International Journal of Current Engineering and Technology
©2017 INPRESSCO®, All Rights Reserved

E-ISSN 2277 – 4106, P-ISSN 2347 – 5161
Available at http://inpressco.com/category/ijcet

Research Article

A Design Pattern of Multi Agent based Simulation Model

Mohamed Younis Mohamed Alzarroug* **and W.Jeberson**

Dept. of Computer Science, Sam Higginbottom University of Agriculture Technology and Sciences Allahabad, Uttar Pradesh, India

Accepted 12 April 2017, Available online 22 April 2017, **Vol.7, No.2 (April 2017)**

Abstract

The software process simulation modeling plays a great curiosity nowadays amid the canvassers and the practitioners as it is a method for exploring the complex business and the policy questions. Though the simulation modeling is employed in several fields for many years, it has just lately employed in the field of software development and evolution procedures. At present, the software process simulation modeling just a start which is employed to tackle various concerns from the strategic management of software development, to support the process enhancements, to software project management training. The span of the software process simulation applications extend from the narrow concentrated parts of the life cycle to the long term product evolutionary models with wide organizational influences. This paper offers a work synopsis in the area of software process simulation modeling. In addition, it recognizes the queries and the concerns that the simulation can be employed to deal with the scope. Simulation modeling or modeling and simulation is an attracting multi-disciplinary subject which sketches out various approaches and tools. The simulation modeling is applied to study the performance of the real world schemes in the field of defense, health, manufacturing, communication and transportation. The study of the utilization of methods and tools of simulation modeling and the progress of the novel forms of these is an affluent basis of research exactly. Simulation modeling has nearly attained its fiftieth bicentenary observed by the progress of General Program (GSP) which is the initial simulation modelling language and the initial published effort in simulation modeling. At present, several canvassers and practitioners are engaged in the advancement and the application in this area.

Keywords: *MABSM, SPSM, E M.*

1. Introduction

The software projects have been affected by several concerns like soaring costs or the agenda overruns and low quality in product for pretty several periods. One of the significant issues for these kinds of troubles is unmanaged risk (Dapeng and Qing Wang Liu1, 2009). Risk management in software plays a vital role for effective management of the software project. The Canvassers and the practitioners have projected several methods for software risk management both efficiently and methodically. Among those methods a leading effort in managing the software risk was introduced by Barry Boehm (Miguel, 2015). In this structure, the managing the software risk includes two main phases with three subsidiary activities such as the risk assessment phase includes risk identification, risk analysis and risk prioritization activities, whereas the risk control phase includes risk management planning, risk resolution and risk monitoring activities. This structure described the fundamental activities for the software risk management has been embraced by several other risk management forms and procedures. Soft risk management is commonly ignored in the real world project management even though there includes several techniques. A research by the Project Management Organization revealed that the risk management is the least experienced when related to all the other project management disciplines in the field of IT.

For the last few decades the software industry has been attacked by several forms of schedule and cost overruns moreover the low quality in product distributed by the software development institutions in the sector of commercial as well as the administration. Simultaneously, the raise in demand of the customer for `enhanced, sooner, cheaper' and the raise in the software intricacy have considerably `raised the bar' for the software developers to enhance the performance. In the real software projects, the threats are frequently controlled by the perception of project managers and the entire procedure of risk management is not often pursued. One of the prime causes for this fact is the deficit of realistic methods by the project managers and the apparatus to efficiently deal with the threats of software. The conventional techniques and replicas for the software risk management are employed not often in the actual

*Corresponding author: **Mohamed Younis Mohamed Alzarroug; Dr.W.Jeberson** is working as Professor software projects as they are more general to direct the behaviors of the operational risk management or else their applicability is restricted to few particular situations.

The industry has attained the assistance based on the structure of the surplus of case apparatus, latest computer languages and the latest and classy machines. A main query now is `how the tools, technologies and the people team up to attain these confronting objectives?' A smart answer to this query is alteration in the software development procedure or the software organization. The probable variation needs a considerable resource to execute and hold considerable implications on the organization either excellent or not. How can the institutions attain insights into the probable answers to these troubles and their probably influences on the institution? A field of study that has tried to handle these queries and holds few triumphs in forecasting the effect of few intended answers is the software process simulation modeling.

The software process simulation modeling plays a great curiosity nowadays amid the canvassers and the practitioners as it is a method for exploring the complex business and the policy questions. Though the simulation modeling is employed in several fields for many years, it has just lately employed in the field of software development and evolution procedures. At present, the software process simulation modeling just a start which is employed to tackle various concerns from the strategic management of software development, to support the process enhancements, to software project management training. The span of the software process simulation applications extend from the narrow concentrated parts of the life cycle to the long term product evolutionary models with wide organizational influences. This paper offers a work synopsis in the area of software process simulation modeling. In addition, it recognizes the queries and the concerns that the simulation can be employed to deal with the scope

Simulation modeling or modeling and simulation is an attracting multi-disciplinary subject which sketches out various approaches and tools. The simulation modeling is applied to study the performance of the real world schemes in the field of defense, health, manufacturing, communication and transportation. The study of the utilization of methods and tools of simulation modeling and the progress of the novel forms of these is an affluent basis of research exactly.

Simulation modeling has nearly attained its fiftieth bicentenary observed by the progress of General Simulation Program (GSP) which is the initial simulation modelling language and the initial published effort in simulation modeling. At present, several canvassers and practitioners are engaged in the advancement and the application in this area.

There includes several forms of simulation. In this paper it is limited to the discrete and stochastic process oriented simulation. This embraces nearly all the simulations conversed at the Winter Simulation Conference. It keeps out the Monte Carlo-type simulations in a spreadsheet (sampling survey, financial and risk analysis, and so on). Moreover, it keeps out the equation based numerical solvers, for instance, the differential equation solvers and the other equation based replicas. Though it is not clearly conversed, it comprises the training simulations and the man-in-the-loop simulations like many accomplished by the military (Taylor, 2011).

1.2 Simulation concepts

A depiction of a system or process is termed as a model. A simulation model is a depiction which integrates the time and the variations which takes place over time. In SPSM, SD and DES are the most frequently employed methods. In addition, hybrid simulation (SD plus DES) is employed for software simulation (Robert, 2012). A discrete event model facilitates to denote the sequential interdependence which takes place amid the behavior in a software procedure.

The variation which occurs only at the discrete points in time but not in a continuous manner is termed as discrete model. A model integrates the logical, arithmetical and structural view of the organization or process. A discrete event model conversed mainly in many papers in the Winter Simulation Conference is the one which depends on the idea of state, events, behavior and processes. Time is a vital part. A discrete event model is the one in which the state alters only at the discrete times is termed as event times. When an event arises, it triggers the recent events, behavior and processes. Theoretically, the state of a model state is a (lengthy) vector, i.e., a list of value which is enough to describe the entire system state at any point in a time. Practically, a state of a model is described implicitly by the interior status of the entire entities utilized in the simulation software package (Carson, 2004).

An event is an instant happening that varies the model's state. An instance for this is an arrival event for a client at a bank and a service completion event for the same client. An activity signifies time duration like a service time or an inter arrival time which is instigated by an event in conjunction with the model in a definite state. For instance, when the arrivals are described by a probability distribution of inter arrival times, subsequently when one arrival arises (an event), the model produces a novel inter arrival time (an activity) which subsequently generates the other arrival event.

Primary events are the events which are driven by the data. Few instances include the arrival times the service completion times. In terms of simulation, the primary events are arranged to arise at a few future time, computed from the data and the statistical conjectures. For instance, if we consider that the inter arrival times are exponentially distributed with mean ten minutes, and after that at the time of an arrival, the model sketches a fresh exponential sample, includes it to the current time, and programs the resulting future time as the instant of the subsequent arrival event. Secondary events are those which are produced within by the model logic. For instance, in a waiting line model, when a server is available and there is a waiting entity (a person or a product) ahead of the queue then a 'service begin' event is arranged to happen instantly.

An entity describes an object in the model. The dynamic entities are formed at zero or else at other times by means of an arrival event. Dynamic entities typically signify some real world object that is flowing by means of a system. Few instances comprise the automobiles in a manufacturing model, the pallets or the cases in a warehouse model, the passengers in an airport model and the telephone calls in a communications model. Entities hold the standard and customized traits that individualize the entity.

An entity which offers service to the dynamic entities is termed as a resource. A resource typically holds a finite capacity signifying few system constraints. One such example is a worker or a team of workers performing a task, a machine, or a vehicle. In few models, resources hold user defined resource states and distinctive characteristics like the downtimes and accessibility schedules. Nearly every discrete event models are stochastic in nature which means it comprises few components which are modeled as a statistical distribution. This instigates random variation into a model, building it into a statistical or a sampling experiment. In general, when one or more components are stochastic (i.e., the intertribal or service times), then the outcomes of the model are stochastic, requiring few forms of statistical analysis to sketch valid conclusions. Nearly every simulation software packages comprises automatic compilation of performance statistics, and trouble free compilation of custom statistics

2. Techniques in Simulation Model Design

The process of model design may be categorized into the subsequent stages:

- conceptual model design or pre model formulation escorted by the pre formation of a knowledge base,
- declarative model design or description based on the mathematical equations which cast away the unnecessary factors,
- functional model design or the formulation of mathematical laws where the object obeys,
- calibration of the model, creation of knowledge base and reason of study,
- problem investigation especially by means of computer experiment,
- Evaluation of the outcomes of model investigations with practice outcomes and the outcomes attained from the other model.

Possibly the toughest general difficulty in simulation is to determine the precise technique to employ for generating a model. Finally, how the starting point is found out? While the discipline of software engineering has come out to react for this question for software, generally, the modelers also come across these kinds of troubles: how does one engineer models? As there includes several modeling methods for simulation, one is frequently in uncertainty as to which model approach to utilize and under what circumstances it should be employed.

2.1 Conceptual Models Design

In the account of a problem existing thoughts are constantly presumed. Though, it is predicted that the fresh ideas will also be formed at the time of solution process. The concepts are yet little value until they are articulated, generally by means of the medium of language or block schemes. It is merely then that the individuals converse their thoughts to the others and can assist in solving the problem

- To design a conceptual model they employ certain techniques as rule as mentioned below:
- Preference of main agents and main subsystems they act,
- Preference of main values described states of studied system,
- Design of a block-scheme of interacting agents and subsystems which comprises flows for the fixed values.

Currently these techniques are prescribed only for very simple models. In formal model each entity being modeled in the real world holds a noticeable and one to one equivalence to an object in the model (like the agent, subsystem, value). As the conceptual model plots openly from the entities in real world to the objects in computer based model, they build it simple to design and execute the systems. The resulting systems are simpler to utilize as they are semantically apparent to the users who recognize the problem area previously.

2.2 Declarative Models Design

A model is said to be declarative when the current state of the system decides the measures of agents and the mode where the state will be altered. The declarative model specifies the responses to the states. On the contrary, the imperative model specifies a subsequent state in accordance with the given state. The dynamic imperative models produce the series of states without indicating how they are attained whereas the declarative models denote the actions performed and the variations made. The declarative modeling approaches facilitate the modeler to comprise the qualitative deliberations with no loss in accuracy. Practically, they build up the models in such a manner where the simulation outcome comprises numerical series, which are expressively related with few statistical data series.

2.3 Functional Models Design

The aim of the functional model of a system is to explain the operations performed on the system. The functional model comprises functional entities of dissimilar types. Every form of entity is described by the function it holds.

2.4 Constraint Models Design

Every constraint (for instance, x + y = z, the triangle is within the circle) confines the probable values in which variables can acquire, it signifies few partial information regarding the variables of interest.

Every entity in a declarative model is defined initially with the constraints which describe the nature and the relationships amid them. The search program code is made secede from the model description. To design a model based upon the constraints it is essential to transform its description from the usual language to the language of constraints. For effectual problem solving selecting the right model and the right constraint satisfaction algorithm is very important.

2.5 Multi-models Design

This design is an expansion of the object oriented design. The prime contribution of the object oriented method for simulation is mapping amid the physical objects and the digital earth. Classes are coupled to construct the hierarchical structures by means of two probable connectors, generalization or aggregation. An object is an instance of a class. Each object comprises features and techniques. Attributes signifies the object properties whereas the techniques denotes the functions and procedures which function upon the attributes. Attributes is of the following two forms namely a) Static and b) Dynamic. A static attribute is will not changes over time. A dynamic attribute signifies a component of the state vector related with the object. The simulation is primarily related with the dynamic attributes. A model is formed by building the objects and relating them by means of message passing.

Technology of the multi model design comprises the subsequent stages:

- Conceptual model design for an investigated life support scheme,
- Breaking the system into a hierarchy of multi model (abstractions),
- Recognition of models to denote those abstraction levels.

Few levels may be declarative or functional in nature and few might merge these elements. A multi model is a set of individual models coupled wholly to encourage intersection of levels. Recognition is a method of the selection of a suitable model for every level. When the selection satisfies based upon statistical data this process comprises processes of data collection, and model calibration – alteration of model parameters. After the model has been intended, it ought to be executed. The way that one executes a model is based upon the type of model.

After designing the model it ought to be executed. The method that one executes a model is based upon the type of model.

3. Methods in Execution of Simulation Models

A simulation is the method of imitating the performance of few system or circumstances using an analogous model, situation, or apparatus either to attain information more suitably or to train the personnel. Computer simulation holds remarkable effect on all the parts of life as it is expensive or exorbitant to construct real physical systems. Simulation facilitates you to build decisions and lift up hypothetical circumstances.

3.1 How to develop a simulation model

A simulation model comprises the following components: system entities, input variables, performance measures and functional relationships. A simulation study is in fact as excellent as the simulation model. It includes the following five steps:

Step1. Identify the problem. Detail the problems with the conventional system. Generate the needs for the intended system.
Step2. Formulate the problem. Pick the bounds of the system, the problem or a component thereof, to be analyzed. Describe the overall goal of the research and some particular concerns to be addressed. Describe the performance measures - quantitative criterion based on which the dissimilar system configurations will be related and ranked. Recognize, concisely at this phase, the configurations of interest and formulate the hypothesis about the performance of the system. Determine the time frame of the analysis, i.e. if the model will be employed for single time decision (capital expenditure) or a time period on a regular basis (air traffic scheduling). Recognize the end user of the simulation model, for instance, the corporate management vs. a production supervisor. The problems ought to be framed as specifically as probable.
Step3. Accumulate and process the real system data. Accumulate data on the system specifications (for example, bandwidth for a communication network), input variables and also the performance of the conventional system. Recognize the sources of randomness in the system, which means the stochastic input variables. Pick a suitable input probability

distribution for every stochastic input variable and assess the consequent parameters. Software packages for distribution fitting and selection comprises Expert Fit, Best Fit, and add-ons in few typical statistical packages. These aids pool the goodness-of-fit tests, for example, χ2 test, Kolmogorov Smirnov test, and Anderson Darling test and parameter estimation in an easily accessible format. Standard distributions like exponential, Poisson, normal, hyper exponential are simple to model and simulate. Though several simulation software packages comprise various distributions as a typical feature, topics relating to the random number generators and creating the random changes from several distributions are relevant and should be looked into. The empirical distributions are employed when the standard distributions are not suitable or do not match the existing system data. Triangular, uniform or normal distribution is employed as an initial guess when no data exist.

Step4. Formulate and build up a model. Build up the schematics and network diagrams of the system (How does the entity flows through the system?). Interpret these conceptual models to the simulation software acceptable form. Authenticate that the simulation model implement as proposed. The verification methods comprises the traces, changing input parameters over the acceptable range and assessing the output, substitute the constants for random variables and physically checking the outcomes, and animation.

Step5. Validate the model. Evaluate the performance of the model under specific circumstances with the performance of the real system. Carry out the statistical inference analysis and obtain the model analyzed by the system experts. Evaluate the confidence that the end user situates on the model and deal with the problems if any. For most of the simulation research, practiced consultants advocate a structured presentation of the model by the simulation consultants prior to the audience of management and system experts. This not only assures that the model considerations are accurate, complete and reliable; however it improves the confidence in the model.

4. How to choose the simulation software

Though a simulation model can be formed by means of general purpose programming languages which are recognizable to the consultants that exist over several platforms, and cheap, most simulation analysis these days is implemented by means of simulation package.

Table 1: Simulation Packages

Types of simulation package	**Examples**
Simulation languages	Arena (SIMAN), AweSim (SLAM II), Extend, GPSS, Micro Saint, SIMSCRIPT, SLX Object-oriented software: MODSIM III, SIMPLE++ Animation software: Proof Animation
Application oriented Simulators	Manufacturing: Auto Mod, Extend+MFG, FACTOR/AIM, QUEST, Taylor II, ManSim/X, MP$IM, ProModel, WITNESS Communications/system: COMNET III, NETWORK II.5, OPNET Modeler and Planner, SES/Strategizer, SES/workbench Business: BP$IM, Extend+BPR, Process Model, Service Model, SIMPROCESS, Time machine Health Care: Med Model [9]

The simulation packages are of two types namely a) simulation languages and b) application-oriented simulators (Table 1). The simulation languages are more flexible than the application oriented simulators. Conversely, the languages need changeable quantities of programming knowledge. The application-oriented simulators are easy to study and have the modeling constructs nearly associated to the application. Majority of the simulation packages integrate animation this is admirable for interaction and is employed to debug the simulation program, a 'correct looking' animation, though, it is not assured as a valid model. More significantly, the animation is not a replacement for the output psychiatry.

4.1 Software process simulation modeling (SPSM)

This is a valuable technique employed to deal with various concerns in software engineering as it was instituted into this field at the end of the year 1980, it is employed in several software development or maintenance projects with changeable process scales and organization situations in the past decades. It enforces planning, organizing, controlling, enhancing the process of the software, and offers the software practitioners (experts and managers) powerful tools and identifiable advantages.

4.2 Agent-based simulation model design (ABMS)

It is an innovative modeling paradigm and it is one of the stimulating practical enhancements in modeling the innovation of relational databases. It assures to hold far reaching consequences on the way that traders utilize systems to support decision making and the canvassers utilize electronic laboratories to assist their research (Charles, 2008).

Agents

There is not general concord on the accurate description of the term agent in the framework of ABS. It is the main issue for argument and infrequent debate. The topic is higher than the scholastic one, as it frequently surfaces when one builds a claim that their model is agent based or when one is striving to determine if such claims made by others holds legality. There includes major insinuations of the term agent based when employed to explain a model based upon the model abilities or the potential abilities which is achieved by means of comparatively minor alteration. Few modelers comprise any form of independent component, if it is a software constituent or a model to be an agent. Some authors insist that a behavior of a component is adaptive in order for it to be deliberated as an agent. The agent comprises both the level rules for behavior and also a high level set of rules to vary the rules. The base level rules offer reply to the environment, whereas the rules to vary the rules offers alteration. Based upon the view of computer science the agent highlights the necessary agent trait of the autonomous behavior.

Modularity

Agents are modular or self-contained. An agent is a particular, discrete individual with a group of features, behaviors, and decision making ability. The modularity need signifies that an agent holds a boundary, and one can simply decide if something (i.e., an element of state of the model) is branch of an agent, is not the branch of an agent, or is a feature shared amid the agents (Betim cico, 2011).

Autonomy

The agent is autonomous and self-directed. It can perform autonomously in its environment and in its interactions with the other agents, particularly from a narrow situation range that are of interest which takes place in the model. When we refer to the behavior of the agent, we refer to a process which relates the information the agent senses from its environment and communications to its choices and actions.

Sociality

The agent is social, communicates with the other agents. The general agent interaction protocols comprises contention for space and crash evasion, agent identification, interaction and information substitute, influence, and other domain or the application detailed mechanisms.

Conditionality

The agent comprises a state which changes over time. Similar to a system has a state comprising of the group of the state variables; an agent also holds a state which signifies its condition, the key variables related with its present condition. The state of an agent comprises a set or a subset of its features. The state of an agent based model is the communal states of every agent together with the state of the environment. The behavior of the agent is conditioned on its state. Which means, the richer the group of the probable states of the agent, the richer the group of behaviors that an agent holds.

The agent frequently holds more properties, which can or cannot be deliberated as a vital property for agency. An agent holds clear objectives which drives its behavior, not essentially the objectives to increase as much as criterion against which to evaluate the efficiency of its choice and events. An agent possesses the ability to study and acclimatize its behaviors in accordance with its knowledge. A frequent cause for system modeling as an ABS is to deliberate the agent learning and variation. At the individual level, learning and adaptation can be represented as the behavior of the agent. The individual learning and adaptation needs an agent to hold memory as a dynamically modernized quality of the agent. In the population level, modification can be modeled by facilitating agents to enter and leave the population, with more successful agents raising their relative numbers in the population ultimately.

Agent Relationships

Agent-based modeling relates itself with the modeling agent relationships and agent communications as much as it does the modeling agents and the behavior of the agent. The primary concerns of the modeling agent communications are indicating who is or who could be, related to who, and the dynamics managing the mechanisms of interactions (Macal1, 2010). For instance, an agent based model of Internet growth comprises mechanisms which indicate who relates to who, why and when. The frequent topologies for denoting the social agent interactions are mentioned below:

(1) Soup. A non-spatial model where the agents do not holds location attribute.
(2) Grid or lattice. Cellular automata epitomize the agent interaction patterns and the existing local information by a grid or lattice; cells which surround the agent nearly are its neighborhood. The location of the agent is the grid cell index.
(3) Euclidean space. Agents move in two dimension or three dimension spaces. The location of the agent is its relative (grid) or geospatial (lat-long) coordinates.
(4) Geographic Information System (GIS). Agents moves and communicate with the sensible geo-spatial landscapes. The location of the agent is a geographical unit (for instance, zip code) or the geospatial coordinates.
(5) Networks. Networks are either static (the links are pre-specified) or dynamic (the links ascertained endogenously). The location of the agent is relative to the other nodes in the network.

It is not at all an issue about the agent interaction topology which is employed in an agent based model to link the agents; the vital concept is that the agents only communicate at any specified time with the confined number of other agents out of every agent in the population. This concept is executed by describing a local neighborhood and restraining the interaction to fewer agents that are positioned in that neighborhood. This is not to state that the agents are required to be located in near proximity to one another spatially to be capable to communicate. The network topology facilitates the agents to be related based upon the relations besides the proximity. For instance, an agent could be a member of several networks such as proximity, communal, family relationship, ideological etc.

Autonomous agents

A distinct most significant describing the feature of an agent is its ability to perform individually, which means to carry out on its own with no external direction based upon the conditions it encounters. The agents are endowed with behavior which facilitates them to take own decisions. In general, the agents are active, instigating their actions to attain their internal goals, rather than simply passive, reactively replying to other agents and the environment. There is no common contract in the writing on the exact definition of an agent away from the vital property of autonomy. Jennings (2000) brings out a computer science description of an agent which highlights the vital quality of the autonomous behavior. In this point of, a behavior of the component ranges from simple and reactive 'if-then' rules to complex behaviors modeled by the modified artificial intelligence methods. The base level rules offers more passive reactions to the environment, while the 'rules to vary the rules' offers more dynamic and adaptive facilities. In the perspective of the practical modeling, depending upon how and the why agent models are really created and expressed in applications, we assume the agents to hold definite vital characteristics:

An agent is a self-contained, modular, and exclusively identifiable individual. The modularity necessity means that an agent holds a boundary. One can simply find out if something is a part of an agent or else not the part of an agent, or is a shared attribute. The agents hold the features facilitate the agents to be identified from and acknowledged by other agents.

An agent is independent and self-directed which performs autonomously in its environment and in its communications with the other agents, at least over a narrow range of conditions which are of interest in the model. An agent holds behaviors which link the information sensed by the agent to its choices and events. The information of the agent move towards via the interactions with the other agents and with the environment. The behavior of the agent is identified by anything from easy rules to abstract models like the neural networks or the genetic programs which relate the agent inputs to outputs by means of adaptive mechanisms. An agent holds a state which changes over time. Similar to a system holding a state comprising the compilation of the state variables, an agent also has a state which signifies the necessary variables related with the present situation. The state of an agent comprises of a set or a sub-set of its features. The state of an agent based model is the communal states of every agent together with the state of the environment. The behavior of an agent is conditioned on its state that is the richer the set of probable state of an agent, the richer the set of behaviors that an agent can hold. In an agent based simulation, the state at any time is all the data required to shift the system from the point ahead.

An agent is social holding dynamic interactions with the other agents which influence its behavior. Agents possess protocols for communication with the other agents like for communication, transfer and space contention, the capability to react to the environment and others. Agents hold the capability to identify and discriminate the features of other agents.

Agents may also possess other useful characteristics

- An agent may possibly be adaptive, that is by holding the rules or more abstract mechanisms which change the behaviors. An agent holds the capability to study and adjust its behaviors in according to its accumulated experiences. Erudition needs few type of memory. Moreover the adaptation at the individual level, populations of agents may perhaps be adaptive by way of the process of selection, as individuals are well apt to the environment proportionately raise in numbers.
- An agent may probably goal directed, holding objectives to attain (not necessary the objectives to rise) based upon its behaviors. This facilitates an agent to evaluate the result of its behaviors corresponding to its goals and modify its responses and behaviors in upcoming interactions.
- Agents may probably be heterogeneous. Not similar to the particle simulation that deliberates comparatively homogeneous particles like the idealized gas particles, or molecular dynamics simulations which model the individual molecules and their communications, agent simulations often assume the full range of agent diversity across a population. The characteristics and the behavior of the agent may fluctuate in their level and sophistication, how much data is deliberated in the decision of the agent, the agent's internal models of the outside globe, the view of the agents probable reactions of other agents based upon its actions, and the level of memory of the previous events the agent retains and employs in taking decisions. Agents may also be endowed with distinct amounts of resources or collect dissimilar resources levels on account of the interactions of the agent, further differentiating agents. A characteristic agent structure is shown in the Figure 1. In an agent based model, the whole thing is related with an agent may be an agent attribute or an agent approach which functions on the agent. Agent features may static, invariable at the time of simulation, or dynamic, variable as the simulation progresses. For instance, a static attribute is the name of an agent; a dynamic attribute is the memory (past interactions) of an agent. The technique of an agent comprises behaviors like the rules or more abstract representations like the neural networks, which relate the situation of the

agent with its action or a set of potential events. An example is the approach that an agent employs to recognize its neighbors.

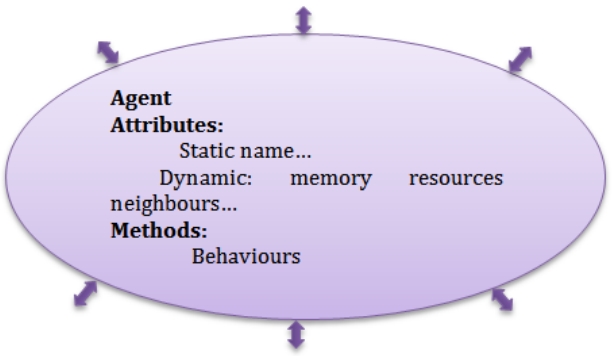

Figure1: A typical structure of an agent

5. Emergence in Agent-based Models

One of the main reasons for the agent based modeling is its capability to capture emergence. Emergence is the vital or the appealing behavior revealed by the agents or the process under study, which is not explicitly modeled at the outset, but which appears as a result of agent interactions when the model is run. Emergent behavior is often unexpected. How to identify unexpected behavior in simulation output is an area of further research. Even simple agent-based models that are completely described by simple, deterministic rules and based only on local information can self-organize and sustain themselves in ways that have not been explicitly programmed into the models. Emergence can be illustrated by simple agent based models like the Life and Boids. Several complex models of the kind which the people are probable to developed to denote the real world phenomenon can also reveal the developing behavior ensuing from the agent interactions. The agent based modeling algorithms is based upon emergence which led to optimization techniques such as ant colony optimization and particle swarm optimization, which have been used to solve practical problems.

5.1 How to perform agent based modeling

Thinking through an Agent Model

Agent based model development pursues the general steps of emerging any model with the additions of the agent related tasks as illustrated in figure 2. It is valuable to inquire a series of agent specific queries prior to develop an agent based model as in Table 1. It is also useful to continue to ask various questions at the time of development, as the ABS development is an iterative procedure of refinement. The answers to these questions help define the scope and level of detail, granularity, appropriate to modeling the system. They signify the resources need for effectively finishing the project as well as help identifying possibly the bottlenecks to development.

Modeling Agent Systems

Identifying agents, accurately specifying their behaviors, and appropriately representing agent interactions are the keys to developing useful agent models. One begins developing an agent-based model by identifying the agent types (i.e., classes) along with their attributes. Agents are generally the decision makers in a system as they are human, organizational or automated. Once the agents are defined, the behavior of the agent is specified. One needs to hold a theory of agent behavior as a base for modeling agent behavior. For instance, a normative model where the agents attempt to optimize a well-defined objective can be a useful starting point to eventually developing more descriptive and domain-specific behavioral heuristics. Alternatively, one may initiate with a bounded rationality model or a generic behavioral heuristic, such as anchoring and adjustment, to describe agent behavior, or a formal behavioral modeling framework such as the Belief Desire Intent (BDI) model may be appropriate. In addition to agents, an agent-based model consists of agent relationships. One defines the agent relationships and then adds the methods that control which agents interact, when they interact and how they interact

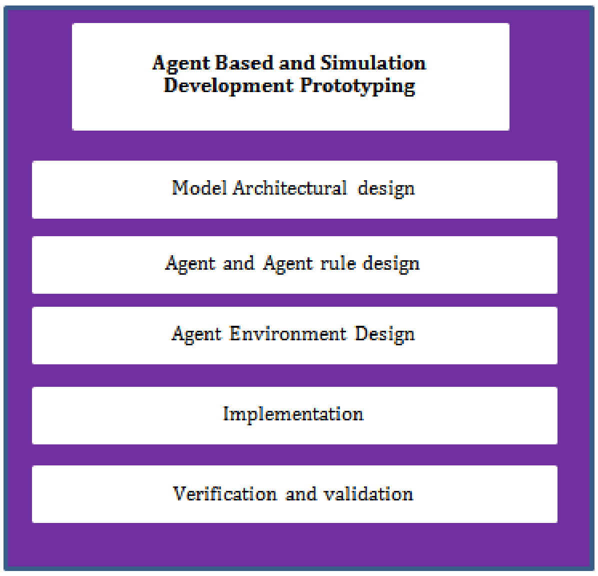
Agent Based and Simulation Development Prototyping
Model Architectural design
Agent and Agent rule design
Agent Environment Design
Implementation
Verification and validation

Figure2: Agent based model and simulation (ABMS) development process

5.2 The Need for Agent Based Modeling

The reason for the agent based modeling as becoming common nowadays is because we exist in an increasingly complex world. Initially, the systems that ought to examine and model are becoming more complex based upon their interdependencies. The conventional modeling tools are not applicable more enough nowadays. An instance for the application area is the deregulation of the previously centralized electric power industry where the agents are rapidly free to make pricing and investment choices in accordance with their individual criterion. Next, few systems have constantly been tricky for us to sufficiently model. Modeling economic markets has conventionally based upon the ideas of the perfect markets, homogeneous agents, and long run equilibrium as these conjectures made the troubles systematically and tractable computationally [16]. It is capable to relax some of these conjectures and take a more realistic view of these economic systems by means of ABMS. Third, the data are convened and organized into databases at excellent granularity level. Micro data can now prop up the individual based simulations. And finally, the power of computation is progressing rapidly. Then the large scale micro simulation models are computed which would not have been conceivable just a couple of years ago

5.3 How to think about agent-based modeling

Structure of an agent-based model

A classic agent based model comprises the following three factors:

• Agents, their traits and behaviors.
•Relationships of the agent and techniques of interaction. A fundamental topology of connectedness describes how and with whom the agents are interacted.
• The atmosphere of the agent. The agents reside and communicate with their environment, in addition to the other agents.

A. Agent-driven Model Design

It is the initial strategy. It matches up to the bottom up design. The focus relies entirely on the agents, their decision making and their performance. The environment and interaction models are included when desired in design of the agent.
A.1 Basic Strategy: The following procedure can be described:
1) Agent observation and coarse behavior description: The modeler views the real world agents and obtains coarse behavior depiction from its observations. Observations are substituted by the literature effort or the operational of theory about the behavior of the agent or decision making .
2) Categorize agents and determine the location of heterogeneity:
The coarse behavior description are examined for ascertaining how many classes or forms of agents are vital for the model and to which level the agents should be dissimilar. The position of heterogeneity may be on the phase of parameter settings, dissimilar activities or yet entirely the dissimilar classes.
3) Decide about agent architecture:
Depending upon the coarse behavior depiction that deals with the agents superficially, the modeler has to choose about the agent architecture. A behavior describing technique may be selected, for instance the perception action rules with or without the internal state depiction or an architecture which is explicitly beached on goals and on the configuration and assortment of plans or even employing plan generation from the initial principles or a detailed cognitive model. This selection is based upon the intricacy and the flexibility of the necessary agent behavior.
4) Formalize agent behavior/goals:
This step comprises satiating the actual behavior into the agent architecture. This is carried out by analyzing, elaborating and refining the coarse behavior description build up in the initial steps.
5) Include interactions and environmental features when required, specific elements of the environmental model or the structures of interaction are included as the behavior of the agent requires comprising specific perceptions, messages or includes manipulations of environmental entities. As these aspects are included in a few ad hoc approaches, re-factoring ought to needed like merging dissimilar environmental entity classes. Moreover, few considerations about the required heterogeneity are necessary.
6) Examine if essential macro phenomena are adequately reproduced: At any time, the validity of the model has to be tested when it is testable. We consider that the focus on the agents paves the way to the behavior of the agent close to validity. However, a prime effort will be testing if the interplay of agents and their environment outcomes in a suitable behavior of the macro level.

This process is a real agent-driven bottom-up technique for developing a multi-agent simulation. Features which relate the agents to others or the environment are significant in so far they are persuading the behavior of the agent.

B. Interaction-driven Model Design

There includes simulation domains where a focus on interactions is more suitable than a purely agent driven design. The model which concentrates on the performance of organizational design is one such example. In the following there is a need to introduce the interaction-driven design.
1) Basic Process: One can devise the subsequent basic process for interaction driven model design:
1) Identify actors or the entities and interactions amid them: Instead of observing the individual real world agents, the modeler is taking the perspective of bird and examines who is interacting and how.
2) Course description of the protocols and their circumstances, constraints, etc. The recognized actors and interactions are refined to protocols going from common concepts of interaction to atomic exchanges of information and manipulations of the environment.
3) Derive agent behavior for creating the interaction elements (messages, signals, actions) and include

environmental entities like shelter objects, to the model when required for interaction. In this step something such as a finite state machine based language can be employed to specify agent interactions as state transitions.
4) Execute the behavior of the agent and examine if the intended interactions and their outcome on the macro level are actually produced by the overall system. It must also be tested if the agent level behavior is plausible or valid based upon the available data.
In the interaction driven method, agents are principally observed as black boxes for creating the suitable messages, information, etc. The general procedure may be further developed into some form organization driven model design motivated by similar methods and methodologies from agent-oriented software engineering. After that, the analysis of the organizational structures forms the starting point for all system analysis as it creates the structural backbone of interactions.

C. Environment-driven Model Design

The third strategy to examine is driven by a focus on the environment.
1) Basic Process: In analogy to the previously discussed design strategies, the starting point of the environment-driven design is an analysis of the environmental structure. Based on this, the agent interface and its behavior definition are determined. The steps are in particular:
1) Identify relevant aspects (global status, global dynamics/ local entities) of the part of the model that represents the environment.
2) Determine the primitive actions of the agent and the reaction of the environmental entities to these.
3) Determine what information from the environment must be given to an agent so that it can appropriately select and perform its actions.
4) Choose on an agent architecture which is appropriate to attach perceptions and actions of the agent suitably for essentially producing the behavior of the agent. At the same time the elements of the internal agent status are settled.
5) Use a learning mechanism for determining the actual agent behavior. A reward function for providing feedback to the agents' actions has to be found. The reward schema also tackles questions such as when and how often to provide feedback to the agents, whether all agents learn based on a shared reward or individual reward is given to the agents.
6) Implement the environmental model including reward function if needed.
7) Specify and implement the behavior of the agent program or agent interfaces in combination with the chosen learning mechanism.

5.3 Advanced of agent-based modeling

An agent-based modeler includes various advanced abilities habitually in the field of distributed computing execution, artificial algorithms and machine learning algorithms, geographical information systems i.e., GIS, associations to the relational databases, version control schemes (particularly if there includes several developers functioning on a project), and integrated development environments i.e. IDE. It is valuable to construct a core model initially which comprises these potential as associations or 'stubs' to assure that the design of core model is suitable and to confirm that scaling up the design emerges possible. The agent based modeling and software toolkits generally offer sophisticated capabilities like these things.

5.4 Implementation of AGENT based modelling

Agent-based modeling can be performed by means of general, all-purpose software or programming language or it can be carried out by means of specially designed software and toolkits which addresses the unique needs of agent modelling. This modeling is carried out in the small, on the desktop, or in the large by means of large scale computing cluster, or it is performed at any scale amid these extremes. The projects often begin small, utilizing one of the desktop ABMS tools, and then mature in stages into the large scale ABMS toolkits. When one starts to construct their original agent model by means of that technique which is most common with, or the technique that one find it simple to study specified their background and knowledge. The execution alternatives are also discriminated to build up an agent based models based upon the software employed. Spreadsheets like Microsoft Excel, in several ways provide the simple method to modeling. It is simple to build the models with spreadsheets than with several other tools, however the resulting models normally permits limited agent diversity, restrict agent behaviours, and holds feeble scalability when related to other techniques. Few macro level programming is also required by means of the VBA language. The common computational mathematics systems like MATLAB and Mathematica are also employed effectively. Moreover, these systems offer no exact abilities for modeling agents. The general programming languages like Python, Java, and C++, and C are employed however the growth from the scratch can be prohibitively classy specified that this requires the growth of several available services previously offered by specialized agent modelling tools. Nearly all the large-scale agent-based models utilize specialized tools, toolkits, or growth environments in accordance with the motives possessing to do with usability, simple to learn, cross platform ability, and the requirement for classy abilities to attach to the databases, GUI and GIS.

Conclusion

Possibly the toughest general difficulty in simulation is to determine the precise technique to employ for generating a model. Finally, how the starting point is found out? While the discipline of software engineering has come out to react for this question for software, generally, the modelers also come across these kinds of troubles: how does one engineer models? As there includes several modeling methods for simulation, one is frequently in uncertainty as to which model approach to utilize and under what circumstances it should be employed

References

Dapeng Liu, Qing Wang, Junchao Xiao (2009), The Role of Software Process Simulation Modeling in Software Risk Management: a Systematic Review, Third International Symposium on Empirical Software Engineering and Measurement, IEEE.

Miguel Wanderleya.B, Júlio Menezes Jr.A, Cristine Gusmãoa, Filipe Lima (2015), Proposal of risk management metrics for multiple project software development, Vol.64, pp. 1001 – 1009

http://mirabilisdesign.com/new/2015/10/10/modeling-of-software-process-using-visualsim-architect/

S J E Taylor, T Eldabi, G Riley, R J Paul, M Pidd, Simulation modelling is 50! Do we need a reality check?

Robert Maidstone (2012), Discrete Event Simulation, System Dynamics and Agent Based Simulation: Discussion and Comparison

John S. Carson II (2004), Introduction to Modeling and Simulation, Proceedings of the Winter Simulation Conference

N.N. Olenev, Modeling and Simulation Techniques, Systems Analysis And Modeling of Integrated World Systems - Vol. I.

http://home.ubalt.edu/ntsbarsh/simulation/sim.htm

Preeti, and Manju (2015), Modeling, Simulation and Optimization, Journal of Basic and Applied Engineering Research, Vol.2, No.14, pp.1168-1171,

R Ahmed, T Hall, Wernick, Robinson, and M Shah (2008), Software process simulation modelling: A survey of practice, Journal of simulation, Vol.2, pp.91-102.

He Zhang, Barbara Kitchenham, and Dietmar Pfahl (2008), Reflections on 10 Years of Software Process Simulation Modeling: A Systematic Review, Springer-Verlag Berlin Heidelberg.

Charles M. Macal. Michael J. North (2008), Agent-Based Modeling nd Simulation: Abms Examples, Proceedings of the 2008 Winter Simulation Conference.

Eva Cipi, Betim Cico (2011), Simulation of an Agent Based System Behavior in a Dynamic and Unpredicted Environment, World of Computer Science and Information Technology Journal, Vol. 1, No. 4, 172-176.

CM Macal1, and MJ North (2010), Tutorial on agent-based modelling and simulation, journal of simulation, Vol.4, No.3, pp.151-162.

https://www.theorsociety.com/Pages/ORMethods/AgentBasedModelling/AgentBasedModelling.aspx

Eric Bonabeau (2002), Agent-based modeling: Methods and techniques for simulating human systems, PNAS, Vol.99, No.3.