nuMedia group

2536 Kilmarnock Cres.
North Vancouver, B.C.
Canada V7J 2Z5
ring: 604.834.5801
www.numediagroup.com
email: joseph@numediagroup.com

nuMedia
group
a division of interactive publishing corporation

RANDOM GENERATOR PROJECT

a butterfly
stirring its
wings in
Beijing
can effect
the weather
in New York

Description of the Proposed Project:

Prime objective: To provide the first in a series of object libraries for the next generation of multimedia content development tools that emulate the laws governing the natural world.

There are three main components to the RG project. The Swarm simulations, the MiddleWare and the External Commands (hooks) to the target applications (i.e. Director, Cocoa, Java, mTropolis etc.) The Project will use Multiple Autonomous Agent Technology to create virtual environments that will automate the production and development process.

The science behind the project is based on Complex Adaptive Theory, where complex global behavior is generated by the interaction of simple local rules. (i.e. flocking behaviour, traffic, genetic reproduction, etc.) and will initially be built around simulation software, Swarm Software, being created by the Sante Fe Institute and used by research scientists[1] around the world. The tools will be a series of object libraries for the creation of graphics, animation and sound elements that can be streamed from a remote server and/or sent as real time scripts or code to client applications and languages.

The modelling formalism that Swarm adopts is a collection of independent agents interacting via discrete events.....Swarm is intended to be a useful tool for researchers in a variety of disciplines, especially artificial life. The basic architecture of Swarm is the simulation of collections of concurrently interacting agents: with this architecture, we can implement a large variety of agent based models."

The Sante Fe Institute

By extending the libraries available to Swarm software and by creating MiddleWare[2] connecting to the most prominent multimedia applications, Internet browsers and Internet specific languages we will be able to randomly generate the media elements over a switched network.

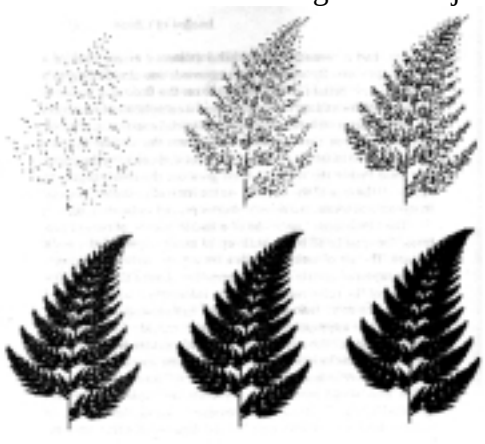
The goal is to create object oriented Virtual Worlds that run on a remote server which can be modified, added to and participated in by multiple end users. This research will lead to the capability of creating a series of Virtual Worlds that run for years and have initially hundreds, then thousands and ultimately millions of participants.

---

[1] Swarm is a software package for multi-agent simulation of complex systems being developed at The Santa Fe Institute and being used by NASA, MIT, the Brookings Institute, the US Military, the US Geological Survey and many others to enable computer simulation modelling in such diverse research fields as artificial life, genetics, financial structures, population growth , anthropology and many and varied other research topics in numerous disciplines around the world.

[2] MiddleWare is the software application that connects the Swarm(s) running on the remote Server to the client applications on the user's networked device.

A key concept underlying this project is that of using fractal algorithms to create the simple rules that determine the agents or objects behaviour. For example:



THE CHAOS GAME.
Each new point falls randomly,but gradually the image of a fern emerges. All the necessary information is encoded in a few simple rules.

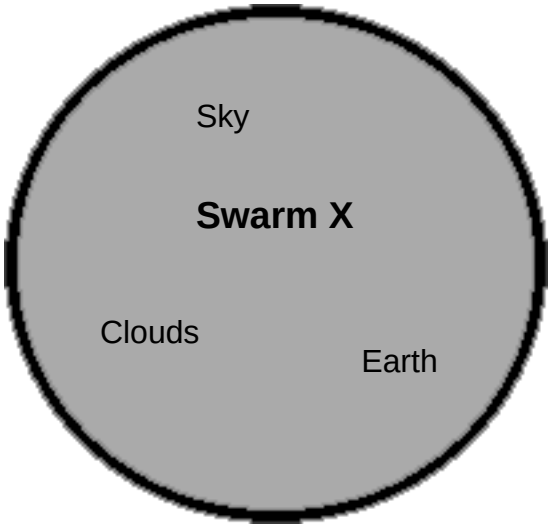
This is an example of how Michael Barnsley using what he called, "collage theorem" produced a fern from a simple fractal equation.

from: CHAOS, Making a New Science
by James Gleick, Penguin Books, 1987 pg. 238

The Random Generator Project

The project will first create a number of interacting Swarms (simulations) to create a larger Swarm consisting of numerous interacting agents[3]; sky, earth, grass, water, clouds, wind, light, shadow, temperature and a creature. This Swarm will provide the data that will be processed by the MiddleWare and sent as commands to Macromedia Director. Hooks (External Commands) will be written to Director so that the graphic, animation and sound information is pushed by the swarm and displayed at the client site. The end result is the graphic and animation production takes place on the fly.

In the first Swarm, Swarm X, there will be the interaction between the earth, the sky and clouds.



A Swarm is comprised of independent agents that interact via discrete events.

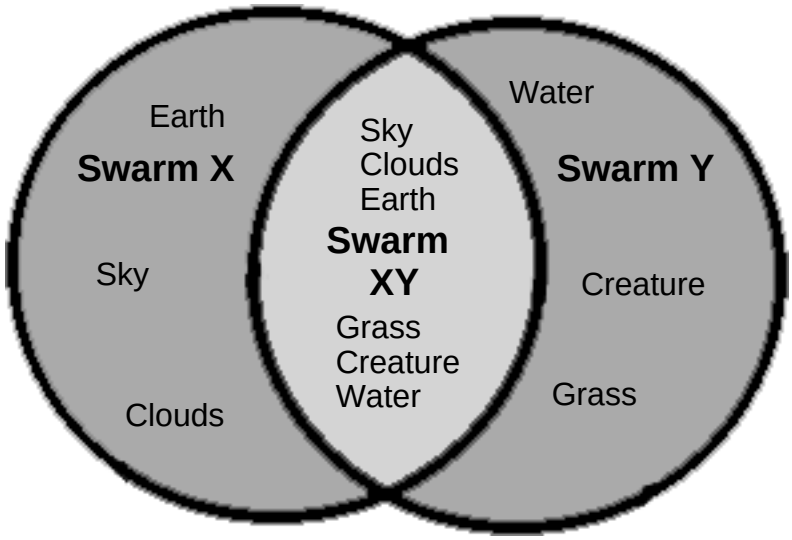
[3] What follows is a description of the approach that we will take to build the first iteration of the software. Besides the agent/objects that are included in this description we will create numerous other agents (i.e. birds, trees, ferns etc.) but for the sake of simplicity we have limited the number of agents in this description to illustrate the principal concept of the work to be done.

Each agent (or object) has a small number of rules to define it and its actions. For example, the rules for the earth, the sky and clouds would define size, shape, colour, texture, boundary, movement etc. It is the interaction of the various agents that creates the simulation. Swarms themselves can be autonomous agents within larger Swarms comprised of two or more Swarms.

In Swarm X the sky is the first agent and will give spatial dimension to this Swarm and subsequent Swarms. The earth will be the second agent and we will set rules to define its shape and dimension, including point of view (POV). The third agent for Swarm X is cloud. Clouds will have a variety of rules depending on the type of cloud.
The problem for our purposes is to determine the rule set which will define the individual agents. What are the rules, how complicated or how simple. Each agent will require an experimental process to determine the minimal rule set to define the individual agent. As well, the rule set will have to account for the interaction of other agents and Swarms.

For example, there are two possible approaches; we can set definite rules that define the agents sky and earth relationally or we can arbitrarily set the boundaries of the sky as the display coordinates of a computer screen (640x480, 600x800 etc.) and have the earth boundaries definable within the sky. Ultimately we want to explicitly represent an emergent structure that acts cohesively as a single agent but if we use arbitrary rules that limit the agent(s) there is the possibility that we will not be able to create a realistic simulation environment. There is also the possibility that if we use real world mathematical rules for agent definition and behaviour we will end up with too much information to deal with and thus not be able to communicate with the clients effectively.

We therefore have to build the first Swarms using both approaches to determine which will be the best approach. It may be that a combination of approaches will give us the desired result and that can only be determined through experimentation.



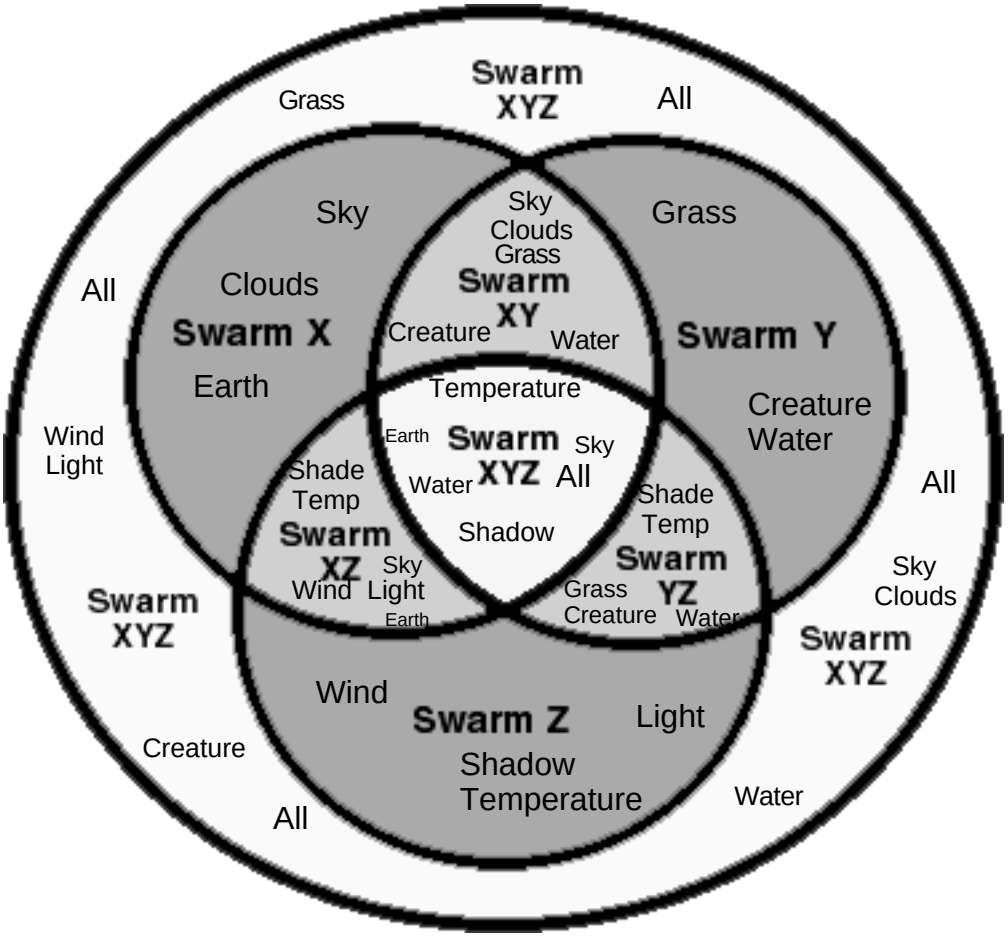
Individual Swarms (X&Y) combine to form a larger Swarm (XY)

The individual agents can generate events that affect itself and other agents. Further, the Swarms as agents can affect the agents within itself, within a connected Swarm as well as the new Swarm created by the interaction of the two independent Swarms. The simulation runs when a schedule of events is given to the various agents. This in effect creates the representation of time within the simulation. The unique capability of this form of modeling is that through the combination of multiple Swarms one can model agents and/or Swarms

that can build or destroy themselves thus creating a 'living' environment ---- A Virtual World.

Object Oriented Technology

The Swarms and subsequent libraries will be written in Objective C and C++. Individual agents will generally be of a generic class and be given individuality through their instance variables. In the example of clouds, there will be a generic cloud but one can create individual instances of clouds by setting the individual parameters. (i.e. Cumulus Nimbus, Cirrus etc.) The design approach will be such that we will create modifiers using inheritance so that one can set the 'look and feel' of the agents thus enabling the author/artist to design these objects to their own specifications.



Multiple Swarms form one larger Swarm (XYZ) where the multiple agents & Swarms interact.

The Scene (putting it all together)

The RG Project will create a series of interacting Swarms, collectively we are calling them: "The Scene".

We have chosen a simple scene to animate as it is the underlying rules that will provide the capability for generating specific agents or objects regardless of their size, shape, colour etc. For example, the agent for the earth resides in one swarm and the agent for grass in another; the rule governing the placement of grass requires that it only 'grow' on the earth. The grass swarm defines how it can move but it is the wind swarm that governs the effect. Once we have the underlying rules set we can experiment with different agents, i.e. tall grass, short grass; cumulus cloud, cirrus cloud. Boundary instances will be defined using fractal geometry, the details will be random but the fractal dimension will be constant.

Visible Agents

A: Sky

B: Clouds

C: Grass

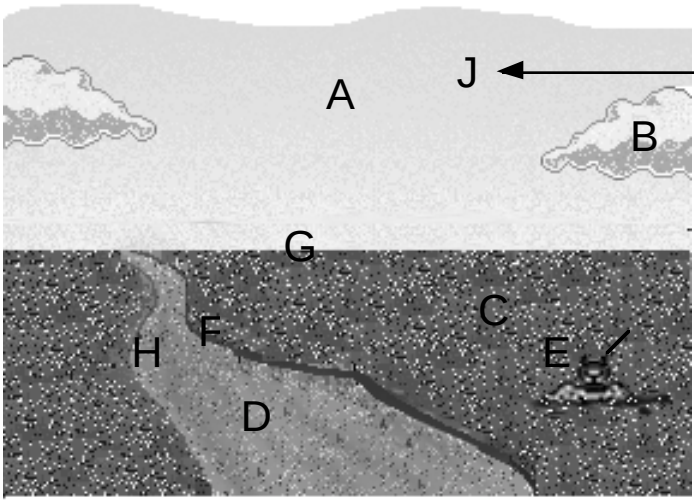
D: Brook

E: Creature

F: Land/Water boundary

G: Sky/Land boundary

H: Water/Land boundary



The Scene

Invisible Agents

J: Wind

K: Light

M: Shadow

N: Temperature

O: Earth

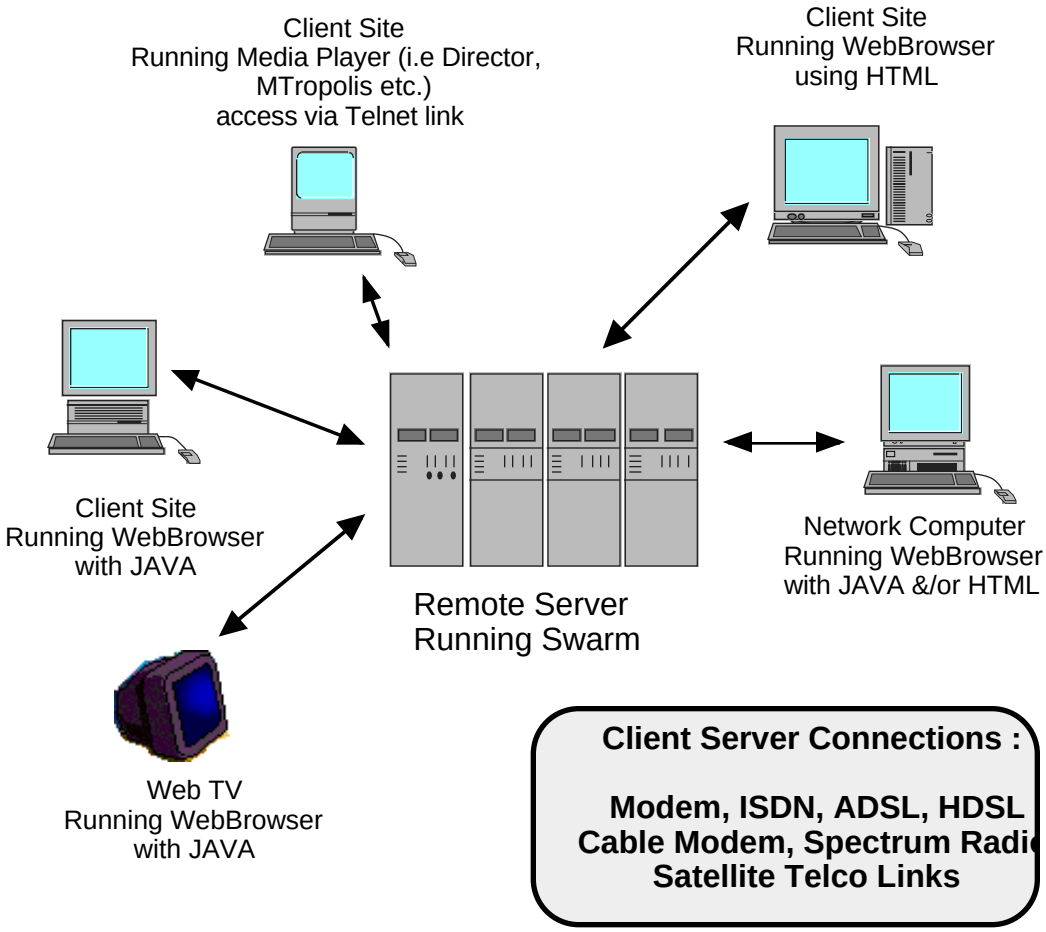
Multiple Swarms are translated into an animated scene on the fly via MiddleWare.

MiddleWare

Once we have created the component Swarms that comprise "The Scene" and set their schedules of events to run the simulation we will then use a feature of Swarm called the "probe" facility to observe, track and record the actions of the individual agents. From this data collection we will then interface with Director through a MiddleWare application which will be written in Visual Basic and C++.

As opposed to the timeline[4] design and production techniques that predominate in multimedia production today, the random generator library will be a series of independent but interactive agents that 'live' within the computer simulation environment that resides remotely on the host server and 'feeds' the simulation data to the graphic, animated (and ultimately video) and sound elements at the targeted client side. Ultimately these object libraries will be large enough to generate full motion animated worlds and later hybrid animation/video worlds using technologies like VRML etc. The possibility of each and every user having a different swarm running is simply a matter of computational power and bandwidth.

Client/Server & Bandwidth



The Swarms will run on a Linux Server with the client running the media player or browser.

It is a truism that the more bandwidth the better but some of the new product offerings by the telco, satellite and cable companies provide sufficient bandwidth to deliver compelling multimedia content. Combine that with the ever increasing speed and performance of chip

[4] i.e. the score in Macromedia Director

technology and you have a delivery mechanism for distributed multimedia. Although WebTV is around the corner and will drive the demand for this type of technology there is still the capability for lower bandwidth content designs accessible over modem connections, Maybe North America, Europe, Japan etc. will have broadband to the curb but the ROW market may be using modems for some time to come. This technology, though, is best suited to multi megabit switched networks.

Risk

The greatest technological risk is not so much the question of can we randomly generate the media elements on the fly but can we create a software environment that can do it over a specified bandwidth and to a target CPU running at X megahertz. We may find that the target CPU must be running at 700MHZ in order for the software to work, in which case this will delay the commercialization of the consumer products. We would then have to modify our approach and target higher end machines used by the larger multimedia production companies as well as the 3-D rendering houses in the film industry while we are waiting for the consumer market to catch up. Moore's law will play a factor in this, we expect the machines suitable for this level of processing to be on the market by the time we come to market.

There are numerous risks attached to each and every agent that we create, how complicated or simple should the rules be for the agents, if the aggregate of rules of the interacting objects requires too much processing power the whole system could bog down and fail. We will have to discover the limits of the system and then work towards creating "cheats" that will approximate a real world simulation. A combination of fuzzy logic and generational AI software (used in robotics where the AI software learns from its past experiences) will enable us to modify and correct some of these inherent difficulties.

There are market and financial risks associated with the project. If we find that by the time the first iteration of the software is complete that it is only deployable on higher end machines then we will have to raise more equity capital to reposition the software and our marketing approach. Having this in mind we will design certain of the agents (i.e clouds, grass, etc.,) so that we can create very specific agent production software that can be sold to the development houses.