with 2 objectives example.

After running the experiment with parameters that are presented in table 13, one of the generated Pareto Fronts is shown in figure 24 and, it shows SOA architecture for BP4 with new architecture and new QoS values.

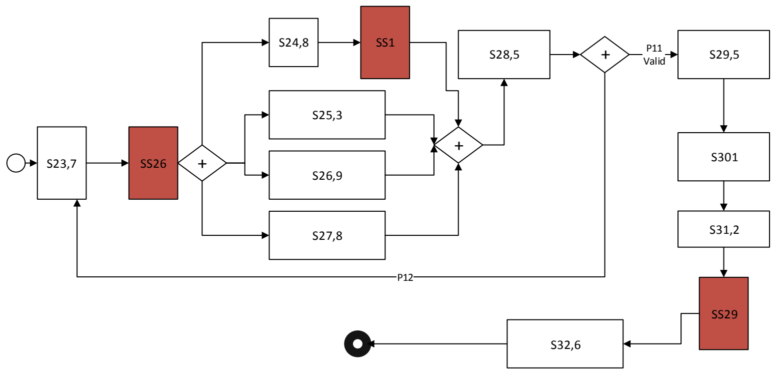


Figure 24: BP4- optimized Service Composition example.

The QoS of the Generated SC for BP4 is shown in table 16and 17.

QoS	**Value**
Cost	49
Confidentiality severity (CS)	0.132723
Integrity Severity (IS)	0.217739
Availability (AS)	0.2275306

Table 16: BP4 optimized SC with 4 objectives example.

QoS	**Value**
Cost	49
Security Severity(SS)	0.192664

Table 17: BP4 optimized SC with 2 objectives example.

It can be noted that 3 Security Services (SS) have been added to the architecture, and each SS is provided from the same provider of the original functional service, furthermore the generated SC has an enhanced security with small increasing in cost.

7.5 Experiment Results

This Section illustrates the results that have been generated through the experiment different runs and the conclusions that can be derived from these results, in this research median of Hyper Volume (HV), Average of the generated objectives and the median of the consumed time have been calculated.

The experiment is run 20 times for all run cases at a Lenovo Think Pad T420 Laptop, 4 GB memory, Windows 7 OS (64 bit), and the possessor is Inter (R) core (TM) i7-2640M, with CPU 2.8 GHz.

Hyper Volume indicator consists of a group of measurements and it is used to evaluate the performance of search algorithms in multi-objective optimization [60]. HV is sensitive to any type of improvements and it measures guarantees that any solution set that achieves the maximum QoS value contains all Pareto-optimal objective vectors[61].

The HV indicator results of our experiment for 25,000 and 100,000 evaluations are presented in tables 18 and 19, the median of HV values is calculated using electronic online calculator [62] for 20 time run for each sub process with different run configurations.

No. concrete services	**10 concrete services**		**100 concrete services**	
No. Objectives	**4 objectives**	**2 objectives**	**4 objectives**	**2 objectives**
BP1	0.335566	0.659704372	0.484440134	0.316518
BP2	0.661825631	0.822967988	0.629232439	0.5571
BP3	0.401097211	0.830508	0.401097211	0.728
BP4	0.123328075	0.62356711	0.5914692625	0.7408
WB	0.042398	0.3884	0.178668234	0.3389

Table 18: HV values for running NSGA-II with 25,000 evaluations.

No. concrete services	**10 concrete services**		**100 concrete services**	
No. Objectives	**4 objectives**	**2 objectives**	**4 objectives**	**2 objectives**
BP1	0.0525	0.4556	0.4641	0.8754
BP2	0.6991	0.8116	0.6826	0.5521
BP3	0.5199	0.6933	0.5799	0.6036
BP4	0.5255	0.6705	0.5393	0.7268
WB	0.0652	0.3172	0.1727	0.569

Table 19: HV values for running NSGA-II with100, 000 evaluations.

The average of the objectives values are shown in table 20 and 21 for all run cases.

	10 concrete services				100concrete services			
	25,000 evaluations.		100, 000 evaluations.		25,000 evaluations.		100, 000 evaluations	
	Security	Cost	Security	Cost	Security	Cost	Security	Cost
BP1	0.24765	3.27163	0.23642	3.87865	0.23792	7.62121	0.23752	6.53591
BP2	0.28015	3.63734	0.28525	3.00523	0.26144	5.59839	0.26004	11.6698
BP3	0.26863	21.6044	0.26359	27.6561	0.26037	32.0383	0.26165	30.3034
BP4	0.28756	27.9471	0.28577	26.276	0.28688	33.9934	0.28567	32.2432
WB	0.26397	77.0267	0.26331	66.7976	0.26456	78.798	0.26331	66.7976

Table 20: objectives average values for running NSGA-II with two objectives.

10 concrete services								
BP	25,000 evaluations.				100, 000 evaluations.			
	C	I	A	Cost	C	I	A	Cost
1	0.31206	0.22667	0.20428	7.47993	0.30958	0.22815	0.20736	7.55854
2	0.32196	0.23442	0.22229	6.80308	0.32209	0.23377	0.22109	6.69969
3	0.31998	0.23218	0.22127	32.9632	0.31964	0.23217	0.22138	36.7353
4	0.32216	0.23481	0.22272	51.2425	0.30902	0.22954	0.19132	28.64273
WB	0.32519	0.23577	0.22951	99.8744	0.32097	0.23251	0.22735	81.8418

Table 21-a: objectives average values for running NSGA-II with four objectives and 10 concrete services.

100 concrete services							
25,000 evaluations.				100, 000 evaluations.			
C	I	A	Cost	C	I	A	Cost
0.31041	0.22651	0.18711	8.79571	0.30793	0.22679	0.18324	8.81855
0.32286	0.23372	0.22204	7.9814	0.32264	0.23344	0.22228	8.0959
0.31963	0.23198	0.2201	36.8649	0.31917	0.23151	0.21817	34.8145
0.32286	0.23375	0.2223	46.8274	0.30737	0.22681	0.17642	9.32793
0.32617	0.23554	0.23093	101.875	0.32536	0.23555	0.23082	96.4221

Table 22-b: objectives average values for running NSGA-II with four objectives100 concrete services.

From the presented results in table 20 and 21-(a and b), it can be concluded that:

- The risk severity values are enhanced that is all CIA severities values is less than 0.5 in all cases.
- The cost value for the 2 objectives case is better than using 4 objectives because the cost is as important as the 3 security objectives together, whereas in 4 objectives the cost is 1 of 4 objectives so it becomes less emphasized in the optimization, and so when the 3 security goals are separated they are improved faster at the expense of the cost objective.
- Increasing number of alt from 10 to 100 will increase the cost value and does not affect the security severities in both (25,000 and 100,000 evaluations), and this leads to the fact that the software engineer must reduce the search space by limiting the alternatives (concrete services) before calling the GA, in order to avoid overwhelming the algorithms with too many alternatives.

- It is noted that a better security can be achieved in a short time based on the note that CIA severities values are almost same for the same Business Process (BP) in different nun attributes.
- It can be noted that increasing the number of evaluations does not affect the results, which means that optimizing two objectives using NSGA-II can be achieved with 25,000 evaluations.

The time indicator is calculated through the experiment run and then the median of the time is calculated for 20 runs using electronic online calculator [62], time indicator is listed on tables 22 and 23

No. concrete Services	**BP length**	**10**		**100**	
No. Objectives	-	**4**	**2**	**4**	**2**
BP1	4	787	758	1014	734
BP2	10	23365	9355	18885	7991
BP3	8	20338	13951	18645	13455
BP4	10	21146	21918	23094	15207
WB	32	66893	344433	44641	379436

Table 23: median of time values for running NSGA-II with25, 000 evaluations.

No. concrete Services	**BP length**	**10**		**100**	
No. Objectives	-	**4**	**2**	**4**	**2**
BP1	4	3219	8571	3516	7251
BP2	10	107859	44472	8932	31671
BP3	8	96995	57103	85346	60330
BP4	10	2112	825956	2067	152207
WB	32	405117	391237	206238	57779

Table 24: median of time values for running NSGA-II with 100, 000 evaluations.

It can be noted from tables 22 and 23 that:

- Increasing the number of evaluations, increases the optimization time in all cases.
- Changing the number of concrete services from 10 to 100 required additional time slots to find the Pareto front.
- It can be concluded that there is a direct proportion between the service composition length (number of abstract services) and the optimization time for example WB requires the highest time and BP4 requires lowest.

Chapter 8

Threats to validity

Validity is defined by Luo, et al in [63] as “The validity of a study denotes the trustworthiness of the results, to what extent the results are true and not biased by the researchers’ subjective point of view” , the result of empirical studies can be biased by deferent factors that may affect the results of the experiment.

This chapter discusses this research threat of validity in all research phases by presenting 4 types of validities, construct, internal, conclusion and external validities [64]:

8.1 Threats to construct validity

Construct validity reflects that there is a bias between the real operational that has been studied and what is investigated in the research questions.[64]

In this research, the fixed GA parameter setting for NSGA-II algorithm can be considered a factor of construct validity. The core objective of this research is finding the optimized SOA based on security and cost and it is not studying GA behaviour by changing Jmetal default parameters. In the future we may work to change GA parameters and see the differences.

Additionally there is no direct method to estimate the security risk, we adapted DREAD model to measure service security instead of relying on service providers information about service security.

There are another threat to construct validity is measurement bias [64]. Measurement bias can be caused by the quality indicators that were used to assess the generated Pareto front. Hyper volume, time, median, averages of objectives are taking in consideration in evaluating Pareto fronts.

8.2 Threats to Internal Validity

Internal validity caused if the researcher neglected the effect of the research factors on each other, if there is a caused relations between research factors that ignored. [64]

In Multi- objectives research problems there is a possibility to have internal threat validity because this research depends on JMetal's implementation and the efficiency of GA can be changed by different implementations.

To eliminate internal validity in this research we divide the security in to 3 factors strength the security and make the security severities values clear for the service requestors, additionally we take in consideration that the added security services in the mutation process may increase the risk of one security goal against others.

8.3 Threats to External and conclusion Validity

This threats related to generalization results, if the findings of the experiment are interested to other people outside the investigated case.[64]

This validity may be caused in this research from the input data that was synthesis data and it was generated based on historical data, security expert's opinion and WSDL information from service registry**.**

Furthermore to reduce the conclusion threats validity; the case study contains all service composition patterns and covers all available cases, we can claim that our results are comprehensive and can be used by software engineers in the field.

Chapter 9

Validation

This chapter aims to verify the research results, we consult a security expert to validate our results and we follow the architecture Trade-off Analysis Method (ATAM) in [36]

ATAM method is used for testing an architecture based on a well-known scenarios based on a certain goals.

In our validation process we select a security expert, then we meet him and illustrate our research goals and asked him to validate a sample of the result. After that we present three service composition (SC) from our generated Pareto Fronts to him.

Our results claim that we enhance the security of these SCs, we asked the security expert to evaluate these SCs based on his experience.

The Evaluation Sheet contains the evaluation Goals, service attributes, Service Compositions, controls, Security requirements, Threats List, and security services.

The evaluation goals contains the main goals confidentiality, integrity and availability, these goals are divided in to sub goals.

Table 1 below shows the Evaluation goals.

Confidentiality	storage Confidentiality
	Transitions Confidentiality
	Authorization
	service Confidentiality
Integrity	storage Integrity
	Transitions Integrity
	Authorization
	service Integrity
Availability	storage Availability
	Transitions Availability
	service Availability

Table 24: Evaluation Goals

Service attributes, threat list and security services are listed in before in the previous Chapters.

The security requirements for BP4 that is used as an example is shown in table 25 below:

Class	Confidentiality	Integrity	Availability
T23	5	5	5
T24	5	5	5
T25	5	5	5
T26	5	5	5
T27	5	5	5
T28	5	5	10
T29	5	10	10
T30	10	10	5
T31	10	10	5
T32	10	10	5
Where availability take in consideration Recovery time objective and Point objective. Scoring: 0-> this security objective is not important for services under this class. 5-> this security objective has important but not critical. 10-> this security objective is very critical and very important.			

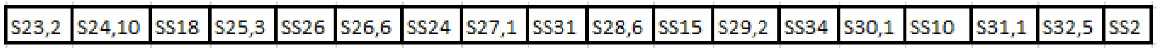
Table 255: the security requirements

The controls table is added by the security expert, he add some controls that may affect his decision in determining the security strength of the given SCs.

Main Security Objective	Controls
Confidentiality	Encryption
	Access Control
	File Privileges if exist
Integrity	Encryption
	Digital Certificate
	Test
Availability	Backup
	Failover
	High availability
	Disaster recovery
	Redundancy
	Test

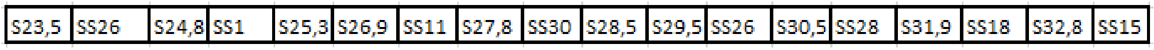
Table 266: controls table (Added by security expert)

Three SCs we selected from the generated Pareto Fronts for BP4 as shown in figure 27-29 and table 27 shows the QoS for the three SCs from the results and the validation results.



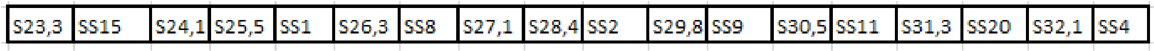
S23,2	S24,10	SS18	S25,3	SS26	S26,6	SS24	S27,1	SS31	S28,6	SS15	S29,2	SS34	S30,1	SS10	S31,1	S32,5	SS2

Figure 25:SC1



S23,5	SS26	S24,8	SS1	S25,3	S26,9	SS11	S27,8	SS30	S28,5	S29,5	SS26	S30,5	SS28	S31,9	SS18	S32,8	SS15

Figure 26:SC2



S23,3	SS15	S24,1	S25,5	SS1	S26,3	SS8	S27,1	S28,4	SS2	S29,8	SS9	S30,5	SS11	S31,3	SS20	S32,1	SS4

Figure 27:SC3

SC	Results (0-1)			Validation Range (0.5-0)		
	C	I	A	C	I	A
SC1	0.2	0.3	0.32	0.5	0.15	0.45
SC2	0.28	0.3	0.35	0.15	0.15	0.45
SC3	0.21	0.38	0.56	0.3	0.15	0.1

Table 277: SCs QoS reslts and validation

In table 27 the Result column presents the Risk for each security Goal that is generated using Aggregation Rules and DREAD model as mentioned before. And the validation column presents the security values of the three security goal that are estimated by the security expert.

Then Results represents risk which mean the lowest is better and validation represent security that means highest is best.

In SC1 it can be noted that confidentiality and integrity values is agreed with the validation values but there is a difference in the Availability value, this difference may be caused by the differences in the used controls and because the security expert studies security without taking cost in consideration.

But in SC2 and SC3 it can be notes that the results are agreed for all security objectives.

The Full evaluation table is added to the appendices of this thesis.

Security expert profile:

We consult Mr Mohammad AbdelRaziq who works as a Information Security Manager at Palestine Telecommunications (Paltel) Group from June 2011 – Present. Where Paltel Group is one of the largest companies at Palestine.

Mr AbdelRaziq has the following certificates in security:

- ISACA: CRISC, Certified in Risk and Information Systems Control in 2011.

- CEH-Ec-Council: CEH, Ethical Hacking in 2006
- CISM-ISACA: CISM, Information Security Management in 2005
- CISA-ISACA: CISA, Information systems Audit in 2004

Mr AbdelRaziq manages and performs audit engagements in Information systems, Telecommunication (mobile and landline), Information Security, and Project Management. And audits engagements include but not limited to: software development, networking, data centres, databases, pre-paid and post-paid billing, interconnect, IN, NGN Services, MSCs, and Mediation. Further More he provides consultation services in Information Systems and Security and project management based on standards and best practices such as: COBIT, PMI Standards, and ISO 27001. He is also Specialties in business analysis, strategic alignment of IT functions, value delivery, application development, project management, and information security in PG companies. [65]

Chapter 10

Conclusions and Future Works

In this thesis, we present the YAFA-SOA optimizer; an optimization tool that implements GA for optimizing SOA to find the optimized Service composition in term of security and cost.

Service security is measured in term of Risk severity. We adapt the DREAD model for Security risk assessment by suggesting new categorizations for calculating DREAD factors based on a proposed service structure and service attributes.

Service composition security risk is measured by implementing the aggregation rules from the local security risk values of the aggregated services in the composition.

The experiment results showed that applying multi-objective GA is feasible to find the optimized security and cost in Service oriented architectures.

We found that adding security services to the generated composition reduces the risk severity of the generated composition and enhances its security in terms of confidentiality, integrity and availability (CIA).

Dividing security to three objectives (CIA) gives the Software Engineer an accurate view for the security measurements and enhances the achievement of security objectives.

The variation of GA Pareto optimal SOA solutions candidate can be considered a break point for starting the services selection process and it helps software engineer to save time and effort in designing SOA.

Running the experiment with 100 number of concrete will give worse solution than running it with 10 concrete; it is concluded that the search space must be reduced by software engineer by eliminating unwanted solutions before running the GA. Dividing the problem in to sub problems reduces the run time of the experiment but it doesn’t affect the Pareto front solutions.

In the future we will work to run the problem by implementing more meta-heuristic search algorithms, and compare the behaviour of these algorithms in deferent parameters input.

Additionally we will work to run the experiment in a larger case study (Enterprise solution) in order to circulate results in SOA.

References

1. Subashini, S. and V. Kavitha, *A survey on security issues in service delivery models of cloud computing.* Journal of Network and Computer Applications, 2011. **34**(1): p. 1-11.
2. Harman, M. and B.F. Jones, *Search-based software engineering*. Information and software Technology, 2001. **43**(14): p. 833-839.
3. Vitti, P.A.F., et al., *Current Issues in Cloud Computing Security and Management.* SECURWARE 2014, 2014: p. 47.
4. Saripalli, P. and B. Walters. *QUIRC: A Quantitative Impact and Risk Assessment Framework for Cloud Security*. in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*. 2010: IEEE.
5. Liu, F., et al., *NIST cloud computing reference architecture.* NIST special publication, 2011. **500**: p. 292.
6. Brian, H., et al., *Cloud computing.* Communications of the ACM, 2008. **51**(7): p. 9-11.
7. Vmware, *Cloud Computing.* 2016.
8. Amazon, *Amazon EC2 - Virtual Server Hosting.* 2016.
9. Microsoft, *The cloud for modern business.* 2016.
10. google, *High-Performance, Scalable VMs- compute engine* 2016.

11. KLEYMAN, B. *Explaining the Community Cloud*. 2014 1/4/2013]; Available from: http://www.datacenterknowledge.com/archives/2014/10/13/explaining-community-cloud/.

12. Buyya, R., C. Vecchiola, and S.T. Selvi, *Mastering Cloud Computing: Foundations and Applications Programming*. 2013: Access Online via Elsevier.

13. Zo, H., D.L. Nazareth, and H.K. Jain, *Security and performance in service-oriented applications: Trading off competing objectives.* Decision Support Systems, 2010. **50**(1): p. 336-346.

14. Newcomer, E. and G. Lomow, *Understanding SOA with web services (independent technology guides)*. 2004: Addison-Wesley Professional.

15. Fareghzadeh, N. *Service identification approach to SOA development*. in *Proceedings of World Academy of Science, Engineering and Technology*. 2008: Citeseer.

16. Thomas, E., *Service-oriented architecture: concepts, technology, and design.* Prentice Hall, 2005. **31**: p. W3C.

17. w3c. *Web Services Architecture*. 1/2/2014]; Available from: https://www.w3.org/TR/ws-arch/.

18. oasis. *SOA-RM*. 2016 1\2\2016]; Available from: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.

19. Beal, V. *Web services*. Available from: http://www.webopedia.com/TERM/W/Web_Services.html.

20. Curbera, F., et al., *Web services platform architecture: SOAP, WSDL, WS-policy, WS-addressing, WS-BPEL, WS-reliable messaging and more*. 2005: Prentice Hall PTR Englewood Cliffs.

21. Hinton, H., M. Hondo, and D.B. Hutchison, *Security patterns within a service-oriented architecture. November 2005*.

22. Russell, N., et al. *Workflow resource patterns: Identification, representation and tool support*. in *Advanced Information Systems Engineering*. 2005: Springer.

23. Karatas, F., L. Fischer, and D. Kesdogan, *Service composition with consideration of interdependent security objectives.* Science of Computer Programming, 2015. **97**: p. 183-201.

24. Jaeger, M.C., G. Rojec-Goldmann, and G. Muhl. *Qos aggregation for web service composition using workflow patterns*. in *Enterprise distributed object computing conference, 2004. EDOC 2004. Proceedings. Eighth IEEE International*. 2004: IEEE.

25. Zeng, L., et al., *QoS-aware middleware for web services composition.* Software Engineering, IEEE Transactions on, 2004. **30**(5): p. 311-327.

26. Lowis, L. and R. Accorsi, *Vulnerability analysis in SOA-based business processes.* Services Computing, IEEE Transactions on, 2011. **4**(3): p. 230-242.

27. Harman, M., et al., *Cloud engineering is search based software engineering too.* Journal of Systems and Software, 2012.

28. Konak, A., D.W. Coit, and A.E. Smith, *Multi-objective optimization using genetic algorithms: A tutorial.* Reliability Engineering & System Safety, 2006. **91**(9): p. 992-1007.

29. Räihä, O., *Applying genetic algorithms in software architecture design.* 2008.

30. Srivastava, P.R. and T.-h. Kim, *Application of genetic algorithm in software testing.* International Journal of software Engineering and its Applications, 2009. **3**(4): p. 87-96.

31. Sharma, C., S. Sabharwal, and R. Sibal, *A survey on software testing techniques using genetic algorithm.* arXiv preprint arXiv:1411.1154, 2014.

32. Deb, K., et al., *A fast and elitist multiobjective genetic algorithm: NSGA-II.* IEEE transactions on evolutionary computation, 2002. **6**(2): p. 182-197.

33. Harman, M., S.A. Mansouri, and Y. Zhang, *Search-based software engineering: Trends, techniques and applications.* ACM Computing Surveys (CSUR), 2012. **45**(1): p. 11.

34. Frey, S., F. Fittkau, and W. Hasselbring. *Search-based genetic optimization for deployment and reconfiguration of software in the cloud*. in *Proceedings of the 2013 International Conference on Software Engineering*. 2013: IEEE Press.

35. Erl, T., *SOA Design Patterns* 2009.

36. Alkussayer, A. and W. Allen. *A scenario-based framework for the security evaluation of software architecture*. in *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*. 2010: IEEE.

37. Zevin, S., *Standards for security categorization of federal information and information systems*. 2009: DIANE Publishing.

38. Wolter, C., M. Menzel, and C. Meinel. *Modelling Security Goals in Business Processes*. in *Modellierung*. 2008.

39. Microsoft. *Threat Modelling* [cited 2014; Available from: https://msdn.microsoft.com/en-us/library/aa302419.aspx#c03618429_011.

40. OWASP. *Threat Risk Modeling*

8 March 2015 [cited 2015; Available from: https://www.owasp.org/index.php/Threat_Risk_Modeling.

41. Viduto, V., et al., *A novel risk assessment and optimisation model for a multi-objective network security countermeasure selection problem.* Decision Support Systems, 2012. **53**(3): p. 599-610.

42. Yautsiukhin, A., et al. *Towards a quantitative assessment of security in software architectures*. in *13th Nordic Workshop on Secure IT Systems, Copenhagen, Denmark*. 2008.

43. de Gramatica, M., et al., *The Role of Catalogues of Threats and Security Controls in Security Risk Assessment: An Empirical Study with ATM Professionals*, in *Requirements Engineering: Foundation for Software Quality*. 2015, Springer. p. 98-114.

44. Microsoft. [cited 2015; Available from: https://msdn.microsoft.com/en-us/library/ff648644.aspx.

45. Ivanovic, D., P. Kaowichakorn, and M. Carro, *Towards QoS Prediction Based on Composition Structure Analysis and Probabilistic Environment Models.*

46. Leitner, P., W. Hummer, and S. Dustdar, *Cost-based optimization of service compositions*. 2011.

47. Sun, P. *Optimal selection of composite web service based on QoS and risk evaluation*. in *E-Business and E-Government (ICEE), 2011 International Conference on*. 2011: IEEE.

48. Kolesnikov, S.S., et al. *Predicting quality attributes of software product lines using software and network measures and sampling*. in *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*. 2013: ACM.

49. Ye, Z., X. Zhou, and A. Bouguettaya. *Genetic algorithm based QoS-aware service compositions in cloud computing*. in *Database Systems for Advanced Applications*. 2011: Springer.

50. Canfora, G., et al. *An approach for QoS-aware service composition based on genetic algorithms*. in *Proceedings of the 7th annual conference on Genetic and evolutionary computation*. 2005: ACM.

51. Leitner, P., W. Hummer, and S. Dustdar, *Cost-based optimization of service compositions*. Services Computing, IEEE Transactions on, 2013. **6**(2): p. 239-251.

52. bizagi. *transactional process* 2014 [cited 2015; Available from: http://www.bizagi.com/processcentral/Documents/df236880-7e8e-49b8-9ffe-f84ac8483f2f/docs/Transactional%20Process%20-%20Description.pdf.

53. Jmetal. 3/1/2014]; Available from: http://jmetal.sourceforge.net.

54. Model, B.P., *Notation (BPMN) version 2.0.* OMG Specification, Object Management Group, 2011.

55. Ericson, D., et al., *Airline Reservation System.*

56. A.ALANAZI, *Flight Reservation System.* 2011.

57. oracle. *Fusion Middleware Security and Administrator's Guide for Web Services*. Sep, 10, 2015]; Available from: https://docs.oracle.com/cd/E17904_01/web.1111/b32511/intro_security.htm #WSSEC2343.

58. Microsoft. *Threats and Countermeasures for Web Services*. 14/9/2015]; Available from: https://msdn.microsoft.com/en-us/library/ff650168.aspx.

59. Nebro, A.J. and J.J. Durillo, *jMetal 4.5 User Manual*. 2014.

60. Auger, A., et al. *Theory of the hypervolume indicator: optimal μ-distributions and the choice of the reference point*. in *Proceedings of the tenth ACM SIGEVO workshop on Foundations of genetic algorithms*. 2009: ACM.

61. Zitzler, E., D. Brockhoff, and L. Thiele. *The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration*. in *Evolutionary multi-criterion optimization*. 2007: Springer.

62. alcula. *Statistics Calculator: Median*. 1/3/2016]; Available from: http://www.alcula.com/calculators/statistics/median/.

63. Luo, X., et al., *Examining multi-dimensional trust and multi-faceted risk in initial acceptance of emerging technologies: An empirical study of mobile banking services*. Decision Support Systems, 2010. **49**(2): p. 222-234.

64. Yu, C.-h. and B. Ohlund, *Threats to validity of research design*. Retrieved January, 2010. **12**: p. 2012.

65. *Mohammad AbdelRaziq linkedin profile*. 2016; Available from: https://www.linkedin.com/in/mohammad-abdelraziq-hasan-msis-cism-cisa-ceh-pmp-crisc-9704498?trk=hb_ntf_ACCEPTED_YOUR_CONNECTION_REQUEST.

Appendices

								SC										
SC1	**S23,2**	**S24,10**	**SS18**	**S25,3**	**SS26**	**S26,6**	**SS24**	**S27,1**	**SS31**	**S28,6**	**SS15**	**S29,2**	**SS34**	**S30,1**	**SS10**	**S31,1**	**S32,5**	**SS2**
Service Id	S23,2	S24,10	SS18	S25,3	SS26	S26,6	SS24	S27,1	SS31	S28,6	SS15	S29,2	SS34	S30,1	SS10	S31,1	S32,5	SS2
Service Class	T23	T24	Know_yo	T25	Reduce	T26	Use_m	T27	access_co	T28	securit	T29	Prevent	T30	securit	T31	T32	Encryp
Service Functionality	select_fligh	change_fligh	0	change_fl	1	change	1	check_flig	1	check_flig	Use_mul	confirm_c	0	get_dif	deny_sa	Pay_In	Send_	1
Service Provide	X	C	1	Z	1	C	1	Z	1	Y	1	X	1	R	1	Y	Y	1
Service Cost	9	9	0	9	0	8	0	10	0	7	0	9	1	9	1	10	10	0
Service Provide Rate	2	1	8	4	2	1	4	4	8	4	0	2	8	1	0	3	3	4
Packet filtering	1	0	C	1	Z	1	C	1	z	0	2	0	R	1	1	0	0	Y
Redundancy	1	1		1		0		1		1	Y	1		1	R	1	1	R
Failover History	0.13	0.15		0.14		0.04		0.34		0.26		0.03		0.18		0.23	0.23	
Backup	1	1		1		1		1		1		0		1		1	1	
Encryption Storage	1	1		1		1		1		1		0		1		1	1	
Encryption Transaction	0	1		0		1		0		1		1		1		1	1	
Tested	1	1		0		1		0		1		1		0		1	1	
Logging Enabled	1	0		1		1		1		1		1		1		1	1	
TLS	1	1		0		1		1		1		0		1		1	0	
Digital Signature	1	0		0		1		1		1		0		0		1	1	
Authentication	1	1		1		0		1		0		1		1		1	0	
SC2	**S23,5**	**SS26**	**S24,8**	**SS1**	**S25,3**	**S26,9**	**SS11**	**S27,8**	**SS30**	**S28,5**	**S29,5**	**SS26**	**S30,5**	**SS28**	**S31,9**	**SS18**	**S32,8**	**SS15**
Service Id	S23,5	SS26	S24,8	SS1	S25,3	S26,9	SS11	S27,8	SS30	S28,5	S29,5	SS26	S30,5	SS28	S31,9	SS18	S32,8	SS15
Service Class	T23	Reduce_ses	T24	Using_Str	T25	T26	Use_leas	T27	IPsec	T28	T29	Reduce_s	T30	Secure_th	T31	Know_yo	T32	Use_m
Service Functionality	select_fligh	1	change_fligh	1	change_f	change_fl	1	check_flig	0	check_flig	confirm_	1	get_differ	1	Pay_Invo	0	Send_co	1
Service Provide	Z	1	X	1	Z	C	0	Y	1	X	Z	1	C	1	C	1	R	0
Service Cost	1	0	6	0	9	3	0	3	-1	7	3	0	1	0	5	0	1	0
Service Provide Rate	3	2	2	5	4	1	5	3	7	2	4	2	1	3	1	8	1	2
Packet filtering	1	Z	1	X	1	0	C	0	Y	0	0	Z	1	C	0	C	1	Y
Redundancy	1		1		1	1		1		1	0		0		1		0	
Failover History	0.08		0.27		0.14	0.21		0.06		0.15	0.04		0.02		0.22		0.02	
Backup	0		1		1	1		1		1	1		0		1		1	
Encryption Storage	1		1		1	1		1		1	1		0		1		1	
Encryption Transaction	1		1		1	1		1		1	1		1		0		1	
Tested	0		1		1	1		0		1	0		0		1		1	
Logging Enabled	1		1		1	0		1		0	1		0		1		1	
TLS	1		0		1	1		1		1	1		1		1		1	
Digital Signature	0		0		1	1		0		1	0		0		1		1	
Authentication	0		1		1	0		1		1	0		1		1		0	
SC3	**S23,3**	**SS15**	**S24,1**	**S25,5**	**SS1**	**S26,3**	**SS8**	**S27,1**	**S28,4**	**SS2**	**S29,8**	**SS9**	**S30,5**	**SS11**	**S31,3**	**SS20**	**S32,1**	**SS4**
Service Id	S23,3	SS15	S24,1	S25,5	SS1	S26,3	SS8	S27,1	S28,4	SS2	S29,8	SS9	S30,5	SS11	S31,3	SS20	S32,1	SS4
Service Class	T23	Use_multipl	T24	T25	Using_Str	T26	Encode_o	T27	T28	Encryptio	T29	Restrict_t	T30	Use_leas	T31	Throttle_	T32	Prevent_
Service Functionality	select_fligh	1	change_fligh	change_fl	1	change_f	0	check_flig	check_flig	1	confirm_	0	get_dif	1	Pay_Invo	0	Send_	0
Service Provide	Y	0	F	X	1	Z	1	Z	Y	1	X	1	C	0	Y	1	R	1
Service Cost	1	0	1	1	0	3	-1	10	1	0	1	-1	1	0	4	0	1	1
Service Provide Rate	3	2	5	2	5	4	2	4	3	4	2	3	1	5	3	5	1	8
Packet filtering	1	Y	1	0	X	0	C	1	1	Y	1	X	1	C	0	Y	1	R
Redundancy	1		1	1		1		1	1		0		1		0		0	
Failover History	0.23		0.11	0.45		0.25		0.34	0.44		0.22		0.12		0.07		0.06	
Backup	1		1	1		0		1	1		0		0		1		0	
Encryption Storage	1		1	0		1		1	0		0		1		0		1	
Encryption Transaction	1		1	1		0		1	1		0		1		1		1	
Tested	1		0	0		0		1	1		1		1		1		0	
Logging Enabled	1		1	1		1		1	1		0		1		0		1	
TLS	1		1	0		0		1	0		1		1		1		1	
Digital Signature	1		1	1		1		1	1		1		1		0		1	
Authentication	0		1	0		1		1	0		1		0		1		1	