Analysis – An Extended Practice

CmpE 202-01

Spring 2018

Professor:

Mohamed E. Fayad <m.fayad@sjsu.edu>

Students:

Shreya Shah <Shreya.shah@sjsu.edu>

Dhruthi Kuram <kuramdhru@gmail.com>

Abstract

Analysis is defined as a detailed examination of anything complex to understand its nature or to determine its essential features: a thorough study. To analyze means to break a topic or concept down into its parts to inspect and understand it, and to restructure those parts in a way that makes sense to you. Analysis is a concept it does not have beginning or end.

Goal of Analysis is **Understanding**- A logical basis for a course of action. In addition, it is an Enduring Business Theme (EBT) as well as Business Object (BO). Any person or a group of persons (Any party) (Any Actor) can do analysis on any scenario or even can customize the scenario with further understanding and analyzing the scenarios by adding or removing any methodologies (Any Mechanism) defined by them. Analysis is done on any general situation (Any Context) considering the problem statement.

The main motivation of this paper is **to design and create a stable architecture by using the principle of software stability model with the comparative study of traditional model**. A traditional pattern is time variant, unstable and difficult to use which eventually makes it impossible for an extended use. The main objective for this paper is to give a stable pattern on the analysis that are more flexible and practical. Major benefit of Analysis is that it allows the reader to understand your point of view, but also to be able to form their own perspective and opinion about your ideas.

Keyword: Comparative Study, Theme, Scenarios, Traditional Model, Stable Model (EBT and BOs) Core Knowledge, Non-Functional Requirements, Qualitative and Quantitative Measurements, Analysis, Understanding, Context

1.0 Introduction

The term analysis can be defined in a very wide range of context. Every day we are bombarded with problems to solve and decisions to make. Everyone in the world must have applied Analysis once in their lifetime directly or indirectly. Analysis depends on how people use the changes to make the life and culture better. It is a form of expository writing in which the writer separates a subject into its elements or parts. It involves a careful examination and evaluation of details in the text. To analyze, one need to examine the nature and structure of something, especially by separating it into parts, to understand and explain, the job involves gathering and analyzing data. The first step of any situation is to define and analyze the problem, the second is to analyze somebody = psychoanalyze. When analyzing a scenario, we will need to consider elements such as context, setting, characters, plot, literary devices and themes. Analyzing information is all about checking sources and accuracy of information. With rapidly changing world that we live in, to remain relevant, requires professionals to learn continuously. One cannot attain knowledge without understanding and knowledge is useless if you cannot apply it to methods of analysis.

2.0 Context and Scenarios

2.1 Context

The context for this project involves Right Information at Right Time. This follows the theme that is "Analysis is an extended practice". This includes anything involving actors, legal actors, region or group of people indulge in benefactor to others. An example Scenario can be a group of students working on a research paper assisting professor having a huge executional issue.

2.2 Scenarios

2.2.1 Scenario 1 – Analysis of new intern joining a IBM for summer internship

A new intern named John enters into the Tech Company like IBM with new high dreams and good enhanced skillset of technologies like JAVA, SQL works on a project which impresses the Manager with the big data analysis report and expects to get offered for a full-time opportunity and running in the race with the other interns to perform their best. There is always an amount of work, your talent and dedication in work which brings you the success. Also, other lot of factors which matters to get you on the good position in any company. In such service oriented company, the hiring process is long and there even lot of layoffs in the company every year when there are drop of projects in the company.

2.2.2 Scenario 2 – Robbery Case at San Jose State University

A student of 3rd Semester named Charles studying in San Jose State University suffering from mental health issues often with that state of mind attempts to rob a female walking on the street by steeling her phone. University Police Officer Mr. James investigates the matter, they felt the state of alert and announced to the university to be alerted of their stuffs and interrogate the suspect and does further investigation. This caused a state of havoc in the nearby surrounding between the students. But with further studies on this the UPD resolves the matter of fact and ask the university to set up a free mental checkup session for such students suffering from such illness.

2.2.3 Scenario 3 – Customer Satisfaction Issue with an employee working in McDonalds.

Customer named Mr. Smith comes to McDonalds and order for Mc Cheese Burger but gets it delayed and the burger served was burnt. He complains to the Manager Mr. Robert about the unsatisfactory service provided from the employee. Manager calls the employee and asks for the explanation which in normal scenario every person will do. After checking in the computer and seeing that this was the first complain registered for this employee and hearing the explanation, the manager excuses the employee giving the first warning and ask him to be alerted while on duty. In such customer to business service system, the acquisition of customer's knowledge specially understanding these

customers including their needs, aims and expectation. Therefore, we need a system that facilitates the proper understanding of the customer's knowledge.

3.0 Traditional and Stable Models

3.1 Traditional Model

3.1.1 Description

A traditional class diagram is an illustration of the relationships and dependencies among classes in the Unified Modeling Language (UML). Here, a class defines the methods and variables in an object, which is a specific entity in a program representing that entity. Traditional class diagrams are useful in all forms of object-oriented programming (OOP). In a traditional class diagram, the classes are not arranged in any single format but are connected by the attributes and define attributes and types of association. A traditional class diagram is shown in **Figure 3.1.3a** which describes the different situations of new intern, customer dissatisfaction at McDonalds. The classes can correlate with each other based on associations and there needs to be a logical flow to the diagram. However, the biggest issue with a traditional class diagram is that it is not feasible to be implemented in the real-world software systems as it has very extensibility and applicability.

3.1.2 Scenario 1 Model

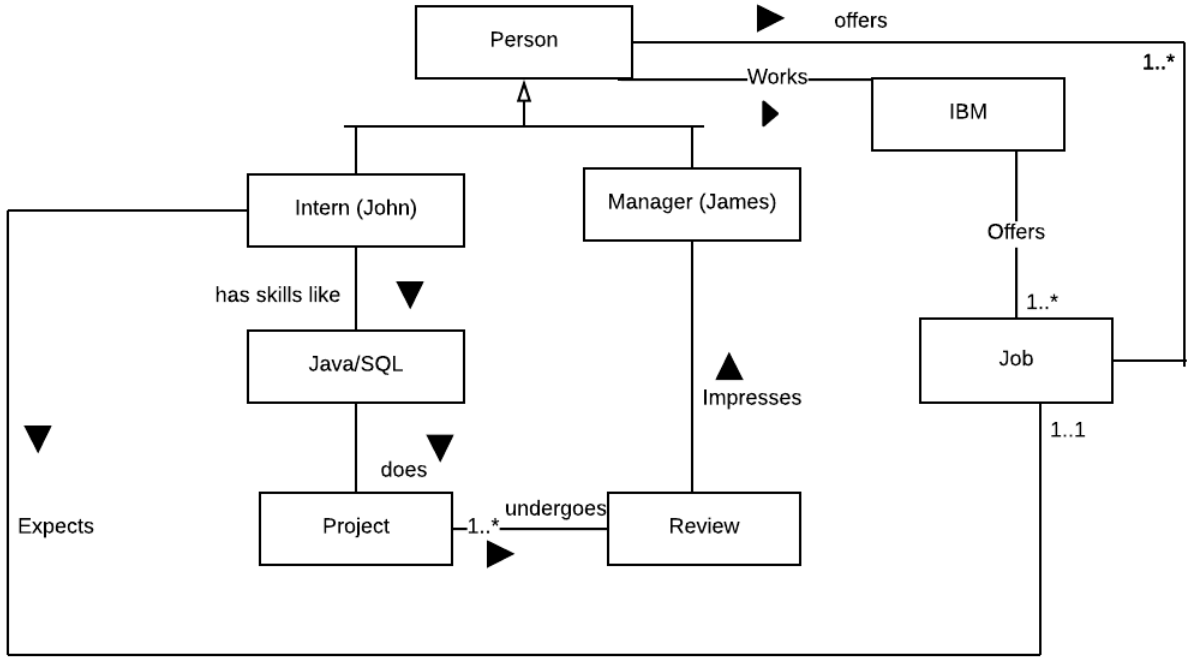

Figure 3.1.2a. Traditional Model for New Interns Entering MNC like IBM

3.1.3 Scenario 2 Model

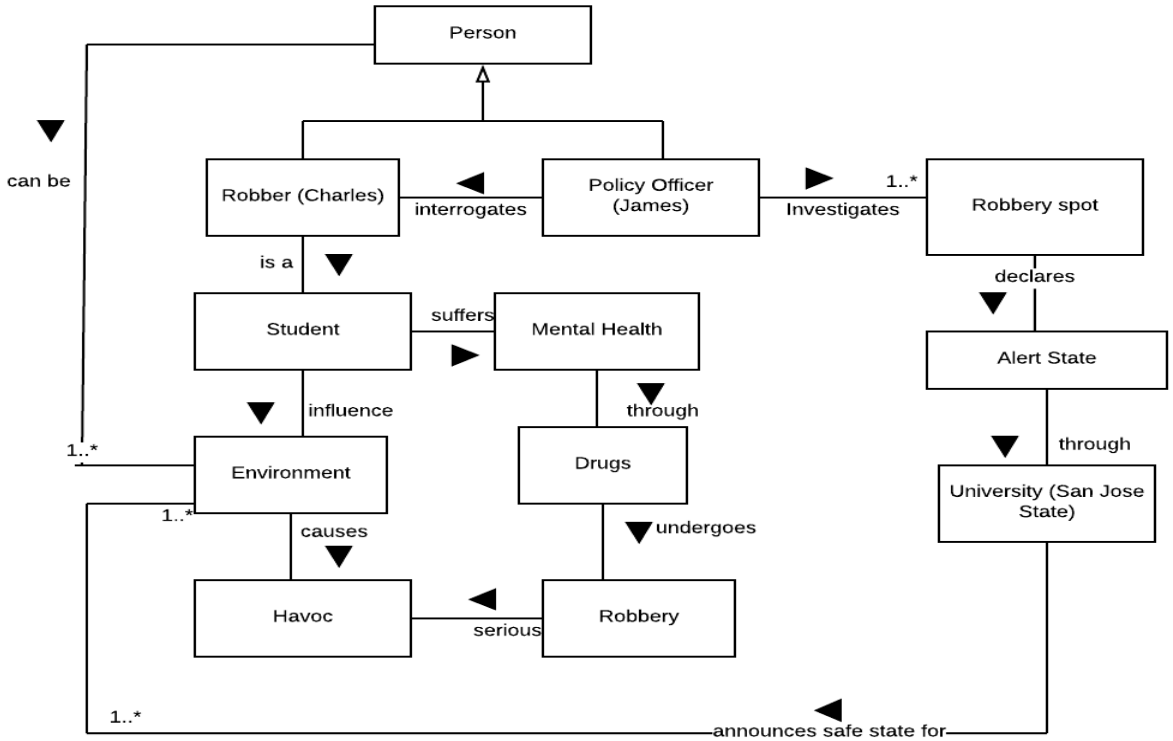

Figure 3.1.3a. Traditional Model Robbery Case at San Jose State University

3.1.4 Scenario 3 Model

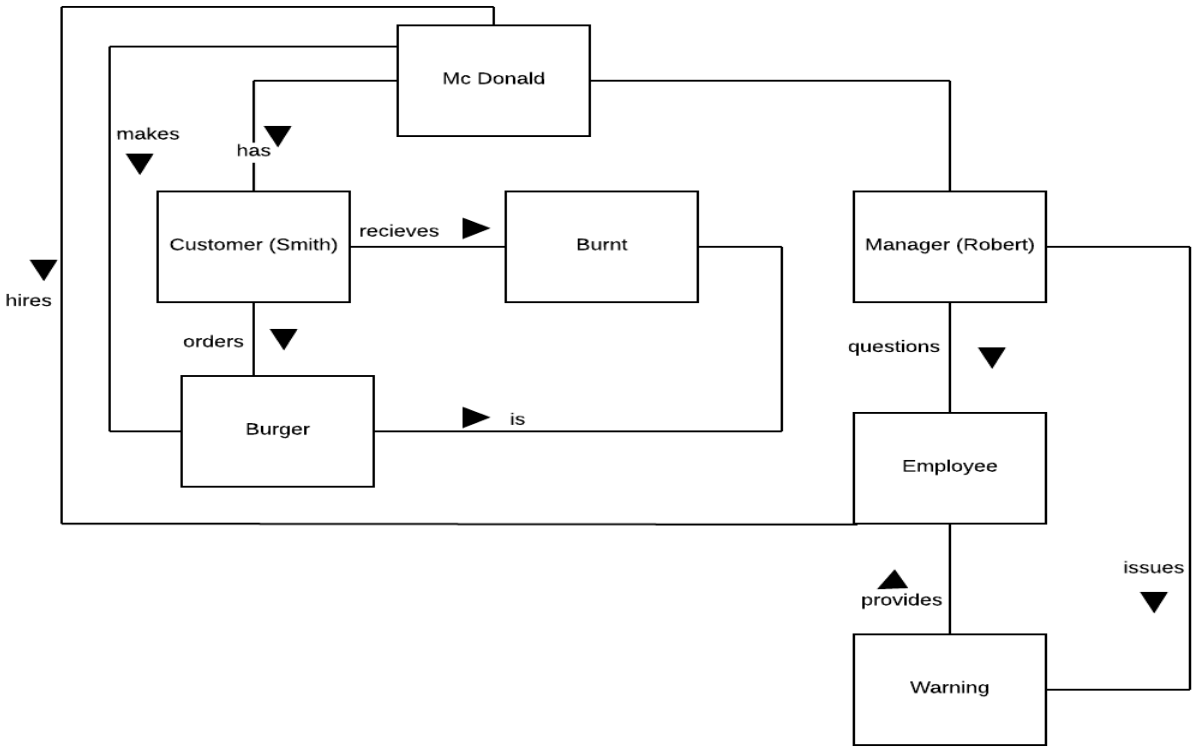

Figure 3.1.4a. Traditional Model for Customer Dissatisfaction at McDonalds

3.2 Stable Model

3.2.1 Description

The primary objective of Software Stability Modeling (SSM), is to develop a model that is reusable across a multitude of applications and retards it obsolescence. Properly constructed, such a model will maintain its core functionality indefinitely. To make this incredible feat reality, we distill the concepts, such as stress in this case, that we are looking at into their base components. There will always be at least one element or overarching objective that is timeless and intangible, which we will call our Enduring Business Theme (EBT). Around the core EBT, we add classes specific to the concept, though these also remain intangible. These are called Business Objects (BO). With these two, the model becomes a pattern to which classes representing physical objects can be "hooked" onto. These tangible objects, called Industrial Objects (IO), are not included as a part of the pattern because they are generated on an as needed basis depending on the context of the software application being designed. The illustration gives a graphical representation of this description.

3.2.2 Model Core

The following figure shows the stable design pattern for analysis:

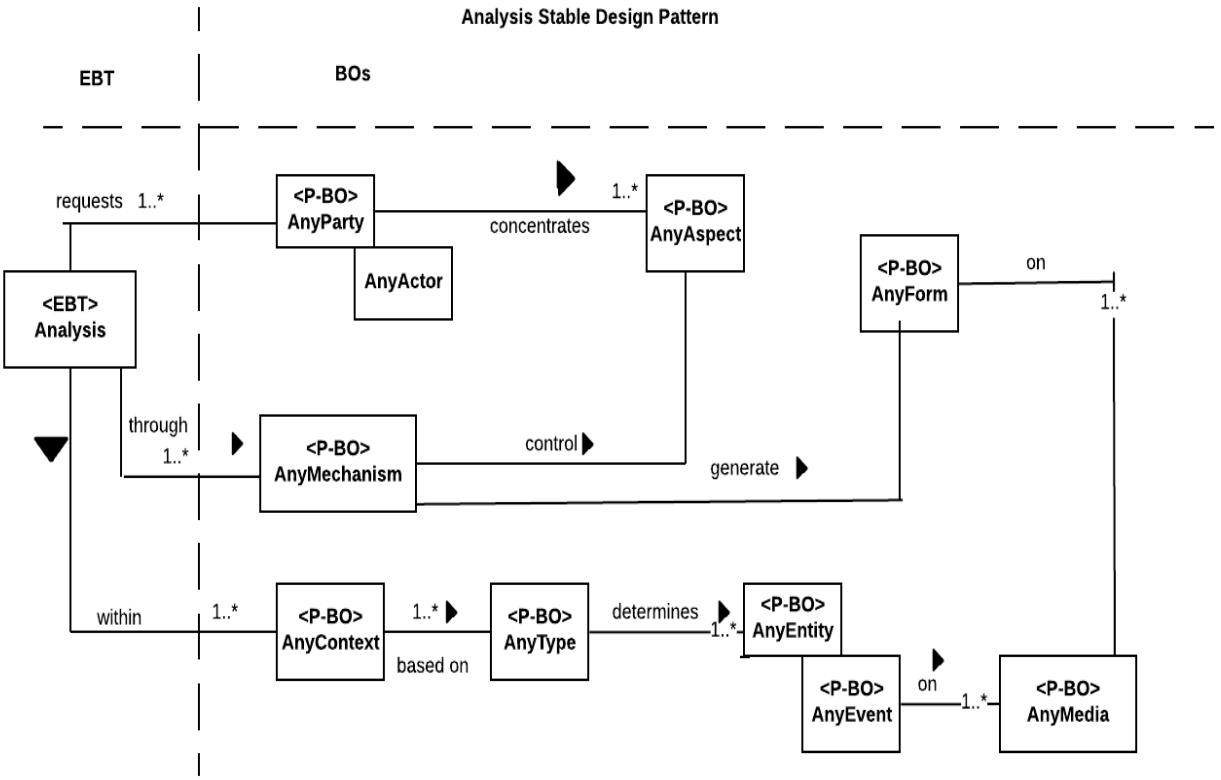

Figure 3.2.2a. Core Knowledge for Analysis

3.3 Stable Model Application

3.3.1 Scenario 1 - Analysis of new intern joining a IBM for summer internship

The following figure illustrates the model for the scenario described in section 2.1.

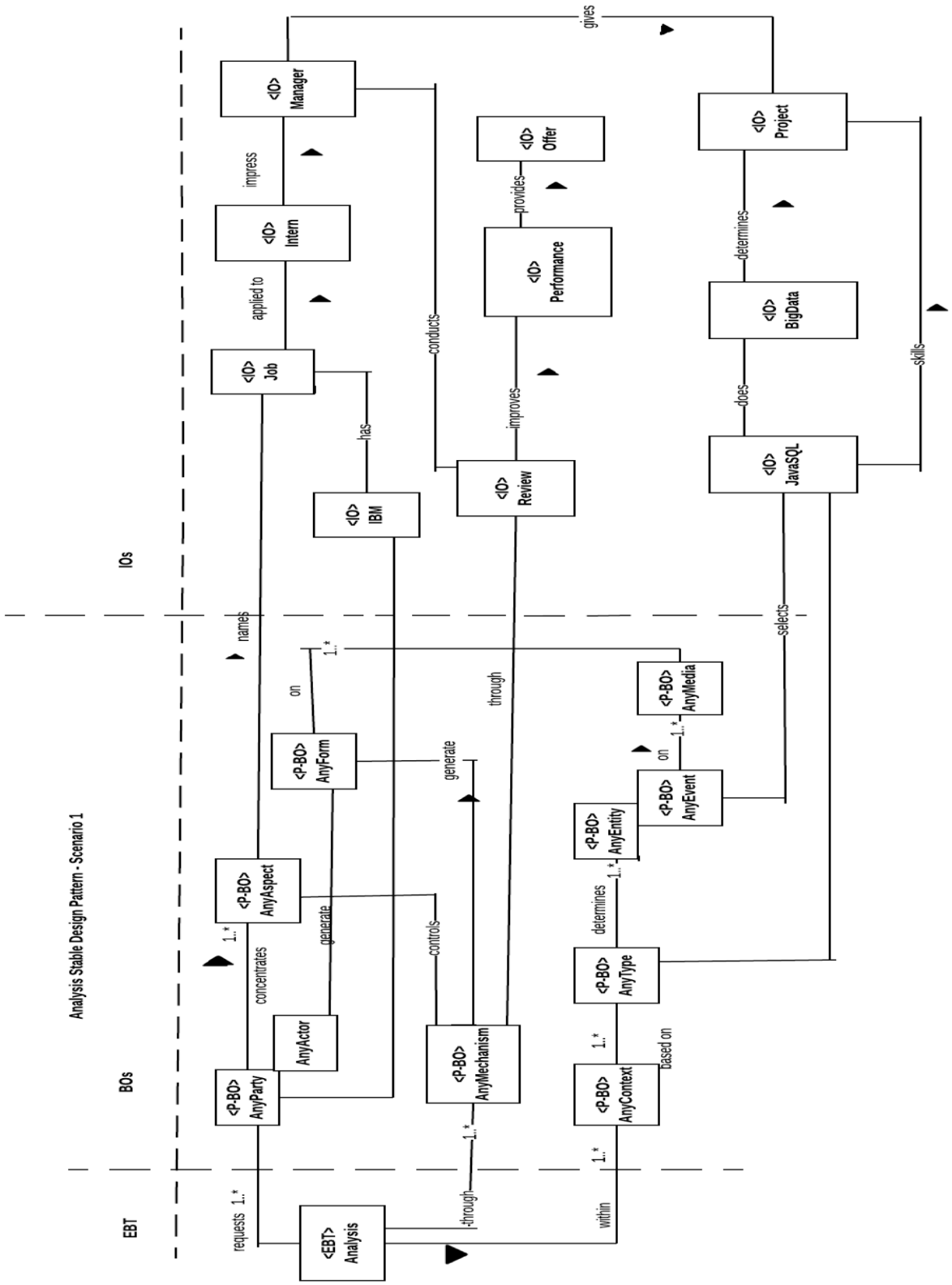

Figure 3.3.1a. Stable Model connected to IOs for scenario 1

3.3.2. Scenario 2 - Robbery Case at San Jose State University

The following figure illustrates the model for the scenario described in section 2.2.

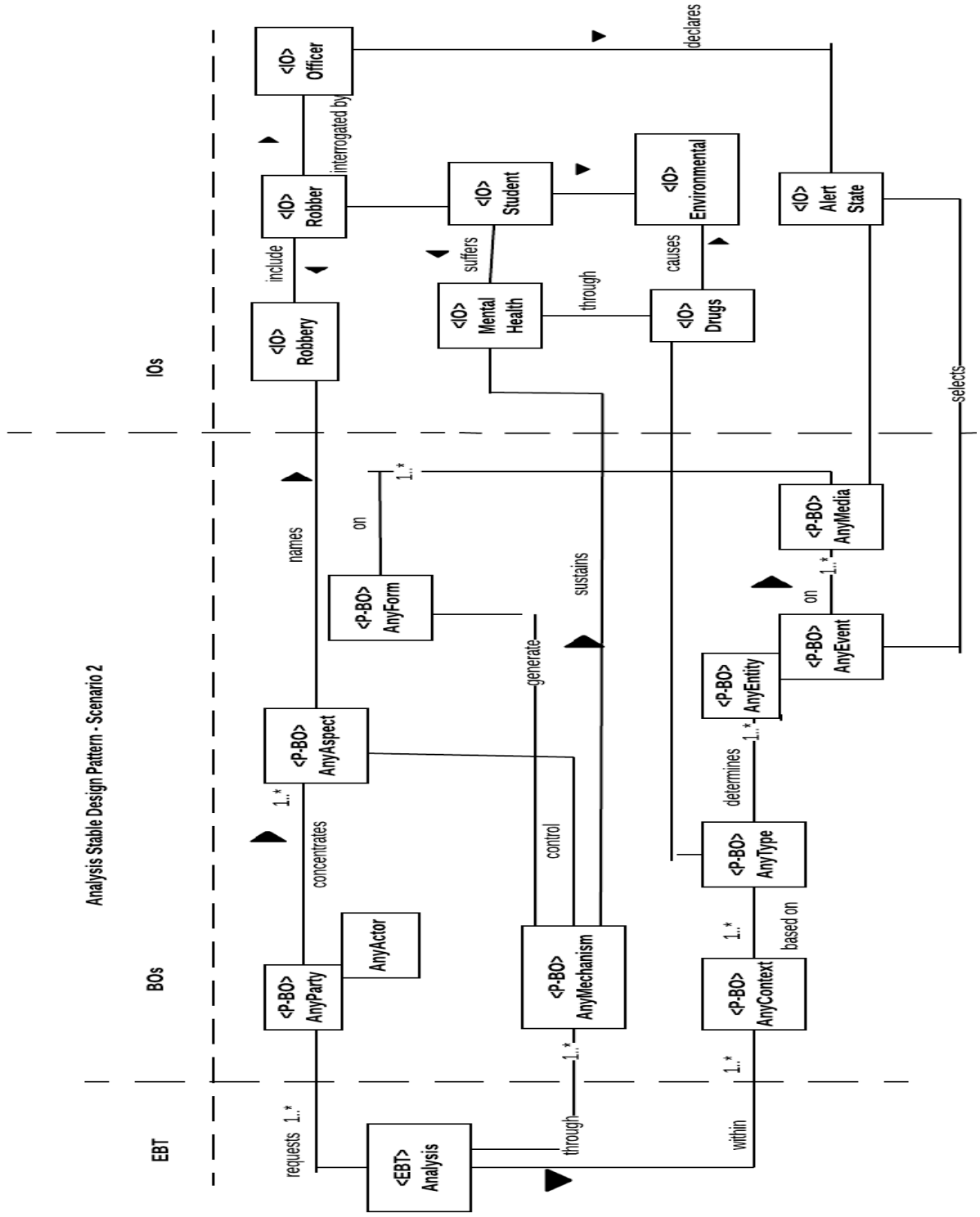

Figure 3.3.2a. Stable Model connected to the IOs for scenario 2

3.3.3 Scenario 3 - Customer Satisfaction Issue with an employee working in McDonalds.

The following figure illustrates the model for the scenario described in section 2.3

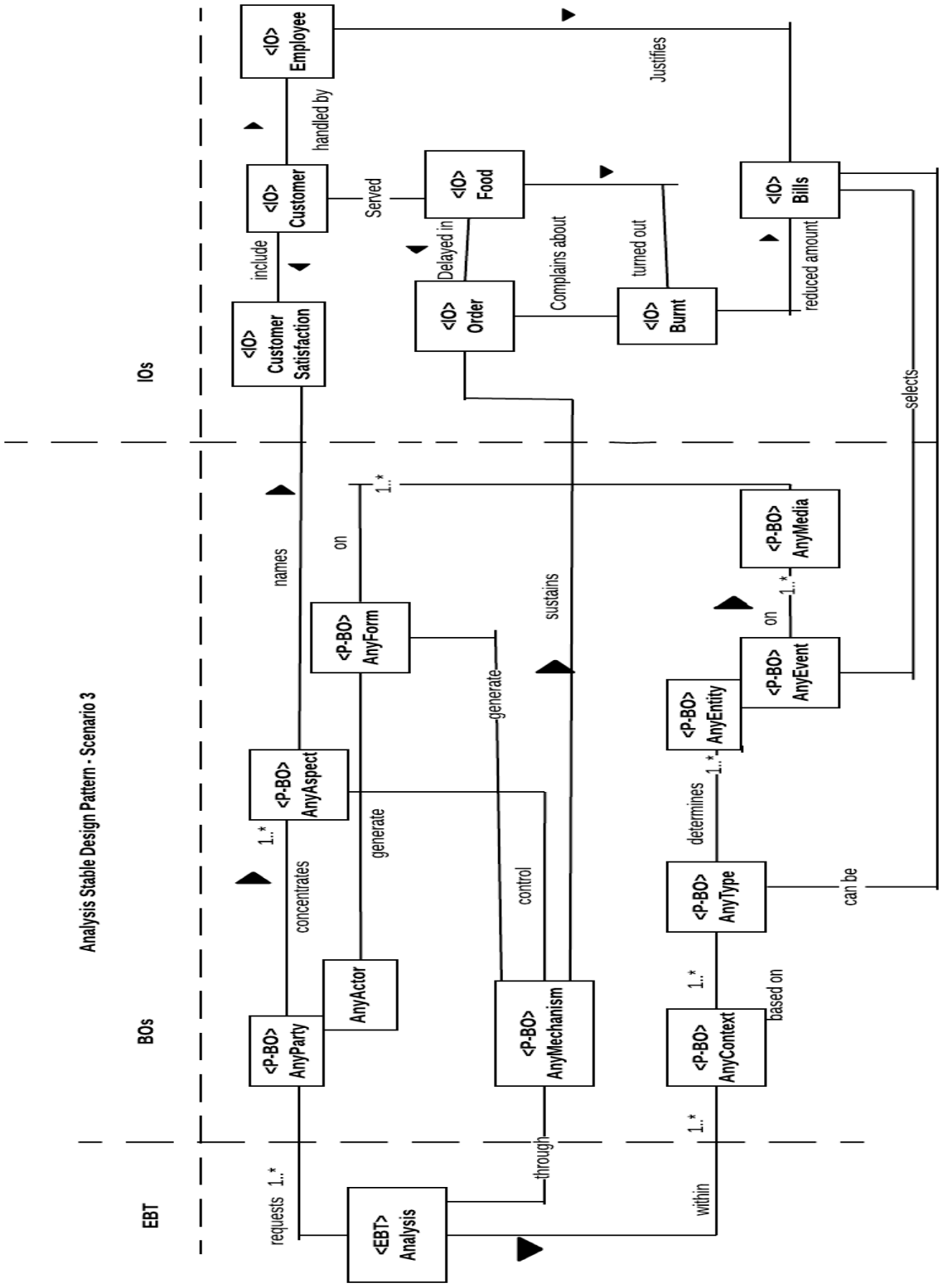

Figure 3.3.3a. Stable Model connected to the IOs for scenario 3

4.0 Comparing Models

4.1 Non-Functional Requirements

The following are the non-functional requirements or quality factors that pertain to analysis:

4.1.1 Consistency

Consistency in analysis is the conformity in application of something, typically that which is necessary for the sake of logic, accuracy, or fairness. Any analysis that is carried out should be consistent to avoid lack of proper understanding. An argument made should be analytical in all respect and its tone and tenor should be even reliable, uniform and regular.

4.1.2 Completeness

Analysis carried out should be complete and wholesome and it should mean something that is understandable, comprehensive, meaningful, logical and self-explaining. An incomplete analysis may lead to undesirable consequence and eventually a waste of time and event.

4.1.3 Accuracy

Analysis performed must be accurate and within context considering the same set of scenarios for a specific condition. Accuracy matters the most as without precision and accuracy the study or research is not considered to that mark.

4.1.4 Understanding

Analysis is the process of assimilating knowledge and ability to judge a particular situation or subject. For any analysis to be effective, it should be easily understandable. People who want to understand something is concrete and meaningful should easily understand what is being analyzed.

4.1.5 Synthesis

Synthesis means producing new things from the old findings or totally a new invention. It contains new challenges for proving a fact or denying a fact. So, analysis in this case helps to get the fresh findings from the analyze topics.

4.1.5 Abstraction

Abstraction means extracting the information from the concepts or topic. The quality of dealing with ideas rather than events. So, when we do analysis we abstract the smallest information from the concepts and provide the evidences for the findings to support the analysis on any topic.

4.2 Weights

The following are the criteria for comparing the traditional and stable models. The weights are distributed out of a total of 90 points.

Consistency (20 points) - it scores high the more tangibility it has because tangible objects can give specifics about the object while abstract objects can only give you a general idea about the objects that can implement it.

Completeness (15 points) - it scores high if the model has more classes that are tangible than stable. Stable classes are factual meaning they are lasting no matter the circumstance. Tangible classes change for every situation, so they appeal more to opinion.

Accuracy (10 points) - it scores higher the more you can add to the model without having to completely modify the model

Understanding (15 points) - it scores higher the more scenarios and contexts the model can cover

Synthesis (20 points) - it scores higher the more varied the model is in terms of describing the pattern

Abstraction (10 points) - it scores higher the more complete the model is in terms of describing the pattern

4.3 Comparison

Based on the weights in section 4.2, the table below describes and provides scores for the traditional and stable models.

Table 4.3a. Weighted comparison of traditional and stable models

Criteria	Weight	Traditional Model	Score	Stable Model	Score
Consistency	20	The traditional model is not very consistent because it is not adaptable to the changing need	8	The stable model is more consistent because they are more adaptable to the changes in any of the system	18
Completeness	15	The traditional model has less completeness as challenges and constraint are not that easier to determine.	7	The stable model is more complete as challenges and constraint are very easy to determine and are all applicable for all applications	12

Accuracy	10	The traditional model is very less accurate because since it cannot adapt to the changes that might occur in the future for any scenario.	5	The stable model is more accurate because it can adapt to as many scenarios as possible. Any field of knowledge has analysis and the stable model works to cover each one with that same model.	8
Synthesis	20	The traditional model is more synthesized as they are more tangible and have very specific details of each classes.	17	The stable model is less synthesized as they are intangible and have a brief detail of each classes	8
Abstraction	10	The traditional model is less abstract as they contain a full detail of the system.	2	The stable model is more abstract as we use general enduring concepts and hence results pattern can be used for many other application	8
Total	75		39		54

4.4 Measurability

For this section, there is one qualitative and one quantitative metric that is used to measure the two models. They are explained below:

Quantitative: # of classes with 3 or more client classes

This metric measures how many classes are associated with more than two classes. The formula for this is given as follows:

C3P = TC - (C2 + C1 + CS)

C3P - Number of Classes with 3 Plus Associations to other classes
TC - Total amount of Classes
C2 - Number of Classes with 2 Associations to other classes
C1 - Number of Classes with 1 Association to another class
CS - Number of Classes that only associate with themselves

Traditional:

TC = 10
C2 = 5

C1 = 0
CS = 0
C3P = 10 - (5 + 0 + 0)

C3P = 5

Stable:
TC = 10
C2 = 3
C1 = 3
CS = 0
C3P = 4

As a result, the stable model has less classes that are associated with 3 or more other classes. This means that the classes in the stable model don't need to interact as much with each other and the system can still work. The more a class uses another class, the more it may depend on that class to perform.

Qualitative: Consistent

Consistent refers to how well a model is made to fit any scenario the model is meant to illustrate. This metric is measured by how many classes are missing from the model which the system needs to fit all the scenarios possible or that are needed to represent the whole system.

Number of Missing Classes in Traditional Model: 1…*
Number of Missing Classes in Stable Model: 0

Therefore, the stable model is more consistent because it is more complete than the traditional model. There are no classes missing in the stable model such that all the scenarios that can be illustrated would need for the system to work. Unlike the stable model, the traditional model is not complete. One can add one to many classes in there that another scenario might need for the system to work. Therefore, the traditional model is not very consistent.

5.0 Discussion and Analysis

5.1 Abstraction

Considering the weighed scores generated both of traditional and stable approaches of pattern making, the stability model for analysis is far more stable, superior, complete and flexible for a wide range of applications. Similarly, this model and its dynamics are easy to understand and comprehend even by inexperienced pattern developers. Conversely, a traditional model although easy to understand, is not fully accurate and its approach may fail to address universality of domains and application scenarios that might lie outside the periphery of core of the problem, i.e. investigating the problem and performing analysis on the same.

5.2 Applicability

The Analysis is an Enduring business concept as well as a Business Object which is accurate, flexible and reusable with an ability to extend its advantages to a diverse number of applications where the concept of analysis is involved. Analysis can exist in different domains including

software science, politics, medicine, personal and professional areas, business and in those areas where there are any problems into any situations.

5.3 Impacts

Finally, the analysis stability model introduces a whole new solution to a lot of existing problems in the related area. It allows the subject to apply analysis in various domains while reducing the trouble of introducing new model for each different domain. For this reason, the impact of using stable model in contrast to traditional model is easily detected due to the accuracy and capability of using it in different context.

6.0 Conclusion

The results show that the Stable Model is the better of the two for developing any kind of system over a long period of time. Stable models can be used in any scenario under any context with any tangible objects that change over time. In contrast, Traditional models are good for building specific systems and they do not last because the objects used can change over time. This means that system built off traditional models must be re-structured every time there is a new development. Analysis is a concept that continues to exist. Therefore, it needs a stable model to make all systems that focus on it consistent throughout.

7.0 References

[1]. https://en.oxforddictionaries.com/definition/us/analysis.
[2]. Marshall Cline, and Mike Girou. (2000). “Enduring Business Theme”, Communications of the ACM, Vol.43, No.5, pp.101-106.
[3]. Model-based Software Reuse Using Stable Analysis Patterns. by Haitham Hamza, Mohamed E. Fayad - Computer Science and Engineering Dept.
[4]. A Pattern for Stress and its Resolution by Mohamed Fayad & Charles Flood.

Appendix A - crc for traditional model – Scenario-1

The following are the CRC cards that apply for the traditional model.

AnyActor(Intern)		
Responsibility	**Collaborations**	
To undergo the task assigned and work for the betterment	**Client**	**Server**
	Person	call()
	skill	develop()

		contains()
Attributes: name,type,age		

AnyActor(Manager)		
Responsibility	**Collaborations**	
To trained the new person and manage the project in many phases	**Client**	**Server**
	Person	call()
	Review	undergo()
		posses()
Attributes: name, position, maintenance		

AnyMechanism(Review)		
Responsibility	**Collaborations**	
The process of analyzing any object and giving points to that person or objects	**Client**	**Server**
	Manager	process()
	Project	assign()
		comment()
Attributes: points, comment		

AnyEvent(Job)		
Responsibility	**Collaborations**	
To perform the work best suited in	**Client**	**Server**
	IBM	Issue()

	Intern	Interrogate()
		declare()
Attributes: jobId, description, positionName		

AnyAspect(Internship)		
Responsibility	**Collaborations**	
To gain work experience or satisfy requirements for a qualification	**Client**	**Server**
	IBM	qualify()
	Manager	manages()
		build()
Attributes: supervisor, taskId, manager		

AnyType(Skills)		
Responsibility	**Collaborations**	
The ability to do something well or expertise in any area	**Client**	**Server**
	Intern	develops()
	Project	maintains()
		contains()
Attributes: skillName, skillType, levelOfExpertise		

AnyForm (Project)		
Responsibility	**Collaborations**	
An individual or collaborative enterprise that is carefully planned and designed to achieve a particular aim.	**Client**	**Server**
	skill	require()
	Review	generates()
		produces()
Attributes: definition, application, type		

Appendix B- crc for traditional model – Scenario-2

The following are the CRC cards that apply for the traditional model.

AnyActor(Police Officer)		
Responsibility	**Collaborations**	
To do analysis on robbery and issue alert state	**Client**	**Server**
	Robber	interrogate()
	AlertState	announce()
		Inspect()
Attributes: state, protocol		

AnyActor(Robber)		
Responsibility	**Collaborations**	
To suffer mental issues and is a student	**Client**	**Server**
	Robbery	include()
	PoliceOfficer	classify()

		mask()
Attributes: Status, Mentalstate		

AnyMechanism(Crime)		
Responsibility	**Collaborations**	
To commit a crime and indulge in bad habits	**Client**	**Server**
	MentalHealth	sustain()
	Aspect	control()
		frighten()
Attributes: Affectedperson,Status, Mentalstate		

AnyEvent(IssueAlert)		
Responsibility	**Collaborations**	
To announce alert state and to interrogate the case	**Client**	**Server**
	AlertState	issue()
	Officer	suspect()
		planning()
Attributes:Interrogate,IssueAlertState		

AnyAspect(Robbery)		
Responsibility	**Collaborations**	
To harass and create havoc	**Client**	**Server**

among people	Robbery	command()
	Robber	commit()
		admit()
Attributes:CreateTension,Crimes,Disturbance		

AnyType(Drugs)		
Responsibility	**Collaborations**	
Affects the health and mental condition of the person who consumes drugs	**Client**	**Server**
	Mental Health	impacts()
	Robbery	undergo()
		addiction()
Attributes: supply,drugType,drugConsumptionAmt		

AnyForm (Havoc)		
Responsibility	**Collaborations**	
Creates kaotic environment and panics public	**Client**	**Server**
	Environment	causes()
	Robbery	suffer()
		creates()
Attributes: tension, state, protocol		

AnyContext (University Area)		
Responsibility	**Collaborations**	
Creates a safe and harmonious environment for students to study	**Client**	**Server**
	Environment	protects()
	AlertState	declares()
		commands()
Attributes: state,name,type		

Appendix C - CRC for stable

The following are the CRC cards that apply for the stable model.

EBT (Analysis)		
Responsibility	**Collaborations**	
To remove one or more parties from another through the use of one or more mechanisms	**Client**	**Server**
	AnyParty	analyze()
	AnyMechanism	consider()
	AnyContext	learning()
Attributes: analysisType, conductDescription, presetList		

BO (AnyParty)		
Responsibility	**Collaborations**	
To carry out Analysis on other party	**Client**	**Server**
	Analysis	investigate()

	AnyAspect	control()
		request()
Attributes: Aspect,terms of Analysis		

BO(AnyMechanism)		
Responsibility	**Collaboration**	
To perform Analysis on any given problem that will affect other parameters	**Client**	**Server**
	Analysis	provide()
	AnyAspect	influence()
	AnyForm	consist()
Attributes: operation, analysis		

BO(AnyAspect)		
Responsibility	**Collaboration**	
To carry out Analysis on any aspect of problem	**Client**	**Server**
	AnyParty/AnyActor	generates()
	AnyMechanism	supports()
		determine()
Attributes: lifescience, scrutiny		

BO(AnyContext)		
Responsibility	**Collaboration**	
To perform Analysis irrespective of any context	**Client**	**Server**
	Analysis	depend()
	AnyType	carry()
		specify()
Attributes: referred,Contextual		

BO (AnyType)		
Responsibility	**Collaboration**	
To categorize the type of analysis being performed for the naming of one to more entities or events	**Client**	**Server**
	AnyContext	encapsulate()
	AnyEntity	organize()
	AnyEvent	determine()
Attributes: typeName, typeId, typeDescription		

BO(AnyEntity)		
Responsibility	**Collaboration**	
To present itself for one of the mechanism that needs for the analysis pattern.	**Client**	**Server**
	AnyType	determine()
	AnyMedia	relation()
		provide()

Attributes: entityID, entityName, functionList

BO (AnyEvent)		
Responsibility	**Collaboration**	
To perform the analysis for one to many parties based on the type of analysis	**Client**	**Server**
	AnyType	begin()
	AnyMedia	end()
		present()
Attributes: eventId, details, startTime, endTime		

BO (AnyForm)		
Responsibility	**Collaboration**	
To perform analysis in any form and in any field irrespective of problem	**Client**	**Server**
	AnyMechanism	includes()
	AnyMedia	perform()
		support()
Attributes:		

BO(AnyMedia)		
Responsibility	**Collaboration**	
To visualize and implement the analysis patterns	**Client**	**Server**
	AnyEvent	relying()
	AnyEntity	display()
	AnyForm	generate()
Attributes:Visualize,Enactupon		

Appendix D - Programming Skeletons

Traditional Model - Scenario 1 (A New Intern joining IBM for Summer Internship)

```
public abstract class Intern{
    private string name;
    private int age;
    private string type;

    /**

     *Intern joins a company
To undergo the task assigned and work for the betterment
*/



    public abstract void call(person);
    /**
     * person can be inten/manager
     * intern has skills required for the project
     * managerinterviews and selects interns according to the project requirements
     */

    public abstract void develops(skill);
    /**
     * skills required for the job
     */
    public abstract void contains();

}
```

Figure a. Scenario 1 class Intern

```
public abstract class Manager{
    private string name;
    private string position;
    private string maintenance;

    /**

     *Mnageer hires intern
     *
To trained the new person and manage the project in many phases


*/



    public abstract void call(person);
    /**
     * person can be inten/manager
     * intern has skills required for the project
     * manager interviews and selects interns according to the project requirements
     */

    public abstract void undergo(review);
    /**
     * skills required for the job
     */
    public abstract void posses();

}
```

Figure b. Scenario 1 class Manager

```
public abstract class Review{
    private string points;
    private string comment;


    /**

     *Mnageer hires intern
     *Manager reviews
     *To trained the new person and manage the project in many phases



*/



    public abstract void process(manager);
    /**
     * The process of analyzing any object and giving points to that person or objects
     * manager interviews and selects interns according to the project requirements
     */

    public abstract void assign(project));
    /**
     * skills required for the job
     */
    public abstract void comment();

}
```

Figure c. Scenario 1 class Review

```
public abstract class Job{
    private int JobId;
    private string description;
    private string positioname;

    /**

     *Mnageer hires intern
     *Manager reviews
     *To trained the new person and manage the project in many phases


*/


    public abstract void issue(IBM);
    /**
     * The process of analyzing any object and giving points to that person or objects
     * manager interviews and selects interns according to the project requirements
     */

    public abstract void interrogate(intern));
    /**
     * skills required for the job
     */
    public abstract void declare();
    /** declare if the the candidate is suitable for the job

     */

}
```

Figure d. Scenario 1 class Job

```
public abstract class Internship{
    private int TaskId;
    private string supervisor;
    private string manager;

    /**

     *Mnageer hires intern
     *Manager reviews
     *To trained the new person and manage the project in many phases


*/


    public abstract void qualify(IBM);
    /**
     * The process of analyzing any object and giving points to that person or objects
     * manager interviews and selects interns according to the project requirements
     */

    public abstract void manages(manager));
    /**
     * skills required for the job
     */
    public abstract void skills();
    /** declare if the the candidate is suitable for the job

     */

}
```

Figure e. Scenario 1 class Internship

```
public abstract class Skills{
    private string skillname;
    private string skilltype;
    private string levelofexpertise;

    /**

     *Mnageer hires intern
     *Manager reviews
     *To trained the new person and manage the project in many phases


*/


    public abstract void develops(intern);
    /**
     * The process of analyzing any object and giving points to that person or objects
     * manager interviews and selects interns according to the project requirements
     */

    public abstract void maintains(project);
    /**
     * skills required for the job
     */
    public abstract void contains();
    /** declare if the the candidate is suitable for the job

     */

}
```

Figure f. Scenario 1 class Skills

```
public abstract class Project{
    private string definition;
    private string application;
    private string type;

    /**

     *Mnageer hires intern
     *Manager reviews
     *To trained the new person and manage the project in many phases


*/


    public abstract void requires(skill);
    /**
     * The process of analyzing any object and giving points to that person or objects
     * manager interviews and selects interns according to the project requirements
     */

    public abstract void generates(review)
    /**
     * skills required for the job
     */
    public abstract void produce();
    /** declare if the the candidate is suitable for the job

     */

}
```

Figure h. Scenario 1 class Project

Traditional Model - Scenario 2 (Robbery case at San Jose State University)

```
package program;

public class PoliceOfficer {
    private String state;
    private String protocol;

    public void interrogate(Robber current)
    {
        this.protocol = current.impose();
    }
    public String announce() {
        return this.protocol;
    }

}
```

Figure i. Scenario 2 class PoliceOfficer

```
package program;

public class Robber {
    private String status;
    private String mentalstate;

    public void interrogate(Robber current)
    {
        this.protocol = current.impose();
    }
    public String include() {
        return this.protocol;
    }

}
```

Figure j. Scenario 2 class Robber

```
pa 202 program;

public class Robbery {
    private String createtension;
    private String Crimes;
    private String Disturbance;

public void cause(Tension impact)
{
    impact.prompt(restriction);
}
public void names(String)
{
    this.restriction = hold;
}
}
```

Figure k. Scenario 2 class Robbery

```
package program;

public class IssueAlert {
    private String interrogate;
    private String isseualertstate;


    public string issue()
    {
    if (this.securityLevel > 0 && this.securityLevel <= 10)
    {
        return this.conditionList.get(2);
    }
    else if (this.securityLevel > 10 && this.securityLevel <= 20)
    {
        return this.conditionList.get(1);
    }
    else
    {
        return this.conditionList.get(0);
    }
    }
    public String interrogate()
    {
        return "alert level : " + this.securitylevel;
    }
    public  String terminate()
    {
        return "none";
    }
```

Figure l. Scenario 2 class IssueAlert

```java
package program;

public class Crime {
    private String affectedperson;
    private String status;
    private String mentalstate;

    public void sustain()
    {
    return this.title + "type $ " + this.typedefinitionnumber + " : " this.definition;
    }
    public String control() {
        return this.crime;
    }

}
```

Figure m. Scenario 2 class Crime

```java
public abstract class Drugs{
    private string supply;
    private string drugtype;
    private int drugconsumptionamt;

    /**
     * drugs create destruction and addiction and is danger for society when cinsumed in larger amounts
     * specific drugtype
     * supply of the drugs which has to be monitered
     */


    /**
     * To visualize and implement the analysis patterns on drugs
     */



    public abstract void impact(mentalhealth );
    /**
     * public safety - will impact mentalhealth
     * health coniditon monitered
     */
    public abstract void undergo(mentaltype);
    /**
     * victim undergoes mental stress
     */
    public abstract void addiction();

}
```

Figure n. Scenario 2 class Drugs

```
public abstract class Havoc{
    private string tension;
    private string state;
    private int protocol;

    /**
     * drugs create destruction and addiction and is danger for society when cinsumed in larger amounts
     * specific drugtype
     * supply of the drugs which has to be monitered
     */


    /**
     * To visualize and implement the analysis patterns on drugs
     */



    public abstract void causes(environment);
    /**
     * public safety - will impact mentalhealth
     * health coniditon monitered
     */
    public abstract void suffer(robbery);
    /**
     * victim undergoes mental stress
     */
    public abstract void creates(havoc);


}
```

Figure o. Scenario 2 class Havoc

```
public abstract class University{
    private string name;
    private string state;
    private string type;

    /**
     * Creates a safe and harmonious environment for students to study
     * place where student trust university
     * when created havoc students panics




     */


    /**
     * To visualize and implement the analysis patterns on drugs
     */



    public abstract void protects(environment);
    /**
     * public safety - will impact mentalhealth
     * health coniditon monitered
     */
    public abstract void declares(alertstate);
    /**
     * victim undergoes mental stress
     */
    public abstract void commands(students);


}
```

Figure p. Scenario 2 class University

Stable Model (With Scenario 2 as example – Robbery case at San Jose State University)

```
public abstract class Analysis {
    private int analysistype;
    private string conductdescription;
    private ArrayList<String> presentList;


    /**
     * adds analysis to the subject
     */
    public abstract void  analysis(Anyparty partybeinganalyzed);
    /**takes one of the restrictions and applies o the analysed party*/


    public abstract void learning(AnyMeachanism);

    /**
     * Reurns a restriction based on analysis

     */
}
```

Figure q. Scenario 2 class Analysis

```
public abstract class AnyAspect {
    private int lifescience;
    private string scrutinity;



    /**
     * to carry out analysis on anyaspect
     * based on operations
     * analysis done on anyaspect
     */


    public abstract void generate(Anyparty );

    /**
     * Reurns a restriction based on analysis
     * controls anyaspect in terms of analysis being carried out

     */
    public abstract void supports(Anymechanism);

    public abstract void determine();


}
```

Figure r. Scenario 2 class AnyAspect

```
public abstract class AnyContext{
    private int referred;
    private string contextual;


    /**
     * to carry out analysis on anycontext
     * based on operations
     * analysis done on anycontext
     */


    public abstract void depend(analysis);

    /**
     * Reurns a restriction based on analysis
     * controls anyaspect in terms of analysis being carried out

     */
    public abstract void carry(AnyType);

    public abstract void specify();


}
```

Figure s. Scenario 2 class AnyContext

```
public abstract class AnyEntity{
    private string entityname;
    private int entityID;
    private string functionlist



    /**
     * present itself for one of the mechanism that needs for the analysis pattern.


    public abstract void encapsulate(analysis);

    /**
     * Reurns a restriction based on analysis
     * categorize the type of analysis being performed for the naming of one to more entities or events
     */
    public abstract void determine(AnyType);

    public abstract void relation(Anymedia);

    public abstract void provide();


}
```

Figure t. Scenario 2 class AnyEntity

```
public abstract class AnyEvent{
    private string startname;
    private int eventID;
    private string details;
    private string endname;




    /**
     * To perform the analysis for one to many parties based on the type of analysis
     */



    public abstract void begin(AnyType);

    public abstract void end(Anymedia);

    public abstract void present();


}
```

Figure u. Scenario 2 class AnyEvent

```
public abstract class AnyMechanism {
    private int operation;
    private string analysis;



    /**
     * to carry out analysis on anymechanism
     * based on operations
     * analysis done that will affect other parameters
     */


    public abstract void provide(Analysis );

    /**
     * Reurns a restriction based on analysis
     * controls anyaspect in terms of analysis being carried out

     */
    public abstract void influence(Anyaspect);

    public abstract void consist(Anyform);


}
```

Figure v. Scenario 2 class AnyMechanism

```
public abstract class Anyparty {
    private int aspect;
    private string termsofanalysis;



    /**
     * to carry out analysis on anyparty
     */
    public abstract void  investigate(Analysis partybeinganalyzed);
    /**takes one of the restrictions and applies o the analysed party*/


    public abstract void control(AnyAspect );

    /**
     * Reurns a restriction based on analysis
     * controls anyaspect in terms of analysis being carried out

     */
}
```

Figure w. Scenario 2 class AnyParty

```
public abstract class AnyMedia{
    private string visualize;
    private string enactupon;



    /**
     * To visualize and implement the analysis patterns
     */


    public abstract void rely(AnyEvent);

    public abstract void display(Anyentity);

    public abstract void generate(Anyform);


}
```

Figure x. Scenario 2 class AnyMedia

```
public abstract class AnyType{
    private string typename;
    private int typeID;
    private string typedescription



    /**
     * to carry out analysis on anytype
     * based on operations
     * analysis done on anycontext
     */


    public abstract void encapsulate(analysis);

    /**
     * Reurns a restriction based on analysis
     * categorize the type of analysis being performed for the naming of one to more entities or events
     */
    public abstract void organize(AnyEntity);

    public abstract void determine(Anyevent);


}
```

Figure y. Scenario 2 class AnyType