République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed KHIDER - BISKRA
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
Département d'Informatique

N° d'ordre :.....
Série :.....

Mémoire

Présenté en vue de l'obtention du diplôme de Magister en Informatique
Option**: Synthèse d'Images et Vie Artificielle**

Titre

Computer Simulation of an Immune Response against Virus Infection using Artificial Life Techniques

Présenté par: KHALDI Belkacem

Soutenu le: 26/04/2012

Devant le jury:

Djedi Nour Eddine	Professeur	Université de Biskra	Président
Cherif Foudil	Maitre de Conférences A	Université de Biskra	Rapporteur
Babahenini Med Chaouki	Maitre de Conférences A	Université de Biskra	Examinateur
Moussaoui Abdelwahab	Maitre de Conférences A	Université de Setif	Examinateur

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed KHIDER - BISKRA
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
Département d'Informatique

N° d'ordre : .....
Série : .....

Mémoire

Présenté en vue de l'obtention du diplôme de Magister en Informatique
Option **: Synthèse d'Images et Vie Artificielle**

Titre

Simulation d'une Réponse Immunitaire contre une Infection de Virus en utilisant les Techniques de la Vie Artificielle

Présenté par: KHALDI Belkacem

Soutenu le: / /2012

Devant le jury:

Djedi Nour Eddine	Professeur	Université de Biskra	Président
Cherif Foudil	Maitre de Conférences A	Université de Biskra	Rapporteur
Babahenini Med Chaouki	Maitre de Conférences A	Université de Biskra	Examinateur
Moussaoui Abdelwahab	Maitre de Conférences A	Université de Setif	Examinateur

Abstract: The simulation of the Immune System (IS) is extremely complex due to the high mechanisms and interactions existed behind it; however great models have been issued to better understand these mechanisms and interactions. Our work is motivated by this challenge for the ultimate aim to enrich the existing IS models that are recently taking place. The current presented work is an attempt to simulate the first humorol immune response initiated in the Lymph Node organ against both T-Independent and T-Dependent antigens. The model is developed under the AnyLogic simulation tool and it's a Multi-Agent system based model in which the behavior of the constitute agents are modeled using the Statecharts formalism. The results issued from our AnyLogic simulation respects several immunology experimentations (B-Cell activation, proliferation, differentiation and antibody generation).

Key words: Simulation, first humoral immune response, Lymph Node, AnyLogic, Multi-Agent system, Statecharts.

Résumé: La simulation du système immunitaire (SI) est extrêmement complexe due aux hautes mécanismes et d'interactions qui existent derrière lui; cependant des modèles considérables ont été élaboré afin de mieux comprendre ces mécanismes et interactions. Notre travail est motivé par ce défi pour le but d'enrichir les modèles existants du SI qui ont été récemment utilisés. Le présent travail est une tentative pour simuler la première réponse immunitaire humorol monté dans les lymphatiques organes contre les antigènes de type T-indépendants et celui de type T-dépendantes. Le modèle est un système multi-agent développé sous l'outil de simulation AnyLogic dans lequel les comportements de chaque agent du système sont modélisés en utilisant le formalisme Statecharts. Les résultats issus de nos simulation AnyLogic respectent plusieurs expérimentations immunologiques (activation des cellules B, la prolifération, la différenciation et la génération d'anticorps).

Mots clés: Simulation, première réponse immunitaire humorale, les ganglions lymphatiques, AnyLogic, System Multi-Agent, Statecharts.

الخلاصة : إن التحديات التي يمنحها محاكاة نظام المناعة بسبب الآليات والتفاعلات بالغة الصعوبة التي تميزه، أدت إلى تطوير العديد من نماذج الكمبيوتر لفهم أفضل لهذه الآليات والتفاعلات. و الدافع من وراء العمل الذي نحن بصدد تقديمه هو إثراء قائمة هته النماذج التي بدأت تأخذ مكانها في الآونة الأخيرة. إن بحثنا هذا هو محاولة لمحاكاة الاستجابة المناعية الخلطية الاولى ضد كل من مستضدات ذات النوع T-Independent و ذات النوع T-Dependent و التي تجري خصائصها في أعضاء العقد اللمفاوية. النموذج المقدم هنا هو نظام متعدد الوكلاء تم تطويره باستخدام أداة المحاكاة AnyLogic حيث أن سلوك كل وكيل تم نمذجته باستخدام تقنية Statecharts . و تظهر النتائج المتحصل عليها من خلال محاكاتنا لهته الاستجابة أن العديد من التجارب المناعية تم احترامها على غرار (تنشيط الخلايا B، تكاثر الخلايا و تمايزها، وتوليد ألأجسام المضادة).

الكلمات المفاتيح : المحاكاة، الاستجابة المناعية الخلطية الأولى، العقدة الليمفاوية، AnyLogic، نظام متعدد الوكلاء ، Statecharts.

Acknowledgements

First of all, Praise and thank to Allah for helping us to complete this thesis; which after more than two years of work, heir it is finally, it contains more than I hoped for even one year ago. Without the support of many colleagues, friends and relatives, I had not finished it.

I would very much like to thank my promoter and daily supervisor, Dr CHERIF Foudil for his acceptance to work with me within the subject of this thesis, I also appreciate his guidance and support during the preparation of my thesis.

I would like to show my gratitude to Pr. David Harel, Pr. Fillipo Castiglione and Mrs. Hila Amir-Kroll for their advices, documentations and published papers sent to me.

Great recognition also would be shown to the AnyLogic Technical support team for their helpfulness; they have always supporting my work by extending my Evaluation License as I request it.

I would also like to acknowledge the committee members (Pr. Djedi Nour Eddine, Dr. Babahenini Med Chaouki and Dr. Moussaoui Abdelwahab) for their reading and kind attention.

From the non-scientific side, I would like to put my relatives: to my parents, who supported and encouraged me. I thank them for their love and their education, to my darling wife for standing by me all throughout the course of my preparation, and to my cheer brothers and sisters for their supports.

Finally, I hope never to forget the One who stands at the beginning and end of everyone and everything, without whom we could not do anything.

Table of Contents

Abstract ..... **i**

Acknowledgments ..... **ii**

Table of Contents ..... **iii**

List of Figures ..... **vi**

List of Tables ..... **viii**

Introduction **1**

1. Motivation: Immune system features 1
2. Simulating the Immune system 2
3. Contributions 4
4. Organization 5

Chapter I: Immunology background **6**

1. The task of the immune system 7
2. Functional Elements of the Immune System 7
   - 2.1. Organs 7
     - 2.1.1. Bone Marrow 8
     - 2.1.2. Thymus 8
     - 2.1.3. Spleen 8
     - 2.1.4. Lymph Node 9
   - 2.2. Immune Cells and Molecules 9
     - 2.2.1. B-Cells 9
     - 2.2.2. Antibodies 10
     - 2.2.3. T-Cells 11
     - 2.2.4. Major Histocompatibility Complex (MHC) 11
     - 2.2.5. Phagocytes and Their Relatives 12
   - 2.3. Layers of the Immune System 13
     - 2.3.1. Anatomic Barrier 13
     - 2.3.2. Innate Immunity 13

2.3.3. Adaptive Immunity 14
3. Overview of Humoral Immunity Response 15
3.1. Response to T-Dependent Antigens 16
3.2. Response to T-Independent Antigens 17
3.3. Process of Lymphocyte recirculation 17
3.3.1. Structure of Lymph 18
4. Summary 19

Chapter II: Computational Models for Immune System 20

1. The purpose of modeling the immune system 21
1.1. For biological researchers 21
1.2. For computer researchers 21
2. Approaches for modeling the immune system 22
2.1. Top-down approaches 22
2.2. Bottom-up approaches 22
3. Related methods for modeling the immune system 23
3.1. Differential Equation Based Models 23
3.2. Cellular Automaton (CA) based Models 25
3.2.1. Related Immune system simulators based on CA 27
- ImmSim 27
- C-ImmSim, ParImm, SimTriplex and ImmunoGrid 29

3.3. Agent-Based Models (ABM) 31
3.3.1. Related Immune system simulators based on ABM 32
- SIMMUNE 32
- CyCells 33

3.4. Reactive Animation based modeling 34
4. Comparison between different approaches 37
5. Summary 39

Chapter III: Statecharts based Behavior for Agent Based Modeling 40

1. Overview of Multi-Agent Based Modeling 41
1.1. Simulation of Multi-Agent Based Models 42
1.1.1. The Behaviors Module 42
- What is an agent? 42
- Agent Architectures 44
- Interaction 45
- Modeling the agent behavior 46

1.1.2. The Environment Module 47
1.1.3. The Scheduling module 48
2. Overview of the Statecharts Formalism 50
2.1. What are Statecharts? 50
2.2. Basic concepts of Statecharts 51
2.2.1. States 51
2.2.2. Transitions 52
2.3. Advanced concepts of Statecharts 53
2.3.1. The hierarchy of States (nested states) 53
2.3.2. Zooming-in & zooming-out 53
2.3.3. Concurrency (orthogonal states) 54

2.3.4. Connectors 55
3. Statecharts based Agent behavior 57
4. Summary 59

Chapter IV: The AnyLogic Simulation of the first Humorol Immune response against an antigen infection 60

1. The AnyLogic Simulation Tool 61
1.1. AnyLogic Features 61
1.2. AnyLogic Modeling Framework 63
1.3. Agent-Based modeling in AnyLogic 64
1.3.1. Agents 65
1.3.2. Space 65
1.3.3. Communication between Agents 66
1.4. AnyLogic Interfaces 66
2. Model of the Lymph Node humorol immune response 67
2.1. Modeling the Time 70
2.2. Modeling the LN environment 70
2.3. Modeling the Immune Cells agents 71
2.3.1. The Lymphocyte Agent 73
2.3.2. The Plasma Agent 76
2.3.3. The Antigen Agent 77
2.3.4. The Antibody Agent 78
2.4. The Main Classes 80
3. Results, discussion and future work 82
3.1. Results 82
3.2. Discussion & Future works 91
4. Summary 94

Conclusion 95

Bibliography references 98

List of Figures

Fig 1.1 The functional components of the biological immune system 8
Fig 1.2 Immune cells that contribute to immune response 10
Fig 1.3 Structure of an antibody 11
Fig 1.4 Types of phagocytes 12
Fig 1.5 Primary & secondary immune response process 16
Fig 1.6 B-Cell activation process 17
Fig 1.7 Lymph Node schematic structure 19

Fig 2.1 Time development of agent concentration in the immune system model 24
Fig 2.2 A 'chronic infection' settled down to an immune system model with very low concentrations of I 24
Fig 2.3(a) Von Neumann neighborhood 26
Fig 2.3(b) Moore neighborhood 26
Fig 2.4 Cellular Automata Update from time step t-1 to t 26
Fig. 2.5 An overview of the ImmSim model 28
Fig. 2.6 A schematic picture of an APC, a B-cell and a T-cell ImmSim model 29
Fig. 2.7 A schematic picture of antigen and antibody in the ImmSim model 29
Fig. 2.8 Reactive Animation Simulation of organ level in three models 36

Fig.3.1 An agent-based model: The micro level entities, their actions and interactions, and the environment 41
Fig.3.2 MAS Modules 42
Fig.3.3 An agent in its environment 43
Fig. 3.4 An agent as a three phases process 46
Fig. 3.5 Discretized environment 48
Fig. 3.6 Continuous approach for the perception of the agents 48
Fig. 3.7 Simulating a MAS as three modeling modules 49
Fig.3.8 (a) Statecharts composed of: initial state, Simple state A, composed state P and final state 51
Fig.3.8 (b) FSM composed of: initial state S, 4 transit states A,B,C and D and final state F 51
Fig. 3.9 Different types of Statecharts 51
Fig. 3.10 Statecharts transitions 52
Fig. 3.11 Statecharts hierarchy (or-states decomposition) 53
Fig. 3.12 Zooming-in and zooming-out capabilities 54
Fig.3.13 Statecharts concurrency (and-states decomposition) 55
Fig 3.14 Different types of connectors 56

Fig. 4.1 AnyLogic screenshots 62
Fig. 4.2 AnyLogic modeling approaches 62
Fig. 4.3 AnyLogic Framework Architecture 64
Fig. 4.4 AnyLogic main interface 66
Fig. 4.5 AnyLogic visual simulation language 67
Fig. 4.6 Simplified illustration of the Humerol immune response against T-Ind. Antigen 68
Fig. 4.7 Simplified illustration of the Humerol immune response against T-Dep. Antigen 68

Fig. 4.8 Simplified view of the process launched in the LN 69
Fig. 4.9 The AnyLogic simulation LN zones 71
Fig. 4.10 The Statecharts behavior of the Lympocyte Agent 75
Fig. 4.11 The Statecharts behavior of the Plasma Agent 77
Fig. 4.12 The Statecharts behavior of the Antigen Agent 78
Fig. 4.13 The Statecharts behavior of the Antibody Agent 79
Fig. 4.14 The LN simulation main class 81
Fig. 4.15 The LN main root class 81
Fig. 4.16 The initial running experimental simulation 84
Fig. 4.17 The initial running simulation of the root main class 85
Fig. 4.18 A number of T-Independent antigens (green color) are added to the simulation (Entering the LN via afferent area) and then are starting moving 85
Fig .4.19 A simulation of a proliferation phase of an humoral immune response against a T-Inpependent Ag 86
Fig .4.20 A simulation of a differentiation phase of an humoral immune response against a T-Independent Ag 86
Fig .4.21 A simulation of a complete clonal expansion phase (processed in a 3 FCs zones) of the humerol immune response against a T-Indep Ag 87
Fig .4.22 A simulation of a plasma secreting antibodies phase (processed in a Medullar Cords zone) of the humerol immune response against a T-Indep Ag 87
Fig. 4.23 The current highlighted active state for the first T-Independent B-Cell at run time 88
Fig .4.24 Highlighted current active states for different running simulated agents 90
Fig .4.25 Snapshots from the simulation run time of the first humoral immune response against a T-Dep Antigen 91

List of Tables

Table 1.1 Biological defense mechanisms 13

Table 2.1 A comparative overview between different immune system modeling methods 37

Table 4.1 List of the used functions that control the movement of cells from and into LN areas 71
Table 4.2 Properties and functions of the Lymphocyte Agent 74
Table 4.3 Properties and functions of the Plasma Agent 76
Table 4.4 Properties and functions of the Antigen Agent 78
Table 4.5 Properties and functions of the Antibody Agent 79
Table 4.6 Initial values of the parameters used during the experimental simulation 84
Table 4.7 Rate values of the experimental simulation 84

Introduction

The human immune system is one of the most complex, adaptive, highly distributive learning systems that exist in nature. This amazing system operates at multiple levels [1] (molecules, cells, organs, and organisms); for the ultimate goal to defend the human body against foreign pathogens. Each human individual has tens of immune system organs and some 10^{12} immune cells that belong to multiple types and subtypes. These cells produce molecules that act as initiators, regulators, and effectors of the immune function.

The powerful information-processing capabilities of the immune system, such as feature extraction, pattern recognition, learning, memory, and its distributive nature provide rich metaphors for its artificial counterpart. From 1985 to now there has been an increased researches interest in immunity-based techniques and their applications. Some of these models [2] are intended to describe the processes that occur in the immune system to have a better understanding of the dynamic behavior of immunological processes and simulate immune system's dynamic behavior in the presence of antigens/ pathogens.

1. Motivation: Immune system features

The immune system is characterized by several features that have let great researches to be interested in. The following ones which are cited in the work of [3] seek to motivate what takes to formally modeling the behavior of the immune system:

An immune system is a highly distributed communicating system: An immune system consists of a number of independent components (for example cells, antigens, etc) in which each element performs a small, specific task and coordinates one another. The coordinated individual components play an important role in the

production of the immune system behavior via the very high number and kinds of interactions between them.

An immune system is diverse: cells; that are implicated in the immune response and belong to the same class; differ from one another in a manner that entails a cell's ability to non-self detection. Hence the noticeable thing for one might be unnoticeable for others. As consequence an immune system is highly robust due the diversity of its cells.

An immune system is mobile: the raison that the constitute immune system are on continuous circulating in the body aiming to raise its possibilities to successfully detect antigens and to accomplish its recognition abilities; yields to the mobility feature which is an essential for performing the immune system.

Also an immune system is highly dynamic: the dynamic feature of the immune system resides in the changes of the components number of the same kind, as well as the structure of the entire system; that are noticed over time and with regards to circumstances. For instance, some cells may live a few days; other will survive much longer.

2. Simulating the Immune system

Simulating the immune system is a challenge that involves multidisciplinary efforts. In immunology [1], simulators are used to get answers to a variety of questions, including understanding the general behavior of the immune system, the causes of disease, effects of treatment, analysis of cellular and molecular interactions, and simulation of laboratory experiments. Models and simulations have proven useful in studying the roles of single constituents and simple interactions, planning of experiments, testing theoretical assumptions, and even highly abstract tasks such as suggesting theory modifications.

Immune system simulators will be required if there is to be a significant increase in the generation of new immunological ideas, because computational simulation is considerably faster than lab experiments. Simulations may even allow researchers to answer questions that are impossible to explore using wet lab techniques, such as the total interactions of an individual B cell during its lifetime. Normally, however, simulations will be used to assess the validity of immunological theories.

Several works have been taken place to carry out the simulation of the immune system; all of them don't seek to simulate the whole phenomena launched during an immune system response but only a part of it is taken into account due to the very height complexity of the immune system. The most important works are around four main models: (1) the Differential Equations (DE) based models, (2) the Cellular Automata (CA) based models, (3) the Agents based models (ABM) and (4) the Reactive Animation (RA) based models.

The DE based models are traditional top-down approaches based on continuous simulation technique in which a system is modeled by a set of mathematical differential equations that intend to give an average behavior of the modeled system; so only a minimum number of homogenous entities of the system are carried out. These models are difficult to be used in such situation when the behavior of the modeled system is non-linear.

The CA based models are bottom-up approaches seeking to simulate homogenous agents in the microscopic level. The systems to be modeled via these methods are considered as fully discretized dynamical systems based on local interactions; thus space, time, and states of the modeling system are discrete. The resulting simulation that is in exponential increase with the number of entities is generated by a synchronous update of all entities (cells) on the regular spatial lattice.

The ABMs are also bottom-up approaches that are mostly used to simulate a large number of agents interacting with each other in both synchronous / asynchronous way. In this modeling method that has wider model scope than CA based one: Agents are heterogeneous and they are specified at individual level (microscopic); they can sense their environment and can also change the state of its environment

Finally the RA technique which has been recently used to simulate reactive systems those are on continuous interactions with their environments by the use of inputs and outputs. The RA technique seeks to combine state-of-the-art reactivity and state-of-the-art animation by linking two modeling tools: the Statecharts formalism with a front-end animation tools. The Statecharts technique is a visual formalism that can be considered as a combination between bottom-up and top-down approach for the raison that they can enable us describing systems at multiple levels, and zooming

in and zooming out between these levels. The resulting executable reactive machine code generated by compiled Statecharts has been then combined with front-end animation engines such as flash engine or other 3D engines to give more realistic visualization simulation.

3. Contributions

In our work we have tried to take advantage of the different models that have been cited above aiming to give a first attempt to model the immune system response using a Statecharts based agent model in our laboratory. The model proposed in this essay is based on the AnyLogic simulation tool which proved to be a sufficient platform for the modeling of Multi-Agents systems as it allows a convenient Statecharts and state events implementation.

In our proposed model, we have modeled the behavior of the humorol immune first response mounted in the Lymph Node (LN) against both the T-Dependent and the T-independent antigens. The whole system is viewed as a multi-agents system in which: the agents behaviors are Statecharts based models and the interactions between constitutes agents are based on events. The proposed model; that has been developed using the AnyLogic simulation tool under an Evaluation License only; is inspired from the model of the LN B-Cell immune response proposed in the wok of [4] which is based on the RA technique; our model re-take parts of its suggested Statecharts based behavior of the B-Cell Immune Response and adapt it in the AnyLogic simulation tool.

The work cited in this thesis doesn't simulate the whole immune humoral immune response process launched as an antigen is encountered in LN; the profound levels such as the details of the communication signals between B-Cells and encountered antigens and those between B-cells and T-Helper Cells are not taken into account due to the high complexity of these immune interaction. Contrary to the work suggested in [4] in which the proposed Statecharts have a concurrent state (also called orthogonal state; which is one of the most advanced features of the Statecharts formalism that allow the concept of parallelism); ours doesn't support this concept for the raison of the limitation of the AnyLogic simulation tool which is in its actual version doesn't support orthogonal Statecharts. However we have simulated this feature on profiting from the ability of the AnyLogic simulation tool to create multi-

statecharts for the same agent. These multi-statecharts can be executed on parallel manner with the same execution fashion of orthogonal states.

The AnyLogic model of the B-Cells LN first humorol immune system response suggested in this essay; isn't only the first challenge that has been initiated in our laboratory but also it's the first attempt to carry out the simulation of an immune response using the AnyLogic tool.

4. Organization

The present thesis is organized in conventional manner around four chapters in which the first one aims to give an overview of the immunology background; the functional of the immune system, its layers, its different organs and components are briefly presented in this chapter. The second chapter is focusing on the different computational models that have been taken place in order to model and simulate the immune system, a comparison between the different models are also presented at the end of this chapter. The third one seeks to present an overview of the combined model to be adopted in the present work. a presentation of the AnyLogic simulation environment and its features are viewed in the first section of the fourth chapter; which illustrates in a deep fashion our AnyLogic proposed model for the B-Cells LN first humoral immune response; at the end of this chapter a discussion of the resulting simulation results, suggestions and future works are given.

Chapter I

Immunology Background

Immunology is a domain looking first of all to understand how the human body protect himself face to the various micro-organisms present in the environment [5]. It's considered as one of the most domains that provide a research challenge due to the complexity, the adeptness and the high distributiveness features that it characterizes.

In this chapter we are looking for giving an immunology background; in which we present an overview of the main immune system components (organs, cells, molecules …) and their functionalities, followed by illustrating a detail process of mounting a Lymph Node humoral immune response against T-Dependent and T-Independent antigens.

1. The task of the immune system

The immune system is considered as a multi-layer system that protects the body against disease; it's made up of a network of cells, tissues, and organs that work together in a complex and highly regulated way to drive back foreign organisms from the body [6]. The crucial cells that are involved include white blood cells, or leukocytes, which come in two basic types that combine to seek out and destroy disease-causing organisms or substances [7].

The features that characterize the immune system make it amazingly complex; such features are [8]: its capability to remember and recognize millions of different enemies, in addition its remarkable ability to distinguish between the body's own cells (Self) and foreign cells (non-self).

The body's immune system is triggered immediately when the immune defenders encounter cells or organisms that are recognized as "foreign". The foreign cell (which can be a microbe such as a virus or a part of microbe such as molecule) is called "antigen" [8]. The process of attack that is initiated instantly by the immune system is than called "immune response".

2. Functional Elements of the Immune System

As we have mentioned in the previous section, the immune system is a collection of *organs*, *cells*, and *molecules* responsible for dealing with potentially harmful invaders. The following section illustrates the most implicated immune components during an immune response.

2.1. Organs

The immune system is composed of two classified organs [Fig 1.1]: central lymphoid organs and peripheral lymphoid ones [9]. The central lymphoid organs participate in the generation and the assistance of the mature lymphocytes. The organs that are involved are the bone marrow and the thymus. Whereas the peripheral lymphoid organs aim to facilitate the interaction that are taken place between lymphocytes and antigens. These organs incorporate lymph nodes, the spleen, and mucosal and submucosal tissues of the alimentary and respiratory tracts**.**

In the below section a brief overview of each of these organs is presented:

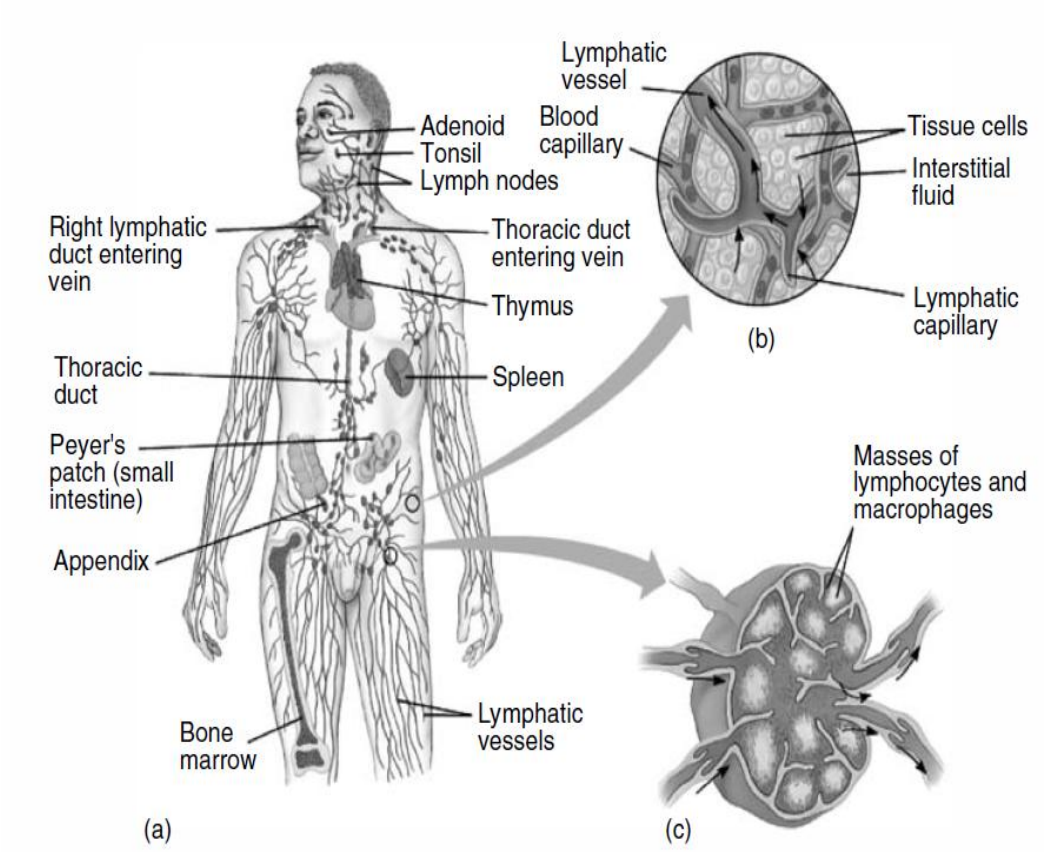


Fig 1.1: the functional components of the biological immune system **[9]**.

2.1.1. Bone Marrow

The bone marrow is the organ where naïve immune cells are initially generated and then derived through the process of hematopoiesis [9]. During this process, two distinguished classes of stem cells are produced: either matures immune cells that perform immunological function, or precursors of cells that migrate out of the bone marrow for the continuity of their maturation process away. The bone marrow also generates B-Cells, natural killer cells, ranulocytes, and immature thymocytes.

2.1.2. Thymus

For abstraction raison [9], the thymus organ is where mature T-Cells are formed. These mature T-cells which are beneficial to the immune response are then released into the bloodstream to perform immunological functions.

2.1.3. Spleen

It's in this organ that [9] foreign substances are captured after they have been brought by migratory macrophages and dendritic throw the bloodstream that enters the spleen.

This organ can be viewed as an immunological "conference center" in which B cells become activated and generate large numbers of antibodies in one of its factories, called the general center. Moreover, old red blood cells are destroyed in the spleen.

2.1.4. Lymph Node (LN)

LNs are distributed throughout the whole body and they are considered as immunologic filter [9] for the lymph fluid. Alike the spleen, an immune response is initiated in the LN via macrophages and dendritic cells that capture and present antigens to T-Cells and B-Cells.

2.2. Immune Cells and Molecules

It's in the bone marrow that all the immune cells are generated as an immature stem cells [7] which than respond to different cytokines and other signals to develop into either ***myeloid*** progenitor cells or into ***lymphoid*** progenitor ones.

The cells that are involved by the myeloid progenitor cells are: *monocytes*, *macrophages*, *neutrophils*, *eosinophils* and *basophils.* Their response to infection is initiated in a premature and nonspecific manner. By Contrast; the lymphoid progenitor cells become small white blood cells known as ***lymphocyte*** which initiate lately a response to infection. B-Cells and T-Cells are the two major classes of this type of progenitor cells.

The [Fig 1.2] shows a hierarchical diagram of the foremost cells that take part in immune response. The following section describes them in some details**.**

2.2.1. B-Cells

B-Cells are specialized lymphocytes that are responsible to generate and secrete substances called *antibodies*, which bind to specific *antigens*. The process of producing these antibodies begins when a B-Cell [8] encounters its triggering antigen and becomes an activated B-Cell which then proliferates and differentiates into either

large cell known as *plasma cell* or into *memory B-Cell*. Each plasma cell [9] is responsible for manufacturing millions of identical antibody molecules and pours them into the bloodstream to attack infected cells.

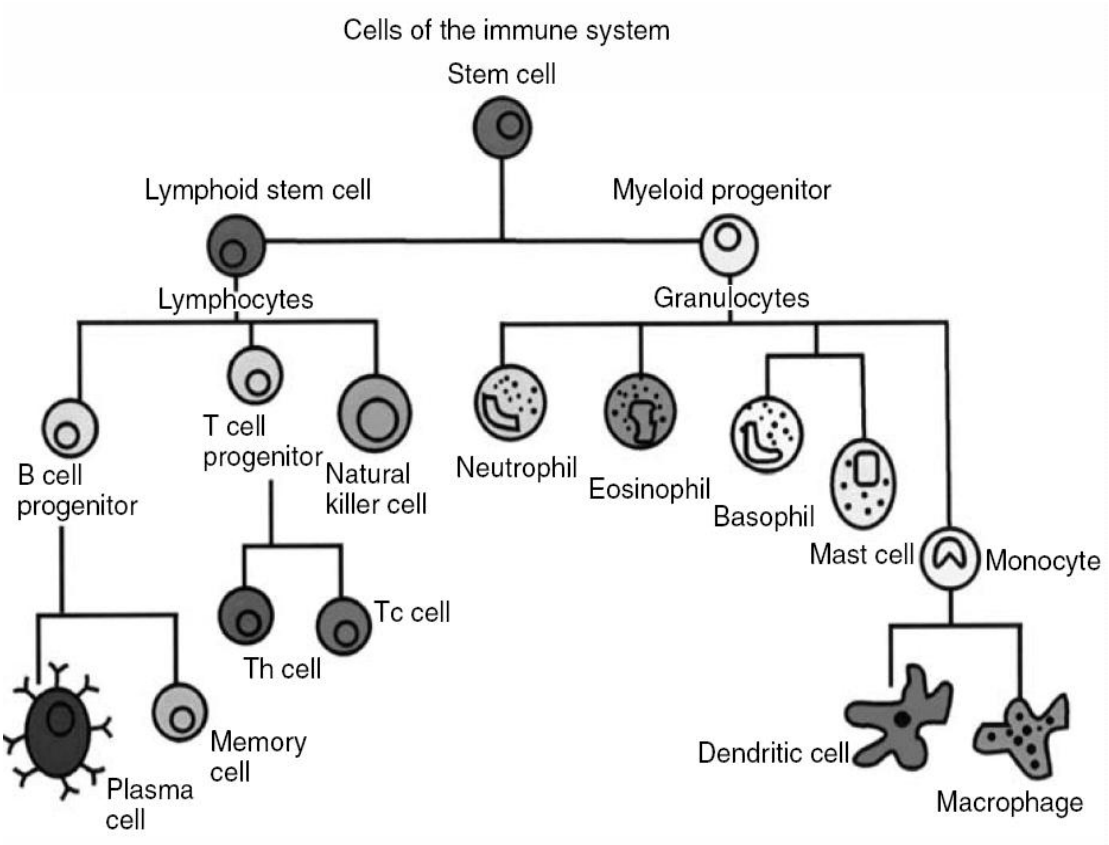


Fig 1.2: Immune cells that contribute to immune response **[9]**.

2.2.2. Antibodies

Antibodies (Abs) are specific molecules that are produced by plasma cells and that they belong to the *immunoglobulin (Ig)* molecules class.

As it's mentioned in the [Fig 1.3]; each antibody is made up of two heavy chains and two light chains [8]. An antigen is recognized when it matches to the antigen variable region, which differs from one antigen to another.

Scientists have identified nine chemically distinct classes of human *immunoglobulins* [8, 9] that play different roles in the immune defense strategy, four kinds of **IgG** and two kinds of **IgA**, plus **IgM**, **IgE**, and **IgD**.

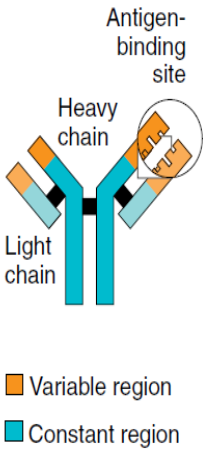


Fig 1.3: Structure of an antibody **[8]**.

2.2.3. T-Cells

T-Cells are particular kind of immune cells that contribute in the immune defences in two essential ways [8]: some direct and stabilize immune responses; others directly attack infected or concerous cells. The major subpopulations of these T-Cells are *Helper T-Cells* (*Th cells*), *Killer T-Cells* (*cytotoxic T Lymphocytes*)and *Memory T-Cells*.Th cells contribute in the immune response by stimulating nearby B-Cells to form antibodies, or by participating in activating other T-Cells. Killer T-Cells directly attack cells that hold certain abnormal molecules on their interfaces. Wheras Memory T-Cells form a pool that will remember earlier immune responses.

2.2.4. Major Histocompatibility Complex (MHC)

MHC molecules are known as proteins [8, 9] that; for the raison to provide a recognizable scaffolding to present a foreign antigen to the T-Cells; they bring sites polypeptides to the cells' exterior surface from inside the cells they are a part of.

The two most important MHC proteins classes that are mentioned in this essayare [9]:

- *MHC I molecules:* they exist in almost every nucleated cell of the body playing a role in presenting antigens to cytotoxic T-Cells.
- *MHC II molecules:* only a few specialized cells types; such as macrophages, dendritic cells, activated T and B cells; can hold this MHC class which presents antigens to helper T-Cells.

2.2.5. Phagocytes and their relatives

Phagocytes are large kind of white cells that are able to eat and absorb microbes and other foreign particles [9]**.** They are categorized into *monocytes, macrophages, granulocytes, neutrophiles, eosinophils and basophils*.

Each category has certain functionalities [8]. For example [Fig 1.4] monocytes develop into macrophages when they migrate into tissues; macrophages act as hunters: riding the body of used up cells and others rubbish, displaying some foreign antigens with drawing the attention of matching lymphocytes, as well churning out a remarkable variety of powerful chemical signals (*monokines*) that are vital to the immune responses; granulocytes contribute mostly in inflammatory and allergy responses, they destroy microorganism with the assist of its granules full of potent chemical; neutrophiles ingest microbes and kill them using its prepackaged chemical; eosinophils and basophiles are kinds of granulocytes that "degranulate," spraying their chemicals onto harmful cells or microbes nearby.

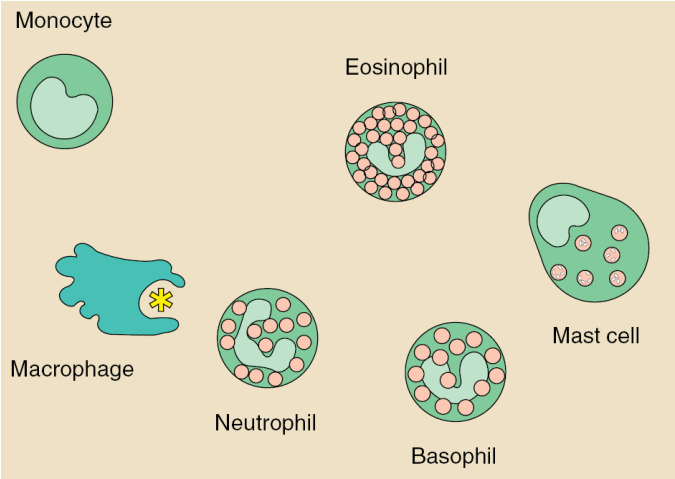


Fig 1.4: types of phagocytes **[8]**.

2.3. Layers of the Immune System

The immune system can be considered as a multilayer system in which different types of defense mechanisms are the constitute component of each layer. Biologic scientists have distinguished three major layers [9]: the anatomic barrier, the innate immunity, and the adaptive immunity. These layers can be classified [Table 1.1] into two main categories: nonspecific and specific defense mechanisms. In the nonspecific category; the same type of immune response is generated for any pathogen entering the body. Whereas in the specific category; a process of recognizing particular pathogens is launched.

Nonspecific defense mechanisms		**Specific defense mechanisms**
First line of defense: Anatomic barrier	**Second line of defense: Innate immunity**	**Third line of defense: Adaptive immunity**
• Skin • Mucous membranes • Secretions of skin and mucous • membranes	• Phagocytic white blood cells • Antimicrobial proteins • Inflammatory response	• Lymphocytes • Antibodies

Table 1.1: Biological defense mechanisms **[9]**.

In the section below, a brief description of each layer is presented:

2.3.1. Anatomic Barrier

Although the anatomic barrier; which is composed of the skin and the surface of mucous membranes; is the first defense line that emphasizes pathogens and inhibits most bacterial growth, many other pathogens can escape this layer and enter the body by penetrating through the mucous membranes.

2.3.2. Innate Immunity

Innate immunity [10] refers to all defense mechanisms against foreign pathogens that individuals are born with. Innate immunity is mainly composed of the following mechanisms:

- Phagocytic barriers. Some specialized cells (like macrophages, neutrophils, and natural killer cells) are able to ingest foreign substances, including whole pathogenic microorganisms. This ingestion has two purposes: to kill the

antigen and to present fragments of the invader's proteins to other immune cells and molecules.

- Inflammatory response. Activated macrophages produce cytokines (hormone like protein messengers), which induce the inflammatory response characterized by vasodilatation and rise in capillary permeability. These changes allow a large number of circulating immune cells to be recruited to the site where an infection occurs.

2.3.3. Adaptive Immunity

Adaptive immunity [9], also called acquired or specific immunity, represents the part of the immune mechanism that is able to specifically recognize and selectively eliminate foreign microorganisms and molecules.

Adaptive immunity produces two types of responses in the presence of pathogens: humoral immunity and cellular immunity. The humoral immunity is based on the synthesis of antibodies by B cells; however, in cellular immunity, T cells cause the destruction of microorganisms that carry invading antigens and those self-cells that have been infected.

- **Humoral immunity**. Humoral immunity is mediated by antibodies contained in body fluids (known as humors). The humoral branch of the immune system involves B cell/antigen interaction, and the subsequent proliferation and differentiation of B cells into antibody-secreting plasma cells. Antibodies function as effectors of the humoral response by binding to antigens and facilitating their elimination.
- **Cellular immunity**. Cellular immunity is cell-mediated; thus, effectors T cells, generated in response to an antigen, are responsible for cell-mediated immunity. Cytotoxic T lymphocytes (CTLs) participate in cell-mediated immune reactions by killing altered self-cells; they play an important role in killing virus-infected- and tumor cells. Cytokines secreted by TDH cells can mediate cellular immunity, and activate various phagocytic cells, enabling them to kill microorganisms more effectively. This type of cell-mediated immune response is especially important in host defense against intracellular bacteria and protozoa.

3. Overview of Humoral immunity Response

The Humoral immunity refers to the production of antibodies and the accessory processes that accompany it in response to an antigen. It's mediated by secreting antibodies which are produced in the lymph node via plasma-cells which are also generated by activated B-cells [10].

The humoral immunity process begins when an encountered antigen is recognized in the body. At this moment and relating to how often the encountered antigen is recognized; one of two responses might be mounted: the primary antigen response or the secondary antigen response one. These two antibodies responses differ quantitatively and qualitatively.

The schema illustrated in the [Fig. 1.5] shows a general overview of the initiating process for those responses. The primary antigen response results from the first exposure to a microbe or an antigen which lead to the activation of unstimulated naïve B lymphocytes [10],[11]. These activated B cells enter so on into the phase called clonal Expansion where large clones of identical cells are produced; the proliferating cells will than differentiate into antibody-producing plasma cells and memory cells. Some of the antibody producing cells migrate to the bone marrow and live in this site for several years; the others circulate in the blood and participate in the process of destructing or neutralizing antigens.

The secondary antigen response is due to the stimulation of memory B cells triggered by the second exposure or more of an antigen [10, 11]. The activated memory B-cells take again the same cycle of proliferating and differentiating processed in the first response with a high level of protection.

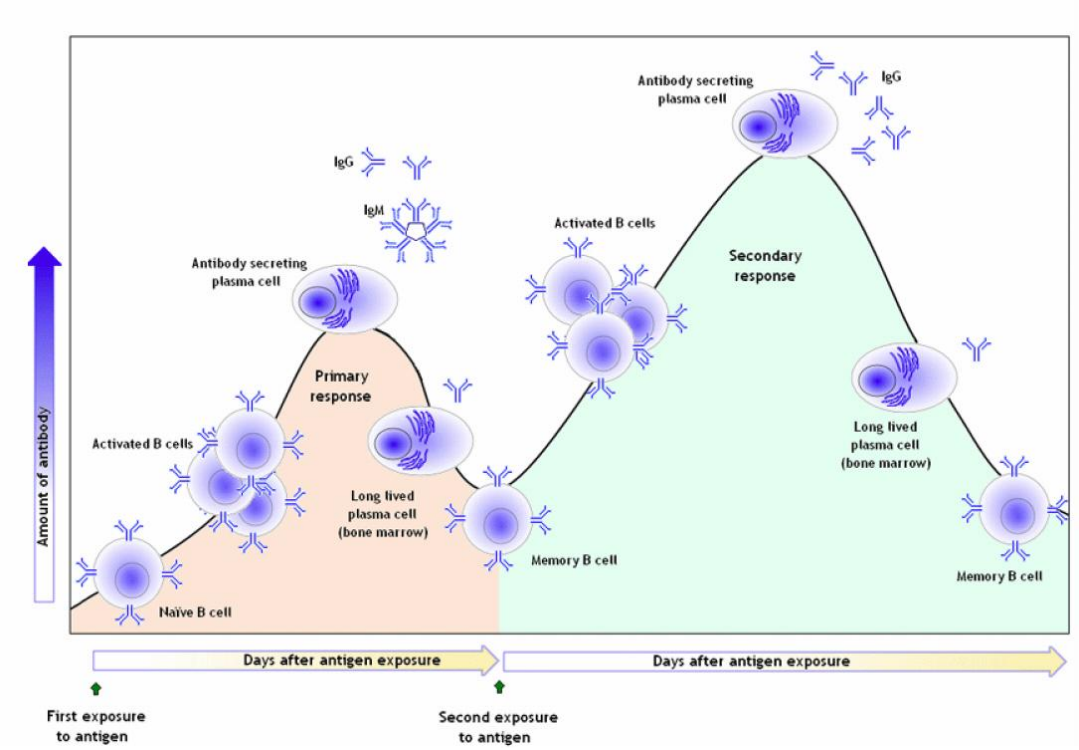


Fig 1.5: Primary & secondary immune response process **[11]**.

The humoral immunity might be stimulated via two kinds of antigens: one is called a T-dependent antigen and the other is known as a T-independent one:

3.1. Response to T-Dependent Antigens

T-dependent antigens, which are the most existing ones, are those that need the implication of the T-helper cells (Th4) in the process of mounting an humorol response. This means, cooperation between the antigen-specific B cells and T lymphocytes must be processed in order for naive B-lymphocytes to proliferate, differentiate and mount an antibody response against these T-dependent antigens.

The entire process is illustrated in [Fig. 1.6]: the interaction between antigens and both helper T-Cells and B-Cells involve sequentially antigen presentation to both T-Helper Cells (via antigen presentation cells (APCs) in conjunction with MHC II molecules) and B-Cells (via B-Cells Receptors (BCRs)) which also express antigen-MHC complexes) [11],[12].

The interaction between helper T-Cells and B-Cells entail the activation of the B cells which enter the cycle of proliferating and differentiating to plasma-cells and memory-cells. This interaction induces a sequence of surface receptor binding and cytokine production that results in B-cell activation, proliferation and differentiation.

(1) Binding of the T cell receptor (TCR) to MHC induces the T cell to produce CD40L, which binds to CD40 on the B cell, producing a major stimulatory signal. (2) CD28 on the T cell then interacts with B7 on the B cell (co-stimulatory signal). Cytokines are also involved.

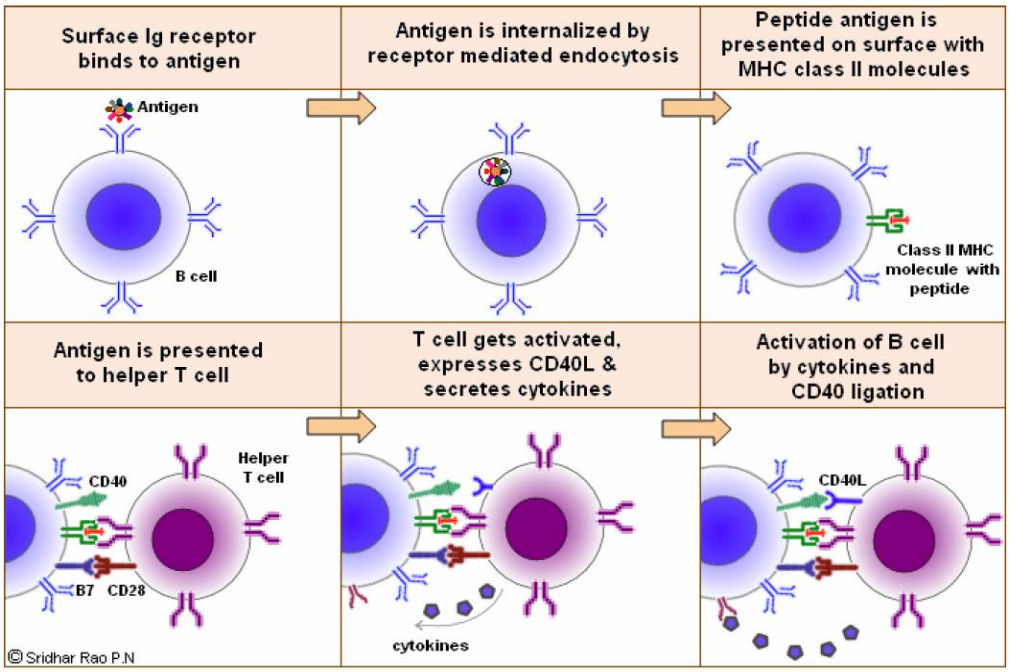


Fig 1.6: B-Cell activation process **[11]**.

3.2. Response to T-Independent Antigens

Contrary to T-Dependent antigens, the T-Independent ones are those that they stimulate mounting the humorol immune response in the absence of Helper T-Cells. The process of activation, proliferating and differentiating of B-Cells is than launched directly without the requirement of helper T-Cells. This is caused by the fact that many non-protein antigens such as polysaccharides and lipids can't be recognized by helper T-Cells for the reason that they are not processed and presented along with MHC proteins.

3.3. Process of Lymphocyte recirculation

We have viewed in the previous section that an humorol immune response can be mounted against two kinds of antigens (T-dependent and T-independent antigens) and we have seen also the process of activating, proliferating and differentiating of lymphocytes (B-Cells). But the question that might be asked heir is where these entire actions do are carried on? In other term in which part of the body these actions will be taken place?

3.3.1. Structure of Lymph Node

We have showed above the immune system organs that are implicated during an immune response against antigens. One among them is the Lymph nodes which are secondary lymphoid organs that are considered as sites of lymphocytes-antigens interaction and lymphocytes-other immune cells interaction. Lymph nodes act as filters that are sampling lymphatic fluid for bacteria, viruses, and foreign particles [4]. It's in these organs that the process of mounting an immune response against an antigen is launched; the meeting and the interaction between the antigen and the various immune cells type implicated in the process of mounting such immune response (including B-Cells, Helper T-Cells) is than entirely orchestrated by the Lymph nodes.

An illustrated schema presented in [Fig. 1.7] shows the different major regions that compose a Lymph node:

- Afferent lymphatic: drain lymph fluid from tissues, including antigen presenting cells (APC) and antigen from infected sites to the lymph node (LN).
- HEV (High Endothelial Venules): the capillary walls where T and B cells enter the LN from the blood. Paracortex: the T cell zone.
- Primary Follicles (PF): where B cells are localized, includes Follicular Dendritic Cells (FDC's).
- Germinal Center (GC): is formed when activated B cells proliferate in the PF.
- Medullary Cords: where plasma cells secrete Antibodies.
- Efferent lymphatics: the only exit from the LN, where activated or re-circulating T and B cells, as well as antibodies (Ab's) leave the LN and join the blood circulation.

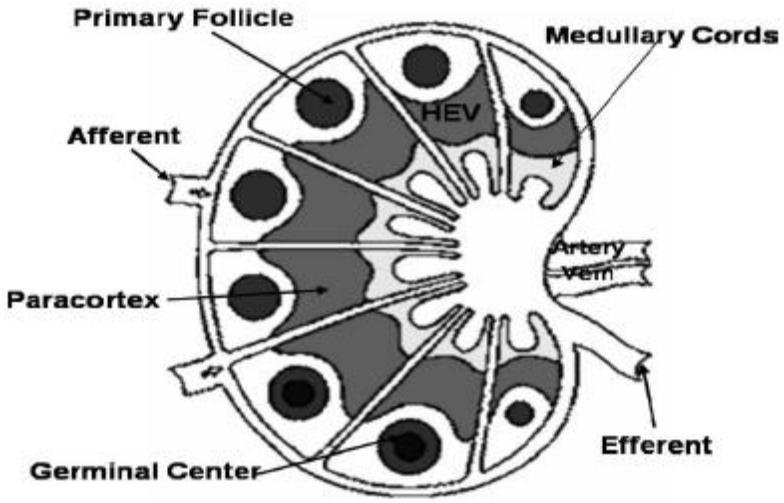


Fig 1.7: Lymph Node schematic structure **[4]**.

4. Summary

In this chapter we have wished-forgiving readers having no or little knowledge of the immune system an overview of topics that are available in basic immunology. However, this chapter doesn't provide enough information for immunologists or specialized readers of the immunology domain.

The purpose of this chapter is to provide an abstract view of the biological immune system and its important mechanism that inspired to develop computational models. For that a sum up of the basic elements of the biologic immune system; in particular, roles of various immune components: organs, immune cells (such as B cells, T cells, and other lymphocytes), and the overall process by which an humoran immune response mounts; are described.

Chapter II

Computational Models for Immune System

From the detailed description of the immune system mentioned in the first chapter, we can see that its mechanisms are very highly complex. Even the biologists are not able to know totally the internal mechanisms of the immune system. As to the engineering applications, only the macro features, such as the distribution, parallelism, self-adaptation, self-organization and so on, are known. Therefore, modeling the immune system is a purpose for both biological and computer researchers for the aim to better understand the immune system and solve the engineering problems.

Many models based on different approaches are developed to model and simulate the immune system; the present chapter focuses on the most well known models having taken place during the last years.

1. The purpose of modeling the immune system

Modeling the immune system is a subset of the larger field of biological simulation. As such, it is related to work in artificial intelligence, artificial life, neural networks, and network simulation [6]. Lots of benefits can be gained from building the model of the immune system for both biological and computer researchers [13]. The below section gives answers for the question: why the immune system should be modeled?

1.1. For biological researchers

Up to now biologists suggest lots of hypotheses on how the immune system confronts outer virus. However, it is still in question that whether these hypotheses sufficiently demonstrate the phenomena observed by us. Therefore, the computer simulation can help biological researchers to further understand the functions of every component and the internal mechanisms of the immune system and verify these hypotheses such as hypotheses about the infection process or simulate the responses of some drugs. Furthermore, it can provide some inspirations for biological researchers to develop some new medicines capable of restraining certain disease and to verify the suitability of the medicines for human body. So computing model with computer not only is cheaper than living tissue, but can minimize the requiring time.

1.2. For computer researchers

For the aim to attempt to bridge the divide between immunology and engineering, a new area of research called Artificial Immune System (AIS) has been released throw the application of techniques such as mathematical and computational modeling of immunology [14]. In this domain, many different aspects of the immune system have been used as inspiration for engineering applications yielding the appearance of the immunity-based algorithms. These algorithms can help us improve the current intelligent algorithms and explore new nature-inspired computing methods that have been applied to a wide range of problems, including computer security, control engineering, robotics scheduling, fault tolerance, and bioinformatics [15].

2. Approaches for modeling the immune system

The immune system can be viewed as a typical Complex Adaptive System (CAS) [13]. Many features of CAS such as emergence, co-evolution, aggregation, variety, simple rules and self-organization, are possessed by the immune system. The immune system has a large amount of nonlinear interactions between cells with simple rules and has the ability of self-regulation with the varying environment. Thus, there are two methods to model the immune system: top-down approach and bottom-up approach.

2.1. Top-down approaches

A top-down approach is essentially the breaking down of a system to gain insight into its compositional sub-systems. It abstracts a complex system into high-level biological functions and quantities to simplify the description and prediction of its dynamics [16].

In this approach an overview of the system is first formulated and specified without detailing any first-level subsystems. Each subsystem is then refined in yet greater detail, sometimes in many additional subsystem levels, until the entire specification is reduced to base elements.

These approaches do not emphasize the microscopic entities explicitly, but estimate the behavior in macroscopic level [13].

2.2. Bottom-up approaches

A bottom-up approach emphasizes the microscopic level, it can be defined as the piecing together of systems to give rise to grander systems. This means that systems are formed by firstly specifying in great detail their individual based elements, then linking together these elements to form larger subsystems, which then in turn are linked, sometimes in many levels, until a complete top-level system is formed. In these approaches [17] biologists seek to understand the components first, then determine how the components could fit together to produce a functioning system.

These approaches require greater computational power in order to simulate a large number of significant entities in real world. The computational complexity is exponential growth with the number of entities in the model [13].

3. Related methods for modeling the immune system

3.1. Differential Equation Based Models

The earliest models of the immune system that were created are differential equation (DE) based models [18]. These models are traditional top-down approaches which use continuous simulation technique rather than discrete event simulation. DE-based models perform interactions based on parameter, population and subpopulation [13]. They have been very popular, and a wide range of immunological phenomena have successfully been simulated using differential equation based models.

DE-based models usually simulate how average concentrations of IS agents (cells, antibodies, cytokines, etc.) change over time and identifying critical parameters of an immune response [19], [20]. A very simple example for such model is a system, which consists of three types of cells: Healthy cells $\boldsymbol{C}$, Infected cells $\boldsymbol{I}$ that can infect healthy cells and IS cells $\boldsymbol{K}$ that kill infected cells. This system can be modeled by the following differential equations [21]:

$$\frac{dI}{dt} = p_{infect} IC - p_{kill} IK - d_I I$$

$$\frac{dK}{dt} = p_{resp} IK - d_K K$$

$$\frac{dC}{dt} = s - p_{infect} IC - d_C C$$

The equations above describe a situation that may be interpreted as follows: Infectious agents of type $\boldsymbol{I}$ transform agents of type $\boldsymbol{C}$ into new agents of type $\boldsymbol{I}$ upon contact with a rate $\boldsymbol{p_{infect}}$. Agents of type $\boldsymbol{I}$ get removed (killed) upon contact with agents of type $\boldsymbol{K}$ with a rate $\boldsymbol{p_{kill}}$. Agents $\boldsymbol{K}$ proliferate upon contact with $\boldsymbol{I}$ with a rate$\boldsymbol{p_{resp}}$ .Agents of all three types die naturally at their specific rates $\boldsymbol{dI}$, $\boldsymbol{dK}$ and $\boldsymbol{dC}$. $\boldsymbol{C}$ type agents are produced at a constant rate $\boldsymbol{s}$.

$\boldsymbol{K}$ could be considered to be an immune system cell type being produced as a response to the appearance of the infectious agent $\boldsymbol{I}$. $\boldsymbol{C}$ may be presenting any possible type of target cell for the pathogen $\boldsymbol{I}$.

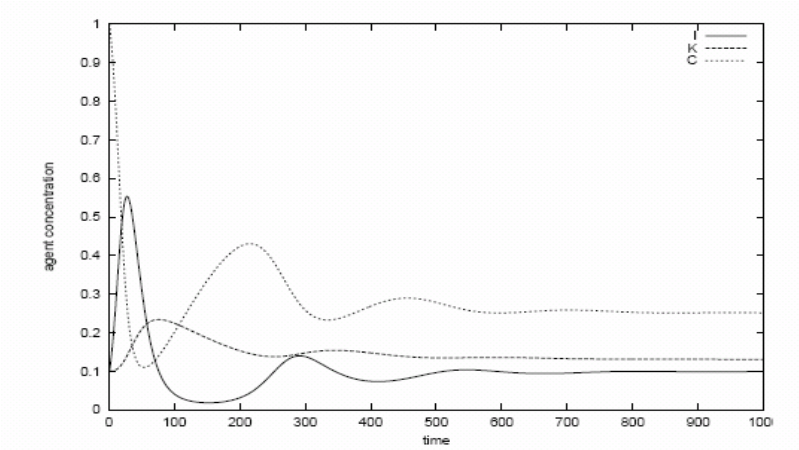


Fig 2.1: Time development of agent concentration in the immune system model **[21]**.

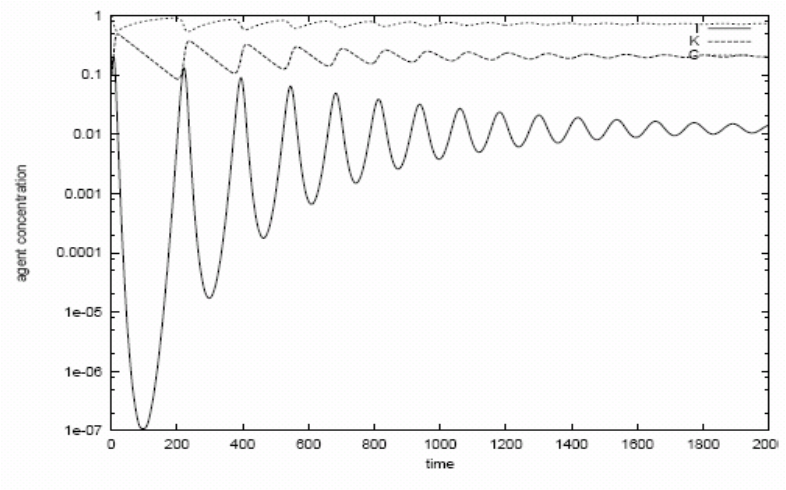


Fig 2.2: A 'chronic infection' settled down to an immune system model with very low concentrations of $\boldsymbol{I}$ [**21]**.

Integrating this system of equations yields different kinds of time development for $\boldsymbol{I}$, $\boldsymbol{K}$ and $\boldsymbol{C}$ depending on the parameters $\boldsymbol{p_{infect}}$, $\boldsymbol{p_{kill}}$, $\boldsymbol{p_{resp}}$ and$\boldsymbol{s}$, the death rates dx and, of course, the initial values of $\boldsymbol{I}$, $\boldsymbol{K}$ and $\boldsymbol{C}$.

[Fig. 2.1] shows the development of the agents for one set of parameters. At first, the number of infected agents $\boldsymbol{I}$ grows while the number of healthy agents $\boldsymbol{C}$ decreases. Then, as the response from the IS' agents $\boldsymbol{K}$ grows, the number of infected agents declines while $\boldsymbol{C}$ recovers. Finally, the system ends up in a steady state that may be interpreted as a chronic infection. [Fig. 2.2] shows in a half-logarithmic plot a system that settles down into a steady 'chronic infection' state after having gone through states where the concentration of $\boldsymbol{I}$ is very low.

A great efforts based on DEs models have been done to model immune system, such as for modeling virus-neutralizing immunoglobulin response [22], dynamics of co-infection of M. tuberculosis and HIV-1 [23], the dynamics of Plasmodium falciparum blood-stage infection [24], change in CD4 lymphocyte counts in patients before and after administration of HIV protease inhibitor indianvir [25], and the differentiation of B lymphocytes under control of antigen [26].

DE-based models of the immune system are still used widely for simulating many different phenomena, with good results. However, they are most often used to simulate one particular phenomenon, and do not simulate the whole immune system. This makes them of limited use when studying immune system models from an artificial intelligence or complex systems point of view [6].

Other problems have also been enumerated with these approaches. Some of them that they assumes large populations of essentially identical entities, which is not the case with biological cells as each cell has a unique life history that defines its interaction with the environment, The DE approach gives only average behavior of the system, and It is difficult to model non-linear behavior [27].

3.2. Cellular Automaton (CA) based Models

Cellular Automata (CA) is a bottom-up approach, which is often used to simulate some natural phenomena [13]. They are fully discretized dynamical systems that are well suited for computer simulations of biological systems [28]. They are defined as a class of spatially and temporally discrete, dynamical systems based on local interactions [29].

In a CA system: space, time, and the states are discrete. Each cell $\boldsymbol{C_i}$, defined by a point in a regular spatial lattice, can have any one of a finite number of states that are updated according to a local rule; i.e., the state of a cell at a given time depends only on the immediately preceding states of itself and its nearby neighbors $\boldsymbol{H_i}$. All cells on the lattice are synchronously updated so as to realize the development of the dynamic system in discrete time steps [24], [30].

In general, we can define cellular automata of any dimension. One, two, and three dimensional automata are often used in science. For an example a two dimensional automata is best represented as a regular spatial lattice or grid. In this

case, cell $C_{i,j}$ is surrounded by cells that form its neighborhood $H_{i,j}$ [31]. Traditionally, there are two possible sizes of $C_{i,j}$'s neighborhood in a two dimensional automaton, namely, $|H_{i,j}|$= 4 in the von Neumann neighborhood and $|H_{i,j}|$= 8 in the Moore neighborhood [32] [Fig. 2.3]. The state of cell $C_{i,j}$ at time t is determined by the state of its neighborhood $H_{i,j}$at time $t-1$ [Fig.2.4]. The function f can be considered as the rule that dictates how a particular state configuration of $H_{i,j}$determines the next state of $C_{i,j}$. More overview of a CA and their classification can be founded in [33].

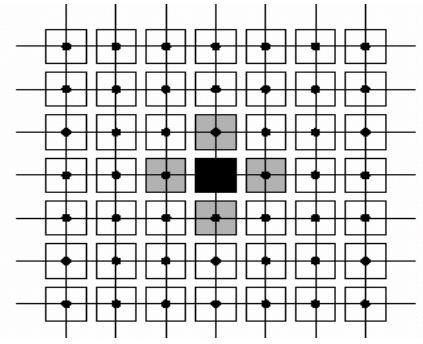


Fig 2.3(a):von Neumann neighborhood **[29]**

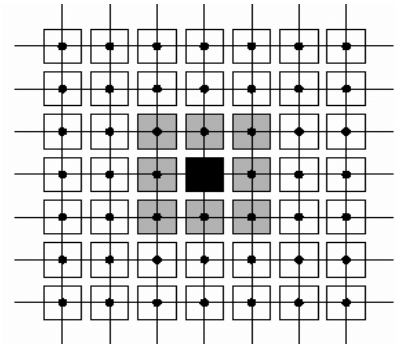


Fig 2.3 (b): Moore neighborhood **[29]**

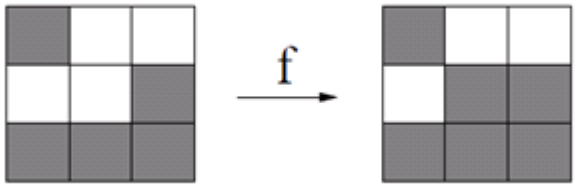


Fig 2.4: Cellular Automata Update from time step t-1 to t **[33]**

The initial idea of using a CA model in immunology was proposed by Kaufman and al [34]. In this model, various cellular populations and interactions were represented by Boolean values. Several subsequent discrete models such as the work of Weisbuch and Atlan [35]; Cohen and Atlan [36]; Chowdhury and Choudary [37]; Sieburg [38] were all developed based on this idea. In the work proposed in [39], the Boolean structure was extended to a cellular automaton which adapted the concept of the 'shape space' [24].

Other CA models for immune system [40]-[42] are based on the work of de Boer and al. [43] which was derived from Jerne's immune network theory [22]. In these CA models, each site of the CA grid represented an idiotype or clone, and the state of the site represented the concentration of that particular clone. The dimensionality of the grid represented the variable characteristics of the clone (e.g. geometric shape and

electric charge), and the size of the grid represented the number of different possible values (e.g. the number of different shapes that were possible). The interaction rules specified that a clone situated at $\vec{x} = (x_1, x_2 \ \ldots \ x_N)$, where N is the dimensionality of the grid, could stimulate the proliferation of clones, thus simulating the phenomenon of cross-reactivity [14].

The next generation of computational immune models based on CA was more ambitious, incorporating significantly more immunological details [44]-[46].The CA grid was used to represent physical space, rather than abstract properties of clones. The simulators incorporated enough detail that one model could be used to study several aspects of immune dynamics or disease [14].

In the next section we will discuss the successfulness applications of this computational immunology generation, which attempted to build general immune system simulators that have been used to investigate a number of immune system phenomena.

3.2.1. Related Immune system simulators based on CA

- **ImmSim**

The first successful model of the immune system was proposed by Celada and Seiden [46]. This model, that is called ImmSim, incorporates many generally accepted theories about the immune system such as interaction between B- and T-cells, affinity maturation and thymic maturation of T-cells [6]. ImmSim was a conceptually important advance, because it developed a general modeling framework that could be used for multiple studies [14]. Scientists consider it as one of the most referenced and peer reviewed IS simulators available [47], and many of the underlying ideas have been used in other models and frameworks. Its basic idea consists in the capture and processing of antigens and how that processing affects the various cell populations [20].

The ImmSim model is a model of a lymph node. It simulates the interactions between lymphocytes, APCs, antibodies and antigen. The lymph node is represented by a two dimensional CA with periodic boundary conditions [Fig 2.5] where each site contains a number of cells and molecules, which interact with each other. The behavior of the system emerges from interactions between cells and molecules [46].

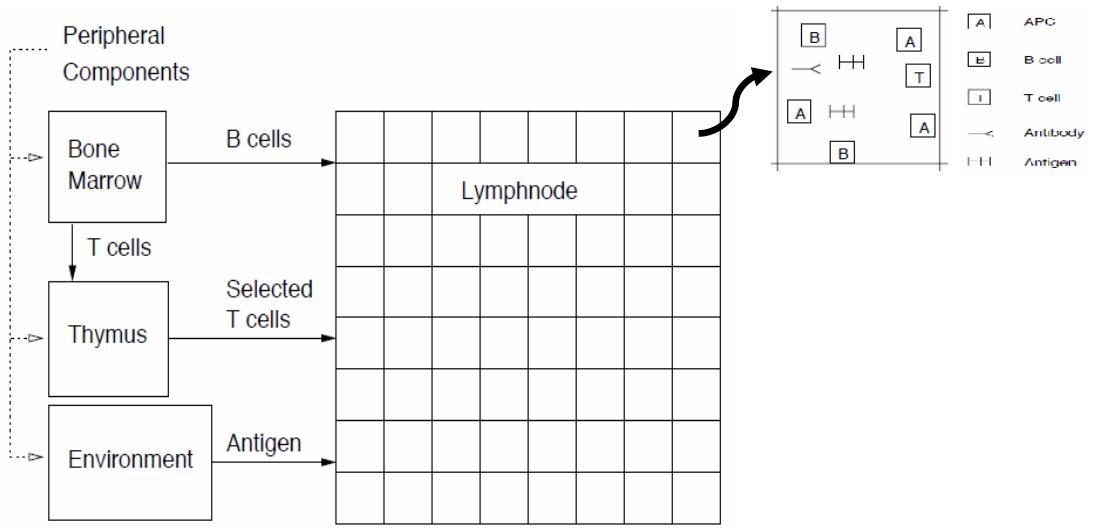


Fig. 2.5: An overview of the ImmSim model. The lymph node is the main component. It is modeled as a grid of sites. The bone marrow, thymus and environment are black box modules called peripheral components. They generate B-cells, T-cells and antigen. A grid site contains a number of entities. Cellular entities are modeled as individuals; molecular entities are modeled as amounts. **[46]**

The ImmSim model represents also the bone marrow, the thymus and the environment as peripheral components which bring new lymphocytes and antigen into the lymph node.

Each grid site contains a number of entities which belong to two groups, cellular and molecular entities. The group of cellular entities (or cells) consists of Bcells, T-cells, plasma cells and APCs, as shown on [Fig. 2.6]. Each cell has its own age and several other properties.

Antigen, antibody and antigen-antibody complexes [Fig. 2.7] are molecular entities or simply molecules. They are modelled as quantities. At each site, only the amount of each possible antigen, antibody and complex is stored.

The original version of ImmSim was written in APL2, it modeled the humoral response [33] which was complemented by a new release version called “ImmSim3” to include Th and Tc cells as well as epithelial cells and cellular response [15], [48]. Although the general idea for the code was good, the fact that the code was written in the APL2 language was beginning to be a real nuisance [49].

This situation was carried out by Bernaschi and Castiglione who came up with a new ImmSim based versions coded entirely in language C. The following section gives more details about their work.

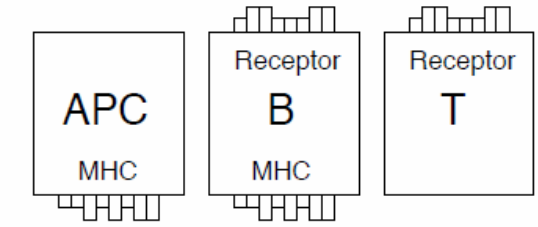


Fig. 2.6: A schematic picture of an APC, a B-cell and a T-cell in the ImmSim model. B- and T-cells have a receptor, APCs and B-cells have MHC on which they present antigenic determinants **[18]**.

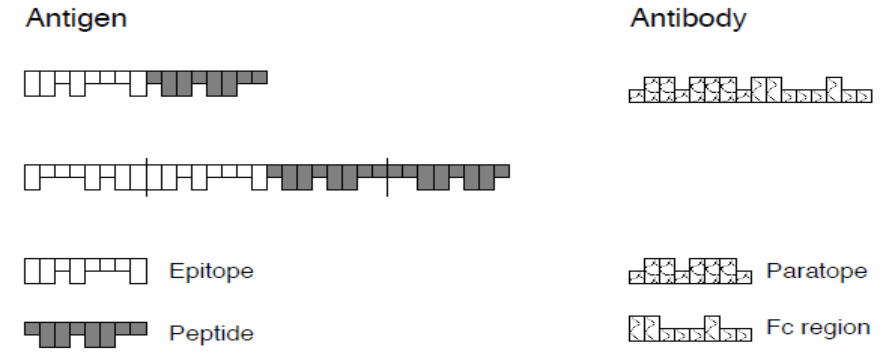


Fig. 2.7: A schematic picture of antigen and antibody in the ImmSim model. Antigen consists of a number of epitopes and peptides, each represented by a bit string. Antibody consists of a fixed Fc region and a variable paratope. The Fc region is identical for all antibodies; the paratope is an antigen specific receptor **[18]**.

- **C-ImmSim, ParImm, SimTriplex and ImmunoGrid**

C-ImmSim and the correspondent parallel variant, ParImm [50], are versions of ImmSim developed by F. Castiglione and M. Bernaschi in the C programming language, with focus on improved efficiency and simulation size and complexity [51].

In these adaptations of the ImmSim model, the IS response is designed and coded to allow simulations considering millions of cells with a very high degree of complexity. The code can resort to parallel processing to run faster; optimized data structures and I/O have allowed stretching the limits of available memory and disk space [14].

Currently, C-ImmSim [52] is the most advanced open source IS simulator based on the original Celada-Seiden automaton, with consistent publication throughput. It simulates both the innate and adaptive immune responses, and it can represent macrophages, dendritic cells, Tcells and B-cells, antibodies, antigens, and some cytokines as agents [22]. The most recent upgrades include, among other features, the

use of three-dimensional shapes, inclusion of the chemotaxis phenomena and consideration of different cell speeds [14].

C-ImmSim uses a bit-string polyclonal lattice model. The "bit-string" refers to the representation of molecules and Receptor-specific interactions, "polyclonal" indicates that multiple clones of different specificity of lymphocytes are represented, and "lattice" means that discrete lattice is used to represent the discrete space [23].

C-ImmSim based simulations have been yielding results with interesting potential. Some examples are work regarding progression of the HIV-1 infection in untreated host organisms [33], scheduling of Highly Active Anti-Retroviral (HAART) for HIV-1 infection [32], simulation of cancer immuno-prevention vaccine concerning its effectiveness and scheduling [53]-[54], and the modeling of Epstein-Barr virus infection [55]. In the correct context, these results have biological relevance, motivating further experiments with the framework [43].

SimTriplex [56] is a specialized cellular automaton in modeling mammary carcinoma, Triplex vaccine and the immune system competition based on a modification of ImmSim framework. It mimics the behavior of immune cells at the cellular level in both vaccinated and in naive mice.

ImmunoGrid [57] is a European Union funded project to establish an infrastructure for the simulation of the IS at the molecular, cellular and organ levels for various applications [43]. Its main objective is the development of a human immune system simulator using common computational platform to help development of vaccines and immunotherapies. The project is a web-based implementation of the Virtual Human Immune System using Grid technologies. It adopts a modular structure that enables easy extensions and modifications [23].

Currently, C-ImmSim and SimTriplex constitute the main part of the system level models. C-ImmSim simulates the immune responses to bacteria and viruses (e.g. HIV-1), and SimTriplex is used to model tumor growth and responses to immunization. The results produced by C-ImmSim and SimTriplex are presented on the web in graphical and text file formats for educational and research users.

3.3. Agent-Based Models (ABM)

Agent based modeling (ABM) is a bottom-up approach that has been applied recently in variety of research areas; such as in social sciences [58], economics [59], Transportation Management [60] and increasingly in Life Sciences; as an interdisciplinary tool to simulate, understand and study the dynamics of complex systems. In this modeling approach, a collection of autonomous decision-making entities called agents is used to model a system. Each agent of the system individually assesses its situation and makes decisions on the basis of a set of rules [60]. These agents interact, co-operate and at times work towards a common goal while keeping their own individual interest at hand [61]. They can sense their environment and can also change the state of its environment. Simulations based on this approach are very closer to the real interaction between entities than the previous methods.

Currently, in life sciences: especially in biology and biomedicine and precisely in immunology very interesting works are based on ABM have been done to simulate phenomena in immunology. for example a Multi-Agent immune model has been used in the Evolutionary and Complex Systems Lab of Nanyang Technological University in Singapore to validate the three stages of HIV infection [13],[62], the diversity and mutation of HIV virus is also emphasized in the School of Computing of Dublin City University in Ireland [13],[63]. Other ABM works have been also done; we cite heir two examples: the first is the one presented by jacob [64]; he proposed a swarm-based approach with 3D visualization to model the immune system in which every individual element is represented by an independent agent controlled by rules of interaction. The second is the platform of CAFISS [65] which stands from Complex Adaptive Framework for Immune System Simulation; CAFISS is an agent based model in which rectangular grids representing spatial locations are used to divide the simulation. The CAFISS simulation is asynchronously updated via multithreading; where every IS cell instance runs in its own thread and interact with other cells by events [19]. In this framework a bit string is owned by each cell and is acted as a cell sensorial input. Antigen receptors or receptors are represented by specific substrings which are activated with variable strength depending on the number of matching bits. CAFISS is developed for the main task to simulate how the immune system confronts HIV, and to display the dynamics of the immune system [13].

Using ABM approaches to model immunology phenomenon is a well suited choice that led to the comprehensive abstract models described in Immune system. In these approaches the immune system is viewed as a complex adaptive system [20] in which every single cell or pathogen is represented by an agent that can interact with other agents to encode some of cell behaviors (cell death, division, cell activation or differentiation).

The modeling based on agents for immunology in particular and CAS in general brings a lot of benefits than the previous models, the most of these benefits [66],[67] are (1) capturing of emergent phenomena that result from the interactions of individual entities, (2) providing natural description of a system composed of behavioral entities and (3) having the flexibility feature which can be observed along multiple dimensions and the discrete feature that characterizes the interactions between agents.

3.3.1. Related Immune system simulators based on ABM

- **SIMMUNE**

Simmune is an agent-based simulator developed by Meier- Schellersheim and Mack [68]; it seeks to model the immune system using molecular and cellular interactions [69]. In this simulator, the both behaviors of the IS agents (cells, molecules) and the IS's challengers (bacteria, viruses) are described on a microscopical scale [68].

It can be viewed also as a hybrid of continuous and ABM techniques for the reason that it combines both molecular such as cytokines defined as continuous quantities with differential equations modeling for their dynamics; and cellular level entities modeled as discrete computational agents [28], [70].

Simmune was firstly developed at the Institute for Theoretical Physics of the University of Hamburg, Germany; but Now, the Laboratory of Immunology of the National Institute of Allergy and Infectious Diseases, NIH, has taken it as part of the bio-computation effort and continues its development within the framework of the new program in systems immunology and infectious disease modeling of the NIAID. The point power in SIMMUNE is that it can be applied not only to simulate IS cells but also any living cell system [28].

The Simmune package comes with three components [21], [71]: (1) Simmune modeler (short: 'simmod'), (2) simulator (short: 'simmune'), (3) and signaling network browser. The simmode offers graphical user interface to define models of multi-cellular system by defining properties of molecules, molecular complexes, enzymatic transformations, Cellular behavior, Extracellular compartments and Simulation parameters, etc. whereas simmune generates the simulation basing on input models created by simmode. During the simulation of a model, three different views, which can be also saved in various data forma, are offered: (1) a concentration over time view, (2) a pseudo 3D view, (3) and a 'slice' view. The first view lists molecule types, cells and how their spatial (average concentration or total number) changes as the simulated time progress, a vision of the simulated extracellular compartment are shown in the second view, and the last one shows a 2D cut through the extracellular compartment and visualize molecule concentration gradients. The signaling network browser is used to investigate the dynamics of the signaling processes contained in the detailed behavior of the cellular biochemistry.

- **CyCells**

For the aim to study intercellular interactions mediated by molecular signales, Warrender has designed a multi-purpose simulator called CyCells [72] in which many of the features are particularly designed for modeling the intercellular infections. CyCells; that it's written entirely in C++; has the flexibility to be used for modeling many different kinds of multi-cellular systems, and allows significant flexibility in choosing cell behaviors and molecular properties [73]. It has a similarity with the Simmune simulator in the way that it can considered as a hybrid simulator in which molecular concentrations are represented continuously whereas cells are represented discretely. To implement CyCells, Warrend has used a three-dimensional square grid in which initial numbers of cells, cell types such as B cells and macrophages, and molecular signals such as cytokines are specified to define models in which explicit representation are used to symbolize each cell of each cell type, and real valued concentrations are used to represent molecular signals [20].

In CyCells, a computational abstraction in Multi-Agent systems known as Sense-Process-Act is used to modal Agent behaviors in which three categories can be segregated: (1) those handling sensing of the external environment, (2) those

updating the internal state, and (3) those implementing concrete actions that affect the agent or its environment or both [73]. So by describing the sense, process, and act functions appropriate for each cell type, it led to define the simulation model. To run this simulation three steps are required [72]: (1) model definition, (2) model initialization, and (3) the simulation execution. In the model definition: behavior of each molecule and cell type to be used in the simulation is specified in the way that molecular types are defined via a name and the appropriate diffusion rates whereas cell types are defined via cell attributes (how those attributes should be initialized and how they should be updated or used to make decision during each time step). Besides to this, a line for every sense, process, or action function used by each cell type, with the related parameter values are included in the modal definition. In the model initialization: the simulation geometry, initial molecular concentrations, and initial numbers of each cell type are specified. In the simulation: three sequences of activities compose one time step simulation: (1) molecular diffusion and decay, (2) updating of each cell according to sense, process, and act functions, and (3) Cell movement. The actions (death, division, differentiation, movement, migration, etc) that can be attached to a Cell affect the composition of the local population.

CyCells have been applied in various immunology phenomena exemplified in particular to study hypotheses about the maintenance of peripheral macrophage population sizes in the lung [74] and to model early infection dynamics of Mycobacterium tuberculosis (Mtb) bacteria [73].

3.4. Reactive Animation based modeling

In the last few years, Prof. Harel and his group have carried out a new project at the Institute of Wizmeman for the aim to model biologic systems throw a formalism called Statecharts. This formalism; which had been invented in the first time by Harel (1984) as a visual language to assist in the development of the avionics system of a new aircraft [75]; have been also used in variety of industries (including telecommunication, aircraft, automobiles, interactive software systems, medical diagnostic systems, aerospace and control application ...) to specify the behavior of complex reactive systems [75], [76]. Reactive systems are systems that change its actions, outputs and conditions/status in response to stimuli from within or outside

it; in other terms, they are on continuous interactions with their environments by the use of inputs and outputs that are either continuous in time or discrete [77].

This visual formalism, which has developed as a language for specifying reactive behavior, views complex systems as a set of objects in which their behavior are represented via a set of states, substates that an object can enter over its lifetime and a set of messages or events causing transitions between one state to another [20], [78]. This led the description of both: how objects communicate and collaborate and how they carry out their own internal behavior under different conditions.

Statecharts can be considered as a combination between bottom-up and top-down approach for the raison that they can enable us describing systems at multiple levels (state may contain sub-states), and zooming in and zooming out between these levels [75]. They can be implemented by tools like Rhapsody (or other similar tools) and they can be compiled into executable reactive machine code (for example, in C++, Java…) which then can be used to generate realistic simulations.

Prof Harel and his group have recently given a new modeling vision that differs to those discussed above (section 3.3.1, 3.3.2 and 3.3.3); they have paid attention that biological systems are naturally reactive systems for the cause that biologic systems can be abstracted to various levels (organisms, organs, cells, molecular mechanisms …) on orthogonal/horizontal interactions with each other [79]. In consequent, they have used Statecharts formalism to model the behavior of biological systems. The resulting executable reactive machine code has been then combined with front-end animation tool to give more realistic visualization simulation and enable natural-looking; this combined technique is called a reactive animation (RA) that seeks to combine state-of-the-art reactivity and state-of-the-art animation by linking advanced tools in the two areas [80].

The most important biological related works based on the RA technique have involved [79] the development of the mammalian pancreas, the differentiation of T cells in the thymus, and the dynamic architecture of the lymph node; their snapshot simulations are illustrated in [Fig 2.8 (A)], [Fig 2.8 (B)] and [Fig 2.8 (C)] successively. In the first work the authors [81] model the early stage of pancreatic organogenesis by linking three components: (1) a Statecharts based reactive model to model the behavior of molecular processes and morphogenesis, (2) a 3D interactive front-end

animation using 3D game studio for visualizing the molecular processes and its interactions, and (3) a Matlab GUI for mathematical analysis. The other two works relate to the immune system: the first simulates the differentiation of T-cells in the thymus gland [75]; in this simulation the authors combine between the Statecharts based model for the biological complexity of the thymus with a 2D interactive front-end animation using pre-recorder flash animations. The second one simulates the development and function of cells in the lymph node [4]; it merge a fully executable, bottom-up computerized model of the Lymph node using the visual language of Statecharts with dynamic 2D front-end animation using flash.

The simulations cited above have proved that the vision to simulate biological system (including immune system) as a reactive systems basing on Statecharts modeling combined by a front-end animation tool; is very closer to the reality and it allow users to interact with the running simulations throw the front-end tier in any level by querying the simulation or modifying it at some point as it runs. It has also become one of the new based projects carried by Microsoft to release a new visual formalism called Biocharts [82] which is specific to model complex biology systems and it is now under development.

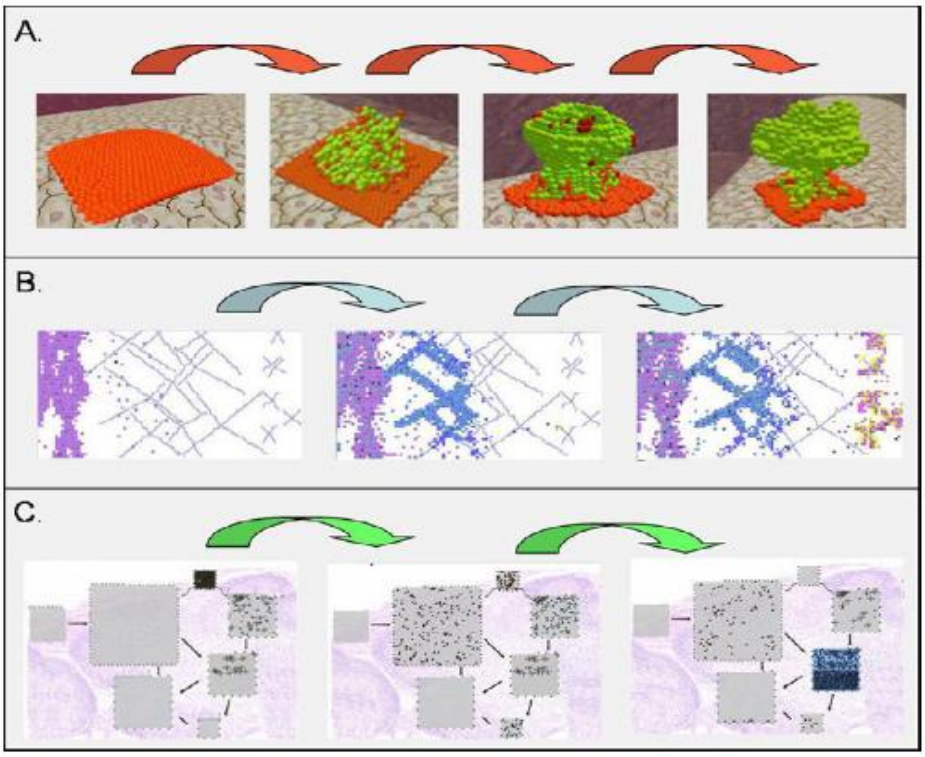


Fig. 2.8: Reactive Animation Simulation of organ level in three models: **(A)** pancreatic organogenesis; **(B)** maturation of T-cells in the thymus; **(C)** development of the lymph node **[79]**.

4. Comparison between different approaches

We have reviewed above different methods on how to model the immune system. In this section, an analyze of these methods is summarized in [Table 2.1] and is detailed as follow:

Modeling Method	**DE**	**CA**	**ABM**	**RA**
Approach	Top-down	Bottom-up	Bottom-up	Top-down & Bottom-up
Level	Macroscopic	Microscopic	Microscopic	Macroscopic & Microscopic
Entities	Homogeneous	Homogeneous	Heterogeneous	Heterogeneous
Time	Continuous	Discrete	Discrete	Discrete
Structuring [(1)]	-	-	compositional	Hierarchical & compositional
Scope of model [(2)]	Limited types of entities	Rules-based	Rules-based	State-based, substate-based and event-based
Concurrency [(3)]	-	Synchronous	Synchronous &asynch.	Synchronous &asynch.
Examples of modeled Systems	virus-neutralizing immunoglobulin response, dynamics of co-infection of Mtb and HIV-1, dynamics of Plasmodium falciparum blood-stage infection	progression of the HIV-1 infection in untreated host organisms, simulation of cancer immunoprevention vaccine , modelling of Epstein-Barr virus infection	infection dynamics of (Mtb) bacteria, validating the three stages of HIV infection, mutation of HIV virus	development of the mammalian pancreas, The Lymph Node B Cell Immune Response, differentiation and activation of T-cells in the thymus gland,
Cost	Low	Exponential increase with the number of entities	High, requiring parallel computing to improve efficiency	Very High, requiring parallel computing to improve efficiency

(1): A language is compositional if the behavior of a system can be specified by modeling a set of interacting sub-systems. A language is hierarchical if models of sub-systems can serve as indivisible, reusable building blocks within a larger system model.
(2): Scope of model represents the entities & interactions that are present in the model.
(3): Concurrency refers to the way in which different parts of a system interact and change. A synchronous state change is a state change where the individual parts of the system change their contributions to the state simultaneously. An asynchronous state change is a state change where different parts proceed independently of each other.

Table 2.1: A comparative overview between different immune system modeling methods**.**

The traditional one is a top-down approach based on DEs that uses continuous simulation technique to model a minimum number of homogenous entities of the system; hence it limits model scope and takes a macroscopic view of the real system to be modeled. DEs-based methods are widely still in use for the raison they are more suitable for verifying the designer's hypotheses, for discovering "what and when" characteristics from an observed biological phenomenon, in addition to its low cost CPU consummation.

The next studied methods are Cellular Automata Based ones. These methods are a bottom-up approach seeking to simulate homogenous agents in the microscopic level. On the contrary of DEs based methods that are continuous based simulation; the CA based models are fully discretized dynamical systems based on local interactions. The resulting simulation that is in exponential increase with the number of entities is generated by a synchronous update of all entities (cells) on the regular spatial lattice.

ABM is also a bottom-up approach that has been applied in variety of research area; it is particularly suitable for simulating a large number of agents that can interact with each other in both synchronous / asynchronous way. In this modeling method that has wider model scope than CA based one: Agents are heterogonous and they are specified at individual level (microscopic). Simulations issued from this method require high parallel computing to improve efficiency.

Finally we have presented the Reactive Animation based model that is one of the recent methods that have been applied in biology simulation. This method; which aim to give more realistic simulation and allow users to interact with the simulation in any level of the system; is a combined approach between state-of-the-art reactivity and state-of-the-art animation. This simulation method needs a very high parallel computing to improve efficiency. Contrary to the other methods, RA based modeling is a combined bottom-up and top-down approach that can simulate very large heterogonous agents in both microscopic and macroscopic level. The agents can be updated in both synchronous and asynchronous way; their behavior and its interactions are specified by defining states, sub-states and events (as a part of the Statecharts formalism); this led to view the system composed by hierarchical levels in addition to its compositional one.

5. Summary

The present chapter outlines the different modeling approaches that have used to simulate an immune system; it illustrates in a detail manner the two well-known approaches (bottom-up and top-down) with its most related models that focus on the modeling and simulation of immunology (DE- based models, CA-based models, ABMs and RA based models). The chapter doesn't detail the architecture of the models involved, because it aims to give a general overview of all kinds of the immunology models that have taken place. At the end of this chapter a comparison between the different models is done for the purpose to get a clear idea about the approach and the model which will be taken in or immune response simulation.

As a conclusion: in our computing simulation to a response immune system against a virus infection; we will use as it will be discussed in the next chapter, a combined model between the Statecharts based behavior modeling; which have been used in RA simulation; with ABM. This combined approach aspires to get profit from the advantages of the two combined models. In our proposed modeling approach we are seeking that it will be the first immune model carried out by an ABM in which agents behavior are Statecharts based models.

Chapter III

Statecharts based Behavior for Agent Based Modeling

As we have viewed in the previous chapter, there are several modeling techniques that have been used in immunology simulation. Our overview is to tack advantage of those techniques by the combination of the Multi-Agent and the Reactive Animation Based models.

Our approach is a Multi-Agent based model in which the Agent behavior is modeled by Statesharts. In the present chapter we introduce both the basic concepts of Multi-Agent modeling and the Statecharts formalism, finally an overview of the works that have proposed a modeling and simulating approaches based on the presented method are then cited.

1. Overview of Multi-Agent Based Modeling

For several years, a wide discussion and research have been taken place in the field of agent-based modeling (ABM) (also sometimes related to the term multi-agent system or multi-agent simulation). The goal of this new class of computational models [83] is how to build complex systems composed of autonomous interacting computing elements called agents who, while operating on local knowledge and possessing only limited abilities, are nonetheless capable of enacting the desired global behaviors; In other term how to take a description of what a system of agents should do and break it down into individual agent behaviors.

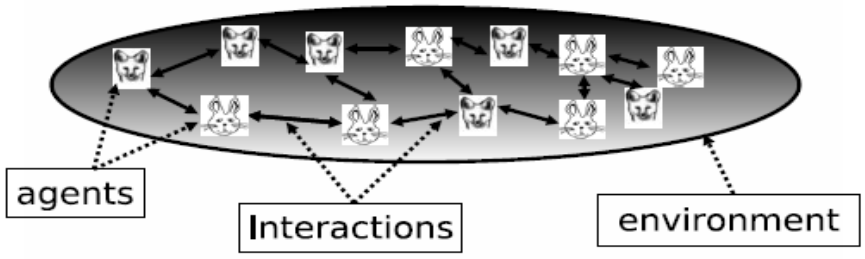


Fig.3.1: An agent-based model: The micro level entities, their actions and interactions, and the environment **[84]**.

In ABM, agents and their behaviors are not the only modeled things; but also the actions and interactions between these multiple agents (as individual entities or collective one such as organizations or groups) can be simulated throw the environment. Thus ABM focuses explicitly [84] on modeling the micro level entities and dynamics of the real system to be modeled (e.g., individual characteristics and behaviors, actions and interactions between the entities and the environment, etc.) [Fig. 3.1]. The process is one of emergence from the lower (micro) level of systems to a higher (macro) level. As such, a key notion is that simple behavioral rules generate complex behavior.

ABMs have been used in an increasingly wide variety of applications, ranging from comparatively small systems for personal assistance to open, complex, mission-critical systems for industrial applications. As examples: from modeling agent behavior in the stock market and supply chains, to predicting the spread of epidemics and the threat of bio-warfare, from modeling the growth and decline of ancient civilizations to modeling the complexities of the human immune system, and many more.

In order to best understand the foundations of ABMS, a brief overview of the fundamental concepts behind this computational model will be given in the sections below:

1.1. Simulation of Multi-Agent Based Models

To simulate MAS, a height level view of agent-based simulating models has been suggested by Fabien Michel, Jacques Ferber, AlexisDrogoul [84]. In their proposition, three correlated modeling activities have been considered for simulating such MAS [Fig. 3.2]. The first one, known as "the behavior module", concerns modeling the agent behaviors; the second one, named "the environment module", defines the virtual place wherein the agents evolve and interact; and finally, "the scheduling module" which is related to the definition of how the other modules (environment and agents) are coupled and managed with taking into account the time factor.

In the following sections we will give an overview of how to model every module among these three modules.

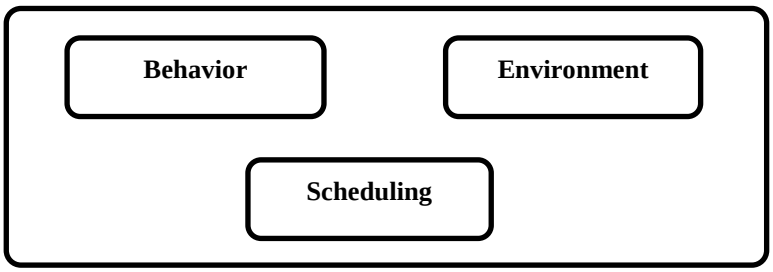


Fig.3.2: MAS Modules **[84]**.

1.1.1. The Behaviors Module

Before we tackle the modeling of agent' behavior an overview of the concept of agent and its related features will be viewed in the sections below:

- **What is an agent?**

The huge discussion and debate that have involved the ABM' subject, had led to a variety definitions of the term of "agent". Unfortunately, some sort of them risked losing all the meaning of the term; whereas others were important as the ones suggested for example by (Wooldridge and Jennings [85]; Ferber [86]; Russell and Norvig [87]). Although there is no universal definition of the term agent, but most of them are focusing that an agent [84]-[87] is essentially a special component(software

or hardware) situated in virtual or real environment that they can act in, and in order to satisfy their design objectives they are capable: firstly to decide for themselves how they autonomously behave basing on their perception, representation and interaction; and secondly to interact with each other both indirectly (by acting on the environment) or directly (via all forms of the encountered social activity: cooperation, coordination, negotiation…)[Fig. 3.3]. So from the definition of the agent given above; in MAS agents have a certain basic features which can be summarized below [88], [89]:

- *Autonomy:* ability to behave alone without the direct intervention of others and has control over its actions and internal state.
- *Sociability:* ability to cooperate with other agents to achieve its tasks in a social point of view.
- *Reactivity:*capability to perceive its environment and to respond in a timely fashion to change the environment.
- *Situatedness:* regarding the fact that the agents are placed into an environment which defines the conditions in which the agents exist, act, and interact.

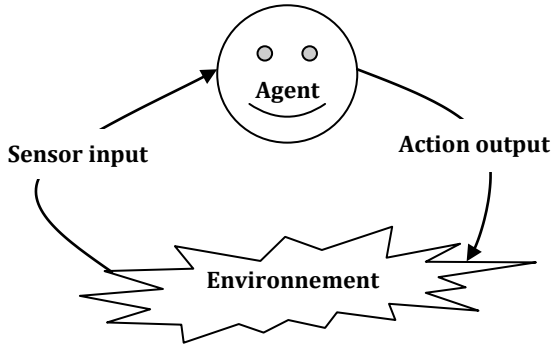


Fig.3.3: An agent in its environment. The agent takes sensory input from the environment, and produces as output actions that affect it. The interaction is usually an ongoing, non-terminating one.

- **Agent Architectures**

The term "Architecture" is habitually used to describe the internal organization of such an agent. The research community specialized in MAS domain has distinguished the following approaches that analyze the architecture of an agent:

a. Reactive Architecture

In this architecture [84], [90] and [91], a mediate decision is made by an agent basing on its perceptions of the state of the environment and the set of its available actions might be performed. The reaction of the agent, which in this case has an implicit representation of its environment and of the other agents, is based on a stimulus–response mechanism triggered by sensor data. Brooks [92] have proposed one of the most-known reactive architecture called "subsumption architecture", in which a set of hierarchy of behaviors are defined using connected layers of state finite machines. Each layer implements a particular goal of the agent, and higher layers are increasingly abstract. In other terms, layers are used to arrange behaviors in a manner that lower layers inhibit higher one. The lower layers have the higher priority. This architecture is better performed in dynamic environment, as well as it's often simple in its implementation.

b. Cognitive Architecture

A cognitive architecture can be used to model cognitive phenomena which are enabled by the definition of the organizational structure of functional processes and knowledge representations [93]. It specifies the different parts of such a system, their functionality, and the interaction between them in a high-level manner [94]. The foundation of this approach takes in account that agents reason from knowledge which is an explicit representations of their environment (states, properties, and dynamics of objects in the environment) and the other agents [84].

Since the eighties to nowadays, researchers in the MAS fields have developed various cognitive architectures in which the most well-known among them are the BDI (Belief-Desire-Intention) [95] based architectures. The hypothesis [89] behind these lasts is that each agent is intending to achieve its goals by performing certain rational actions with regards to its beliefs about world states, its knowledge and those of others, its intentions and those of others. Thus cognitive agents are

characterized by its beliefs which represent the information an agent have about the environment, its goals (desires) that represent the specified corresponding tasks to be accomplished, intentions which represent desires that the agent has committed to achieve, and finally plans that specify some courses of action might be followed by an agent in order to achieve its intentions.

c. Hybrid Architecture

In such situation, it seems to be very complicated to use single agent architecture to model the behavior of an agent; so some solutions have been proposed in which the intention is to combine between the two above cited agent architectures in order to profit from their advantages. Although the resulting combination architecture that is called "hybrid architecture" can built more flexible agents composed of modules that might be treated independently with reactive and cognitive aspects of the agent behavior, the main encountered problem in such architectures is how to ensure a good balance and a good coordination between these diversified modules. The best works developed in this sense are: InteRRap done by Muller and Pischel [96] and the Touring Machine done by Arango's team [97].

- **Interaction**

In MAS, collaboration, negotiation and cooperation between agents will be never achieved without communication that allows agents to interact with system resources and with each other [84], [89]. This key feature can be expressed by special language called "Agent Communication Languages ACL" which the most distinguished among them are those issued from the theory of social sciences linguistics and philosophy language in a hand, and Biology and ethology in the other hand.

The first ACLs kind that is known as speech acts is the traditional model of communication between agents, and it's interpreted via "Message Passing". Actually the most widely used and studied ACL is the **F**oundation for **I**ntelligent **P**hysical **A**gents "FIPA" ACL. FIPA [98] is an IEEE Computer Society standards organization that promotes agent-based technology and the interoperability of its standards with other technologies.

In the second ACLs type, "signal" is used to interpret the interaction between agents. This model of ACL is based on the theory of communication between animals

where they use signals to behave collectively. In the MAS domain, the most used kind of signals is marks [84] which are traces made by agents while they are on movement.

- **Modeling the agent behavior**

For a simplification purpose to not explain in details the existing modeling agent architectures, an agent model has been proposed in which the agent is always in a cyclic three phase process: [Fig. 3.4] perception – deliberation – action. The representation of this cycle has been illustrated in the work of F.Michel, J.Ferber, and A.Drogoul [84]. The work assumes that:

a. Firstly from the current state of the environment, a perception is obtained by an agent. The obtained perception might be a simple raw data structure or more complex one.
b. Secondly, a deliberation (memorization) function starts its process in which the agent makes its internals progress and renew its own representation of the world using the perception obtained before. In this process a specification of the core part of the behavior of an agent and its architecture (reactive or cognitive) is defined. In such situation as memorization process is not needed, and perceptions are harmonized directly to actions, the deliberation process is skipped.
c. Finally, an action is taken by the agent basing on its new internal state and its current perception. The result of the taken action is immediately noticed in the modification of the environment.

Fig. 3.4: An agent as a three phases process **[84]**.

1.1.2. The Environment Module

As a part of a MAS component, the environment is considered also as an essential compound of MAS since all the perceptions and actions that might be taken or had by an agent are broadly defined by it. According to Russel and Norvig [84], [87] and [88], an environment in which agents are situated can be:

- Accessible in the case in which complete, accurate, up-to-date state can be obtained by the agent; Inaccessible In the other hand.
- Determinist in the situation in which the resulting state achieved from performing an action is definite; non-determinist in the opposite situation.
- Static when it remains unchanged except by the performance of actions by the agent; dynamic in contrast.
- Discrete while the number of actions and perceptions is fix and finite; continuous otherwise.

Related to the environment properties cited above, there are two distinguished main approaches to model an environment:

- ***Discretized approach:*** it views the environment as a collection of connected bounded areas in which agent perception/action are defined; the most important platforms that use this kind of approach are those characterized by their based-grid environment formed by a regular grid of cells (or patches) [Fig. 3.5]. This approach is broadly used for the simplicity of its implementation (e.g., StarLogo [84], TurtleKitb [99]).
- ***Continuous approach:*** it computes the range of perceptions/actions basing on each agent which is considered as a reference point [Fig. 3.6]. It is frequently used when accurateness is needed, for an example it has been used for simulating soccer robots (e.g., in RoboCup soccer simulators [86])

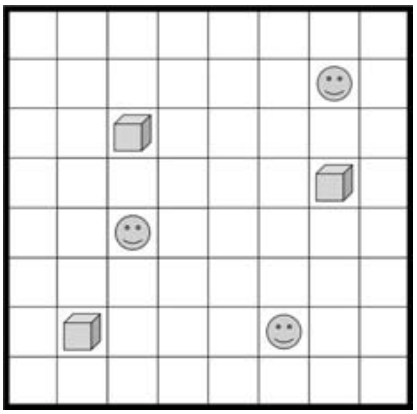


Fig. 3.5: *discretized environment* **[86]**.

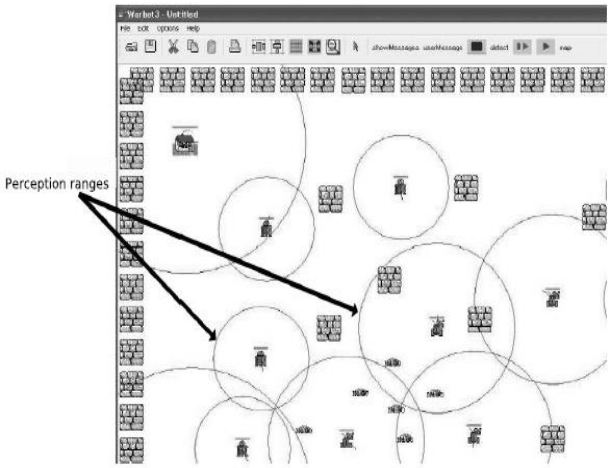


Fig. 3.6: *Continuous approach for the perception of the agent* **[86]**.

1.1.3. The Scheduling module

The scheduling module is viewed as a coupling manner between the agents and the environment with respect to the evolution of time which is one of the crucial parts that should be modeled while simulating MAS. This coupling manner is achieved via a function that defines the evolution of the whole MAS from one moment (t) to the next (t+dt). In fact there are three main approaches that could be used to model the time [86]: (1) continuous time in which the system state is computed for any time stamp via time functions, (2) discrete time where time evolves discretely with respect to constant time intervals, and (3) discrete event-based in which time evolves discretely from one event to the next considering a continuous time line. Choosing one among these approaches needs to take in account:

- ***Modeling the behavior time of the agents***: the majority of models, which represents the behavior time of an agent, associate for simplicity reasons a single instant t to a perception-deliberation-action process cycle. This means that the internal states of an agent are changed instantaneously. In the other hand, certain works in some application domains use a no-instantaneous approach to represent the behavior time of an agent.
- ***Modeling the temporal evolution of the environment:*** in such application domain, taking account of modeling the evolution of an environment in time could be essential for the raison that the environment in addition to its

reaction to the agent inputs may also progress in time according to its own dynamic.

To sum up this section, In the scheduling module the manner, of how the environment and the behavior modules are coupled and managed, is defined with respect to time which is also is specified by choosing a particular time management (continuous time, discrete time or event-based). The environment evolution, the agent perceptions, actions and interactions should also be defined with respect to one another. The [Fig. 2.7] below illustrate a global overview of MAS viewed as a three modeling modules:

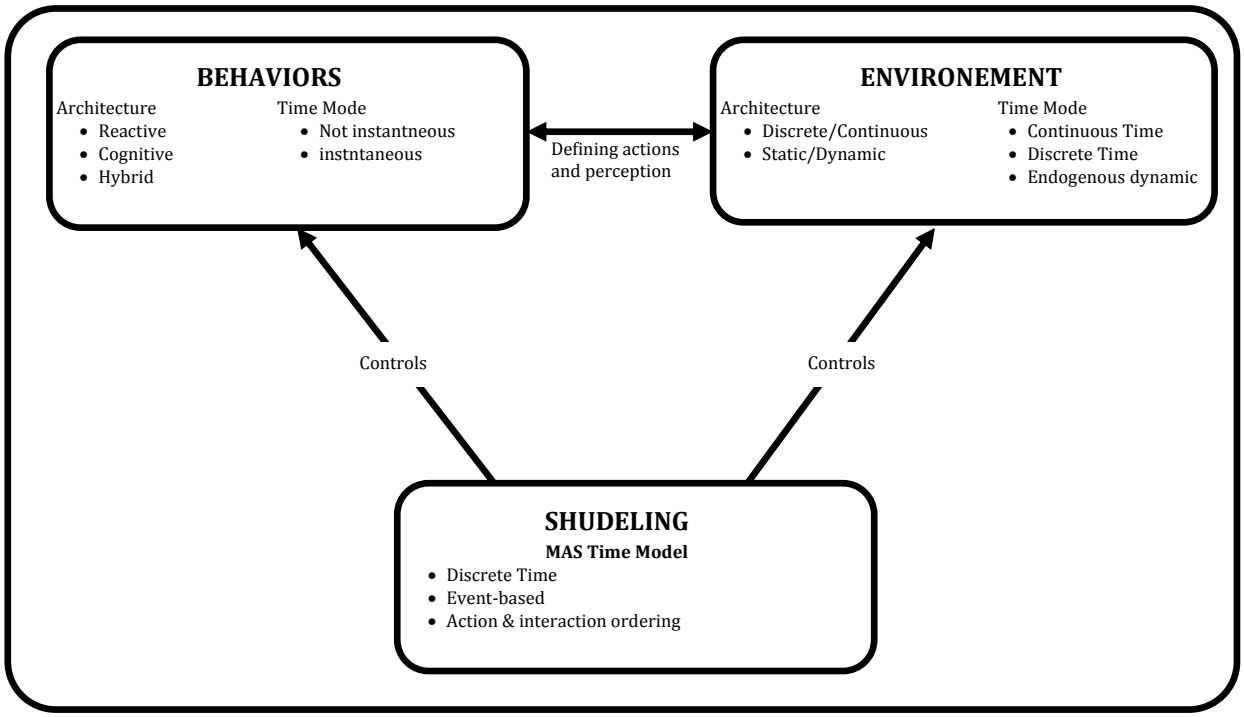


Fig. 3.7: Simulating a MAS as three modeling modules **[86]**.

2. Overview of the Statecharts Formalism

In this section we present an overview of the Statecharts modeling formalism and its related concepts:

2.1. What are Statecharts?

Statecharts are a graphical state-transition formalism based on exchanging messages or events between the system and its environment. They have been developed in 1982 by Prof Harel when he assisted in the development of the avionics system of a fighter aircraft at the Israel Aircraft Industries (IAI). Harel in his article [100] has detailed the full history of how he could create this new modeling formalism which has been used to describe the behavior of reactive systems in a manner that shows how the system reacts to the reception of an event or a specific sequence of events and what events the system generates.

Statecharts are an attempt to overcome the limitations of traditional Finite State Machines (FSMs). So they are essentially FSMs extended into a modular, highly structured, and economical description language that [101], [102]:

- Enable clustering, orthogonality (i.e.: concurrency) and refinement,
- Encourage zooming capabilities for moving easily back and forth between levels of abstraction,
- Incorporate a broadcast communication mechanism, time out and delay operators for specifying synchronization and timing information.
- Include a means for specifying transitions that depends on the history of the system's behavior.

In its short definition [103]:

Statecharts = state diagrams + depth + orthogonality + broadcast communication

The example shown in [Fig.3.8 (a)] illustrates a Statecharts corresponding to the FSM shown in [Fig.3.8 (b)]. The Statecharts is composed of an initial state, a simple state A, a composed state P and a final state.

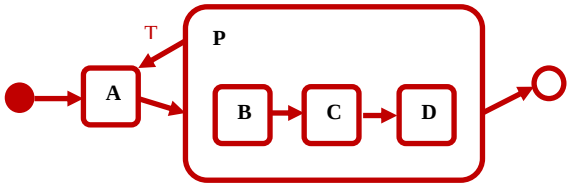


Fig.3.8 (a): Statecharts composed of: initial state, Simple state A, composed state P and final state

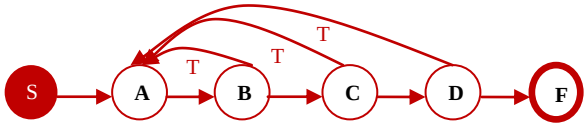


Fig.3.8 (b): FSM composed of: initial state S, 4 transit states A,B,C and D and final state F

2.2. Basic concepts of Statecharts

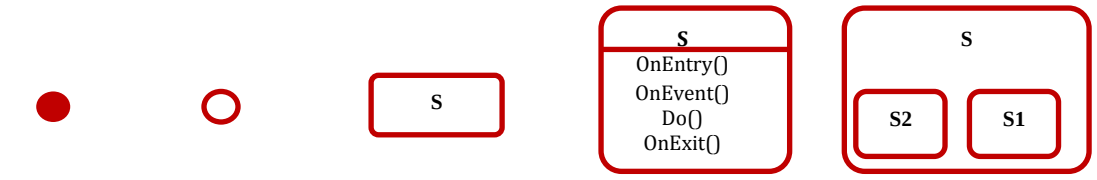
The two basic components of Statecharts are states and transitions.

2.2.1. States

States, which are used for memory purposes, are a condition of an object in which it performs some activity or waits for an event. They are denoted by a rounded square symbol [Fig. 3.9 (c)] that represents different contexts in which system behaviors occur [104]; for example they can represent: a time period during which a predicate is true, an action is being performed, or someone waits for an event to happen [105].

States may have associated actions [Fig. 3.9 (d)]:

- OnEntry: these actions are triggered as soon as the state is entered
- Do: these actions take place during the lifetime of the state
- OnEvent: these actions take place in response to an event
- OnExit: these actions take place just before the state exists



(a) Initial state

(b) Final state

(c) Simple stateS

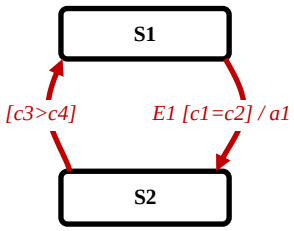
(d) Advanced state S with associated activities

(e) Composite state (superstate) S composed of simple states (substates) : S1 and S2

Fig. 3.9: Different types of statecharts

2.2.2. Transitions

Transitions are relationships between states, drown as a splined arrows, optionally labeled by a trigger that causes actions; the general syntax of such trigger is a triple: *Event [Condition]/Action*. This indicates that an object in the source state will perform certain actions and enter a target state when a specified event occurs and specified conditions are satisfied [106], [107].



- transition from State S1 to State S2: transition taken with performing an action a1 is when the event E1 occurred and the condition c1=c2 is satisfied.
- transition from State S2 to State S1: transition taken each time the condition c3>c4 is satisfied.

Fig. 3.10: Statechartstransitions

As it's sown in [Fig 3.10]; transitions' triggers can combines some events: It may consist of the condition, enclosed in square brackets, or it may consist of the condition only. Thus if the transition is labeled *E1 [c1=c2] / a1*, the action a1 is performed if the condition *c1=c2* is satisfied at the instant the event E1 occurs. If the transition is labeled [c3>c4], the condition c3>c4 is tested at each instant of time when the system is in the transition's source state, and the transition is taken if it's true [108].

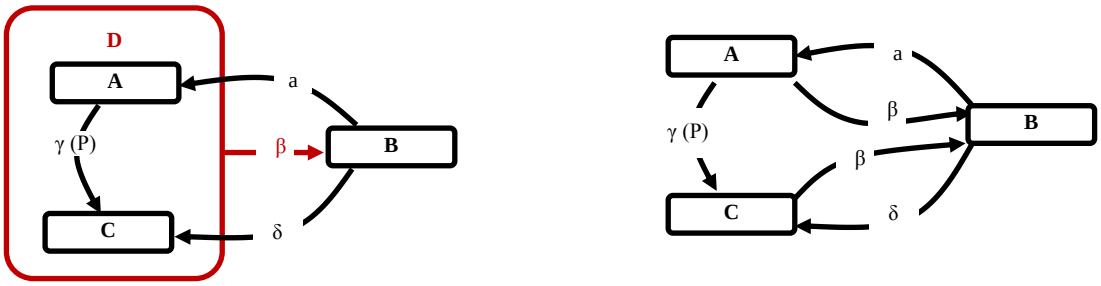
Events are used to represent information that is available for a precise instant in time and then vanish. They may launch actions and they are edge-sensitive, comparable to signals and interrupts; as for example a single ring on a phone, triggers an action which can be either a respond to take the call or ignore it [107]. Events can be generated externally to the Statechart coming from external sources or internally coming from internal sources.

2.3. Advanced concepts of Statecharts

2.3.1. The hierarchy of States (nested states)

Hierarchy (or states-nesting) that allows dealing with highly complex behaviors in elegant and modular manner is one of the most important innovations of Statecharts over the classical FSMs. It is used to group sets of states together by drawing super-states (lower-level states inside a higher-level states) [107], [109] and [110]. The super-state is called also an or-state, and it's the parent of its lower-level states.

The [Fig.3.11 (a)] illustrates how we can cluster states into new super-states yielding the minimization of the number of transitions between states: In [Fig.3.11 (b)] the system is composed of 3 states A, B and C. and as the event β takes the system to State B from either A or C we can create a new super-state D [Fig. 3.11 (b)] that group the two sub-states A.

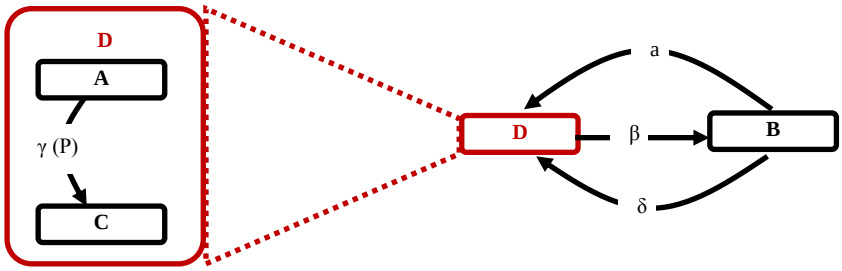


(b): Clustering of states **(a):** A simple statecharts

Fig. 3.11: Statecharts hierarchy(or-states decomposition) **[111]**.

2.3.2. Zooming-in & zooming-out

The example shown above [Fig. 3.11] might also be approached from a different angle of view: it illustrates the capability of Statecharts to view the system as a multi-levels decomposition which yields the ability to zoom-in and zoom-out into the different levels of the system [101]. For example looking inside super-state D disregarding external interface for the time being [fig.3.12 (a)] represent the zooming-in concept ; whereas eliminating inside of D and abstracting [Fig.3.12 (a)] to [Fig.3.12 (b)] illustrate the concept of zooming-out.



(a): zooming-in **(b):** zooming-out

Fig. 3.12: Zooming-in and zooming-out capabilities **[101]**.

2.3.3. Concurrency (orthogonal states)

We have seen in the previous section, the capability of Statecharts to decompose states in X-Or way giving the possibility to view the system in hierarchical levels. Statecharts formalism hasn't only this part of features; it has another improvement residing on the possibility to allow and-decompositions of states [108], [109]. These and-states, which are also called orthogonal components, describe the feature of parallelism inside a Statechart; it allows modeling of concurrent behaviors in the same modal. Orthogonal states are all activated when an and-state is entered and are all deactivated when it is exited.

So the concept of concurrency in Statecharts means [109], [110] that a concurrent state is described as consisting of two or more parallel active sub-states, and to be in such state entails being in all of those sub-states simultaneously.

As it's shown in the [Fig.3.13], to represent graphically a concurrent state, we use the notation of a dashed line that partitions the state into regions (and-components). The figure illustrates an orthogonal state Y consisting of and-components A and D. Both A and D are an Or-state while The first consists of sub-states B and C, and the second consist of sub-states E, F and G. they have both defaults internal transitions; So the combination (B, F) is achieved when default transitions are applied in the absence of any additional information, from the outside of Y. Then the occurrence of event α, imply transferring the system into (F, G) simultaneously which illustrates a certain kinds of synchronization (a single event causing two simultaneous happenings). In the other hand, the appearance of event μ at (B, F) tacks the system to (B, E) affecting only the D component. This illustrates a

certain kind of independence [108], [111] (a transition is taken in an and-component, independently of what might be happening in the others).

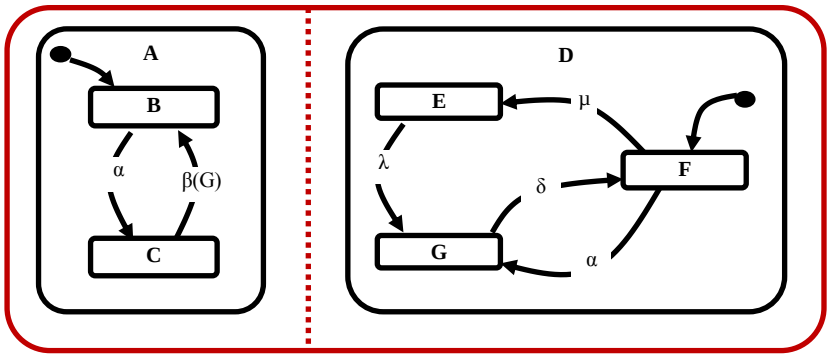


Fig.3.13: Statecharts concurrency (and-states decomposition) **[101]**.

2.3.4. Connectors:

To clarify the specification of such a system, Statecharts allow different kinds of connectors that are used to economize the number of entrance arrows to complicated sub-states [108], [111]. These connectors are presented in the section below.

a. Condition Connectors

A condition connector [Fig 3.14 (a)] (called also C-Connector) is a [101], [104] and [108] conditional pseudo-state that indicates a branch point from where each outgoing transition has a condition (guard). At the occurrence of the event, either, the transition having the true evaluate condition is then taken among the various outgoing ones; or the event is discarded if none of the conditions are true.

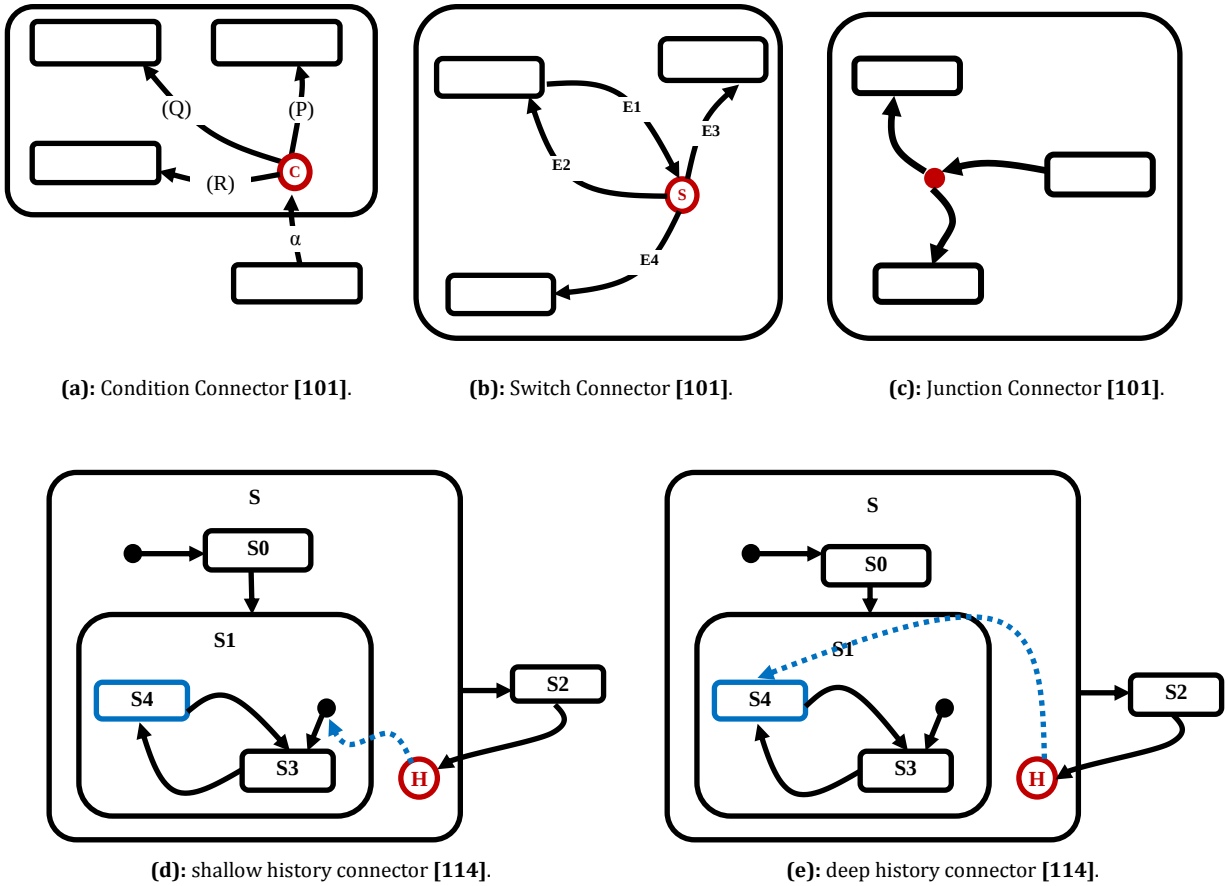
b. Switch Connectors:

The switch connector [Fig 3.14 (b)] (S-Connector) is a [101], [108] branch point that allows a transition to be connected to several different states depending on the value of an event. It's usually used with a set of events (rather than conditions) in which at least one event among them must be appears in all transitions stimulated by this set of events.

c. Junction Connectors

Statecharts allows also another type of connectors called a junction connector [Fig 3.14 (c)]. It's a junction point indicating that several states can transition to the same state on a given event. It joins transition arrows which its labels can be split as

desired. This possibly economizes [108] both the number of lengthy arrows and the number of identical portions of labels presented in the chart.



(a): Condition Connector **[101]**. **(b):** Switch Connector **[101]**. **(c):** Junction Connector **[101]**.

(d): shallow history connector **[114]**. **(e):** deep history connector **[114]**.

Fig 3.14: Different types of connectors

d. History Connectors:

History connectors also called H-connectors are pseudo-states allowing entering the state most recently visited within a given super-state. They are required in such cases; in which the last configuration of a complex state must be known when it was recently reactivated after it was been deactivated previously. This can provide a mechanism for memorizing and returning to previously visited sub-state [106], [112].

History connectors can be located only inside a composite state; they are shown as H or H* in the diagram and are categorized as follow [113], [114]:

- The shallow history H [Fig. 3.14 (d)] stores the last visited state on the same level of hierarchy.

- The deep history H* [Fig. 3.14 (e)] stores all last visited states on the path from the node to the leaf in the hierarchy tree.

So to summarize; the different kinds of connectors cited above are meant to visually emphasize the distinction between different kinds of behavior [108]: a C-connector indicates branching by conditions, S-connector branches by events, and junction connectors are used for the remaining cases; whereas history connectors are used for memorizing the state most recently visited.

3. Statecharts based Agent behavior

In the above sections we have presented the basics of Multi-Agent Simulation and the Statecharts formalism. In this section we will talk about combining the two modeling approaches in which Statecharts formalism is used to model the behavior of agents.

In fact, there are several researches that have been proposed to model agent behavior via Statecharts-based approaches. for instance, State-based programming for agent behavior has been explored to model agent communication and behaviors**,** UML state machine based models have also been used in JADE "**J**ava **A**gent **DE**velopment Framework" [115] platform where SmartAgent have been developed as an extension of the JADE agent behavior model [116], in the work done in [117] the behavior of situated agents have been modeled using a combination between Statecharts standard notation and the free-flow architecture with a focus on reusing roles in different applications. A proposed modeling and distributed simulation approach for discrete event systems (DESs) was suggested in the work of [118]**,** the approach uses the statecharts-based formalism to express complex behaviors of the constitutes entities of the modeled system. Another work done in [119], it aims to describe the ELDATool; which is a Statecharts-based visual tool for a rapid prototyping of Multi-Agent Systems based on the **E**vent-driven **L**ightweight **D**istilled Statecharts-based **A**gents (ELDA) model.

Although the above cited works have intended to use the Statecharts formalism to model agents' behavior and have been a references works for the upcoming ones, but there were other well-known successful woks that have been also became a reference works in how to use Statecharts based behavior for MAS. These works use

the ***AnyLogic*** [120] simulation tool; which is the only simulation environment on hand that allows us to combine different techniques and approaches such as differential equations, discrete events and agent based systems.

For instance: Stephan Emrich and al. [121] proposes a fully agent based modeling of epidemic spread; the work uses the AnyLogic as implementation platform agents characterized especially by its Statecharts tool which can be programmed very conveniently to model the agent's behavior. The modifications and/or extensions of the final model can be also handled in an elegant way.

Peer-Olaf Siebers and al.illustrates in their work [122] how to build a combined Agent Based / System Dynamics Model in AnyLogic; their work shows the ability of AnyLogic to build a simulation model by combining ABM and SD methods in one model using one particular architecture. they highlight the "points of interaction" of agents; which its behavior is based on Statecharts; and system dynamics and try to show that model elements belonging to different approaches live a single space of AnyLogic model and can easily access each other.

David Buxton and al. [123] suggest an Agent-based AnyLogic Simulation to understand dynamic behavior of the aero-engine value chain under future business environments. The value chain encompasses original equipment supply, aftermarket services and consumables supply. Each player in the value chain is represented as an agent, allowing detailed capture of individual business processes, logic, attitudes to risk and responses to changes in the market place. The agent-based model developed here is based on Statecharts formalism that models the business processes, decision rules and exchanges of information and materials involved.

Maxim Garifullin, Andrei Borshchev [124] presents an agent based approach to model consumer market using the AnyLogic Tool. The goal of this paper is to introduce the patterns in consumer market modeling that are most frequently met in the consulting practice. The work presents how we use the Statecharts formalism of AnyLogic to model the behavior of the consumer agents. Many other AnyLogic' works can be found in [125].

So to sum up this section, the AnyLogic simulator tool is actually the only available tool that allows us to build a simulation model using multiple methods: System Dynamics, Agent Based and Discrete Event (Process-centric) modeling.

Moreover, the AnyLogic tool can combine different methods in one model: putting agents into an environment whose dynamics is defined in SD style, use process diagrams or SD to define internals of agents, etc. In addition The AnyLogic tool provides a capability to use Statecharts formalism which can be programmed very conveniently to model the agent's behavior. The modifications and/or extensions of the final model can be also handled in an elegant way. Thus the AnyLogic tool is the suitable environment which can assist us to model our immune response against a virus infection.

4. Summary

We have viewed in this chapter an overview of both the Agent based modeling and the formalism of Statecharts. The basic concepts that characterize these tow viewed techniques are also given. We have cited also at the end of this chapter the works that have intended to use the Statecharts formalism to model agents' behavior. The most important works are those using the AnyLogic simulator which we will use it as a platform in our immune simulation.

Chapter IV

The AnyLogic Simulation of the first Humorol Immune response against an antigen infection

In the current chapter we will illustrate how we model and simulate the immune response against an antigen infection using the AnyLogic simulation tool which we have already proven in the last section of the previous chapter, that it's the suitable tool for our immunology simulation.

As the process being launched by the immune system to response to an encountering antigen in such body is hardly complex to be modeled as a whole system (chapter1), we will focus only on simulating a part of the immune response against an antigen infection. The part to be modeled and simulated concerns the first humoral immune response process that is initiated in the Lymph node wherein antigens are recognized.

This chapter is initiated by an overview of the AnyLogic simulator tool and its basic features, followed by a synopsis of the biologic behavior of the part of the immune system being simulated while an encountering Antigen is recognized. We have destined our study for the humoral immunity first response processed in the Lymph Node against both T-dependant and T-independent Antigens which have been already explained in the first chapter. A proposed model than will be illustrated using the AnyLogic environment tool, finally experimental results of the result simulation will be discussed.

1. The AnyLogic Simulation Tool

AnyLogic is a Java based multi-approach simulation modeling tool developed by **XJ** Technologies [120]. It's actually the only available tool that not only supports the three well-known modeling paradigms: discrete event (DE), systems dynamics (SD), and agent-based (AB); but, it has also the ability to combine between these different approaches to find solutions for problems that would be difficult to formulate into tradition 'single approach' tools. These combination possibilities make it a very interesting tool that has applied for simulating complex systems in a wide range of domain applications including : control systems, traffic, system dynamics, manufacturing, supply chain, logistics, telecom, networks, computer systems, mechanics, chemical, water treatment, military, nonstandard problems, education,…

In the following sections: a brief overview of the AnyLogic features, its architecture framework, and its available tools are presented.

1.1. AnyLogic Features

AnyLogic is innovative professional simulation software based on advanced technologies such as UML, Java, hybrid systems theory, and best numerical methods. These make it full of sophisticated features that have been improved in its current professional release "6.5.1", and which provide to the final users the below advantages [126]:

- ***Reducing development cost and time:*** with regards to its competitors, AnyLogic has a visual development environment [Fig. 4.1] that considerably speeds up the development process of such model saving time and cost. Its development environment comes with a set of object libraries that provide the ability to quickly integrate reused pre-built simulation objects in a hand; and in the other hand, it has the ability to import models from other widely used IDE. In addition the fact that it's based on Eclipse makes it works in multi operating system (Windows, Mac and Linux).
- ***Developing more models with one tool****:* the crucial feature of AnyLogic is that not only allows the development of agent-based, system dynamics, discrete-event, continuous and dynamic system models [Fig. 4.2] but also it allows combinations between these modeling methods all in one tool. In

AnyLogic users can seamlessly integrate discrete and continuous simulations, and they can limitlessly extend their models with custom Java codes, external libraries and external data sources. AnyLogic provides also a powerful experimental framework that supports extensive statistical distribution functions, Monte Carlo simulations and advanced forms of optimization.

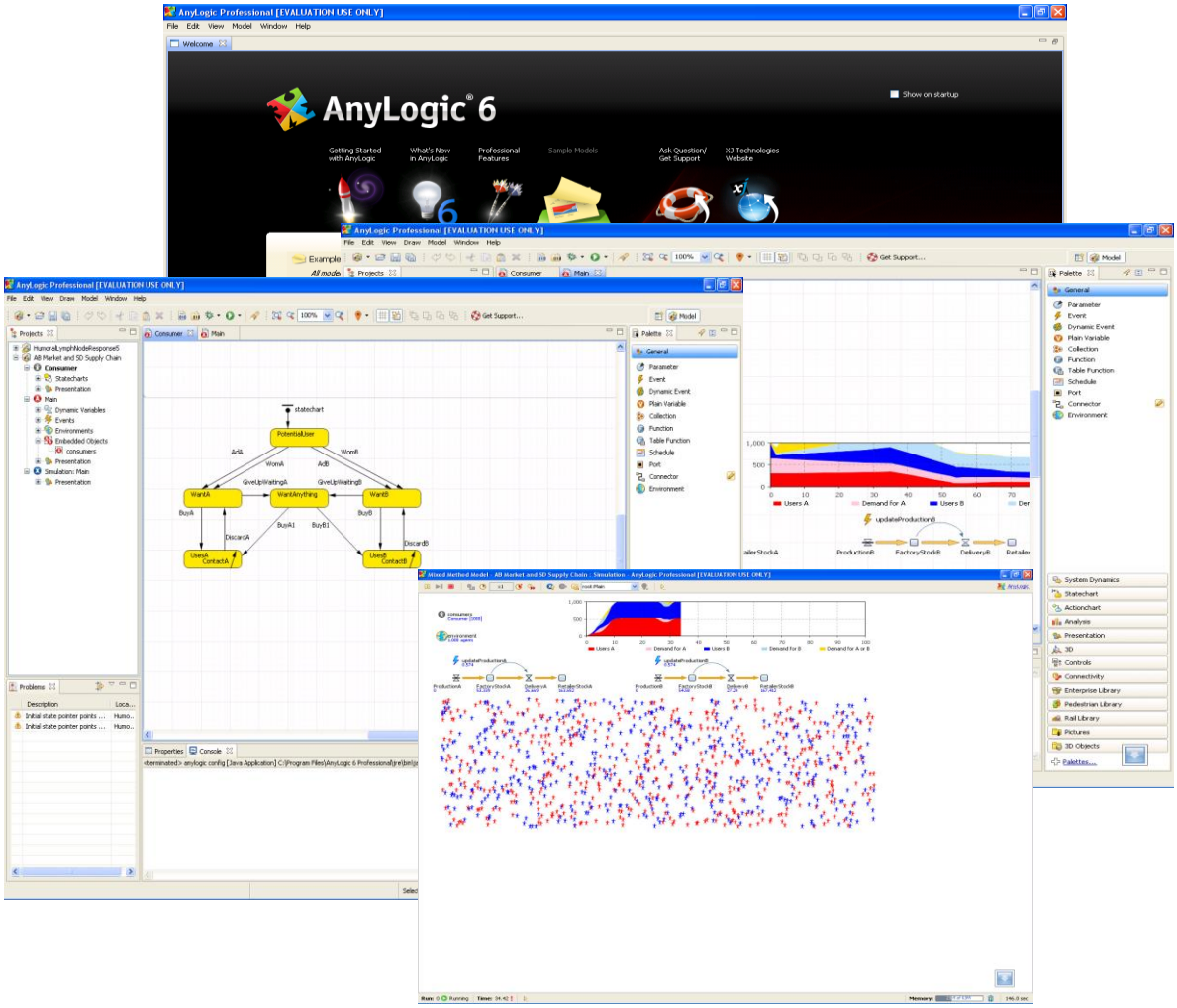


Fig. 4.1: AnyLogic screenshots

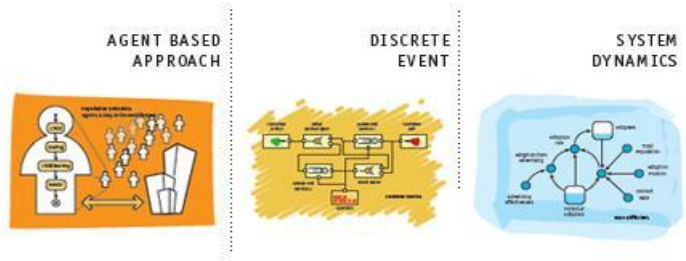


Fig. 4.2: AnyLogic modeling approaches **[126]**.

- ***Improving the visual Impact of models:*** in addition to the sophisticated animation functions that are available in AnyLogic simulation environment and which allow it to develop simulations that are visually rich and interactive; AnyLogic has the ability to export models as Java applets allowing users to run their simulations in a web site.

1.2. AnyLogic Modeling Framework

As it's shown in [Fig. 4.3], the AnyLogic framework based modeling architecture is viewed as superposed layers, in which each layer offers its services to the upper one [127].

The foundation layer is the Java programming language which is considered as a high-level standard cross platform object-oriented language that can "assist" visual modeling language. Besides that the Java Development Kit (JDK) comes with high libraries and packages including UI, graphics, math, animation and others; Java has also the possibility to run applications in Web browsers as Java applets.

The next layer is the UML for Real Time (UML-RT), which is an extension to UML. This last is a general purpose modeling language for specifying, visualizing, constructing and documenting the artifacts of systems; it has a strong set of concepts applicable across domains. The UML-RT is a complete working modeling standard which is specifically fine-tuned for the development of complex, event-driven, real-time systems. Its modeling constructs have rigorous formal semantics that provide for model execution. UML-RT modeling means explicit structural decomposition, clear separation of structure behavior and great degree of reusability. Real world objects are modeled in AnyLogic via a set of "active object" UML-RT classes which may encapsulate other active objects to any desired depth yielding to form a hierarchy of object instances when the model is running. Active objects interact with their surroundings exclusively through interface objects: ports and variables. Ports are used for discrete communication (message passing) and variables are used for continuous communication. The notion of interface makes active object classes highly reusable.

The upper layer is the modeling approach used to model a system. The approach can be a discrete, continuous, or hybrid and belongs to the chosen method

used to model a system; AnyLogic comes with a set of modeling tools that can be used for each approach. For example using discrete approach yields the use of message passing ,message class, port, Statechart , triggering transitions ,timers ,events, threads , event scheduling. Choosing a continuous approach entails the use of: variables, connections, equations, formulas, vector/matrix support, built-In Numerical Methods, using external numerical methods; whereas hybrid approach is allowed via hybrid Statecharts.

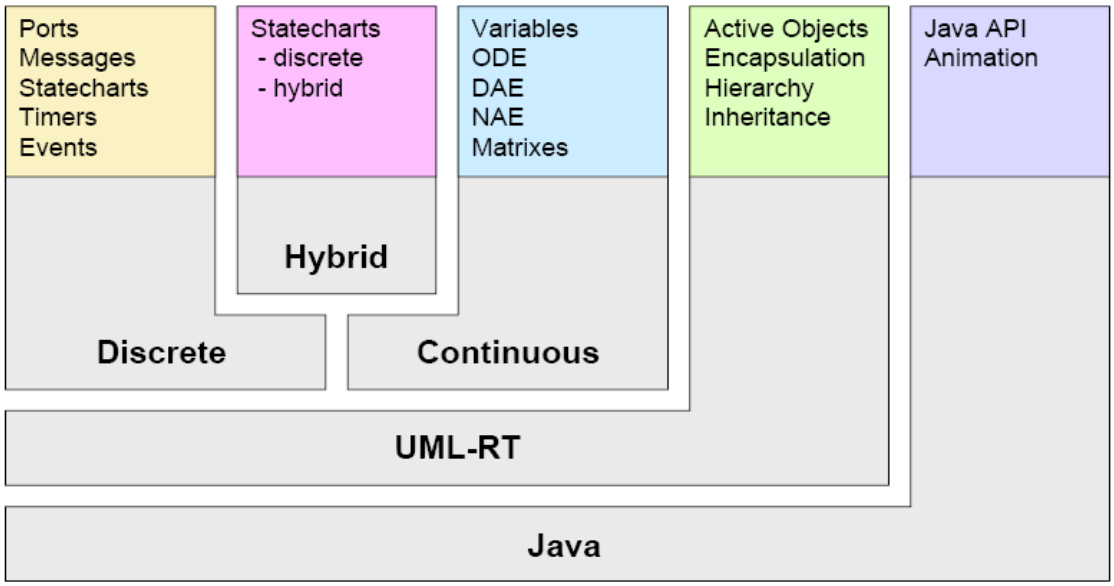


Fig. 4.3: AnyLogic Framework Architecture **[127]**.

1.3. Agent-Based modeling in AnyLogic

AnyLogic has significantly simplified the development of agent-based models due to the visual tools that are available in its development environment. It offers an agent-based modeler design time that can easily model the agents' system and their behaviors, than putting them in a certain environment which can be either continuous or discrete. The agents can form social networks with other agents; can also communicate and send messages to each other; and can move in space, have states, and follow rules.

Unlike the discrete event and system dynamics modeling approaches that are supported in AnyLogic and in which a variety set of libraries are available; AnyLogic hasn't a universal agent-based library that can help reducing the modeler's work.

However AnyLogic simplifies the agent-based models development via offering some reusable design patterns [128]:

1.3.1. Agents

From architectural viewpoint, a typical AnyLogic agent is a special subclass of *ActiveObject* class that extends the latter with services useful for agent based modeling. Belongs to the environment in which the agent is situated, the AnyLogic agent can be either a discrete or continuous type. The user can define any properties, methods or events, and can add behaviors to the agent via the rich components pallet offered by AnyLogic including: UML Statecharts which are used to define agent behaviors, ActionCharts used to define algorithms, Environment objects which help to describe the agent environment and to collect the statistics, Events used to describe occasional or time-certain occurrences. The user can also write specific Java code for special or unanticipated agent models. The defined agents are than embedded into an ActiveObject Main class which is considered as a root class that contains a replicated object of the modeled agents. The main class also may contain one or more Environment constructs to specify properties shared by the replicated agents.

1.3.2. Space

AnyLogic supports two-dimensional continuous and discrete space types for agents:

a. Continuous Space

AnyLogic delivers by the use of this kind of space a set of abilities that can the user implement in its models. Among these abilities we find: setting and retrieving the current agent location, moving the agent with the specified speed from one location to another; executing actions upon arrival, animating agent at its location, establishing connections based on agent layout, and other useful services.

b. Discrete Space

AnyLogic also supports discrete space which is a rectangular array of cells fully or partially occupied by the agents. The cell can only contain one agent at once. The abilities offers by Anylogic for the support of this kind of space includes: distribution of agents over the cells, moving to a neighboring cell or jumping to arbitrary cell,

finding out who are the agent's neighbors (with respect to the neighborhood model), finding empty cells, and other useful services.

1.3.3. Communication between Agents

AnyLogic supports several regular inter-object communication techniques for agents, the user can call methods, send messages via ports, link continuously changing variables, etc.

1.4. AnyLogic Interfaces

AnyLogic comes with a graphical interface [Fig. 4.4], tools, and library objects that allow the user to quickly model diverse areas such as manufacturing and logistics, business processes, human resources, consumer and patient behavior. The object-oriented model design paradigm supported by AnyLogic provides for modular, hierarchical and incremental construction of large models.

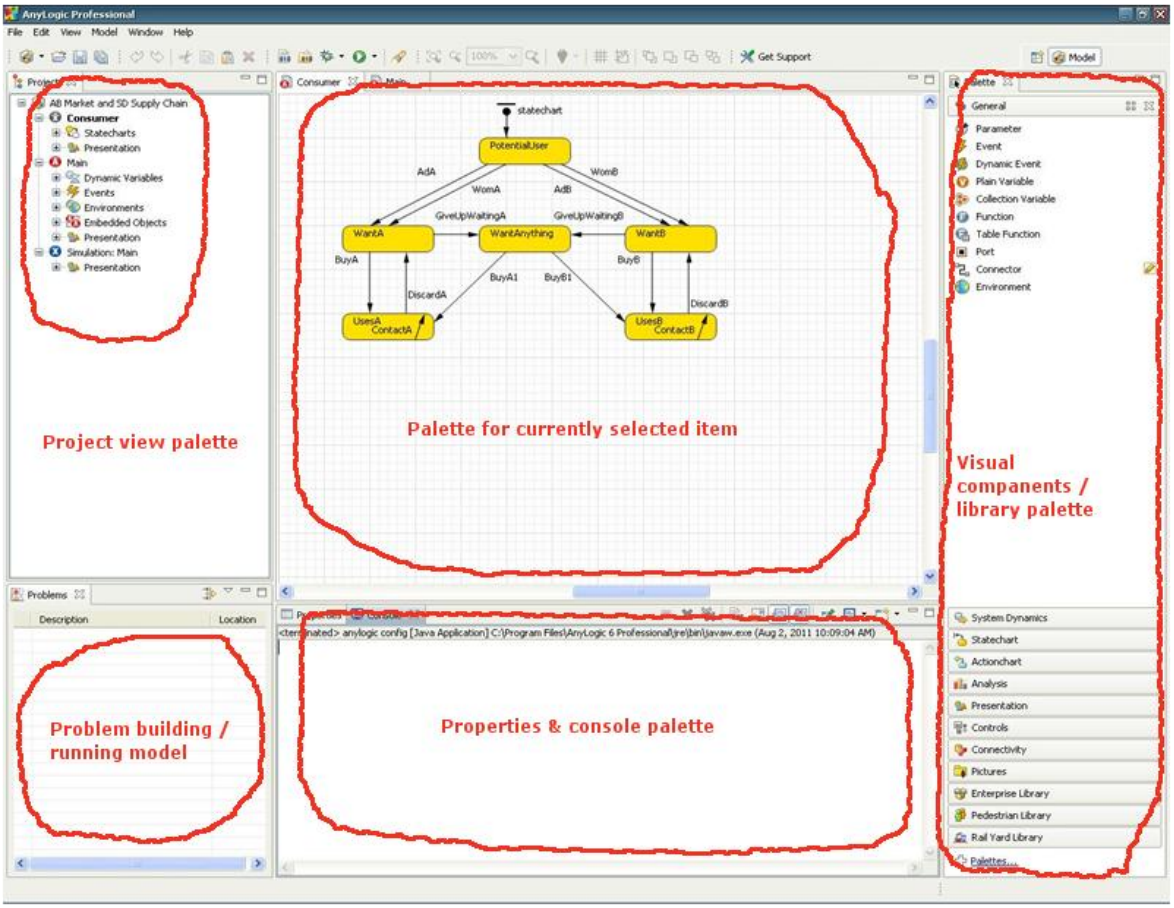
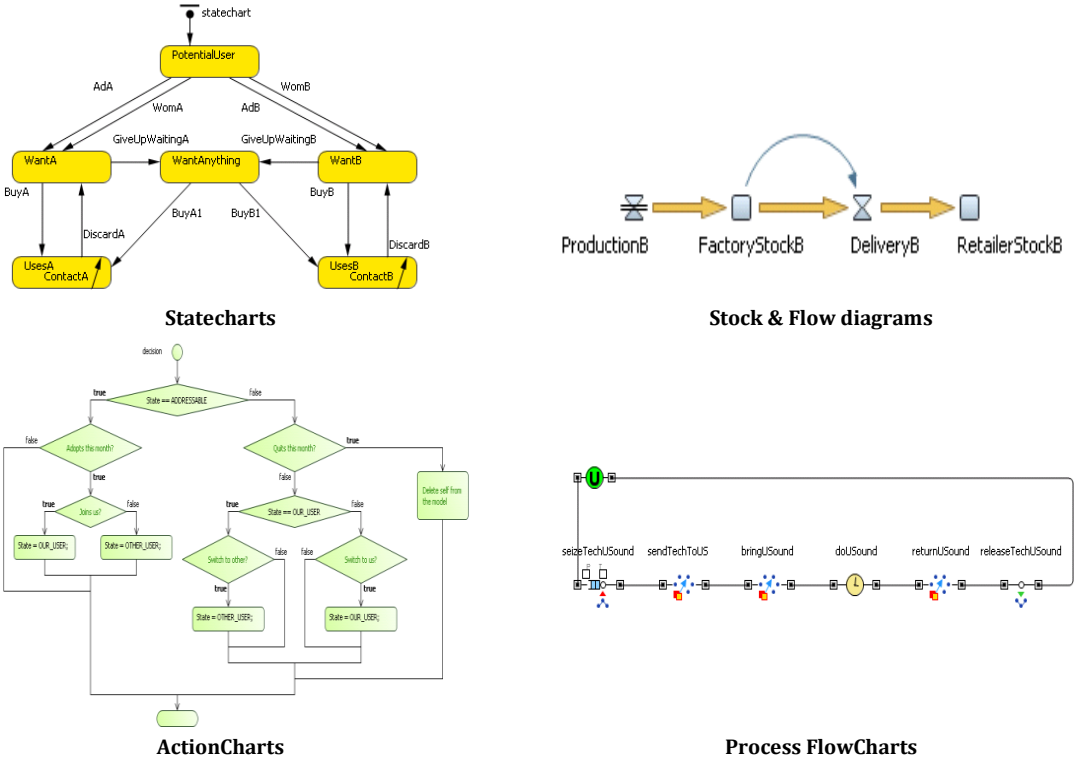


Fig. 4.4: AnyLogic main interface

It's also delivered with the below different visual simulation languages which can be combined together for hybrid modeling [Fig 4.5]:

- **Stock & Flow Diagrams** are used for System Dynamics modeling.
- **Statecharts** are used mostly in Agent Based modeling to define agent behavior. They are also often used in Discrete Event modeling, e.g. to simulate machine failure.
- **Action charts** are used to define algorithms. They may be used in Discrete Event modeling, e.g. for call routing, or in Agent Based modeling, e.g. for agent decision logic.
- **Process flowcharts** are the basic construction used to define process in Discrete Event modeling. Looking at this flowchart you may see why Discrete Event style is often called Process Centric.



Statecharts **Stock & Flow diagrams**

ActionCharts **Process FlowCharts**

Fig. 4.5: AnyLogic visual simulation language.

2. Model of the Lymph Node humoral immune response

We have illustrated in the last section of the first chapter the process of mounting an humoral immune response against T-Dependent and T-Independent Antigens. This process is summarized in the two diagrams (Fig. 4.6 & Fig. 4.7) which simplified the progression of the process cycle.

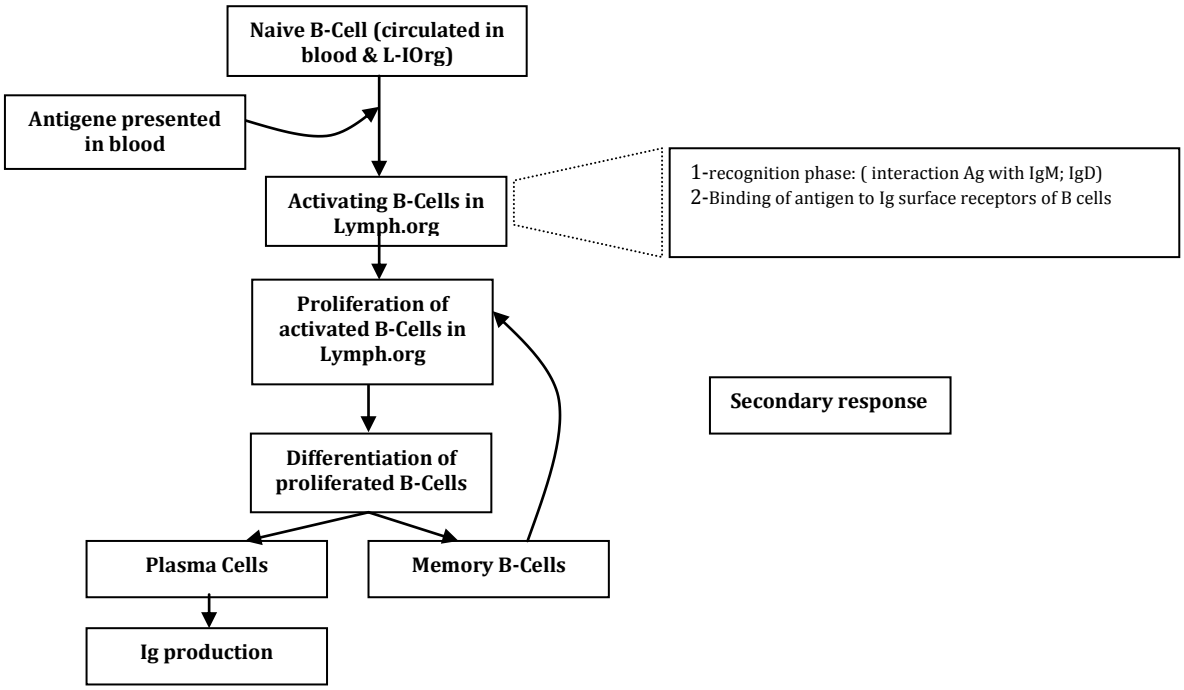


Fig. 4.6: Simplified illustration of the Humoral immune response against T-Ind. Antigen.

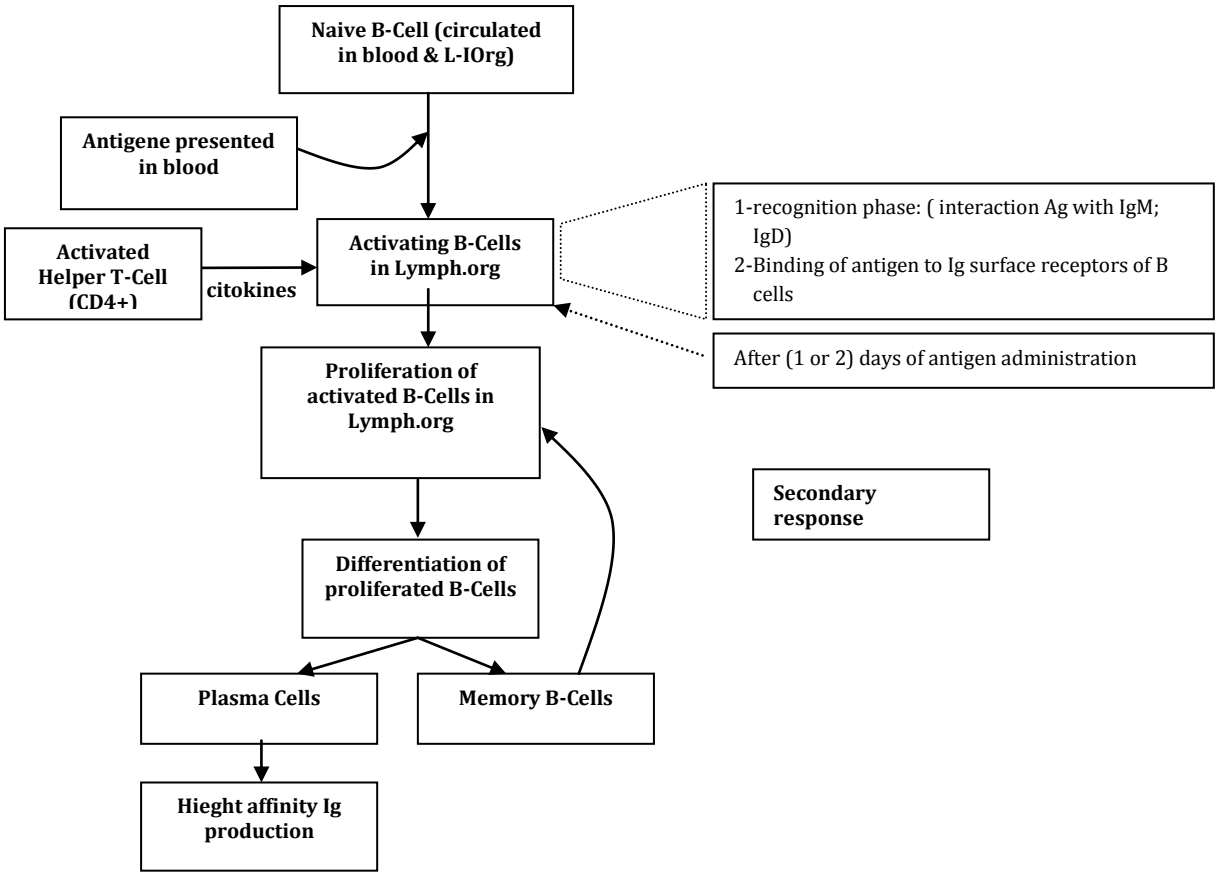


Fig. 4.7: Simplified illustration of the Humoral immune response against T-Dep. Antigen.

In the model developed in this issue we have simplified this process due to the height complexity mechanism founded in it. The modal is an AnyLogic Agent-based model in which the chosen type of agents is ContinuousAgent2D. ContinuousAgent2D is a subclass of the AnyLogic ActiveObject class that offers a full APIs for developing such agent's type. The model focuses on a single, two-dimensional LN from the time of the initial entry of a subset of immune cells together with an immunogenic antigen.

The modeled agents including: lymphocytes (B-Cells, T-Helper Cells, and Memory B-Cells), the plasma cells, the secreting antibodies, and the antigens are modeled with regard to their behavior, movement, location, and interactions. They are in continuous movement in a two-dimensional environment which is represented as a real image of a LN. The environment is divided into different regions that represent the various real LN zones wherein the agents are initially situated and wherefrom they enter or live; the movement of agents between these regions are specified with regard to their behaviors, events and stimulations appeared during their interactions. [Fig. 4.8] gives a simplified process of the whole modeled system from the time that B-Cells enter the LN to the moment that they exit it.

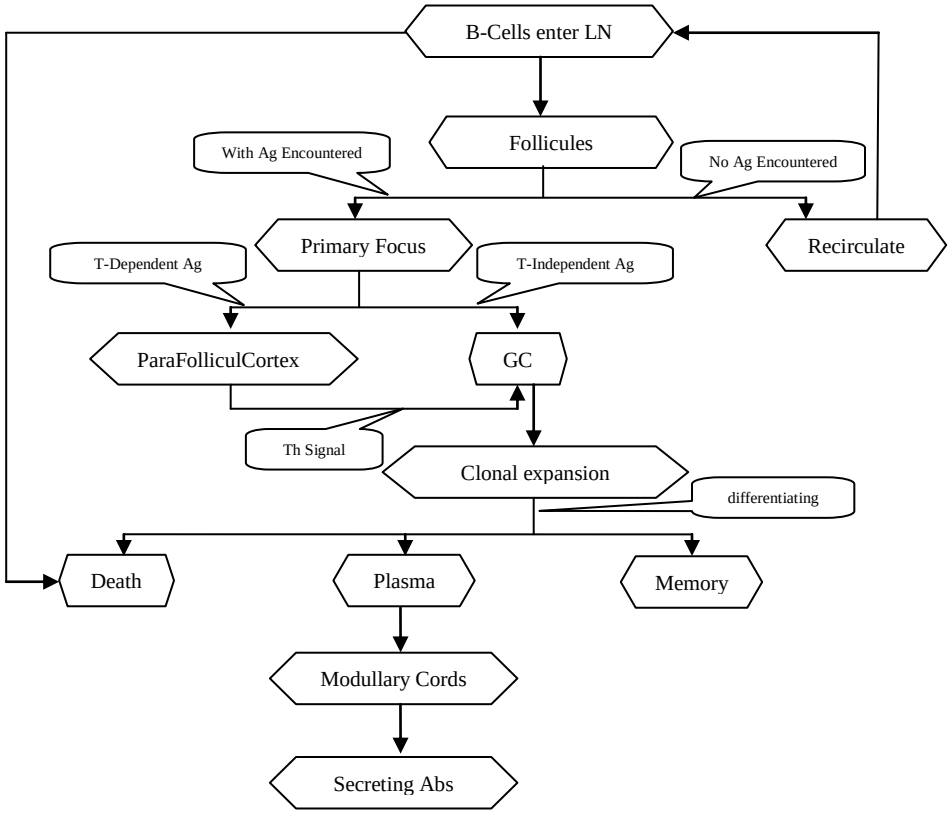


Fig. 4.8: simplified view of the process launched in the LN.

The detailed behind this model is presented in the sections below:

2.1. Modeling the Time

Time is an important factor in every biologic system and in certain cases it's too long to model directly the entire processes that take place during these systems. Immune response is one of these biologic systems for which we should be careful when simulating the time of its processes. In our simulation we have tried to keep the relative times between its different processes correct; for that we have used the AnyLogic simulated time unit (TU) which is fixed to (0,001) and which in our model corresponds to 1 second so an hour is evaluated to (0,36 UT). In immunology many processes, for which time is an important factor, were examined; for example: a typical lymphocyte circulation cycle takes 12–24 hours [4]; normal proliferation takes 8–12 hours [4]; etc. In our model, we attempted calculating the corresponding time values, although adjustments were made to allow for technical obstacles.

2.2. Modeling the LN environment

In our AnyLogic simulation of the LN B-Cell response to a T-Dependent and T-Independent antigens, we have tried to let the simulation realistic. For that it's suitable to use a real image of a LN (taken from [129]) which can show its different constitute regions. LNs can be functionally separated into three major areas each supporting a different cellular environment: the cortex, the paracortex, and the medulla. The cortex zone is mostly composed of B-Cells and various APCs, the paracortex zone where B-Cells and T-Helper Cells interact, and the medulla contains mostly lymphocytes including the antibody producing plasma cells.

To model these regions we have used a set of closed curves each represents a special LN areas. As it's shown in the [Fig. 4.9] the center LN is chosen to be the environment where all the interacted system' agents are distributed. The agents can move continuously between these areas with regards to the biologic experiments. The detailed process of circulation of each agent is presented later.

In the main ActiveObject class of the simulation, we have developed a set of functions [Table 4.1] that is shared by all the agents indicating the movement to one zone to another, for example the function *moveToGC(AgentContinuous2D cell)* move a given cell from its current location to one of the GCs illustrated in the Fig. the path

followed to reach the target location is specified automatically by the AnyLogic move API.

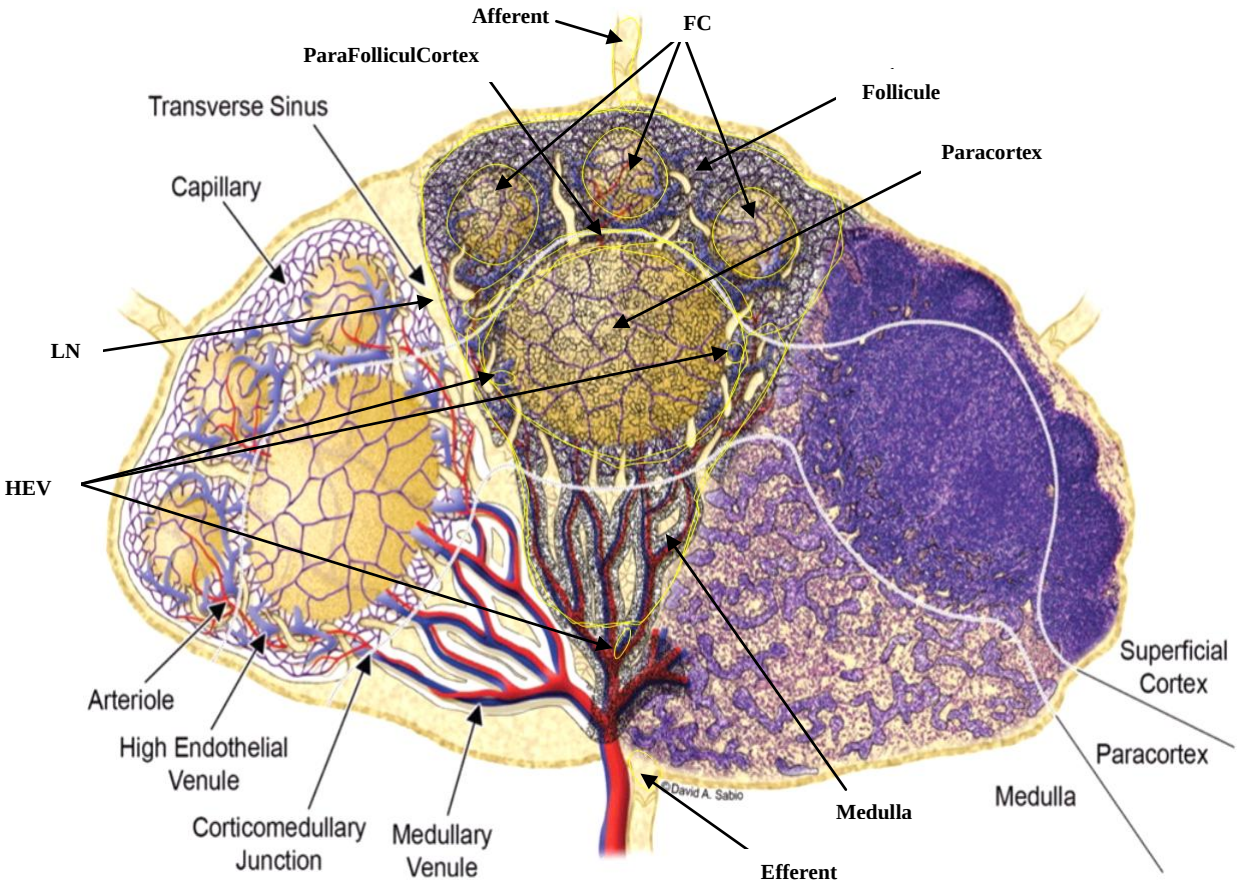


Fig. 4.9: The AnyLogic simulation LN zones.

Function	Description
moveToGC(AgentContinuous2D cell)	Move a given cell from its current location to one of the GCs area.
moveToFollicules(AgentContinuous2D cell)	Move a given cell from its current location to the Follicule area.
moveToParacortex(AgentContinuous2D cell)	Move a given cell from its current location to the Paracortex area.
moveToModullaryCords(AgentContinuous2Dcell)	Move a given cell from its current location to the ModullaryCords area.
moveToParaFolliculesCoretex(AgentContinuous2Dcell)	Move a given cell from its current location to the ParaFolliculesCoretexarea.
moveToEfferent(AgentContinuous2Dcell)	Move a given cell from its current location to the Efferentarea.
setInHEV(AgentContinuous2Dcell)	Sets the location of the given cell in one of the HEVs area.
setInAfferent(AgentContinuous2Dcell)	Sets the location of the given cell in the Afferent area.

Table 4.1: List of the used functions that control the movement of cells from and into LN areas.

2.3. Modeling the Immune Cells agents

As we are simulating a part of the humoral immune response being initiated in the LN against both T-Dependent and T-Independent antigens, the captured developed agents include:

- B-Cells matured to recognize T-Independent Antigen (BCellsTInd).
- B-Cells matured to recognize T-dependent Antigen (BCellsTDep).
- T-Helper Cells (THCells).
- Memory BCells differentiated from proliferated BCellsTInd (BMemoryTInd).
- Memory BCells differentiated from proliferated BCellsTDep (BMemoryTDep).
- Plasma Cells differentiated from proliferated BCellsTInd (PlasmaTInd).
- Plasma Cells differentiated from proliferated BCellsTDep (PlasmaTDep).
- Antibodies secreted from PlasmaTInd (AbTInd).
- Antibodies secreted from PlasmaTDep (AbTDep).
- T-Independent Antigen Presenting Cells (AgTIndAg).
- T-Dependent antigen T-Independent (AgTDepAg).

As the number of agent's type to be modeled in this system is greater than the number allowed in AnyLogic evaluation license which allows only modeling five (5) ActiveObject classes; has let us assembling the five first captured agents (BCellsTInd, BCellsTDep, THCells, BMemoryTInd, and BMemoryTDep) in one ActiveObject class named lympohocyte, gathering the two agents (PlasmaTInd and PlasmaTDep) in Plasma' ActiveObject, coupling also the AgTIndAg and AgTDepAg agents in the Antigen ActiveObject, and assembling the AbTInd and AbTDep in the Antibody ActiveObject.

The behavior of each modeled agent is specified by the use of the AnyLogic integrated Statecharts formalism. Each behavior is divided into two main Statecharts: one to modal the life cycle of the agent, the other modal the location cycle of the agent. The life cycle statecharts focuses on the concerned agent life from its first existence till its death passing from state to another with regards to the biologic experimentations. Whereas the location cycle statecharts concentrates on the current location of the concerned agent who is on continuous movement from LN zone to another basing on the events occurred during its movement in a hand, and on the actual life cycle state in the other hand.

2.3.1. The Lymphocyte Agent

The lymphocyte agent models different lymphocytes including-Cells, Memory B-Cells, and T-Helper Cells. For each of the B-Cells and Memory B-Cells two kinds of cells are taken into account: ones are matured to recognize T-Dependent antigens, the others are matured to recognize T-Independent antigens; the T-Helper Cells are implicated only in the humoral immune response to T-Dependent antigens.

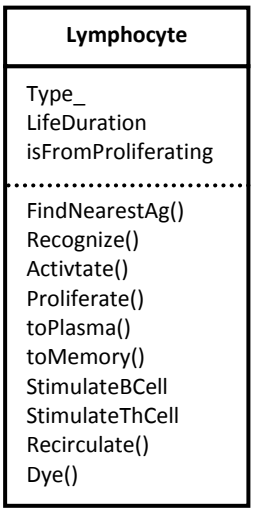
The biologic experimental that has shown in the first chapter illustrates that B-Cells enter permanently into the LN via HEVs and migrate to its zone area (Follicules) where they may meet antigens. If the B-Cell recognizes the encountered antigen, the humoral immune response process will differ belongs to the presented antigen type:

If the antigen is a T-Independent one the B-Cell becomes an activated cell and migrates to the Follicule Center (FC) where it begins what the immunologist called colonel expansion phase. At this moment a large number of B-Cell clones are generated; some of them become a plasma cells, others become a memory B-Cells and the others are died.

Whereas if the type of presented antigen is a T-Dependent: the stimulated B-Cell will be completely activated when it migrates to the paracortex zone and waits for an activated T-Helper Cell which activates the B-Cell after a set of stimulating events interactions between them. The activated B-Cell will than take the same process of colonel expansion followed by a matured B-Cell activated by a T-Independent antigen.

All the lymphocyte cells which either recognize or do not recognize antigen live the LN via efferent zone than restart the recirculation process. They are death after an expiration of their life duration.

In our AnyLogic modal we have developed a Lymphocyte class which extends the AnyLogicAgentContinuous2D subclass. Its properties, methods and Statecharts behavior are given below:



Property/Methode	Type	Description
Type_	Int	Lymphocyte Type: • 0: B-Cell response to T-Independent Antigen. • 1: B-Cell response to T-Dependent Antigen. • 2: Memory B-Cell response to T-Independent Antigen. • 3: Memory B-Cell response to T-Dependent Antigen. • 4: T-Helper Cell.
LifeDuration	Int	Time unit representing the duration Life of the cell (default 500)
isFromProliferating	boolean	Source of Lymphocyte born: • False: cell is previously existed (from the initiation process of the modal). • True: cell is born from the proliferating phase.
FindNearestAg()	Void	Looking on continuous manner for a nearest APC that hold an antigen, the finding method is calculated based on the nearest distance.
Recognize()	Void	Antigen recognizing phase of the current lymphocyte.
Activtate()	Void	Activating phase of the current lymphocyte
Proliferate()	Void	Proliferating phase of the current lymphocyte with regards to the proliferated rate defined by the user.
toPlasma()	Void	Differentiating phase of the proliferated lymphocyte to a Plasma cell, the rate of differentiating is defined by the user.
toMemory()	Void	Differentiating phase of the proliferated lymphocyte to a Memory cell, the rate of differentiating is defined by the user.
Recirculate()	Void	Enter again the LN.
StimulateBCell	Void	Stimulate a B-Cell in order to be activated.
StimulateThCell	Void	Stimulate a Th-Cell in order to be activated.
Dye()	Void	Remove the lymphocyte from the simulation in case that LifeDuration is expired.

Table 4.2: Properties and functions of the Lymphocyte Agent.

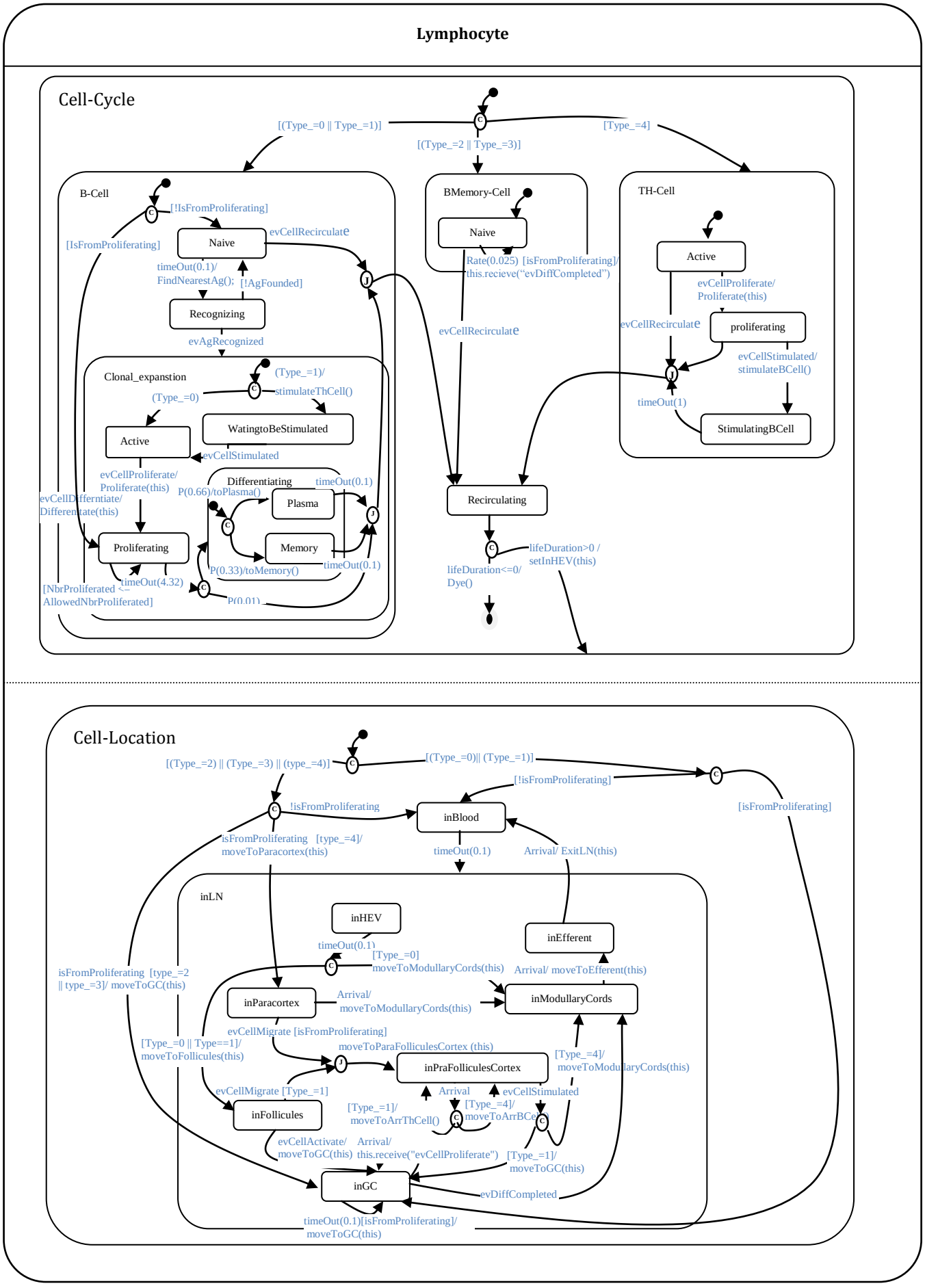
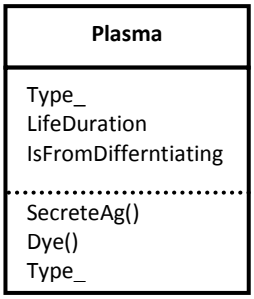


Fig. 4.10: The Statecharts behavior of the Lymphocyte Agent.

2.3.2. The Plasma Agent

Plasma cells are specialized immune cell that secrete special antibodies to kill antigens. They are founded in the LN medulla area where they initiate the process of generating a large number of antibodies. Initially plasma cells are generated in the FCs after the differentiation process of a clonal expansion cycle of an activated B-Cell; they migrate to the modularly cords zone where they begin secreting the specified antibodies and inject them into the blood to kill antigens.

Alike the Lymphocyte agent, we model this aspect by a Plasma class that extends the AnyLogic AgentContinuous2D subclass. The properties, methods and the Statecharts behavior that enrich the Plasma class are illustrated bellow.



Property/Methode	Type	Description
Type_	Int	Plasma Type: • 0: Plasma response to T-Independent Antigen. • 1: Plasma response to T-Dependent Antigen.
LifeDuration	Int	Time unit representing the duration Life of the cell (default 500)
IsFromDifferntiating	boolean	Source of PlasmaBorn: • False: cell is previously existed (from the initiation process of the modal). • True: cell is born from the differentiating phase.
secreteAg()	void	Differentiating phase of the proliferated lymphocytes to one of memory B-Cells or Plasma cells, the rate of differentiating is defined by the user.
Recirculate()	Void	Enter again the LN.
Dye()	Void	Remove the lymphocyte from the simulation in case that LifeDuration is expired.

Table 4.3: Properties and functions of the Plasma Agent.

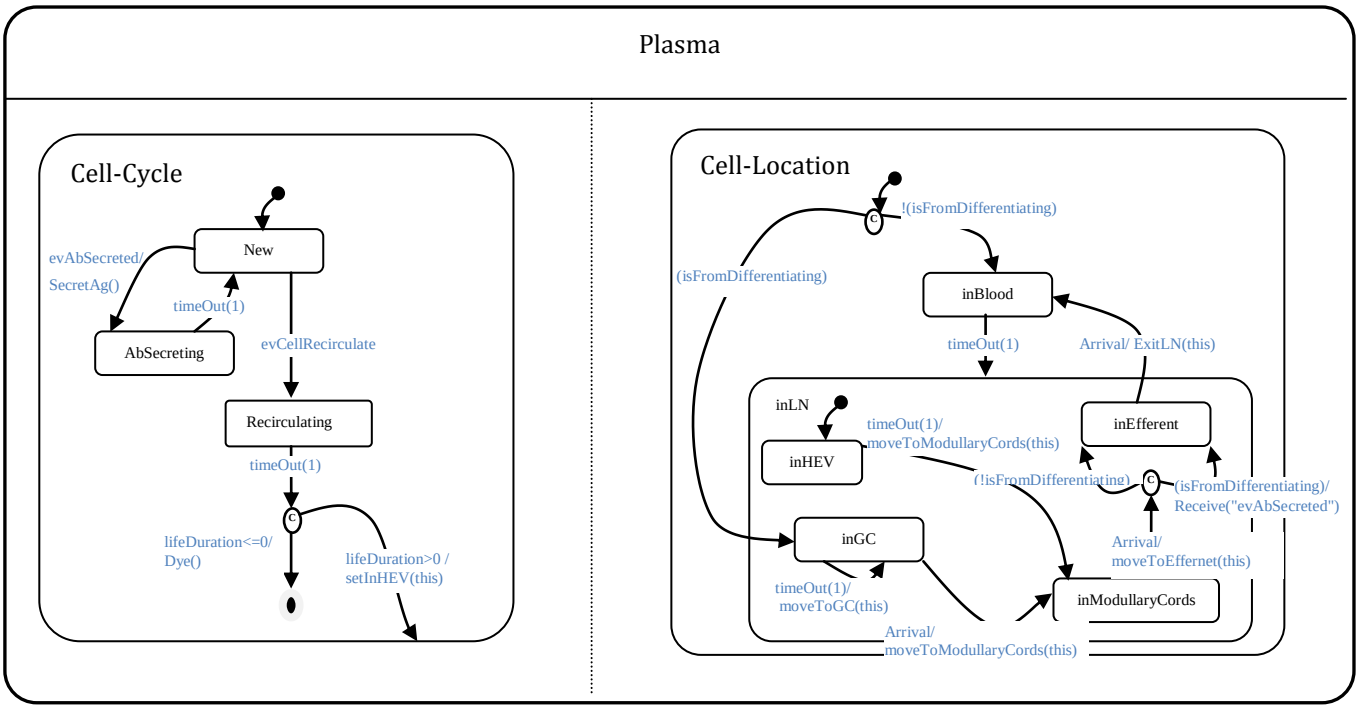


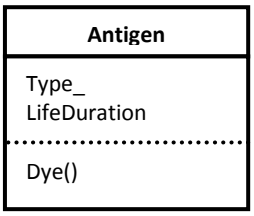
Fig. 4.11: The Statecharts behavior of the Plasma Agent.

2.3.3. The Antigen Agent

Antigens can be recognized directly by B-Cell after a set of chemical interaction between them. They are also recognized by TH-Cells with the assistance of a specialized immune cell called Antigen Presenting Cells (APCs). Antigens are on continuous circulation in our bodies; they are mostly recognized in the LN where they enter it via afferent vessels then exit it via efferent area.

For the raison to simplify our modal the process of how APCs capture an antigen and present it to Th-Cells is not specified heir. In our modal we have modeled two kinds of Antigens: the first a T-Independent antigen (AgTIndpAg), the second is a T-Dependent antigen (AgTDepAg). In our simulation, Antigens are on continuous movement entering and living the LN. Antigens that are recognized by a B-Cell do not re-enter the LN and they are completely removed from the simulation to simulate that the antigen is killed.

Similar also to the modeled plasma cells and modeled lymphocyte cells and as it's mentioned in the following illustrations, we have developed an Antigen class which is an AgentContiuous2D subclass:



Property/Methode	Type	Description
Type_	Int	Antigen Type: • 0: T-Independent Antigen. • 1: T-Dependent Antigen.
LifeDuration	Int	Time unit representing the duration Life of the antigen (default 500)
Dye()	Void	Remove the antigen from the simulation.

Table 4.4: Properties and functions of the Antigen Agent.

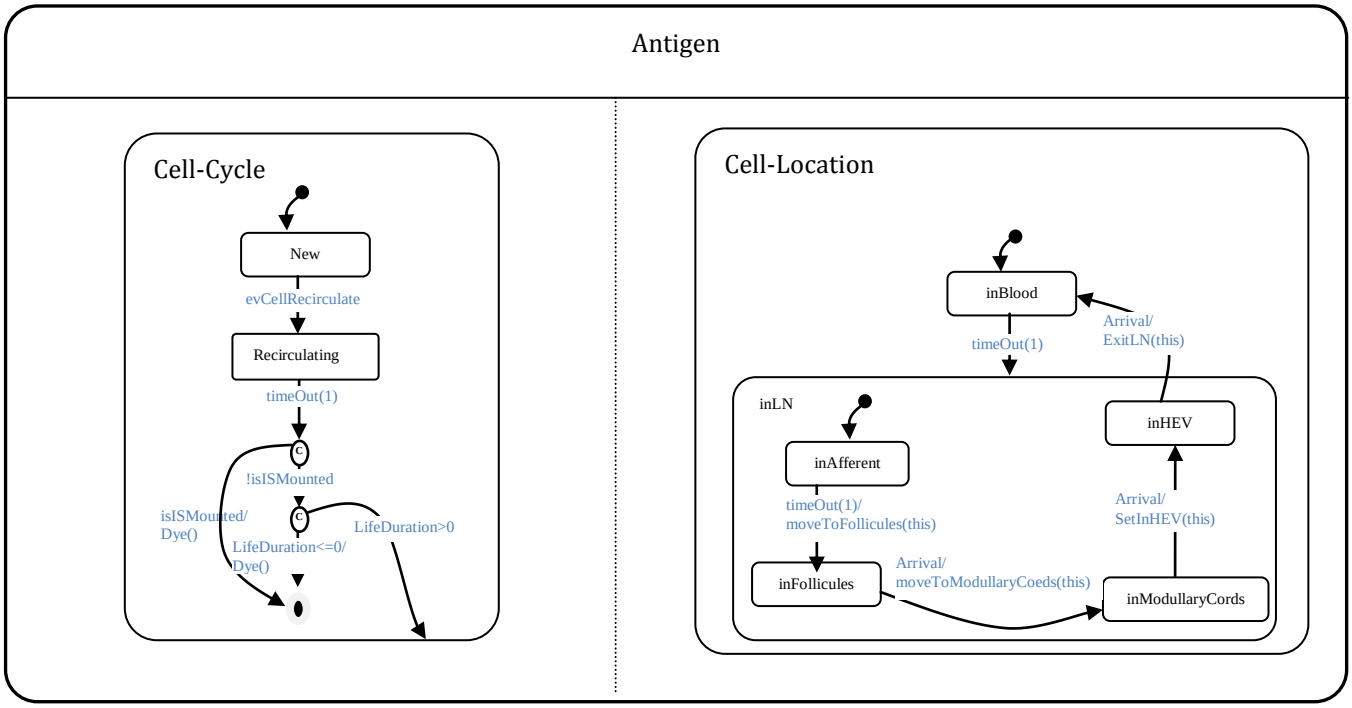


Fig. 4.12: The Statecharts behavior of the Antigen Agent.

2.3.4. The Antibody Agent

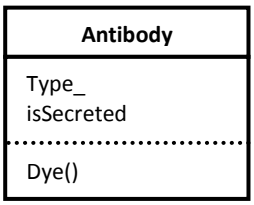
Antibodies (Abs) are specific molecules that are produced by plasma cells.

They are mostly generated in the LN modularly cords zone then are injected in blood via efferent vessel to assist killing antigens. As we have viewed in the first chapter, antibodies belong to variant classes of immunoglobulin (Igs) molecules.

In our simulation not all the Igs classes are modeled, as instances we have only simulated two kinds among them: IgsM that are produced under the direct

stimulation of B-Cells by T-Independent antigens, and IgsG generated by the Th-Cell B-Cell stimulations. The outside antigens killing process is not taken into account heir for the raison that we have only modeled the process initiated inside the LN.

The antibody features and its behavior are modeled via an AgentContinuous2D subclass called antibody class. The below illustrations show its main properties, methods, and Statecharts behavior.



Property/Methode	Type	Description
Type_	Int	Lymphocyte Type: • 0: B-Cell response to T-Independent Antigen. • 1: B-Cell response to T-Dependent Antigen.
isSecreted	boolean	Source of born: • False: Antibody is previously existed (from the initiation process of the modal). • True: Antibody generated by plasma cells
Dye()	Void	Remove the antibody from the simulation.

Table 4.5: Properties and functions of the Antibody Agent.

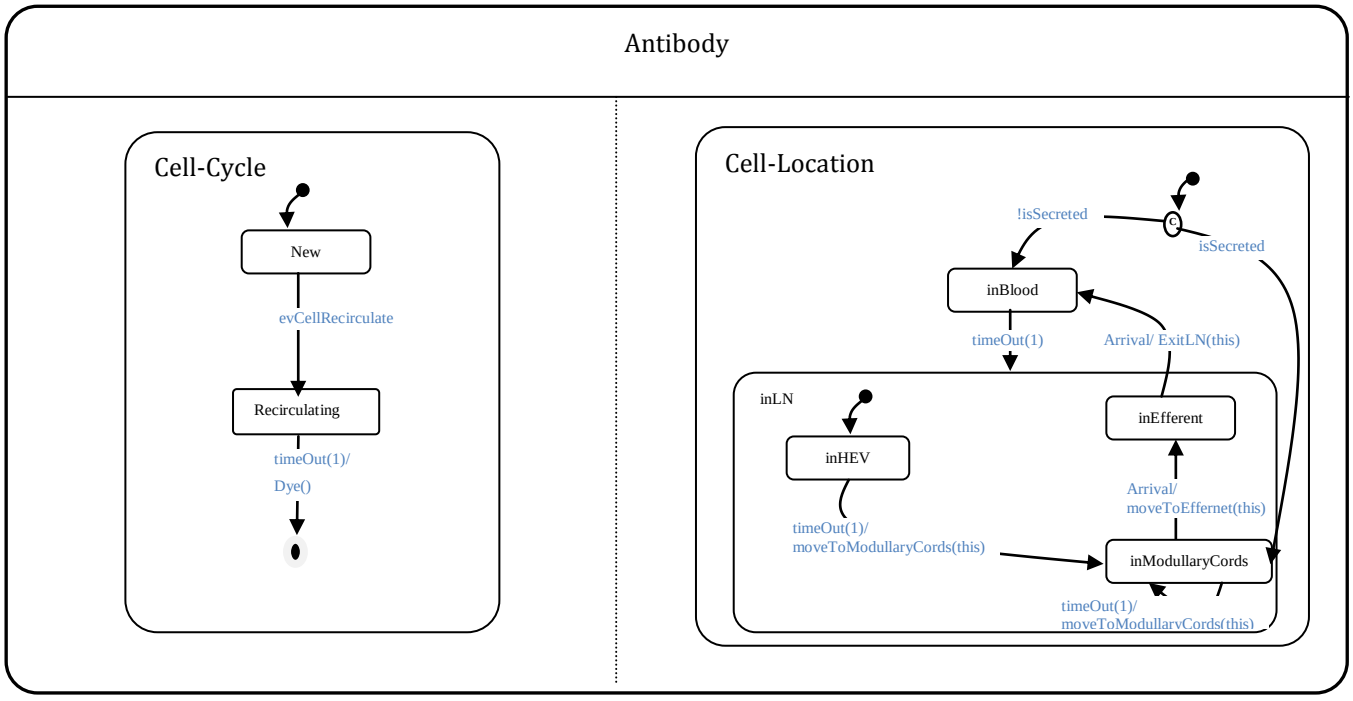


Fig. 4.13: The Statecharts behavior of the Antibody Agent.

2.4. The Main Classes

During the simulation execution all the above different shown agents are situated in the LN environment. The dispatching of each agent inside its specific LN area is initially calculated in arbitrarily fashion then each agent will follow its target LN area with regards: to its type, its current LN zone and the interactions that may take place during its movement.

In order to orchestrate and control the whole modeled system including its various agents with their movement into and from the major LN areas, an AnyLogic main class has been developed in which we have specified (Fig. 4.15):

- The LN environment as it's illustrated in the section (2.1).
- A collection of each modeled agent including (B-Cells, Th-Cells, Plasma Cells, Antigens, and Antibodies).
- A set of variables and functions used to develop the model.
- A set of time statistical analyses about the current happened simulation
- A set of input parameters that the user can change before and during the simulation. The parameters includes (Number of initial agents that compose the system "defined in the initial experimental simulation class (Fig. 4.14)", controls to set and modify the proliferated, differentiated, and antibody secreting rates, and finally controls to inject antigens in the LN).
- A colored legend panel that shows the presentation used to simulate each agent.

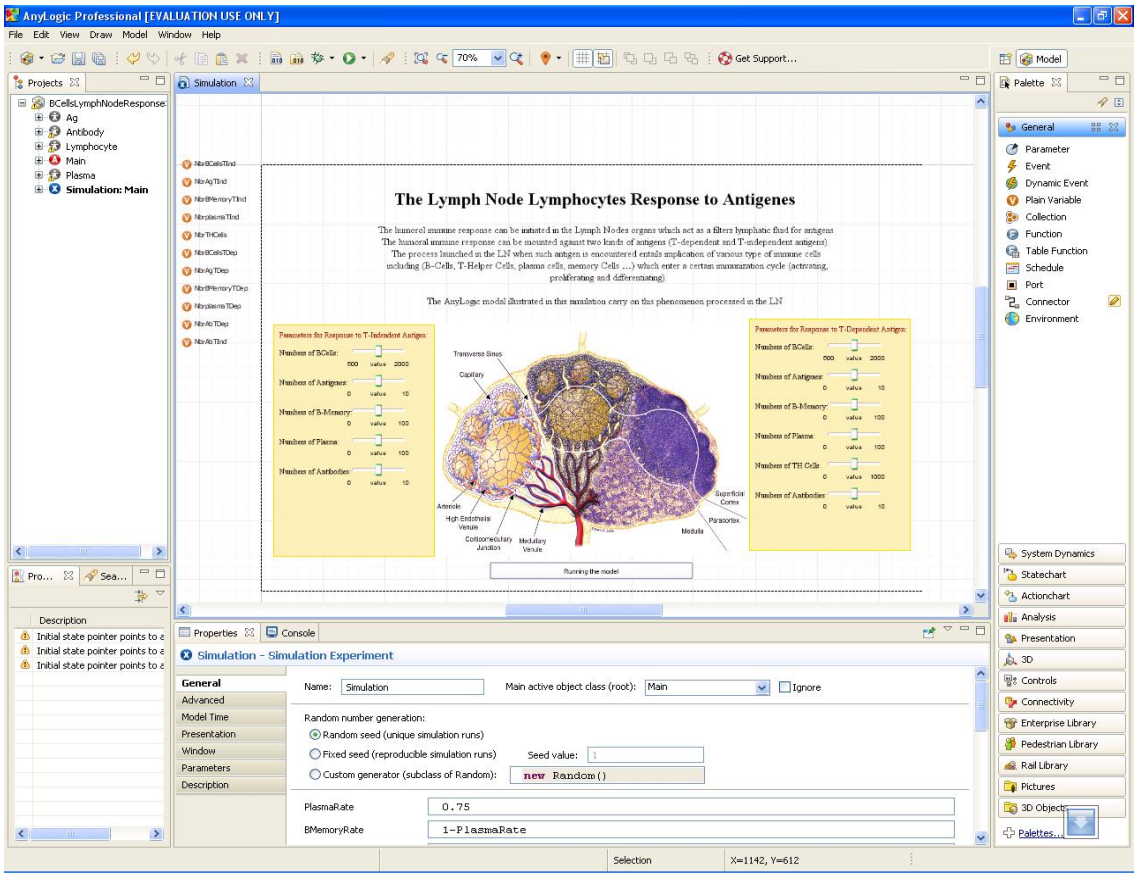


Fig. 4.14: The LN simulation main class.

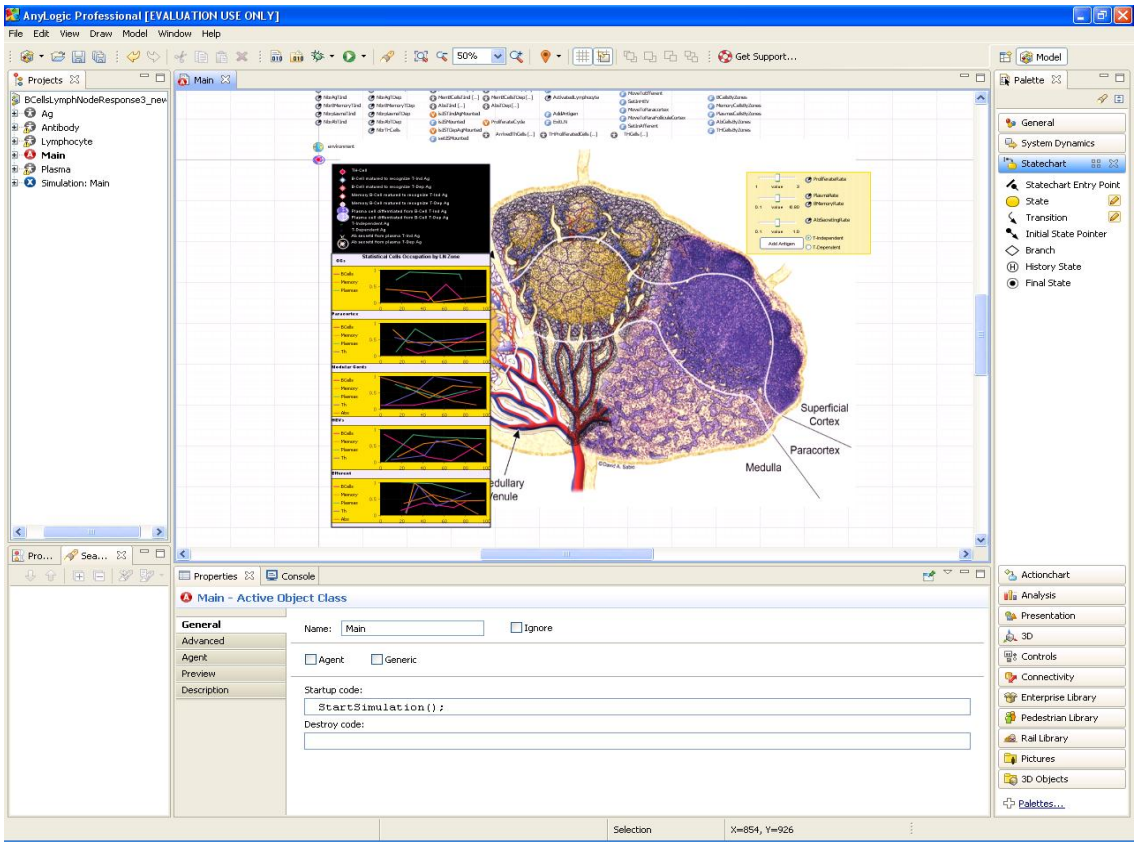


Fig. 4.15: The LN main root class.

3. Results, discussion and future work

3.1. Results

In this section we describe firstly the behavior of the simulator, and then show the type of results it generates. During a typical run of the simulator, a number of emergent behaviors can be seen that result from the rules of the model described in the previous section. At the beginning, the user defines the initial number of different immune response agents that are implicated in the simulation [Fig 4.16], the agents include: Th-Cells, two kinds of antigens (T-Dependent antigens and T-Independent antigens), two kinds of B-Cells each kind is matured to recognize an antigen type, and as also by the same two kinds of B-Memory Cells, two kinds of plasma and two kinds of secreting antibodies.

After all these parameters were specified, the user can switch to the root simulation that shows the initial allotment of the entire implicated agents in their LN zones. For an illustration sample, the [Fig. 4.17] shows an initial result simulation of the modeled system for the parameters' values given in table [Table 4.6]. The running simulation illustrates firstly the random distribution of the concerned defined agents; as we have explained previously each cell agent is located in its specified LN zone. Than the simulation begins showing the movement of each agent from its current location to its target zone with regards to the movement rules defined in its Statecharts location behavior. The user can interact whenever he wants with the simulation interface by a set of available controls: for example he can add T-Dependent or T-Independent antigens to the simulation. In [Fig. 4.18] there is a situation in which some T-Independent antigens start moving after they are injected to the LN via the Afferent zone; if any specified B-Cell that is matured to recognize this kind of antigen has encountered the antigen, the humoral immune response will instantly begin processing from the activation of the specified B-Cells to the clonal expansion phase finished by plasma secreting antibody phase. In our model the details behind the antigen recognition phase isn't taken into account due to the extreme chemical interconnection signals known in this case between an antigen and a B-Cell; in consequence we have only develop an event that periodically calculate the distance between the current B-Cell and all the antigens injected in the LN; if the calculated distance is less or equal to two (2), the concerned B-Cell is then becoming in the active state of the *Clonal_Expansion* composite state that invokes an immediate

migration of the concerned B-Cell to one of the FCs and starts proliferating with a specific modified proliferate rate initially has the value $\rho = (24/6.24) * Ln(2)$ [130]. The proliferation process, which itself involves the creation of an additional instance of the same object, is stopped when the number of the total proliferate B-cells exceeds the allowed total proliferated number which in our model assigned the value ($T_{prolif} = (2^{\rho} - 1) * 100$). The proliferated cells will then either dye with a probability of *1%* [4] or differentiate either to Plasma Cells or Memory Cells; the probability of this differentiating phase is defined by the user (the initial used probabilities are [4]: $(P_{plasma} = 66\%)$ to become a plasma cell and $(P_{mem} = 33\%)$ to become a Memory Cell). After that the generating cells migrate to the Modullary Cords zone where each plasma cell has a user defined probability initiated to *25%* to begin secreting a huge number of antibodies; it's around 2000 Ab's are secreted every second for a few days [4]; in our model the total number of secreted antibodies is fixed to $T_{Ab} = (T_{prolif} * P_{plasma}) / 2$ per plasma cell for the raison of the limitations of the resources (memory and processor frequency) of the computer we have used for the simulation.

The simulation illustrated from [Fig. 4.18] to [Fig. 4.22] mention a situation such this in which a complete first immune response is simulated beginning from the injection of a set of T-Ind antigens [Fig. 4.18] to a proliferation phase shown in [Fig. 4.19] to a differentiation phase mentioned in [Fig. 4.20] and [Fig. 4.21] and finally to a secreting antibodies phase viewed in [Fig. 4.22].

During the simulation the user can also modify the parameters that control the proliferation rate, differentiation rate and antibody secreting rate. The illustrations shown in the last cited figures ([Fig. 4.18] to [Fig. 4.22]) correspond to the rate values cited in the [Table 4.7]. Our model offers also statistical presentations to the user showing him in every time unit the occupation of the total number of each Cell per LN zone; the graphs viewed in the left side of the shown snapshots illustrate the occupation of: the both types of B-Cells, Memory Cells, Plasma Cells, Th-Cells and Antibodies for each of: the GCs zone, the Paracortex zone, the Modullary cords zone, the HEVs zone and the Efferent zone.

parameter	value
Initial number of B-Cells matured to recognize T-Indpendent Antigen	500
Initial number of B-Cells matured to recognize T-Dependent Antigen	500
Initial number of B-Memory Cells matured to recognize T-Indpendent Antigen	50
Initial number of B-Memory Cells matured to recognize T-Dependent Antigen	50
Initial number of Plasma Cells differentiated from B-Cells matured to recognize T-Indpendent Antigen	50
Initial number of Plasma Cells differentiated from B-Cells matured to recognize T-Dependent Antigen	50
Initial number of T-Indpendent Antigen	0
Initial number of T-Dependent Antigen	0
Initial number of antibodies generated from plasma cells differentiated from B-Cells matured to recognize T-Indpendent Antigen	50
Initial number of antibodies generated from plasma cells differentiated from B-Cells matured to recognize T-Dependent Antigen	50
Initial number of Th Cells	500

Table 4.6: Initial values of the parameters used during the experimental simulation.

Rate	Description	value
Proliferate Rate	The rate of generated B-Cells for every activated B-Cell	ρ = (24/6.24) *Ln(2) [130]
Plasma Rate	The rate of differentiated Plasma cells	66% [4]
B-Memory Rate	The rate of differentiated B-Memory cells	33% [4]
Antibody Secreting Rate	The rate of secreted antibodies for a given plasma Cell	25%
Death Rate	The rate of death cell among the proliferated Cells	1% [4]

Table 4.7: Rate values of the experimental simulation

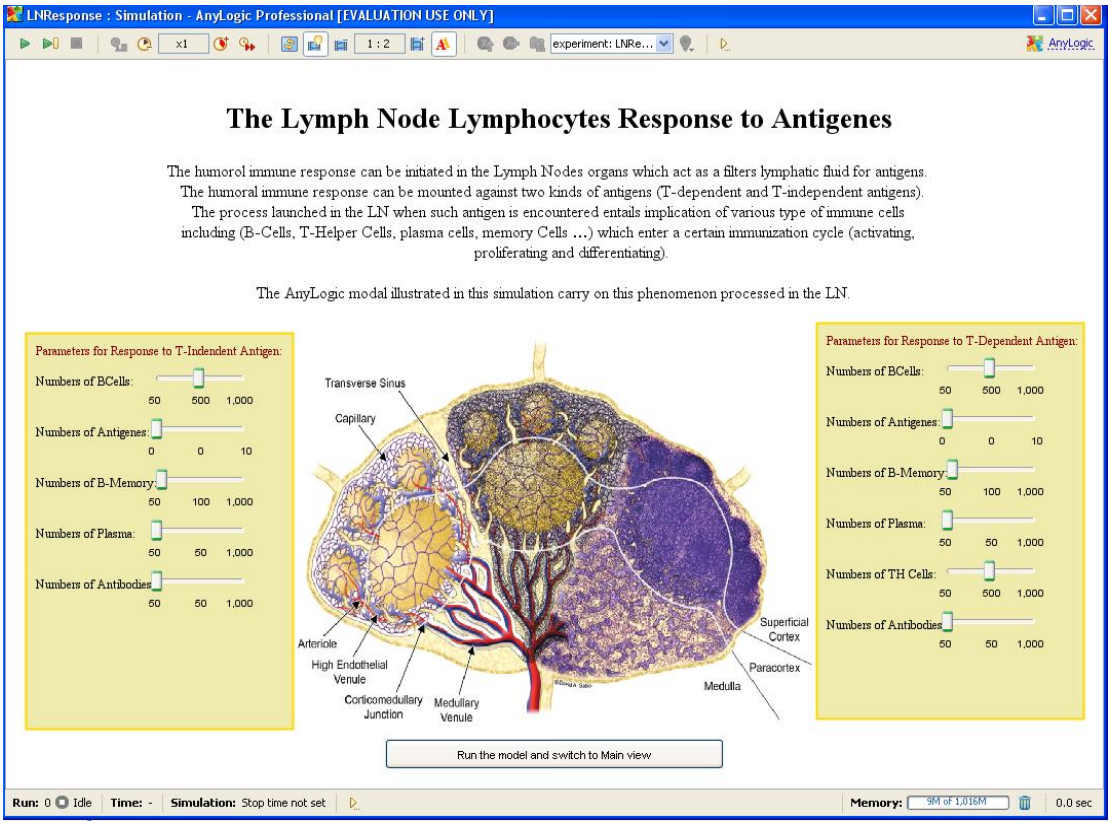


Fig. 4.16 : The initial running experimental simulation

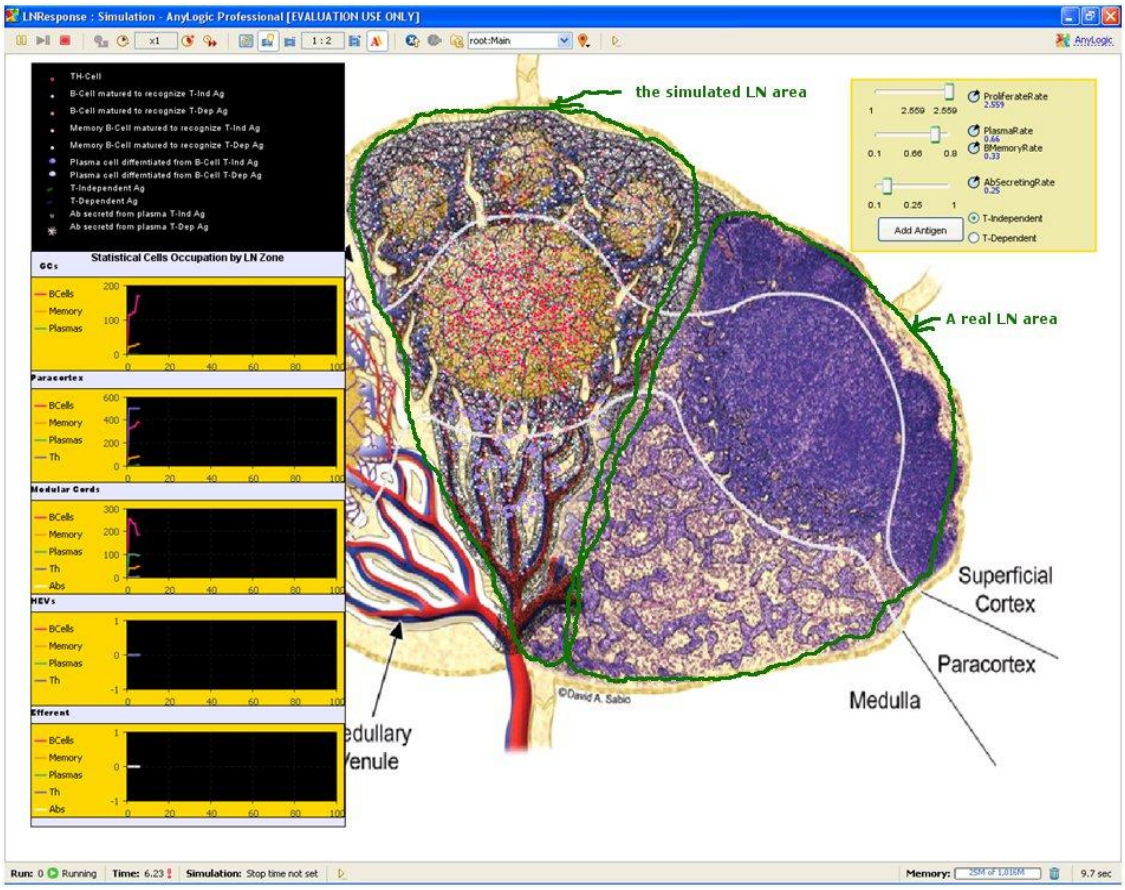


Fig. 4.17 : The initial running simulation of the root main class:
Each defined agent is situated in its LN zone, and it begins moving from its current zone to another with regards to its Statecharts location behavior

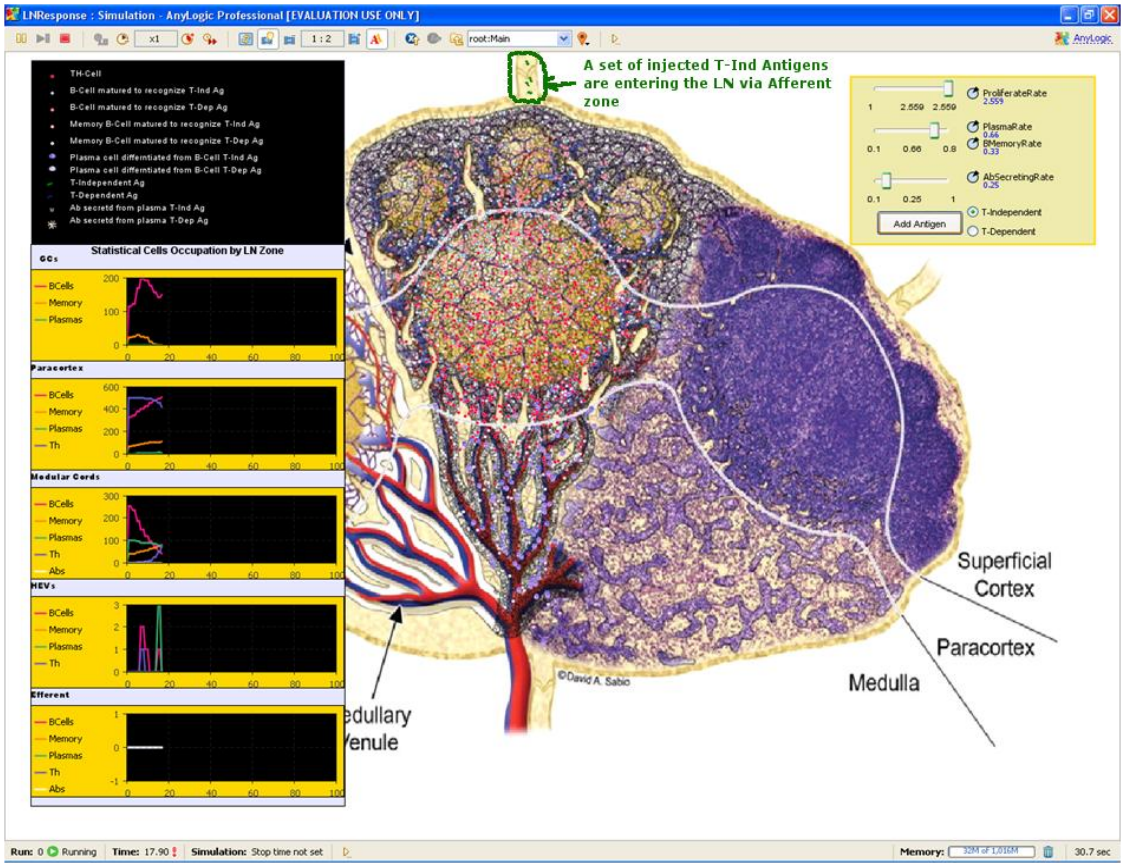


Fig. 4.18 : A number of T-Independent antigens (green color) are added to the simulation (Entering the LN via afferent area) and then are starting moving

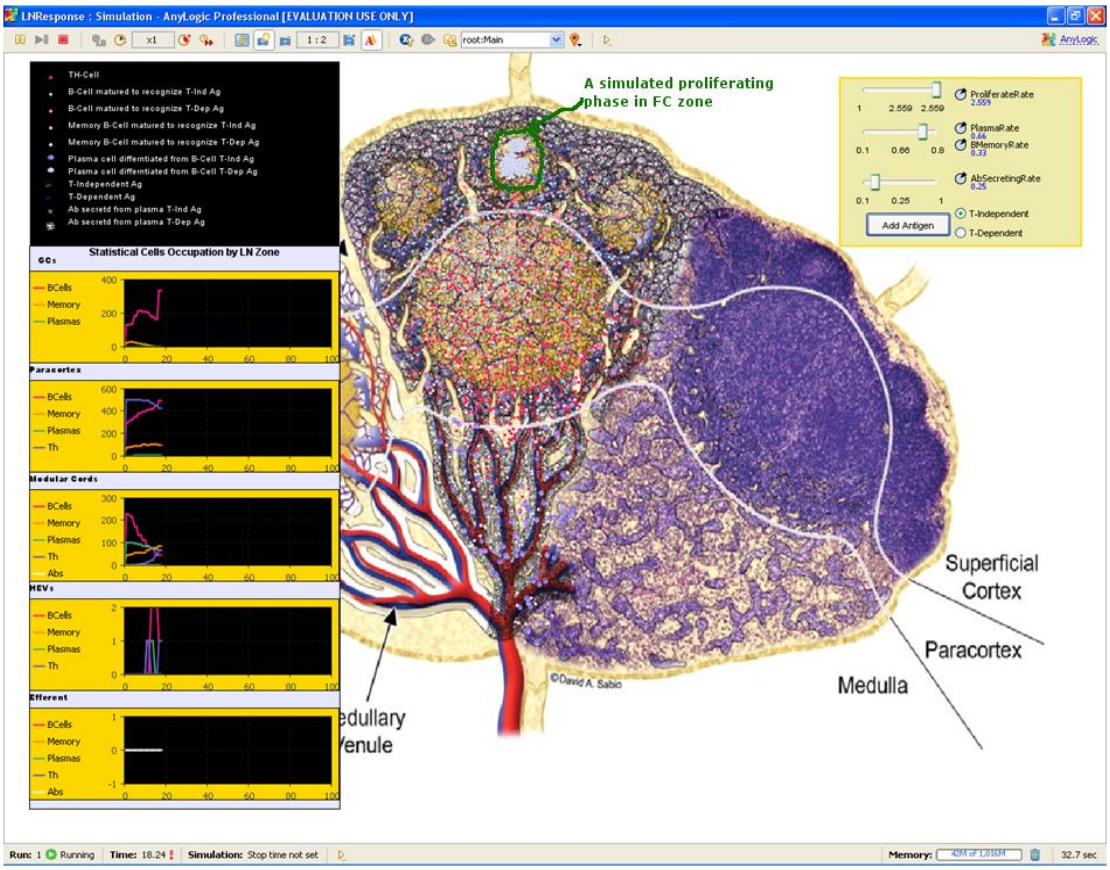


Fig .4.19: A simulation of a proliferation phase of an humoral immune response against a T-Inp Ag.

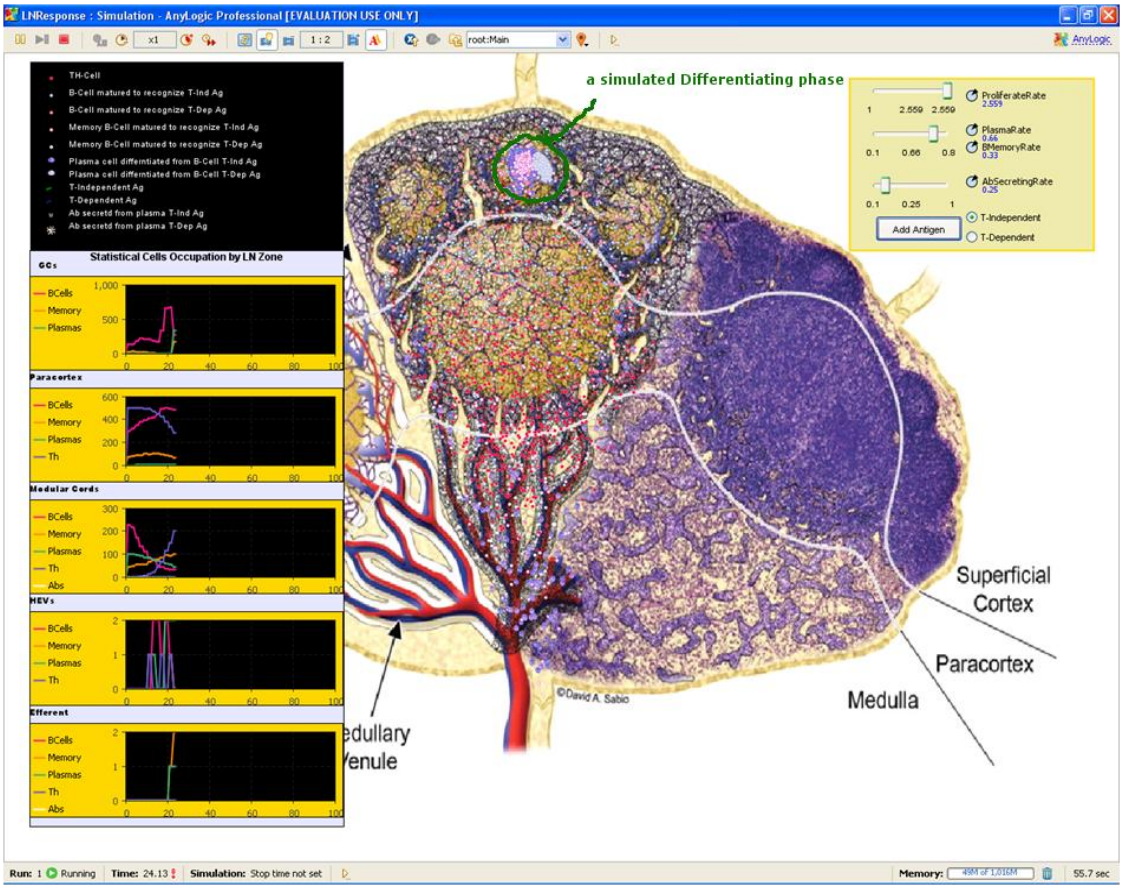


Fig .4.20: A simulation of a differentiation phase of an humoral immune response against a T-Ind Ag.

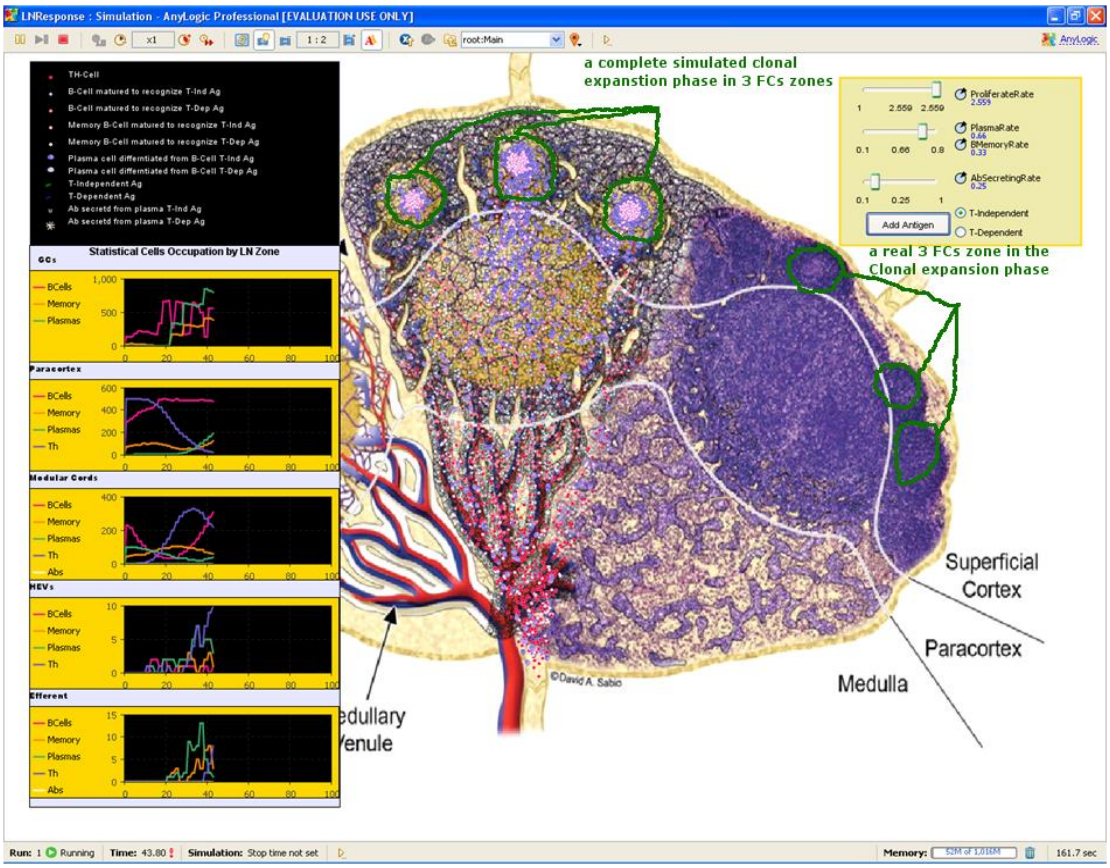


Fig .4.21: A simulation of a complete clonal expansion phase (processed in a 3 FCs zones) of an humoral immune response against a T-Indep Ag.

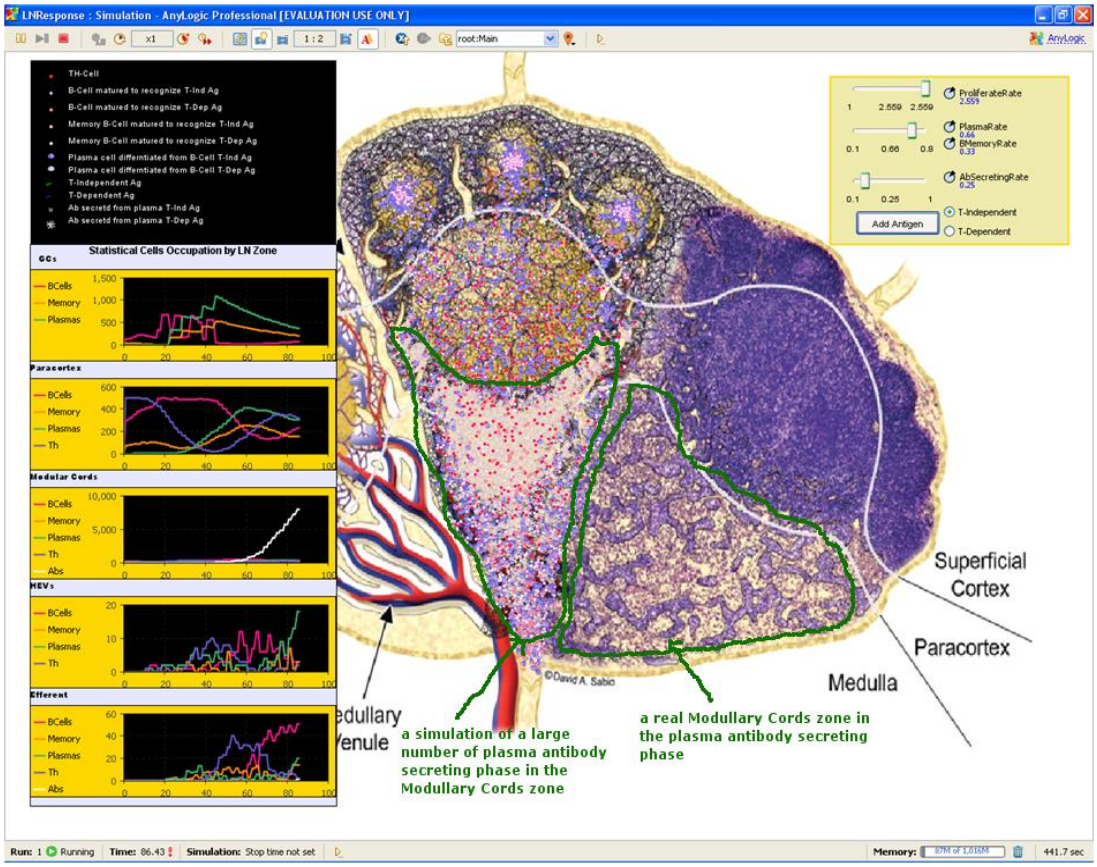


Fig .4.22: A simulation of a plasma secreting antibodies phase (processed in a Medullar Cords zone) of an humoral immune response against a T-Indep Ag.

The user has moreover a possibility to know the current state of any agent via the tools offer by the AnyLogic toolbar. For instance the snapshot mentioned in [Fig. 4.23] highlights the current state of the first generated B-Cell matured to recognize a T-Independent antigen (remember that a B-Cell is a Lymphocyte instance class); the figure shows that the concerned Lymphocyte is actually on parallel composite states: the *Cell_Cycle* one and the *Cell_Location* one. Inside the *Cell_Cycle* composite state the Lymphocyte is in its BCell composite state wherein the current active state is the Naïve state; whereas inside the *Cell_Location* composite state the figure shows that the specified Lymphocyte is actually in the *inLN* composite state and inside it the current active state is the *inFollicules* state.

The [Fig. 4.24] sums up different snapshots of the current active states for different simulated agents.

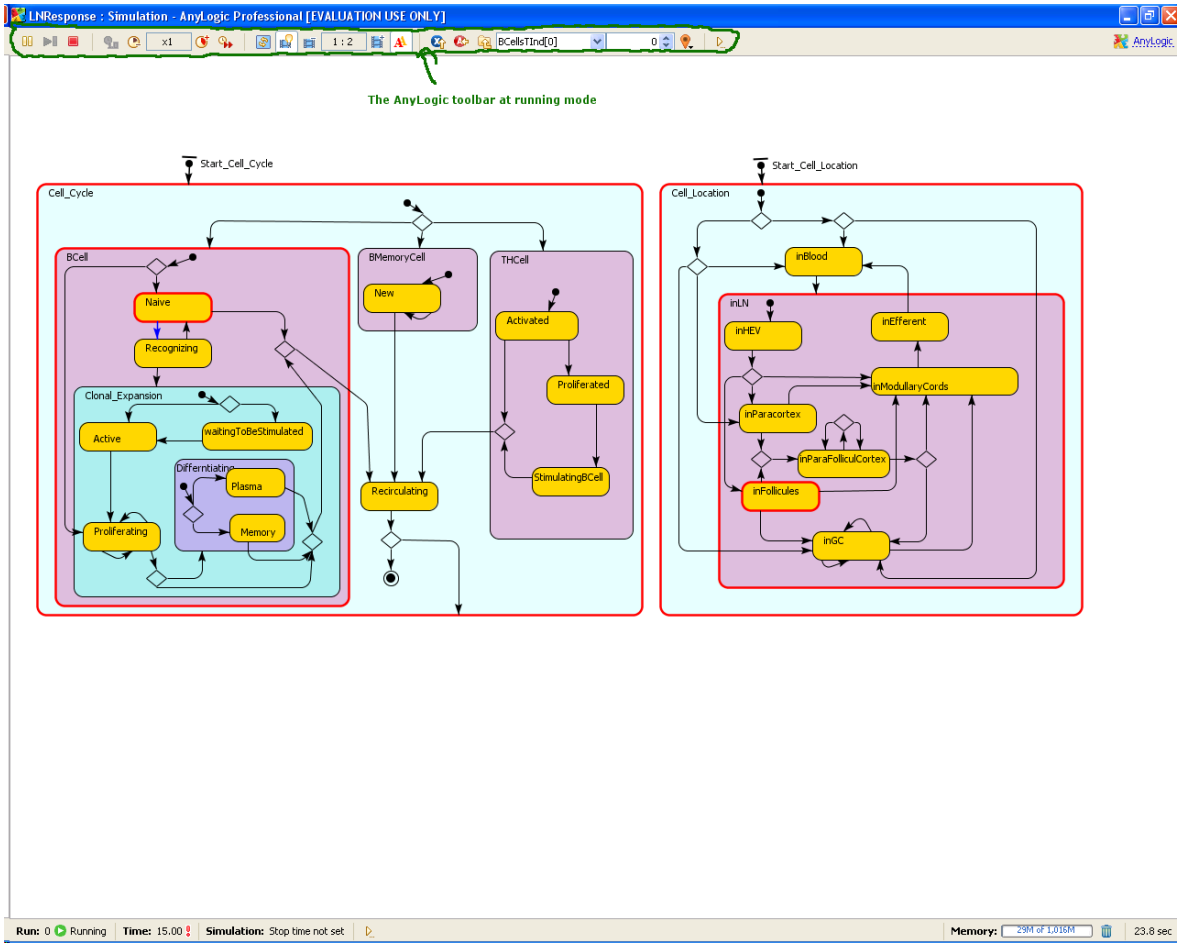
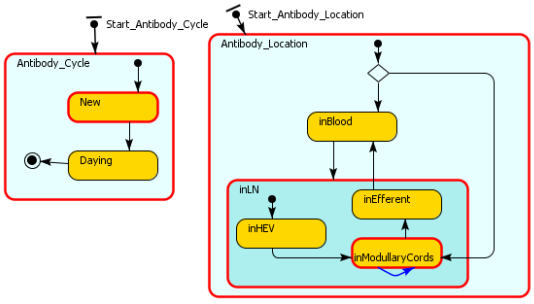
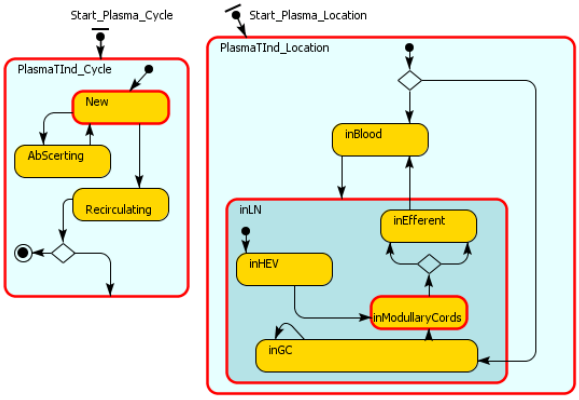


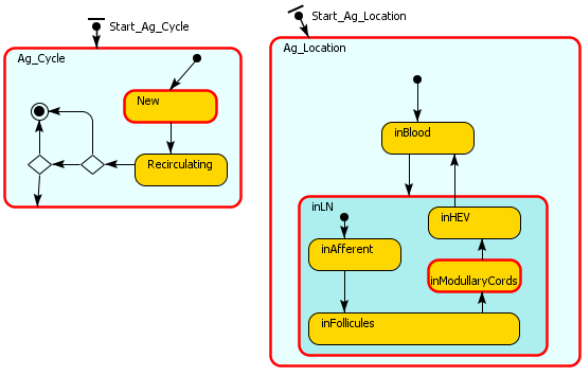
Fig. 4.23: The current highlighted active state for the first T-Independent B-Cell at run time.



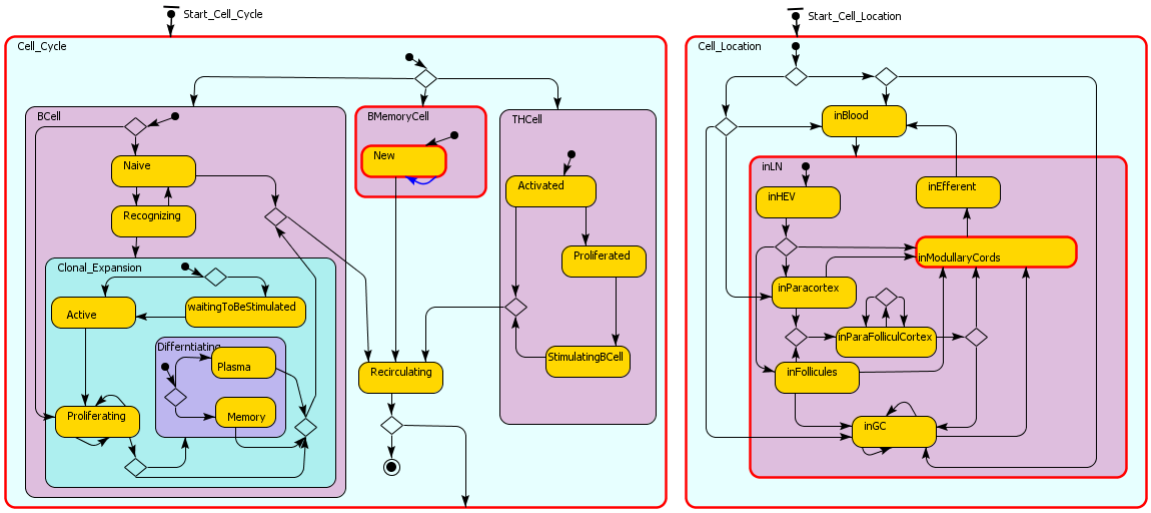
The current highlighted active state for a T-Independent antibody.



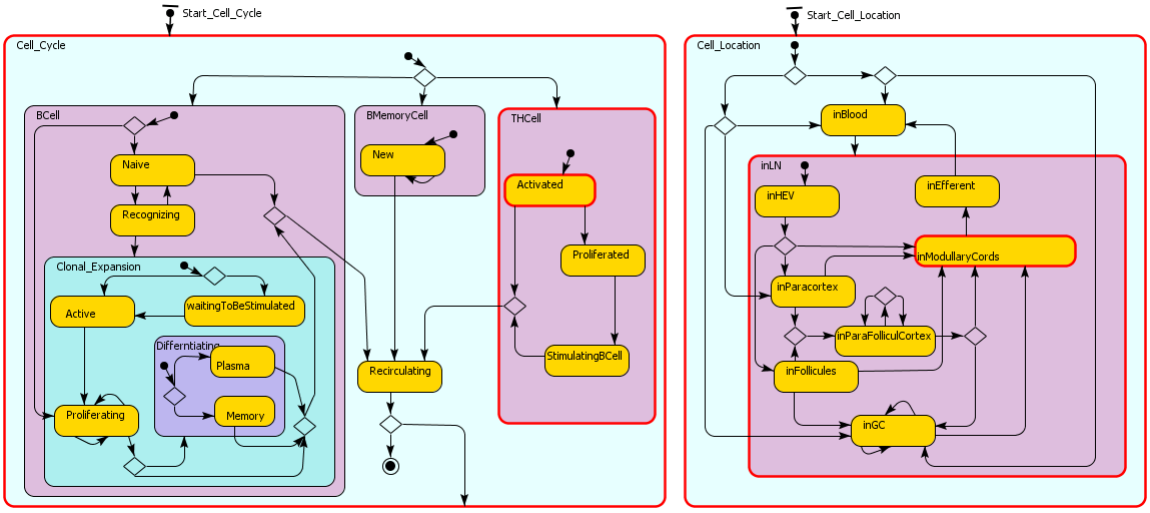
The current highlighted active state for a T-Independent Plasma Cell.



The current highlighted active state for a T-Independent antigen



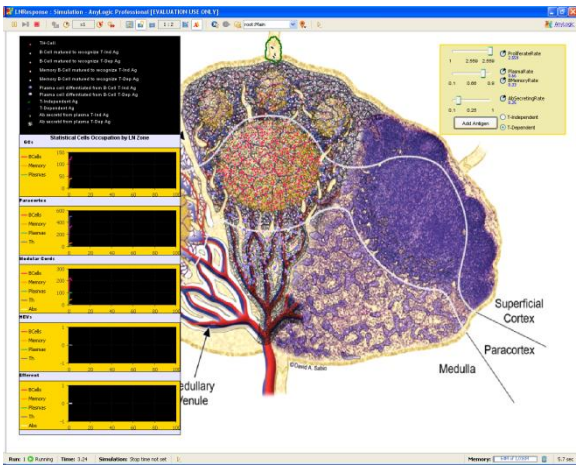
The current highlighted active state for a T-Independent B-Memory Cell.



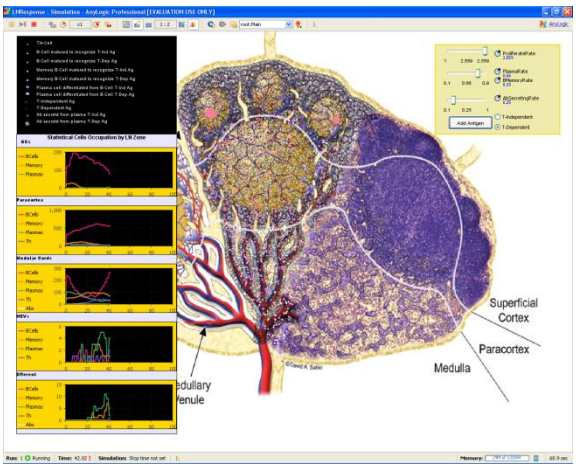
The current highlighted active state for a T-Helper Cell.

Fig .4.24: Highlighted current active states for different running simulated agents

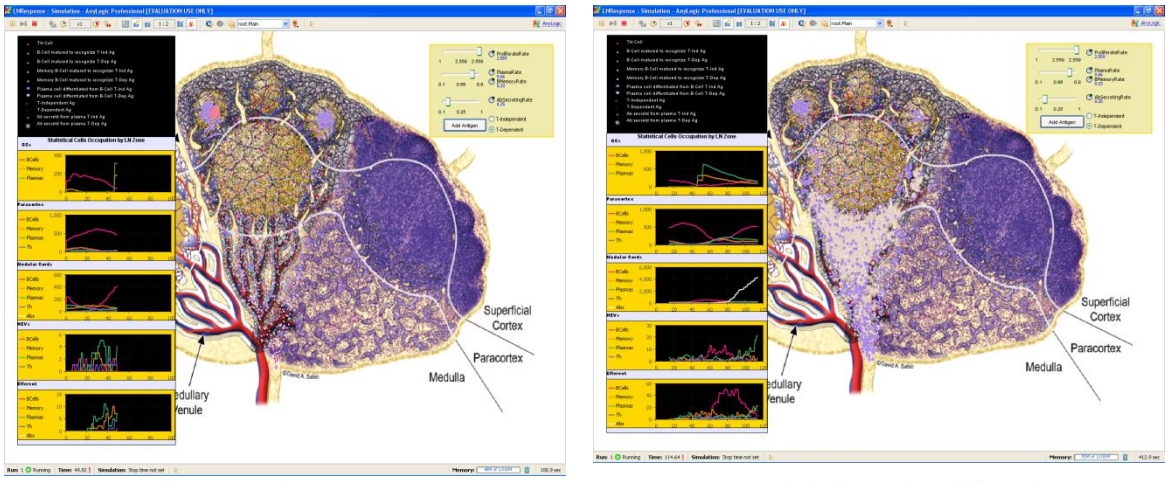
Alike the illustrations simulation that are viewed for mounting a LN first humorol immune response against T-Independent antigens, the user also can inject T-Dependent antigens into the simulation and start viewing the phenomenon emerged from the application of their behavior rules defined previously. This phenomenon is shown in the [Fig. 4.25] which gives various snapshots during the time running simulation.



A number of T-Dependent antigens (midnight Blue color) are added to the simulation (Entering the LN via afferent area) and then are starting moving



A number of activated TH-Cells that have migrated to the ParaFolliculeCortex zone, one among them have stimulated B-Cells and these last started their proliferation phase in FCs zones



A differentiating phase in FCs zones

Secreting antibody phase in the medullar cords zone

Fig .4.25: snapshots from the simulation run time of the first humoral immune response against a T-Dep Antigen

3.2. Discussion & Future works

The study described in this issue demonstrates how we can use an Agent-Based approach for which every agent behavior is controlled completely by the Statecharts formalism to simulate a part of the first LN humoral immune response against antigens. The use of the Statecharts technique; that has been used together with a front-end-animation tool in the work of [4] to serve as an enlightenment of the manner on how a LN computational simulation can be translated into realistic animation; proves that it's a suitable and powerful visual modeling technique to be applied in biologic systems as they are considered as reactive systems. The work presented heir is based on the AnyLogic simulation tool and is considered as the first attempt in our Laboratory even in the entire world to model and simulate such biologic system using the AnyLogic environment.

In our approach we have looked for profiting from the work done in [4]; in which the Statecharts technique has used as a state-of-the-art reactivity to model the development of a LN, we have remodeled completely the LN using the AnyLogic simulation tool with regards to the immunological experimentations. Although we haven't model all the experimental details that are issued from the immunology researches due to its immense complexity, however our simulation results those are compared at run time with a real LN image are closes to the reality.

The results issued during the execution of our AnyLogic simulation model shows that the process of mounting a LN humoral immune response against both T-Dependent and T-Independent antigens is much feet the biologic experiments; all the

phenomenon emerged from the application of the behavior rules defined in the Statecharts of each implicated cell are compared with the real images issued from the immunology experimentations. The obtained results demonstrate also that we were able to transform a part of this static experimental data into dynamical behavior including: cell migration from LN zone to another, cell proliferation, cell differentiation into memory or plasma cells and generation of antibody-producing plasma cells; a statistical analysis of the dynamic occupancy of the different LN zones is also given to the final users in order to illustrate statistics about the total numbers of cells that are actually residing in each LN zone.

As a deep analyze of our LN AnyLogic model which has much been simplified due to the limitation of the AnyLogic evaluate License key available to us in a hand and due to the immense complexity of some immune mechanisms in the other hand, and with regards to the LN model established in [4]; our model haven't detailed the cell interactions signals that can be founded during an immune response. For example: the antigen-BCells interaction signals, the antigen-ThCell interaction signals and BCell-ThCell ones aren't carried out in our model. The model also doesn't take into account the orthogonal states feature used in the work of [4] for the raison that the AnyLogic simulation tool doesn't support in its current release this powerful Statecharts features; nevertheless we have simulated this feature on profiting from the ability of the AnyLogic simulation tool to create multi-statecharts for the same agent. These multi-statecharts can be executed on parallel manner with the same execution fashion of orthogonal states.

A positive view point of our model is that it deals with two kinds of antigens: T-Dependent and T-Independent, it's also developed with one simulation tool that can combine different modeling approaches at once, and which integrates also a 2D and 3D render engine that can animate the simulation in two or three dimensions. Contrary to the model of [4] which dials only with the T-Dependent antigens and it's developed using two different tools: the IBM Rhapsody developer tool to model the cells behavior and the Adobe Flash tool as a render engine to animate the cells behavior.

After we have analyzed our model and although the simulation of such an immune response is very highly complex due to the high complexity of the

mechanism behind it; we believe that we have succeeded to build a simplified AnyLogic model that models the first LN humoral immune response with taking into account a part of the immunology experimentations. Our results obtained during the execution simulation of the modeled system shows that the model respects several immunology experimentations (B-Cell activation, proliferation, differentiation and antibody generation) even that some behaviors such as cell signal interactions, activation of Th-cells, etc.., aren't carried on for which a perspective future work can be initiated to extend the model.

We hope also that other AnyLogic immune researches works can be initiated to involve the other immune organs such Spleen, Bone Marrow and Thymus for the aim to model the entire immune response by gathering piece to piece the models of each immune organ. This also can initiate a collaboration work between our laboratory as a computing simulation research side with hospital immunology laboratories as biology researches side.

As also our model is developed under an Evaluation AnyLogic License which has limited our development, we suggest to the comity of our laboratory to buy a professional or university AnyLogic License in order to much profit from this amazing multi-modeling simulation tool and to join the various great companies, research institutes and research laboratories [131] that use this product in their researches. This would be offering great opportunities not only to the whole members of our laboratory but to the other laboratories of our university to initiate future researches work that can be based on the AnyLogic simulation tool.

Finally it would be a great pleasure for us that our AnyLogic model is the first attempt in our laboratory even in the entire world to initiate a simulation of a first LN humorol immune response against antigens using the AnyLogic simulation tool; as consequence we hope that we have enriched the existing immune models that have taken place and we have opened a research windows for the future extension of our work and for other researches area.

4. Summary

We have presented in this chapter an AnyLogic Agent-Based Model for the first LN humoral immune response against antigens. The whole modeled system is viewed as a multi-agent system wherein the behavior of the constitute agents are modeled via the Statecharts formalism. The model is developed completely using the AnyLogic simulation environment, which gives us the opportunity to work with such combined ABM and Statecharts formalism; it focuses on modeling a part of the first humoral immune response initiated in the LN when an encountered antigen is recognized; it has much been simplified due to the immense complexity behind the mechanisms of such immunology systems. Even though some behaviors such as cell signal interactions, activation of Th-cells, etc.., aren't carried on in our work, however the results obtained from the simulation of our AnyLogic model shows that it respects several immunology experimentations including: B-Cell activation, proliferation, differentiation and antibody generation.

We think that we have succeeded to enrich the existing immune system models by presenting our AnyLogic model which is considered as the first attempt to model such biologic systems in our Laboratory. Our work can be extended for the aim to model the other no-modeled parts of the entire LN humoral immune response in a hand; and in the other hand to model the whole immune response as a big future perspective by initiating research works that intend to model the other involved immune organs such: Spleen, Thymus, Bone Morrow…, and gathering together piece to piece these models to form the complete immune response.

Conclusion

The human immune system is considered as one of the most complex, adaptive, highly distributive learning systems that give a challenge for both immunologist and engineers to work together for the aim to attempt to bridge the divide between them. The simulation of such immune system is extremely complex due to the high mechanisms and interactions existed behind these systems; however great efforts are taking place to better understand these mechanisms and interactions. The researches that have been issued during the last years varies from mathematical simulation models to Cellular Automata (CA) ones arriving to Multi-Agent based ones (ABM) and finally the Reactive Animation (RA) simulation models; each model is based on specific rules that it characterizes from the others.

The reactive animation models, which aim to couple between state-of-the-art reactivity and state-of-the-art animation, have been one of the recent applied approaches for simulating biologic systems. They are based on two combined techniques: the Statecharts formalism to model the behavior of systems and the front-end animation tool to visualize the animation simulation and enable natural-looking. The most known immunology works based on this technique is the works done by the David Harel's group who has attempted to model the maturation of T-cells in the thymus [77] and the development of the lymph node (LN) [4]. This last studies the dynamic development of the LN with a focus on the behavior of a subset of immune cells that enter a single, two-dimensional LN with immunogenic antigens; the model uses a RA technique in which the behavior of the whole system including the behavior of its implicated cells are modeled via the Statecharts formalism; the generated executable code issued from this visual language is then used to visualize the animation front-end part.

At the beginning of our research work, we have intended to extend this RA model trying to use the same technique to model the other no-modeled part, but we have faced a great obstacle due to the no availability in our market of the tools that have been used for modeling this techniques (the state-of-the-art reactivity of the work was done by the IBM Rhapsody Developer tool which is a very expensive commercial software). Even the deep researches done to find another free tool that can allow us dealing with this technique have failed.

So in our work we have obliged to find another technique which we aim to be the first one used in our laboratory, we have tried to remodel a part of the process launched in a LN using another approach that challenge to take advantage from the different existing approaches used to model such an immune system. We have used a Multi-Agent based model in which the behavior of the constitute agents are modeled using the state-of-the-art reactivity part of the RA technique.

Among the available tools that have given us this opportunity to dial with such combined ABM and Statecharts formalism is the AnyLogic simulation tool. This one is an innovative professional multi-approach simulation modeling tool based on advanced technologies such as UML, Java, hybrid systems theory, and best numerical methods.

Our AnyLogic model focuses on modeling the first humorol immune response initiated in the LN as an encountered antigen is recognized. The model dial with two kinds of antigens: the T-Independent antigens that can instantly mount an humorol immune response without the help of T-Helper Cells and the T-Dependent ones that must implicate T-Helper cells for mounting an humorol immune response. Due to the high complexity of the mechanisms behind this immunology system, we have simplified our AnyLogic model trying to not give all the details known from the immunology experimentation.

Unlike the LN model presented in the work [4] in which more immunology mechanism details are taken into account such as the details of interaction signals between different immune cells or the recognizing phase interaction signals (B-Cell vs antigen and Th-Cell vs antigen); our AnyLogic model does not carry on these mechanisms due to the high complexity interaction behind them in a hand, and in the

other hand due to the limitation of the AnyLogic evaluation License where the number of the allowed modeled agent are limited to five (5).

To achieve our wanted goals we have firstly given an overview of the natural immune response and the mechanism behind it; followed by presenting and analyzing the different computational models used to simulate such an immune response; then explaining the technique to be used in our model in which an overview of both Agent based modeling and Statecharts formalism are studied; next finishing by presenting our AnyLogic LN model starting with presenting the AnyLogic simulation tool, then detailing the Statecharts behavior for each implicated agent and finally discussing the obtained results, giving suggestions and opening overview of future works.

We have proven that our AnyLogic simulation issued results of the first LN humorol immune response respects several immunology experimentations (B-Cell activation, proliferation, differentiation and antibody generation) even that some behaviors such as cell signal interactions, activation of Th-cells, etc.., aren't carried on for which a perspective future work can be initiated to extend the model.

We have also proposed a big perspective for future works; which can be founded collaboration between our laboratory as a computing simulation research side and hospital immunology laboratories as a biology research side; for the aim to model the whole immune response by coupling the initiated further AnyLogic researches works that involve modeling the other implicated immune organs such as Spleen, Bone Morrow, and Thymus. As a part of proposing this big research project, we have also suggested to the committee of our laboratory to buy a permanent AnyLogic License key (Professional or University) in order to much profit from this amazing multi-modeling simulation tool and to join the various great companies, research institutes and research laboratories that use this product in their researches. As consequence we view that great opportunities would be offered not only to the whole members of our laboratory but also to the other laboratories of our university to initiate future researches work that can be based on the AnyLogic simulation tool.

Finally we hope that our first attempt in our laboratory to initiate a simulation of a first LN humorol immune response against antigens using the AnyLogic simulation tool will enrich the existing immune models that have been taking place.

Bibliography references

[1] Jamie T. and Uwe A. *"*Towards a conceptual framework for innate immunity*".In 4thInt. Conf. Artificial Immune Systems (ICARIS)* [Online]. Banff, Canada, 2005, pp 112-125. Available: http://ima.ac.uk/papers/twycross2005.pdf.

[2] Johannes T. and Bjorn H. *"*Improved Simulation Algorithms for Agent Based Models of the Immune Response". *In 2nd Int. Symp. Agent Based Modeling and Simulation EMCSR* [Online]. Vienna, 2008. Available: http://www.tcs.uni-luebeck.de/downloads/papers/2008/EM08_L_Textor.pdf.

[3] Raúl M., Rosa S. and Fernando G. *"*On Modelling an Immune System (Sobre un Modelo Computacional del Sistema Inmune)*". Computación y Sistemas* [Online]. 7(4), 2004, pp 249-259. Available: http://www.cic.ipn.mx/portalCIC/s11/vol07-04/art3.pdf.

[4] Naamah S., Irun R. C., and David H. *"*The Lymph Node B Cell Immune Response: Dynamic Analysis In-Silico*".Proc. IEEE* [Online]. 96(8), 2008, pp 1421-1443. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.140.2684&rep=rep1&type=pdf.

[5] Pascal BALLET. "Interets mutuels des systems multi-agent et de l'immunologie applications à l'immunologie, l'hématologie et au traitement d'images". PhD dissertation [Online]. University of Western Brittany, France, 2000.Available:http://doelan-gw.univ-brest.fr:8080/~ballet/these/these_ballet.html.

[6] John Daigle.*"*Human Immune System Simulation: A Survey of Curent Approches*". ACM J.* [Online]. V (N, M), 2006. Available: http://student.johnpdaigle.com/PDFs/immune.pdf.

[7] "Immune System" [Online]. Available:*http://kidshealth.org/parent/general/body_basics/immune.html*[Accessed:Sept, 2011].

[8] U.S. Dep. Health and Human Services. "Understanding the Immune System How It Work*".NIAID Science Education* [Online]: NIH Publ. No. 03-5423, Sept. 2003. Available: http://www.niaid.nih.gov/topics/immunesystem/documents/theimmunesystem.pdf.

[9] Dipankar D. & Luis F. Niño."Immunological Computation Theory and Applications".*CRC Press*, Taylor and Francis Group, 2009.

[10] Abbas, A. K. and A. H. Lichtman.*"*Basic Immunology: Functions and Disorders of the Immune system*".W.B. Saunders Company* [Online], 2004. Available: http://ebookee.org/Basic-Immunology-Functions-and-Disorders-of-the-Immune-System-Repost-_675763.html.

[11] SridharRao. "B cell activation and Humoral Immunity". Dept. Microbiology, JJMMC, Davanger [Online]. Unpublished. Available: www.microrao.com/micronotes/pg/humoral_immunity.pdf.

[12] Kitchen G., Horton-Szar D. "1- Principles of immunology, 2- The functioning immune system" in "Immunology and hematology". *Crash course (2nd edt),* 2007.

[13] Xiang-hua Li & al. "Modeling Immune System: Principles, Models, Modeling, Analysis and Perspective".*J. Bionic Engineering.* 6(1), 2009, pp 77-85,doi: 10.1016/S1672-6529(08)60101-8.

[14] L. N. de Castro, J. I. Timmis. *"*Artificial immune systems as a novel soft computing paradigm*".Soft Computing.* 7(8), 2003, pp 526–544, doi: 10.1007/s00500-002-0237-z.

[15] J. Timmisa & al. *"*Theoretical advances in artificial immune systems*".Theoretical Computer Science*. 403(1), 2008, pp 11-32.doi: 10.1016/j.tcs.2008.02.011.

[16] Silvia Daun, Gilles Clermont. "In silico modeling in infectious disease*". Drug Discovery Today: Disease Models.* 4(3), 2007, pp 117-122.doi: 10.1016/j.ddmod.2007.09.001.

[17] Geoffrey W. Hoffmann.*"*Immune Network Theory*".University of British Columbia* [Online]. 2008. Unpublishedavailable: http://www.phas.ubc.ca/~hoffmann/ni.html [Accessed: Sept, 2011].

[18] Wouter Hanegraaff. *"*Simulating the Immune System". August 10, 2001.Unpublished available: http://www.lymenet.de/literatur/hanegraaff.pdf.

[19] Nuno Fachada& al. *"Agent Based Modeling and Simulation of the Immune System: a review*". Evolutionary System and Biomedical Engineering Lab Systems and Robotics Institute Portugal [Online]. Unpublished available: https://dspace.ist.utl.pt/bitstream/2295/128229/2/article_cut.pdf.

[20] Stephanie Forrest & Catherine Beauchemin.*"*Computer immunology*". Immunological Reviews* [Online]. 216(1), 2007, pp 176–197. Available: http://www-users.cs.york.ac.uk/jtimmis/utm/Papers/forrest-2007.pdf.

[21] Vorgelegt Von Martin.*"*The Immune system as complex System: Description and simulation the interactions of its constituents*"* .PhD dissertation [Online].Dept Physic, University of Humburg, Germany, 2001. Available: http://cdsweb.cern.ch/record/494617/files/cer-002249151.pdf.

[22] Funk G.A and al.*"*Mathematical model of a virus-neutralizing immunoglobulin response*". J. Theoretical Biology*. 195(1), 1998, pp 41–52.

[23] D. Kirschner.*"*Dynamics of co-infection with m. tuberculosis and hiv-1*".Theoretical Population Biology.* 55(1), 1999, pp 94–109.

[24] F.E. McKenzie and W.H. Bossert. *"*The dynamics of plasmodium falciparum bloodstage infection*". J. of Theoretical Biology.* 188(1), 1997, pp 127–140.

[25] D.S. Stein and G.L. Drusano. "Modeling of the change in cd4 lymphocyte counts in patients before and after administration of the human immunodeficiency virus protease inhibitor indinavir*". Antimicrobial Agents and Chemotherapy.* 41(2), 1997, pp 449–453.

[26] P. Klein, J. Sterzl, and J. Dolezal.*"*A mathematical model of b lymphocyte differentiation: control by antigen*".J. of Mathematical Biology.* 13(1), 1981, pp 67–86.

[27] Kalita, J.K. and al."Computational Modeling and Simulation of the Immune System with Modelingand Simulation of Bacillus anthrax*".Int. J. Bioinformatics Research and Applications* [Online]. 1(1, 2, 3), 2005, pp xx–yy. doi : 10.1.1.61.6566.

[28] Francesco Pappalardo and al: *"*Computational simulations of the immune system for personalized medicine: state of the art and challenge*".Current pharmacogenomics and personalized medicine* [Online]. 6(4), 2008, pp 260-271. Available: http://works.bepress.com/ping_zhang/10.

[29] Joana Moreira And Andreas Deutschy.*"*Cellular Automaton Models Of Tumor Development: A Critical Review*".Advances In Complex Systems.* 5(2 & 3), 2002, pp 247-267.

[30] Wolfram, S. "Theory and Application of Cellular Automata". *World Scientific*. 1, 1986, pp 232–246.

[31] Armin R. Mikler, Sangeeta V. andKaja Abbas. "Modeling Infectious Diseases using Global Stochastic Cellular Automata". *J of Biological Systems.* 13(4), 2005, pp 421-439.doi: 10.1142/S0218339005001604.

[32] S. Wolfram. "Statistical Mechanics of Cellular Automata". *Reviews of Modern Physics.*55, 1983, pp 601-644.

[33] Xuan Xiao, Shi-Huang Shao and Kuo-Chen Chou. "A probability cellular automaton model for hepatitis B viral infections". *Biochemical and Biophysical Research Communications* [Online]. 342, 2006, pp 605–610. Available in http://gordonlifescience.org/members/kcchou/paper/BBRC_HBV_CA.pdf.

[34] Kaufman, M.; Urbain, J. and Thomas, R. "Towards a Logical Analysis of the Immune. Response". *J Theor Biol.* 114(4), 1985, pp 527-561.

[35] Weisbuch, G. and Atlan, H. "Control of the Immune Response". *J. of Physics, Mathematical and General.* 21, 1989, pp 189-192.

[36] Cohen, I.R. and Atlan, H."Network Regulation of Autoimmunity: An Automation Model".*J Autoimmun.* 2(5), 1989, pp 613–625.

[37] Chowdhury and al. "A Unified Discrete Model of Immune Response". *J Theor Biol.*145, 1990, pp 207-215.

[38] Sieburg, H.B. "A Logical Dynamic Systems Approach to the Regulation of Antigen-Driven Lymphocyte Stimulation". *In Theoretical Immunology: Part I. A. S. Perelson (eds.),* 1992, pp 273-293.

[39] Neumann, A.U. "Control of the Immune Response by a Threshold Automata Model on A Lattice". *Physica A: Statistical Mechanics and Its Applications.*162, 1989, pp 1-19.

[40] Stauffer, D. and Sahimi, M. "High-dimensional and Very Large Cellular Automata for Immunological Shape Space".*J. of Modern Physics C: Computational Physics and Physical Computation.*4, 1993, pp 401-408.

[41] Stauffer, D. and Sahimi, M. "High-dimensional Simulation of Simple Immunological Models". *J Theor Biol.* 166, 1994, pp 289-297.

[42] Smith, D.J. and al. "Deriving Shape Space Parameters from Immunological Data". *J Theor Biol.* 189, 1997, pp 141-150.

[43] De Boer, R.J., Segel, L.A. and Perelson, A.S. "Pattern-formation in One and Twodimensional Shape-space Models of the Immune System". *J Theor Biol.*155, 1992, pp 295-333.

[44] Seiden PE, Celada F. "A model for simulating cognate recognition and response in the immune system". *J Theor Biol.*158, 1992, pp 329–357.

[45] F. Celada and P. E. Seiden. "A Computer Model of Cellular Interactions in the Immune System". *Immunol. Today.*13, 1992, pp 56-62.

[46] Kesmir C, Boer RJD. "A spatial model of germinal center reactions: cellular adhesion based sorting of b cells results in efficient affinity maturation". *J Theor Biol.* 222, 2003, pp 9–22.

[47] F. Fachada, N.*"SimulIm: an application for the modelling and simulation of complex systems, using the immune system as an example"*, Graduation project report [Online], Instituto Superior Técnico, Universidade Técnica de Lisboa; 2005. Available: https://dspace.ist.utl.pt/bitstream/2295/95112/2/simulim.pdf.

[48] Bezzi M, Celada F, Ruffo S, Seiden PE. "The transition between immune and disease states in a cellular automaton model of clonal immune response". *Physica A.*245, 1997, pp 145–163.

[49] Catherine Beauchemin. "Spatiotemporal Modelling of Viral Infection Dynamics". PhD dissertation [Online], Faculty of Graduate Studies and Research, Dep. Physics, University of Alberta Canada, 2006.Available: http://phymbie.physics.ryerson.ca/~cbeau/docs/these.pdf.

[50] M. Bernachi, F. Castiglione & S. Succi. "A parallel algorithm for the simulation of immune response". *CyteSeerx,* 1997, pp 198-208.doi: 10.1.1.47.7749.

[51] Bernaschi, M., Castiglione, F. "Design and implementation of an immune system simulator".*Computers in Biology and Medicine.* 31(5), 2001, pp 303–331.

[52] *http://www.iac.rm.cnr.it/~filippo/* [Accessed: Sept., 2011].

[53] Motta, S., Castiglione, F., Lollini, P., Pappalardo, F. "Modelling vaccination schedules for a cancer immunoprevention vaccine". *Immunome Research.* 1(1), 2005, pp 5.

[54] Pappalardo, F., Lollini, P., Castiglione, F., Motta, S. "Modeling and simulation of cancer immunoprevention vaccine". *Bioinformatics.* 21(12), 2005, pp 2891–2897.

[55] Castiglione, F. and al. "Simulating Epstein-Barr Virus Infection with C-ImmSim". *Bioinformatics.* 23(11), 2007, pp 1371–1377.

[56] Pier-Luigi L., Santo M. and Francesco P. "Discovery of cancer vaccination protocols with a genetic algorithm driving an agent based simulator". *BMC Bioinformatics* [Online]. 7(1), 2006.Available: http://www.biomedcentral.com/content/pdf/1471-2105-7-352.pdf.

[57] *http://www.immunogrid.org/*[Accessed: Sept, 2011].

[58] Xiaochen Li & al "Agent-Based Social Simulation and Modeling in Social computing". *C.C. Yang et al. (Eds.): ISI 2008 Workshops*; LNCS 5075; pp 401–412, 2008.

[59] Eric Bonabeau. "Agent Based Modelling: Methods and techniques for simulating human systems".*PNAS.* 99(3), 2002, pp 7280–7287.

[60] TRANSIMS.*http://transims.tsasa.lanl.gov/* [Accessed: Sept, 2011].

[61] Ankit Singhal. "An Effective Communication Framework for Inter-Agent Communication In a Complex Adaptive System Simulation With Application To Biology". Master of Science Thesis. Faculty of the

Virginia Polytechnic Institute and State University, Blacksburg, Virginia; 14 Nov 2006.

[62] Guo Z, Tay J C. "A comparative study on modeling strategiesfor immune system dynamics under HIV-1 Infection.*Lecture Notes in Computer Science*, 2005, 3627, 220–233.

[63] Dimitri Perrin, Heather J. Ruskin, and Martin Crane. "An Agent Based Approach to Immune Modelling: Priming Individual Response". *Proceed. World Academy of Science; Engineering And Technology*, 17, 2006.

[64] Christian Jacob, Julius Litorco, and Leo Lee. "Immunity throw swarms based simulations of the Human Immune System". *ICARIS*, 2004, p.400-412.

[65] Tay J C, Jhavar A. "CAFISS: A complex adaptive framework for immune system simulation". *In Proced. ACM Symp. Applied Computing (SAC'05).* 2005, pp 158–164.

[66] Le Zhang &al. "Multiscale agent-based cancer modeling". *J Math. Biol.*58, 2009, pp 545–559.

[67] Luıs C´esar da Costa & al: *"Study of immunological response using a multi-agent system".* Virtual Reality and Scientific Visualization Laboratory & Bioinformatics Laboratory.Petr´opolis, Brazil. Unpublished Available: http://www.lncc.br.

[68] James Mata, Melvin Cohn."Manual for the use of a Cellular Automata Based Synthetic Immune System*".The Salk Institute, Conceptual Immunology Group*, San Diego, 2007. Available:http://www.cig.salk.edu/papers/SIS_manual_wp_M.pdf.

[69] M. Meier-Schellersheim, G. Mack."SIMMUNE, a tool for simulating and analyzing Immune System behavior*". Deutsches Elektronen-Synchrotron (DESY)*, Hamburg, Germany. 1999, Rep. N°: DESY-99-034. Available: http://arxiv.org/abs/cs/9903017.

[70] *http://www.simmune.org*[Accessed: Sept, 2011].

[71] *http://www3.niaid.nih.gov*[Accessed: Sept, 2011].

[72] Christina Elizabeth Warrender."Modeling intercellular interactions in the peripheral immune system". PhD dissertation [Online]. University of New Mexico, 2004. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.110.8563&rep=rep1&type=pdf.

[73] Christina W., Stephanie F., Frederick K. "Modeling Intercellular Interactions in Early Mycobacterium Infection". *Bulletin of Mathematical Biology* [Online]. 68, 2006, pp 2233–2261. Available: http://www.springerlink.com/index/F698424230324675.pdf.

[74] Warrender C., Forrest S., Segel L."Homeostasis of Peripheral Immune Effectors*". Bulletin of Mathematical Biology* [Online]. 66(6), 2004, pp 1493-1514. Available: http://www.springerlink.com/index/L151716920641821.pdf.

[75] Sol Efroni, David Harel, and Irun R. Cohen. "Toward Rigorous Comprehension of Biological Complexity: Modeling, Execution, and Visualization of Thymic T-Cell Maturation". *Genome Research.*13, 2003, pp 2485–2497. Available: www.genome.org.

[76] Irun R. Cohen and David Harel. "Explaining a complex living system: dynamics, multi-scaling and emergence".*J. R. Soc. Interface.* 4(13), 2007, pp 175-182. doi:10.1098/rsif.2006.0173.

[77] David Harel and Michal Politi. "*Modeling reactive systems with statecharts: The statements approach David Harel, Michal Politi*". USA: MCGrow-Hill, 1998.

[78] Hila Amir-Krolla, AvitalSadota, Irun R. Cohenb, David Harel. "GemCell A generic platform for modeling multi-cellular biological systems". *Theoretical Computer Science.* 391(3), 2008, pp 276–290.doi:10.1016/j.tcs.2007.11.014.

[79] David Harel& al. "Concurrency in Biological Modeling: Behavior, Execution and Visualization".*Electronic Notes in Theoretical Computer Science.* 194(3), 2008, pp 119–131.doi: 10.1016/j.entcs.2007.12.009.

[80] David Harel, Sol Efroni, and Irun R. Cohen. "Reactive Animation: Realistic Modeling of Complex Dynamic Systems".*IEEE Computer Society.* 38(1), 2005, pp 38-47.doi: 10.1109/MC.2005.31.

[81] YakiSetty, Irun R. Cohen and David Harel. "Four-Dimensional Reactive Animation Model for the Early Stages of Pancreatic Organogenesis". *National Academy of Sciences*, 105(51), 2008, pp 20374-20379. doi: 10.1073/pnas.0808725105.

[82] Hillel Kugler, AnttiLarjo and David Harel. "Biocharts: a visual formalism for complex biological systems". *J. R. Soc. Interface*. 7(48), 2010, pp 1015-1024. doi:10.1098/rsif.2009.0457.

[83] José M Vidal: *"Fundamentals of Multiagent Systems with NetLogo Examples"*.Unpublished 2010,

available:http://www.damas.ift.ulaval.ca/~coursMAS/ComplementsH10/mas-Vidal.pdf.

[84] Adelinde M. Uhrmacher Danny Weyns. *"Multi-Agent Systems Simulation and Applications"*. New York, USA: CRC Press, Taylor and Francis Group, 2009.

[85] Michael Wooldridge and Nicholas R. Jennings. "Intelligent Agents: theory and practice".*The knowledge Engineering Review.* 10(2), 1995, pp 115-152.

[86] Ferber, J.*"Les systèmes Multi-Agents: Vers une intelligence collective".* Paris: InterEditions, 1995.Available: http://www.lirmm.fr/~ferber/publications/LesSMA_Ferber.pdf.

[87] Russell, S. and Norvig, P.*"Artificial Intelligence: A Modern Approach*". Prentice Hall (3rd edt), 2009.

[88] Michael Wooldridge. "*An introduction to MultiAgent Systems*". England: John wiley & R Sons Ltd, 2002.

[89] Fabio Bellifemine, Giovanni Caire, Dominic Greenwood. "*Developing Multi-Agent Systems with JADE*".England: John wiley & R Sons Lt, 2007.

[90] Charles M. Macal, Michael J. North."Tutorial On Agent-Based Modeling And Simulation Part 2: How To Model With Agents". *In Proceed. of the 2006 Winter Simulation Conference* [Online], 2006. Available: http://www.informs-sim.org/wsc06papers/008.pdf.

[91] NicolettaFornara.*"Interaction and Communication among Autonomous Agents in Multiagent Systems"*. PhD dissertation [Online]. School of Communication Sciences, University of Lugano, Switzerland, June 2003. Availabel: http://doc.rero.ch/lm.php?url=1000,40,6,20050519123617-PD/1_2003COM002.pdf.

[92] Brooks, R.A. "A robust layered control system for a mobile robot".*IEEE J. of Robotics and Automation.* RA-2(1), 1986, pp 14-23.

[93] James S. Albus, Anthony J. Barbera. "RCS: A Cognitive Architecture For Intelligent Multi-Agent Systems*".Elsevier Annual Reviews in Control*. 29(1), 2005, pp 87-99. doi: 10.1016/j.arcontrol.2004.12.003.

[94] Jonathan Bona.*"MGLAIR: A Multimodal Cognitive Agent Architecture"*, PhDdissertation, Department of Computer Science and Engineering, University of New York, June 30, 2010.

[95] Anand S. Rao, Michael P. George. "BDI Agents: From Theory to Practice".*In Proceed.1st Int. Conf. Multi-Agent Systems (ICMAS-95)* [Online], San Francisco, USA; 1995. Available : www.upv.es/sma/teoria/teoria.../bdi%20agents%20from%20theory-Rao.pdf.

[96] Jorg P. Muller, Markus Pischel. "The Agent Architecture InteRRaP: Concepts and Application". *Citeseer* [Online], RR-93-26, pp 99, 1993. Available:http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.45.5965.

[97] Brian A. Coan and al. "The Touring Machine System (Ver. 3): An Open Distributed Platform for Information Networking Applications". *IEEE Comput. Soc.* 97(4), 1993, pp 74-81. doi:10.1109/ISWC.1997.629922.

[98] Foundations of Intelligent Physical Agents (FIPA)."http://www.fipa.org" [Accessed: Sept, 2011].

[99] F. Michel, G. Beurier, and J. Ferber."The TurtleKit simulation platform: Application to complex systems". *In Proceed.1stInt. Conf. Signal & Image Technology and Internet-Based Systems SITIS' 05,* November 2005, pp 122–128.

[100] David Harel. "Statecharts in the Making: A Personal Account".*Communications of the ACM 67*.52(3), March 2009; doi:10.1145/1467247.1467274.

[101] David Harel. "Statecharts: A Visual Formalism for Complex Systems".*Science of Computer Programming.* 8(3), 1987, pp231-274. doi:10.1016/0167-6423(87)90035-9.

[102] David Harel and Michal Politi."*Introduction"* in *"Modeling Reactive Systems with Statecharts: The Statemate Approach".*[Online]USA: McGraw-Hill, 1998, pp 1-14, Available: http://www.wisdom.weizmann.ac.il/~dharel/STM.Book/chapter1.pdf.

[103] Gregory Bryon Storm.*"Statechart Visual Formalism as The Execution: Language For Control Of Real-Time Systems".*Master of Science Thesis [Online]. Faculty of Texas Tech University, May 1995. Available: http://etd.lib.ttu.edu/theses/available/etd-01292009-31295009265801/unrestricted/31295009265801.pdf.

[104] Jonathan Kaye and David Castillo.*"Describing Behaviors Using Statecharts"*in*"Flash MX for Interactive Simulation: How to Construct & Use Device Simulations".* Canada: Thomson Learning Inc. November 2002.Available: www.flashsim.com/pubDown/chaps/2667-ch07.pdf.

[105] Stéphane Faulkner.*"Information System Engineering: Analysis and Design, Section 3.2.3: Object-Oriented Modeling Statechart Diagram".* Class Note of GETI-2100, Catholic university of Louvain (Belgium), 2005-2006. Available :www.delta.cs.cinvestav.mx/~pmalvarez/softeng/.../Curso-Statecharts.pdf.

[106] *"http://jcsites.juniata.edu/faculty/rhodes/smui/statechart.htm*[Accessed: Sept., 2011].

[107] DoronDrusinsky.*"Modeling and Verification Using UML Statecharts: A Working Guide to Reactive System Design, Runtime Monitoring and Execution-Based Model Checking".* UK: Elsevier Inc., 2006.

[108] David Harel and Michal Politi. "*The Behavior View Statecharts*" in *"Modeling Reactive Systems with Statecharts: The Statemate Approach".* USA: McGraw-Hill, 1998, pp 53-72. Available:http://www.wisdom.weizmann.ac.il/~dharel/STM.Book/chapter4.pdf.

[109] J. A. Ferreira and J. P. Estima de Oliveira. "Modelling Hybrid Systems Using Statecharts and Modelica". *In Proceed. 7th IEEE Int. Conf. Emerging Technologies and Factory Automation ETFA'99,* 2, 1999 Barcelona (Spain), pp 1063 - 1069.doi: 10.1109/ETFA.1999.813108.

[110] Łabiak G., Miczulski P. "Uml Statechart sand Petri Nets Model Comparison for System Level Modelling". *Behaviour* [Online]. 27, 2004, pp 5-9. Available: http://lord.uz.zgora.pl:7777/skep/docs/F2015/GLabiakPMiczulskiSbornik03.pdf.

[111] João W.L. and al. "Validation of Statecharts Based on Programmed Execution".*In Proceed. 7th Int. Conf. Computing and Information (ICCI'95).* Trent University, Peterborough, Ontario, Canada, July 5-8, 1995, pp. 870-885.

[112] Bruce Powel Douglass. "UML Statecharts". *Embedded Systems Programming.* Sept 2003.Unpublished available: http://www3.informatik.uni-erlangen.de/Lectures/UMLEmbSys/WS2001/slides/Statecharts.pdf.

[113] *http://www.klangfarbe's_UML-Statechart-Framework-for-Java_at_master_-_GitHub.htm*[Accessed: Sept 2011].

[114] "*Simulation Modeling with AnyLogic: Agent Based, Discrete Event and System Dynamics Methods (Designing state-based behavior: statecharts)*". XJ Technologies [Online].Available:http://www.xjtek.com/files/book/Designing_state-based_behavior-statecharts.pdf.

[115] *http://www.jade.tilab.com* [Accessed:Sept 2011].

[116] Martin L. Griss and al. "SmartAgent: Extending the JADE Agent Behavior Model". *AOSE Workshop*, SCI, Orlando, Florida, July 2002, p 1-10. doi:10.1.1.58.1731.

[117] Danny Weyns,Elke Steegmans and Tom Holvoet. "Combining Adaptive Behavior and Role Modeling with Statecharts". *3th Int. Workshop Software Engin. Large-Scale Multi-Agent Systems (ICSE'04)*. 2004, pp 81-89. doi :10.1049/ic:20040363.

[118] F. Cicirelli, A.Furfaro, A. Giordano, L. Nigro."Statechart-Based Actors For Modeling and Distributed Simulation of Complex Multi-Agent Systems". *In Proceed. 23rdEurop. Conf.Modelling and Simulation ECMS'09*. Madrid, Spain, 2009, p 7. Available : http:// www.scs-europe.net/conf/ecms2009/.../abs_0112_310d27b6.pdf.

[119] Giancarlo F., Alfredo G., Samuele M. and Wilma R. "ELDATool: A Statecharts-based Tool for Prototyping Multi-Agent Systems".*In Proceed. Workshop Objects and Agents (WOA'07)*. Genova, Italy, 2007, p6. doi:10.1.1.144.658.

[120] *http://www.xjtek.com* [Accessed: Sept 2011].

[121] Stefan Emrich, SergejSuslov and Florian Judex. "Fully agent based modellings of epidemic spread using AnyLogic - Multimethod Simulation Software Tool AnyLogic". *EUROSIM 2007* [Online]. Ljubljana, Slovenia, Sept 9-13 2007. Available on http://www.xjtek.com/file/192.

[122] Peer-Olaf Siebers, UweAickelin, Helen Celia, Chris W. "Understanding Retail Productivity by Simulating Management Practices - Multimethod Simulation Software Tool AnyLogic". *EUROSIM 2007*. Ljubljana, Slovenia, September 9-13 2007. Available: http://www.xjtek.com/file/19.

[123] David Buxton, Richard Farr, Bart Maccarthy. "The Aero-engine Value Chain Under Future Business Environments: Using Agent-based Simulation to Understand Dynamic Behaviour". *MITIP2006*. Budapest, 11-12 Sept 2006. Available:http://www.xjtek.com/file/195.

[124] Maxim Garifullin and Andrei Borshchev."Using AnyLogic and Agent Based Approach to Model Consumer Market - Multimethod Simulation Software Tool AnyLogic". *EUROSIM 2007*. Ljubljana, Slovenia, Sept 9-13 2007. Available: http://www.xjtek.com/file/194.

[125] *http://www.xjtek.com/anylogic/articles*[Accessed: Sept 2011].

[126] *"AnyLogic, Multi-Method Simulation Software".* Decision Support Systems, Professional Consulting Services. AnyLogic brochure available: *http://www.xjtek.com.*

[127] *"AnyLogic Product Overview"*. XJ Technologies [Online], 2002. Available: *www.xjtek.com.*

[128] *http://www.coensys.com/agent_based_models.htm*[Accessed: Sept 2011].

[129] Cynthia L. Willard-Mack. "Normal Structure, Function, and Histology of Lymph Nodes". *Toxicologic Pathology* [Online]. 34, 2006, pp 409–424. Available:*http://tpx.sagepub.com/content/34/5/409.*

[130] Paul S. Andrews and Jon T. "A Computational Model of Degeneracy in a Lymph Node".*J. Springer*. 4163(1), 2006, pp 164-177. doi: 10.1007/11823940_13.

[131] *http://www.xjtek.com/anylogic/selected_customers/* [Accessed: Sept 2011].