BATTLE SIMULATION

Battle Combat Simulation

Designed by DEVSJAVA

ECE575 Course Project - Fall 2003

Taekyu Kim

taekyuk@email.arizona.edu

Abstract

This report documents a student project that was done as part of the class "Object-Oriented Simulation/Discrete Event Models (ECE575)". As the name of this class implies, its main goal was to provide students with hands-on training in the simulation concepts of object oriented model designed by DEVSJAVA. The idea of this specific project was to develop the battle combat simulation model that helps a military commander to predict the result of combat for the given different situation and find the most effective way to attack enemy and win the battle in the real world environment.

1. Introduction

Recently, Most of constructive simulation is not multilevel one but single level simulation. It might not be able to well reflect the real world. Because, single simulation have restricted unit object which is not subdivided. It's mean that simulation has low resolution and the fidelity is rough. A unit object not subdivided in single model has constant numerical power, and computed a regular rate during combat. But, in real world, the object is potently affected by commander operating it. It can be never computed regular rate. Hence, everyone who participated in war game simulation knows if high and low echelon unit could practice at the same time, the reliability of military simulation for training will be severely increased. That is the reason to design multilevel simulation. In addition, the trend of future military simulation is synthesis among virtual, constructive and live simulation. We could say that is a kind of multilevel simulation.

The idea of this specific project was to develop the battle combat simulation model that helps a military commander to predict the result of combat for the given different situation and find the most effective way to attack enemy and win the battle in the real world environment. This involves guiding the military commanders to find the most effective way to give the command to attack in the battle. By knowing the current troops power level and the power level of the enemy, this simulation model requires to receive an order to attack from user. After getting the order to attack, this model shall simulate the battle combat with given parameters that are reflecting the real world and show the processes of combat periodically. As in the real world, the combat will be affected by the frequently changing current factors (e.g., changing weather, food supply, and etc.). Thus, the simulation model has to dynamically update its parameters of the world and has to vary its order to attack. The simulation in DEVSJAVA could support powerful method to design of multilevel simulation.

2. Background Knowledge of DEVS

Discrete Event System Specification (DEVS) is a system specification capturing (i.e., modeling) structural architecture and dynamic behavior of dynamic systems based on well-defined mathematical formalism. It also allows building modular and hierarchical model compositions based on closure-under-coupling paradigm. In DEVS, generally, a model is considered as either atomic model or coupled model. The Atomic model is specified as follows:

Atomic model:

$$M = < X, S, Y, \delta_{\text{int},} \delta_{ext,} \delta_{con,} \lambda, ta >$$

where,

X: set of external input events;

S: set of sequential states;

Y: set of outputs;

$\delta_{\text{int}} : S -> S$: internal transition function

$\delta_{ext} : Q \times X^b -> S$: external transition function

$\delta_{con} : Q \times X^b -> S$: confluent transition function

X^b is a set of bags over elements in X,

$\lambda : S -> Y^b$: output function generating external events at the output

$ta : S -> R^+_{0,\infty}$: time advance function;

Q= {(s, e)| s$\in$**S,** $0 \le e \le ta(s)$} is the set of total states where e is the elapsed time since last state transition

Basic models may be coupled in the DEVS formalism to form a coupled model. A coupled model tells how to couple (connect) several component models together to form a new model. Two major activities involved in coupled models are specifying its component models and defining the couplings which create the desired communication networks. A coupled model is defined as follows:

$$DN = < X, Y, D, \{M_i\}, \{I_i\}, \{Z_{i,j}\} >$$

where,

X: set of external input events;

Y: a set of outputs;

D: a set of components names;

for each i in D,
M_i is a component model
I_i is the set of influences for i

for each j in $I_{i,}$

$Z_{i,j}$ is the i-to-j output translation function

3. Concept

The following figure represents relationship between real world and simulation about multilevel simulation. Each echelon could perform simulation for training independently. It's mean that all simulation may be executed with interaction at the same time, or designated only one could be performed. In most of training simulation had designed for particular level. In this model, each model which has different simulation resolution is processed with data exchange.

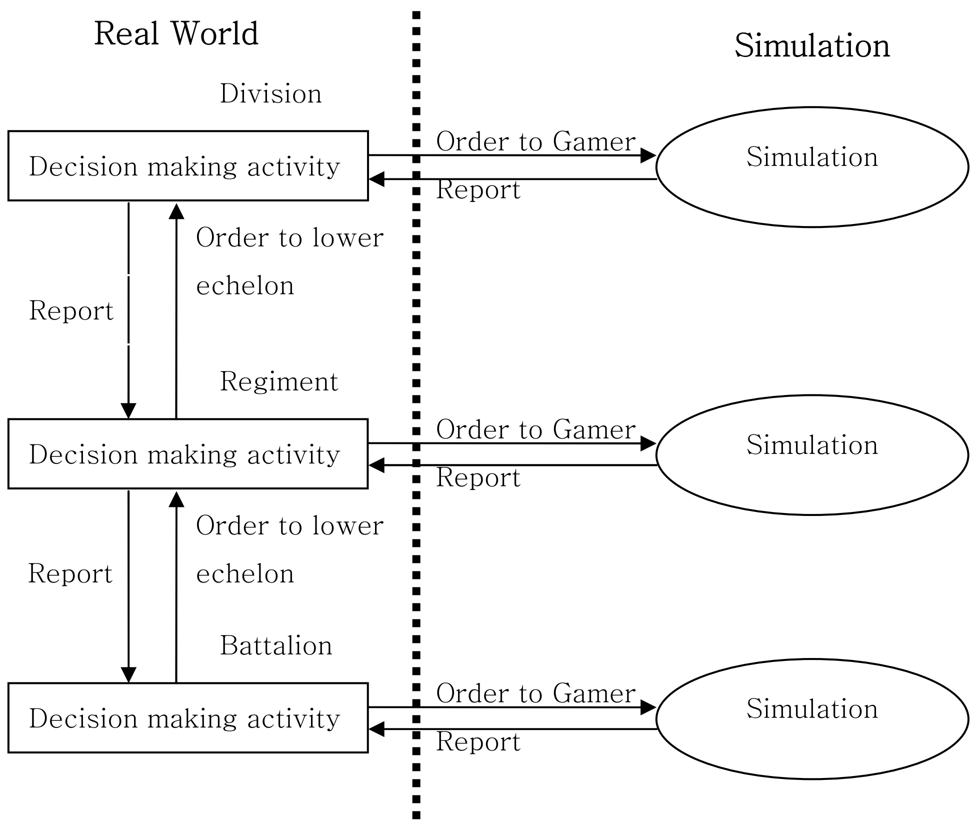


Figure 1. Relationship between real world and simulation

4. System Entity Structure

How we can describes the system in military simulation point of view. In generally, the function of tactics could be well divided, even though those are complexes as depending by much non deterministic variable.

Figure 2. illustrates the part of functions. It is very simple, and somebody might disagree the decomposition of functions, even if the shape is certainly. But what it is important thing is design. This SES model shows the class(Model) hierarchy.

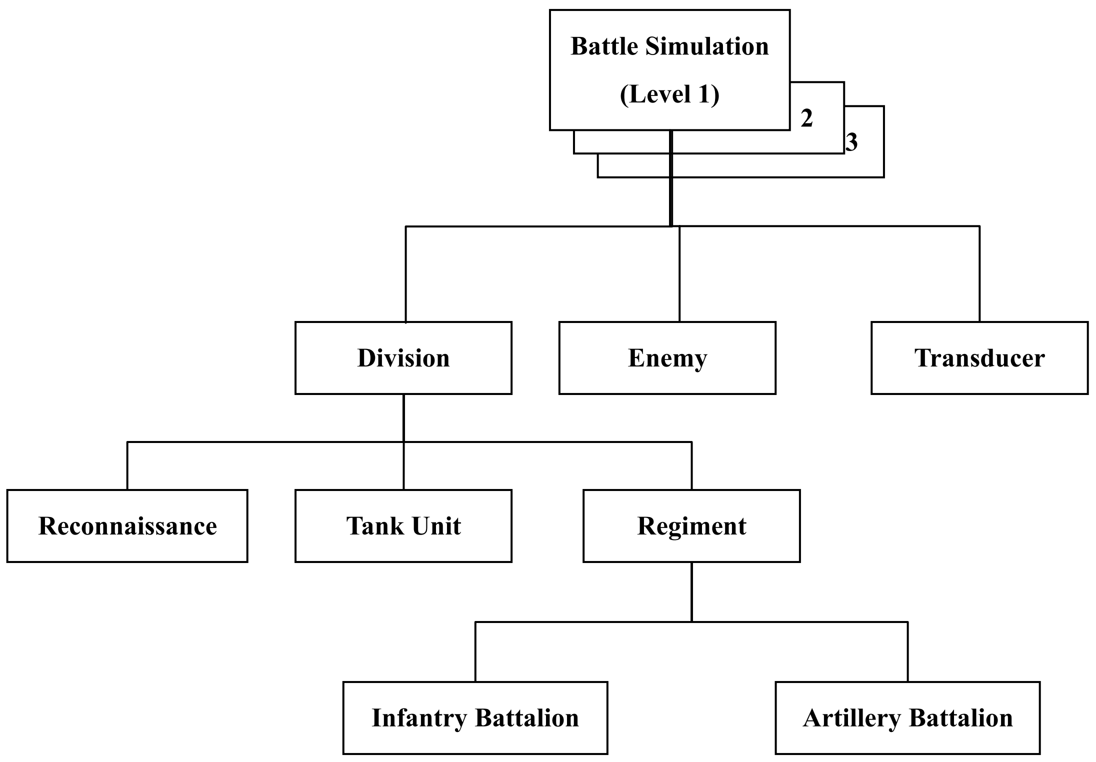


Figure 2. System entity structure of the battle combat simulation model

The top rectangle means multilevel unit that are similarity in tactical function point of view. The difference is that a kind of unit in real world and resolution and fidelity in simulation. The goal of this model is to show that each model which has different simulation resolution and fidelity operates independently with communicate in case of need each other. All the above is designed and implemented by DEVSJAVE.

5. DEVS Based Design

How can we describe the model with DEVS? In SES, 9 entities of upper rectangle have time element except detect entity. Why? In real world, as soon as a unit or people move, closed opposite unit is detected immediately. There are no any actions to detect. But if it is not moving unit but detecting machine, it may have time element. As you know, DEVS atomic model have time element. It is important concept to decide where it will be done. The highest level entity in this model is Battle Simulation which is consisted of 3 DEVS models, one is a coupled model and two are atomic models. So, Battle Simulation is a kind of coupled model by DEVS. This Battle Simulation model represents one of the distributed simulation model. It is highly modular and scalable architecture. It means that it can be easily extended by adding additional Battle Simulation scheme later. Overall architecture is as follows:

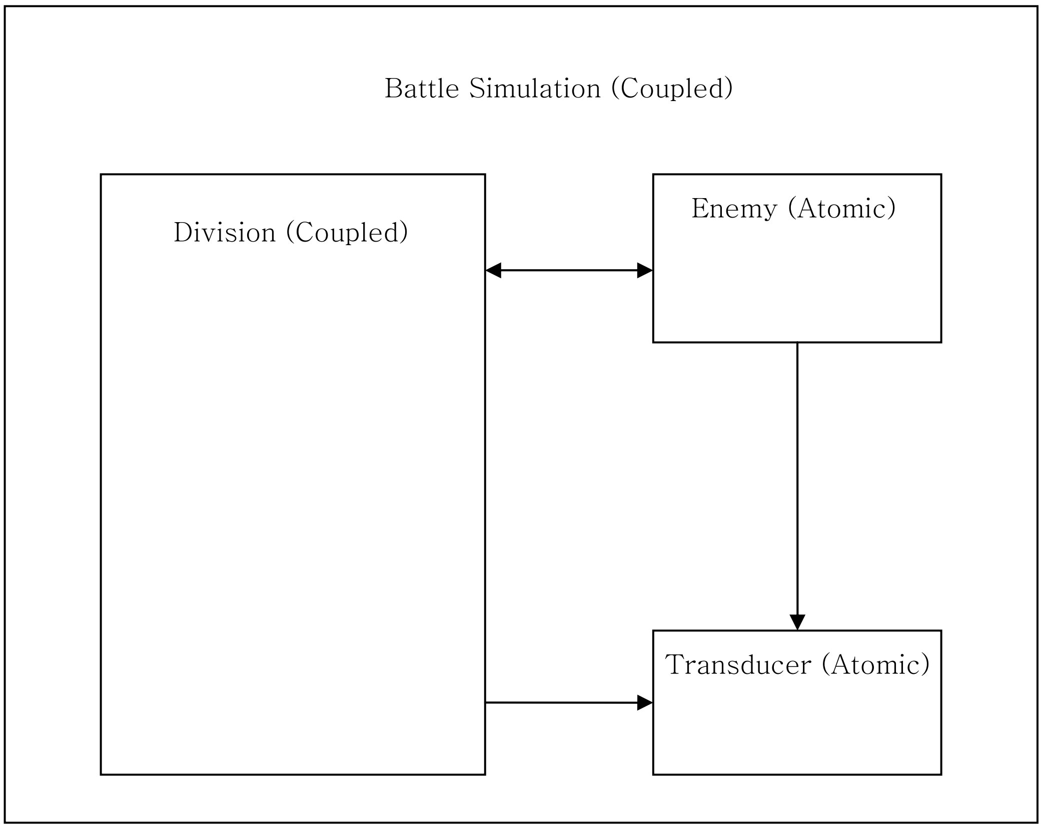


Figure 3. Overall architecture of Battle Simulation model.

Like a Battle Simulation model, the Division model is also consisted of several models, one is a coupled model and two are atomic models. So, I can say that the Division model is a coupled model with Reconnaissance, Tank Unit, and Regiment model. It is highly modular and scalable architecture to be easily extended by adding additional Division scheme later. There is no connection between each model in Division model. It means that the sub-models of Division model don't need to share their states each other. In other word, the sub-models of Division have to get an order and report to Division directly. The detailed view of the Division model is as follows:

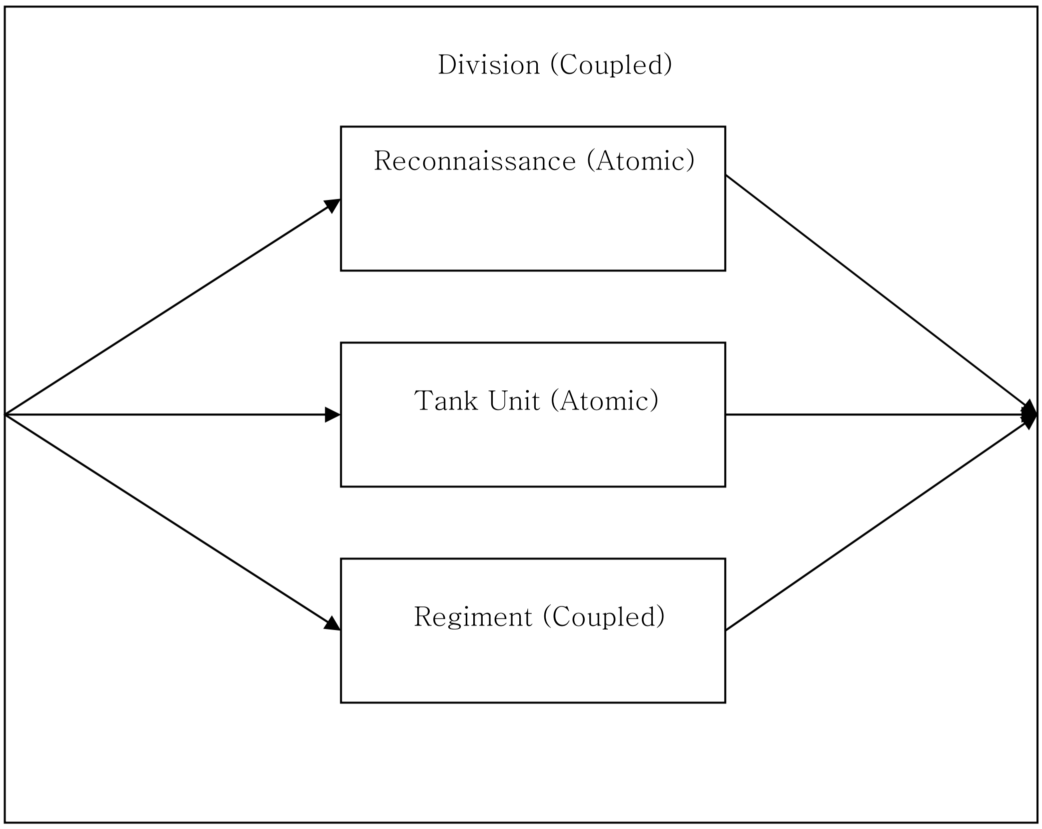


Figure 4. Detailed architecture of Division model.

Next step is about the Regiment model that is a sub-model of Division model. Because Regiment model is also consisted of two sub-models which are Infantry Battalion and Infantry Battalion model, this model can be expressed as a coupled model like Division model. The detailed architecture of Regiment model is as follows:

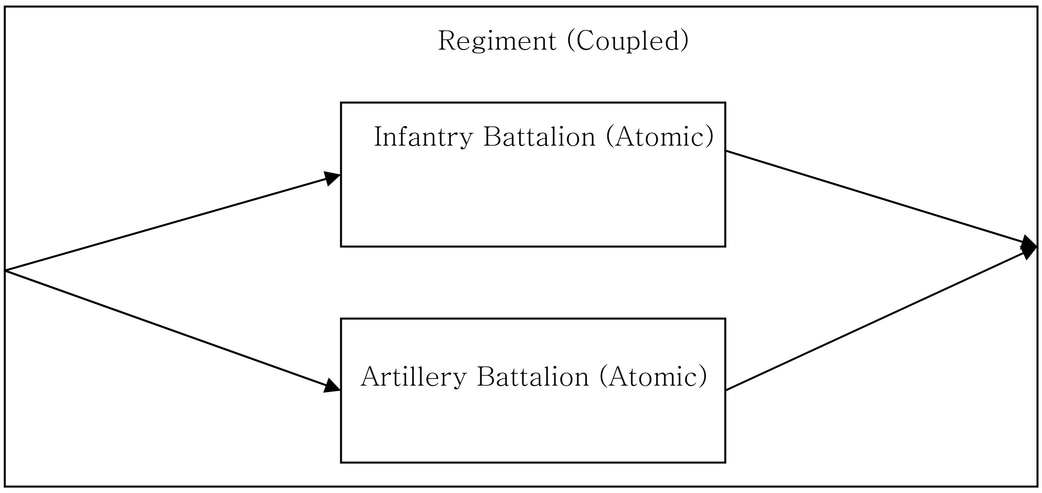


Figure 5. Detailed architecture of Regiment model.

Finally, we can see the detailed overall architecture of the Battle Simulation model by adding the detailed architecture of each coupled model to the overall architecture. In other word, adding Figure 5 to Figure 4 and adding Figure 4 to Figure 3 can derive the detailed overall architecture of Battle Simulation model. The detailed overall architecture of Battle Simulation model is as follows:

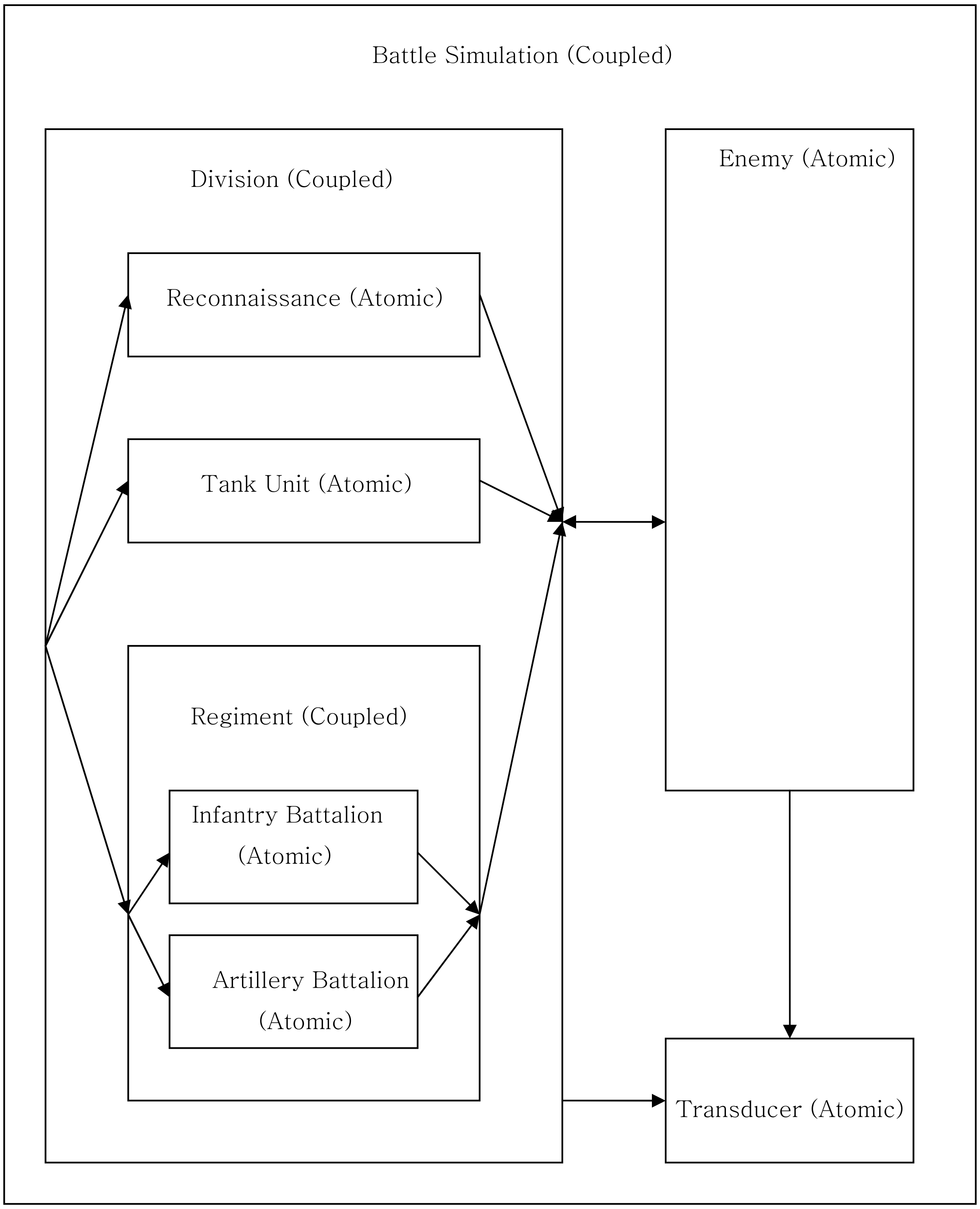


Figure 6. Detailed Overall Architecture of Battle Simulation Model.

6. Scenario

There are 2 scenarios for the command of ordering to attack. Scenario 1 is including the attack one by one command and Scenario 2 is including the attack all to enemy order. These two scenarios make 2 different results. The objective of being separated scenario is to compare the simulation results of two ways of the attack order and achieve the more way to attack for the given environment.

Before starting simulation, each model must have its own power level. Once the power levels are set, the Battle Simulation can be simulated by combating with these power levels. Not only are there default power levels for each model, but also the user can inject the power level of Enemy model by manually to compare the combat results and find the effective way to attack for the different enemy. The coupled model's power level is the sum of the power levels of sub-models. For example, Regiment model has the power level as 200 that is the sum of Infantry Battalion power level(100) and Artillery Battalion power level(100). Default power levels of each model are follows:

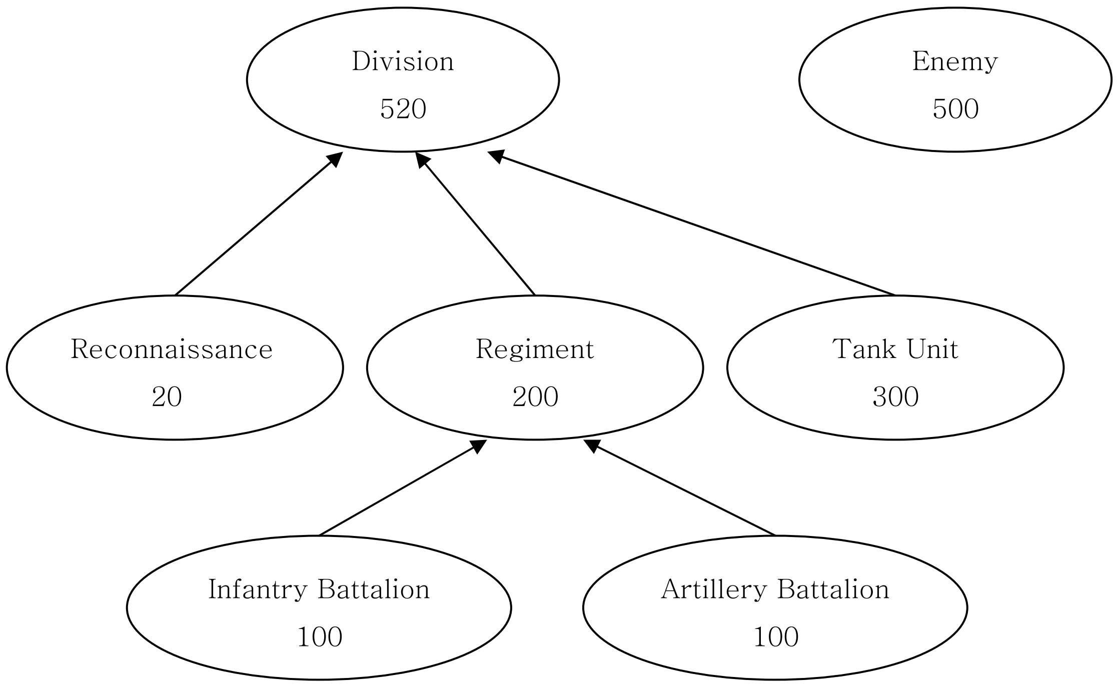


Figure 7. Default power level for each model

6.1 Power level calculation

The power level shall be updated periodically whenever one or more objects(models) fight with enemy. In this time, the power levels for the relevant model decrease by being affected with opposite objects(models). This is exactly as same as the real world expectation. The rule of power level calculation is follows:

■ **Infantry - Infantry**

opposite object / object	Platoon	Company	Battalion
Platoon	1:1	1:3	1:6
Company	3:1	1:1	1:3
Battalion	6:1	3:1	1:1

Table 1. Power level comparison between Infantry and Infantry

■ **Tank - Infantry**

opposite object / object	Platoon	Company	Battalion
Platoon	5:1	3:1	1:1
Company	10:1	5:1	2:1
Battalion	20:1	10:1	5:1

Table 2. Power level comparison between Tank and Infantry

■ **Artillery – Infantry (probability 50%)**

opposite object / object	Platoon	Company	Battalion
Company	3:1	1:0	2:0
Battalion	10:1	5:0	3:0

Table 3. Power level comparison between Artillery and Infantry

6.2 Scenario 1: Attack One by One

step	Action Models	Descriptions
1	Enemy : Inject power level	- inject the enemy's power level - default value is 500
2	Division -> Reconnaissance unit	- order to move and search the enemy
3	Reconnaissance unit -> Division	- report the enemy's information
4	Division : decision making	- decide the way to attack : 1. attack one by one
5	Division -> Regiment	- order to attack the enemy
6	Regiment -> Infantry Battalion	- order to attack the enemy
7	Infantry Battalion -> Regiment	- report the own status periodically - if they are in danger, stop attacking and retreat - request fire support
8	Regiment -> Artillery Battalion	- order to attack the enemy

9	Infantry Battalion -> Regiment	- report the own status periodically - if they are in danger, stop attacking and retreat - request help
10	Regiment -> Division	- request Tank Unit
11	Division -> Tank Unit	- order to combat to help Regiment
12	Tank Unit -> Division	- report the own status periodically - if they are in danger, stop attacking and retreat
13	Win or Loose	- Win : if the power level of the enemy gets to 0. - Loose : if all the combat unit which are Tank unit, Infantry Battalion, and Artillery Battalion are in danger

Table 4. Scenario 1: Attack One by One

6.3 Scenario 2: Attack All to Enemy

step	Objects	Descriptions
1	Enemy : Inject power level	- inject the enemy's power level - default value is 500
2	Division -> Reconnaissance unit	- order to move and search the enemy
3	Reconnaissance unit -> Division	- report the enemy's information
4	Division : decision making	- decide the way to attack : 2. attack all to Enemy
5	Division -> Regiment & Tank Unit	- order to attack the enemy
6	Regiment -> Infantry Battalion & Artillery Battalion	- order to attack the enemy
7	Infantry Battalion -> Regiment Artillery Battalion	- report the own status periodically - if they are in danger, stop attacking and retreat
8	Tank Unit -> Division	- report the own status periodically - if they are in danger, stop attacking and retreat
9	Win or Loose	- Win : if the power level of the enemy gets to 0. - Loose : if all the combat unit which are Tank unit, Infantry Battalion, and

		Artillery Battalion are in danger

Table 5. Scenario 2: Attack All to Enemy

7. Experimental Frame

The Battle Simulation model has its own experimental frame that display pots which is the variation of power levels of Division model(our force) and Enemy model(enemy troops), correlations, and the state transition in addition to the cell plot. As I mentioned before, Division model consists of several atomic models and coupled models. Enemy model would also have consisted with several atomic models and coupled model like Division model. However, the point of view in this Battle Simulation is for division commander in the troops. Further more, the division commander doesn't need to know the detail status of enemy. That is the reason why Enemy model doesn't have any detailed architecture in this Battle Simulation model. Following is the screen shots of the experimental frame designed by DEVSJAVA

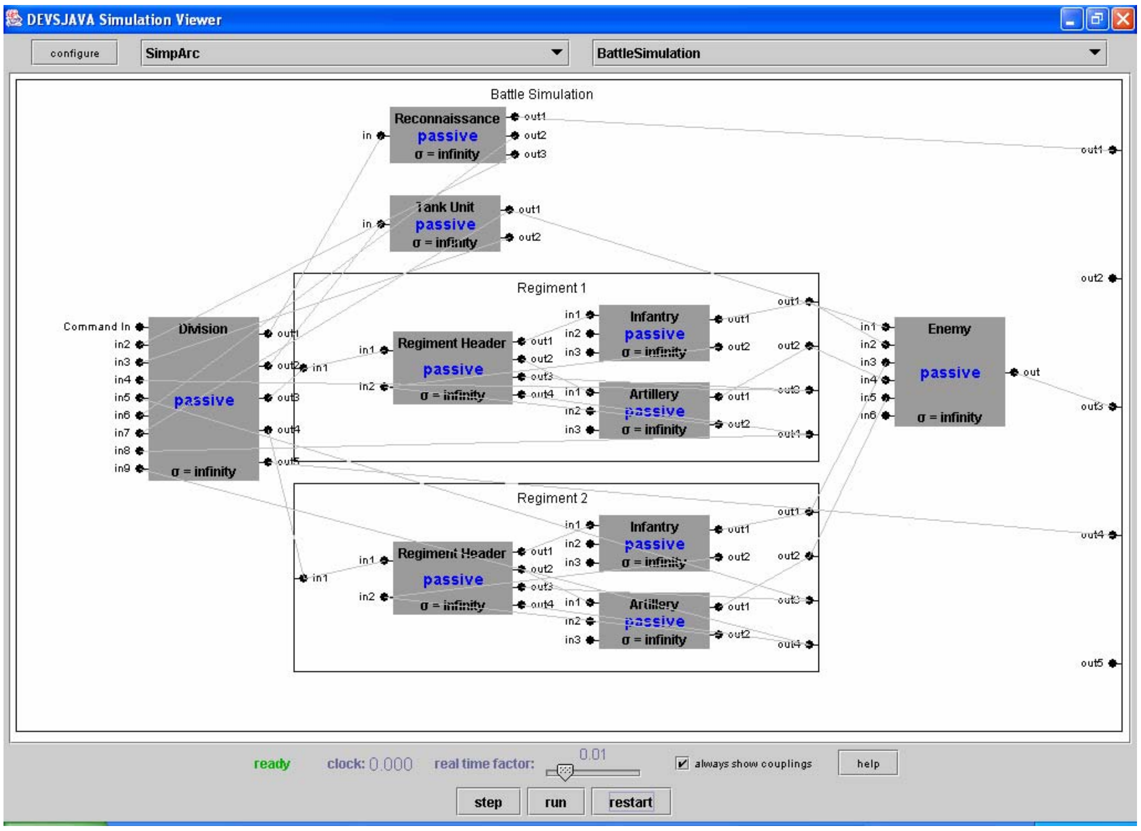


Figure 8. Screen shot of experimental frame

The above figure shows the experimental frame. As you see, there are two coupled Regiment models that are Regiment1 and Regiment2. There is just one

Regiment model in SES and the overall architecture of the Battle Simulation model as I have mentioned so far now. Because I would like to verify that my Battle Simulation model can be extended easily just by adding models which are created as different instances and I want to make the Battle Simulation model more complex, I create two instances of Regiment model. One more thing I need to mention is about the Transducer model in the overall architecture in Figure 3. There is no Transducer model in the above figure. However, in the Battle Simulation model, cell plot is Transducer model and has the functionality of Transducer model. The cell plot gather the power levels of Division and Enemy model and draw the change of power levels as time goes by. Last of models follows the detailed overall architecture in Figure 6. Let me introduce how to simulate this Battle simulation model since it requires several steps for manual input.

How to simulate this model

1. Inject the power level of enemy at the 'in1' of Enemy model
2. Order the reconnaissance search enemy by injecting the order of search enemy at the 'Command in' of Division model.
3. After finding enemy, Reconnaissance returns enemy's power level to Division model. Then, the status of Division changes 'Require Order'
4. Inject one of the order to attack enemy at the 'Command in' of Division model.
   - choose 'Fight one by one' for the scenario 1
   - choose 'Fight all to enemy' for the scenario 2
5. from next to finish, simulate automatically

By following the above steps, I can get the plot result of the simulation. The cell plot that shows the simulation result is follow:

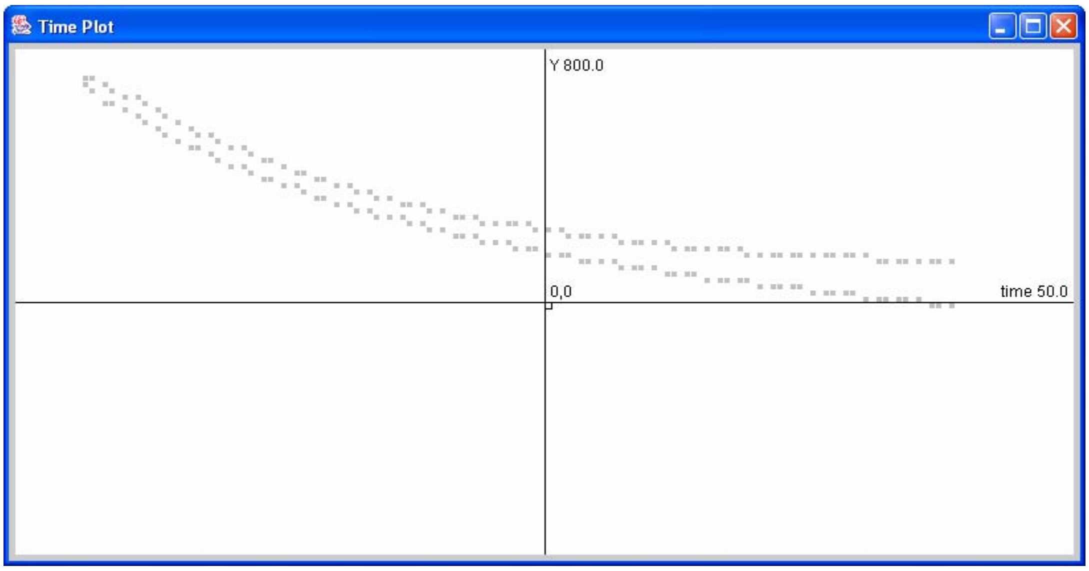


Figure 9. Screen shot of cell plot of simulation result

8. Results & Discussion

The comparison between the two scenarios shall be simulated as the power level of Enemy is changing. I assumed that the power level of the Division is constant as 720. Power level 720 comes from the sum of all the power level of included model in Division.

- Division(720) = Reconnaissance(20) + Tank Unit(300)
  + Regiment1(200) + Regiment2(200)
- Regiment1(200) = Infantry Battalion1(100) + Artillery Battalion1(100)
- Regiment2(200) = Infantry Battalion2(100) + Artillery Battalion2(100)

Following table shows the results of simulation time for the several simulations following the scenario 1 and scenario 2.

Power Level of Enemy	Simulation Time of Scenario 1	Simulation Time of Scenario 2
100	89	13
300	93	23
500	86	38
600	70	49
700	61	90
800	55	50

Table 6. The result of simulation time for the different enemy power level

First, I need to take a careful look at the time result in Table 6. For scenario 1, the simulation time increases during first 2 simulations that is 100 and 300 of enemy's power level. That's an expectable result because the power level of our troops is set as 720 and the power level of enemy increases from 100 to 300. To defeat the enemy that has 100 power level is much easier that 300 power level. However, I can see that the simulation time decreases as the enemy's power level goes up to 500, 600, 700, and 800. Simply, I expect that our troops must win the battle against less power level enemy. But, it is totally false because scenario 1 is following the rule of order of attack one by one. Thus, it's not possible to use all the power at one time to the enemy. As a result, even if the enemy's power level is less than our troops, enemy could win the battle. So, the simulation time can be gotten an unexpected value if the power level of our troops is more than the power level of enemy. In this simulation, I can get the abnormal value when the power level of enemy is 500, 600, or 700. On the other hand, I can get the simply expected simulation time for the scenario 2 that is following the rule of order of attack all to enemy because our troops can use all the power against the enemy at one time. As a result, it is obviously true that our troops which have the more power level than that of enemy must win the battle. Also, the more simulation time is required if the power level of enemy goes up until the power level of our troops. The next chart shows the variation of simulation time for the enemy's power level.

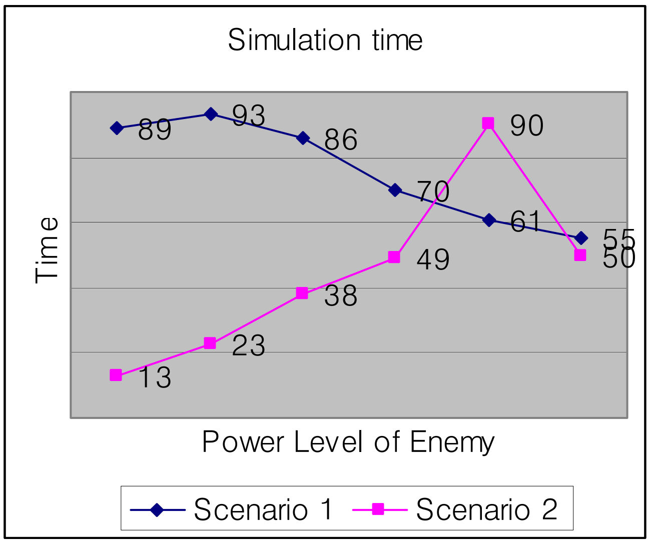


Figure 10. Simulation time for the several enemy's power level

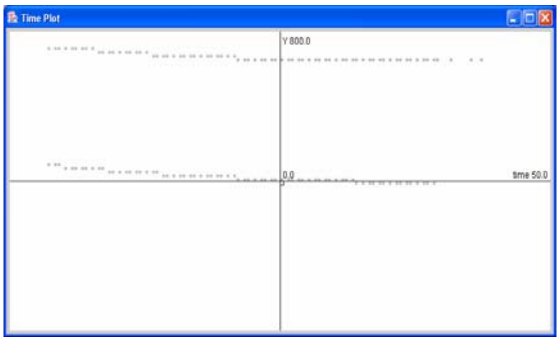
Power Level of Enemy	Battle Result of Scenario 1	Battle Result of Scenario 2
100	Win	Win
300	Win	Win
500	Loose	Win
600	Loose	Win
700	Loose	Win
800	Loose	Loose

Table 7. The battle result for the different enemy power level

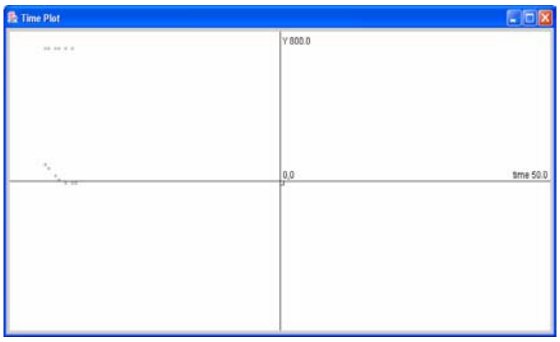
Second, I have to consider the battle result of the simulation. The above Table 7 shows the distinction of the battle results between scenario 1 and scenario 2. For the scenario 2, there is no something special for the battle result. It means that our troops win the battle if the power level of our troops is larger than the enemy's power level and loose the battle if the power level of our troops is less than the enemy's power level. However, I get the different battle result for the scenario 1 comparing to the scenario 2. When the enemy's power level is 500, 600, and 700, our troops loose the battle because our troops can't use all the power at one time against the enemy. Because it is very difficult to expect these results by mathematical or theoretical method, the simulation is very useful.

The next following figures show plots which come from the simulation. These figures represent the power level variation of our troops and enemy as time goes by and the battle result.

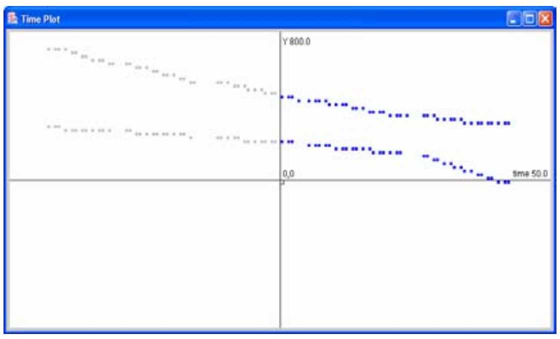
Scenario 1 : Power level of Enemy 100
Battle Result : Win



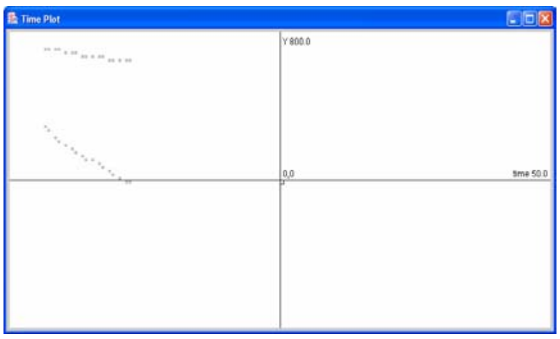
Scenario 2 : Power level of Enemy 100
Battle Result : Win



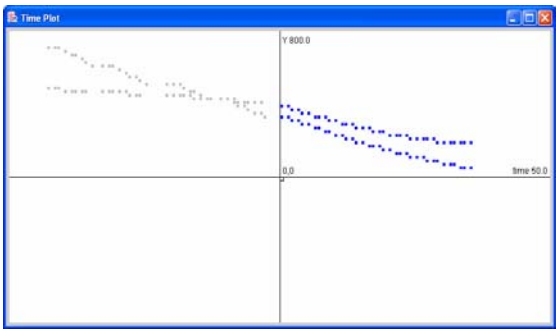
Scenario 1 : Power level of Enemy 300
Battle Result : Win



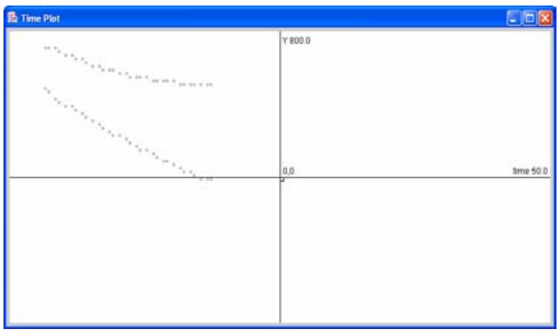
Scenario 2 : Power level of Enemy 300
Battle Result : Win



Scenario 1 : Power level of Enemy 500
Battle Result : Loose

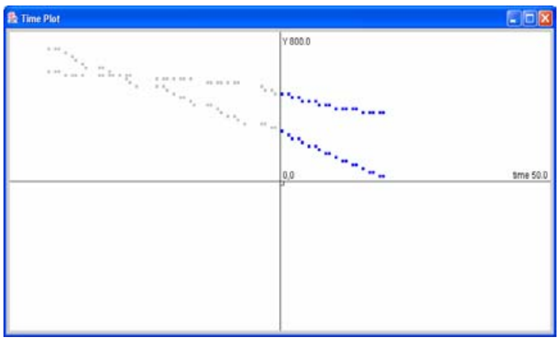


Scenario 2 : Power level of Enemy 500
Battle Result : Win

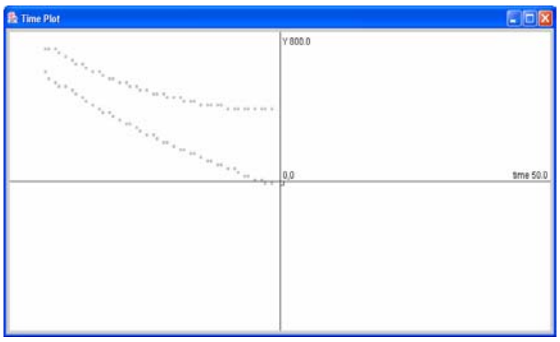


Scenario 1 : Power level of Enemy 600
Battle Result : Loose

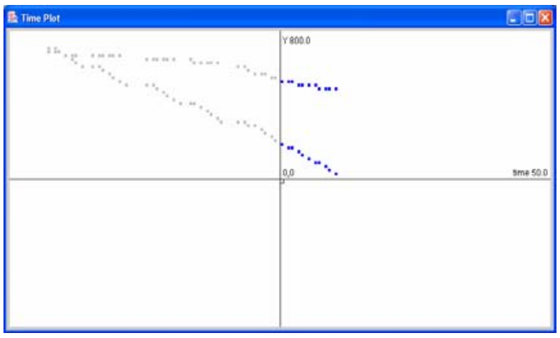
Scenario 2 : Power level of Enemy 600
Battle Result : Win



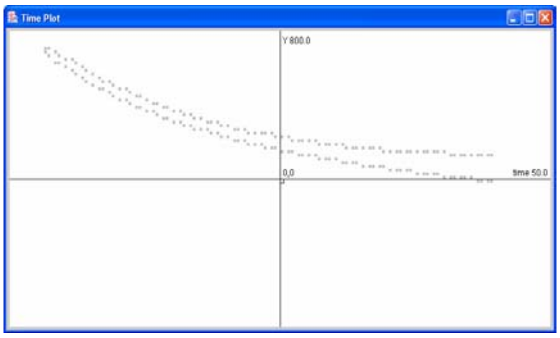
Scenario 1 : Power level of Enemy 700
Battle Result : Loose



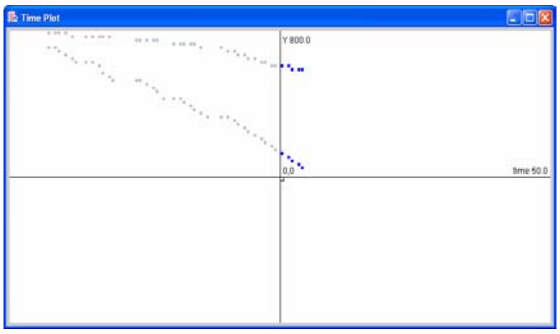
Scenario 2 : Power level of Enemy 700
Battle Result : Win

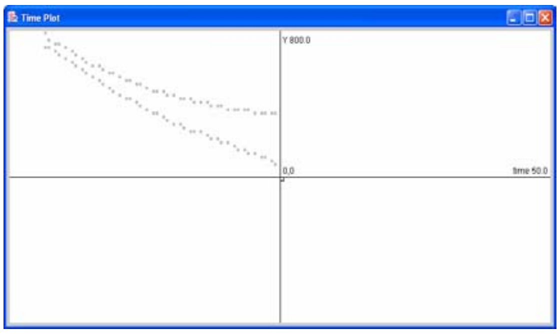


Scenario 1 : Power level of Enemy 800
Battle Result : Loose



Scenario 2 : Power level of Enemy 800
Battle Result : Loose





9. Conclusion and Future works

Modeling and simulation of the Battle Simulation is performed in this project. This Battle Simulation has two distinct scenarios. One is following the order of "attack one by one" and the other is following the order of "attack all to enemy". In general, the Battle Simulation is coupled with 3 models that are Division, Enemy, and Transducer model. Enemy and Transducer model are atomic and on the other hand, Division model is a coupled model that is consisted with two atomic models(Reconnaissance and Tank Unit model) and one coupled model(Regiment model) again. At last, Regiment model is coupled with two atomic models(Infantry Battalion and Artillery Battalion model). Such model hierarchy has been shown before in Figure 2.

Currently, this model can be simulated for the division commander to predict the result of combat for the given different situation and find the most effective way to attack enemy and win the battle in the real world environment. I have two scenarios right now. By selecting one of two scenarios, the simulation can get the different battle result for the same given state. This simulation lets the division commander know what kinds of battle result can be come for each scenario and give him the two choices to order his troops to attack enemy. This simulation works well. However, it is not an ultimate way but the guidance of attack. Because there are advantages and disadvantages for each scenario, it's very hard to say which scenario is good for the given environment. For example, the scenario 2 that is attack all to enemy has the advantages to use all the usable power level to enemy at one time and the disadvantage for defending against another enemy because there is no more extra power to defend. The decision is totally depending on the division commander.

The main objective of this project is the multilevel simulation. It means that everyone who participated in war game simulation knows if high and low echelon unit could practice at the same time, the reliability of military simulation for training will be severely increased. That is the reason to design multilevel simulation. In addition, the trend of future military simulation is synthesis among virtual, constructive and live simulation. I need to develop the current single level simulation to the multilevel simulation with more parameters that are reflection of the real world environment and a lot of choices to command of attack enemy to get the very precise battle result so that win the real battle.

By conducting this project for last few months, I can understand DEVS deeply and learn how to realize some M&S problems into DEVS models, simulate them, and analyze simulation results. Also, I can confirm the meaning of "modular and hierarchical model construction based on closure-under-coupling paradigm." It has been very valuable experience as my first simulation study. Lastly, it is very amazing to get the very reasonable precise result by simulating model that is designed by DEVSJAVA.