INTERACTIVE COMPLEX SYSTEMS SIMULATION

William Bricken
Chris Langton *Los Alamos National Laboratory*, *Santa Fe Institute*
Bruce Sawhill *Santa Fe Institute*
June 1993

Statement of Purpose

The focus of the proposed research is non-linear computational environments (complex and chaotic systems which include a human participant), the new interactivity paradigms that they offer, the underlying mathematical and interface theories, and the interactive experiences available to human participants in complex computational environments. The title Interactive Complex Systems Simulation (ICSS) was chosen to differentiate this research effort from projects at other institutions which concentrate either on purely computational issues or on purely human interface issues. We feel it is necessary and desirable to advance the state of the art on both fronts simultaneously in order to take full advantage of new computational technology and new metaphors for the generation and comprehension of information.

1 Introduction

The advent of electronic computers has brought about a revolution in all areas of science and engineering, and has opened up the possibility for scientific investigations and technological accomplishments of a wholly new kind. Computers have permitted the in-depth study and modeling of systems of great complexity, such as stellar and galactic dynamics, atmospheric processes, biological cells, brains, weather, the human immune system, ecologies, and economies. The importance of understanding such systems is enormous: many of the most serious challenges facing humanity (e.g., environmental sustainability, economic stability, the control of disease), as well as many of today's most difficult scientific questions (such as the nature of intelligence, the origin of life, and computational modeling of biological systems) will require a deep understanding of complex systems. Computers have also provided the ability to address previously intractable practical problems such as large-scale combinatorial optimization, the automatic analysis of complex data, the simulation of environments not accessible by experimental means, and the creation of autonomous learning systems, all of which will have tremendous significance for science and technology.

As research in such areas has progressed, the need for increasingly powerful and sophisticated mathematical models, and their implementation as computational systems, has become paramount. The recent development of massively parallel computers has great potential for addressing these problems. However, powerful hardware is rarely sufficient by itself for making significant progress on the types of problems listed above. At present the primary bottleneck lies in the creation of new computational methods (algorithms, interfaces, analysis tools) that fit these problems in a more natural way than do traditional computational and mathematical methods.

The central research issue is to develop a computational infrastructure that supports the complexity of non-linear modeling while providing a seamless interface to the human attempting to understand the inherent complexity of global systems.

Truly complex systems are characterized by irreducibility, which means that traditional analytic mathematical techniques cannot be used to reduce a system to its constituent parts and still preserve the essential character of the system. Complex systems consist of many parts that behave differently in each other's presence than they do alone, hence invalidating the "divide and conquer" approach that has been so successful in fields such as particle physics. One might consider a single hydrogen atom in isolation and obtain useful insight about the behavior of hydrogen gas, but it is very difficult to consider a single cell in isolation and to obtain useful insight about the dynamics of life, ecology, and evolution. To understand a complex system, we must keep track of the net of interactions that tie the system together, a task that is theoretically well proscribed but practically very difficult.

Before the existence of cheap and powerful computers, such complex systems were simply mathematically intractable. Computers have changed the situation completely by providing the capability for the *synthesis*, as well as the analysis, of complex behavior. When complex behaviors are analyzed in the traditional sense, they are decomposed into individual parts, but in doing so, the interesting behavior vanishes, because it is inherent in the interactions *among* the parts. A tractable way to study complex behavior is to synthesize it, to put all of the parts and their interactions together and let the collective behavior emerge from the resultant dynamics. This synthesis is an inherently parallel activity, in which every part is updating itself simultaneously with all of the others.

Mathematical simulation is not, in itself, sufficient for our comprehension of complexity. In general, complex phenomena are extremely difficult to understand. The synthetic techniques of non-linear modeling must also incorporate the human in the synthesis, they must include a new participatory paradigm for interface. Complex environments must present information dynamics to the human in a manner which calls upon the native, intuitive capabilities of our entire sensorium. Non-linear information is simply too complex to be understood solely by intellectual abstraction. In particular, we must call upon visualization, audio cueing, kinesthetic feedback, and the full dynamics of *participation within the complex environment* in order to develop a deep understanding of irreducible systems.

Computational modeling of global economies, of weather systems, of social organizations requires *both* high performance parallel computing and high bandwidth participatory interface.

What is required, then, is a methodology that:

- can be implemented to take advantage of massive parallel processing,

- efficiently navigates through enormous, multidimensional solution spaces using intelligent pattern-recognition abilities to identify classes of satisficing solutions,

- permits complex behavior to emerge from the interaction of simple parts (rather than being laboriously and most often inadequately preprogrammed),

- is capable of self-adaptation in order to optimize performance with respect to different users and to different problems, and

- provides an extraordinarily wide bandwidth of interaction with human users, to the extent that data becomes experience.

A number of new approaches to computation that have been developed in recent years, inspired by complex problems as diverse as understanding the reaction of the human immune system to disease, piloting a supersonic aircraft under enemy fire without looking at the controls, and developing models of world resource usage. Neural networks, genetic algorithms, lattice gas methods, simulated annealing, and virtual reality are examples of such novel approaches. These approaches have been used both as models of natural phenomena and as nature-inspired methods for solving practical problems. Since new methodologies provide a contrast to traditional computational methods, they lead to a broadening of traditional notions of how information processing can take place.

Innovation in information processing methodologies also leads to an entirely new set of challenges for creating, managing, and interpreting information itself. As computational power increases, we will need to integrate theories of understanding and of experience into our computational methodologies.

In a broad view, this proposal addresses the problem of human interaction with the machines of the future. How will we understand information when it is processed by emergent algorithms, in incomprehensible volume, over inconceivably short times, blind to physical distance, and shifted through the narrow window of a display device?

We suggest an innovative computational architecture, the Process Gas, coupled to an intuitive environment, Virtual Reality, as a testbed for studying interactive complex systems simulation. We then propose five applications to aid in the iterative refinement of this architecture and to provide immediate benefit from the infrastructure development.

2 Interactive Complex Systems Simulation (ICSS)

2.1 Complex Systems and the SFI

The charter of the Santa Fe Institute (SFI) is the study of Complex Systems. SFI has become one of the world's leading institutions for the study of complex systems, attracting researchers from all over the world to work in its interdisciplinary atmosphere and to create new tools and new metaphors to manipulate and describe the common phenomena that underlie different complex systems. The current research programs at the SFI include adaptive computation, theoretical biology, theoretical immunology, economics, artificial life, and time-series forecasting. While each program sponsors research on a particular, specialized topic, there is a great deal of interaction among the programs, as well as mutual feedback between research on specific topics and research on general questions about complex systems. Fostering such interactions is a primary purpose of the Institute.

These interactions provide a means by which commonalities among the various disciplines can be discovered. Ideas and techniques from one discipline (e.g., economics) can often be successfully exploited in another discipline (e.g., artificial intelligence), and these interactions often produce wholly new ideas and insights. The SFI has been the catalyst for a number of very non-traditional collaborations, for example, between economists and physicists applying new results from dynamical systems theory to economic systems; between economists and artificial intelligence researchers applying ideas from economics to machine learning systems; between computer scientists and physicists, applying concepts of formal language theory to complex many-body systems; among computer scientists, biologists, and operations-research scientists studying optimization in computational and in biological contexts; and among scientists from many disciplines bringing different techniques to bear on the problem of economic, social, and environmental sustainability. The important and novel results growing out of these collaborations are attracting more and more interest to the SFI, from academia, industry, and education, as well as from the popular media.

Though the SFI studies many different kinds of complex systems, we have sought universal principles underlying such diverse "complex" phenomena as national economies and mammalian immune systems. Since the founding of the Institute in 1984, we have come to recognize the existence of a number of such universal principles, but none more striking than that many complex phenomena share a common architecture. This commonality is based upon the observation that similar dynamical phenomena arise in many fields, fields which on the surface appear to be completely unrelated, such as transitions to chaos, evolution by punctuated equilibria, economic cycles of prosperity, scaling laws, etc.

The vast majority of the complex natural phenomena we study are characterized by having large numbers of different varieties of "agents" (objects capable of storing, processing, and transmitting information) interacting with and modifying each other, with no single agent having complete knowledge of all of the rest of the agents. These agents are, in general, semi-autonomous and relatively simple. Their local, nonlinear interactions with one another cause the entire collection to self-organize, to develop structures and characteristics at the aggregate level that were not explicit in the individual agents. These agents might be firms in an economy or antibodies in the immune system or quantum states in a lattice of atoms.

Thus, to obtain a practical quantitative understanding of a particular complex system, we must first construct a computational simulation that abstracts the common unifying architecture across complex systems. We can then constrain the simulation to the particular case and observe the resultant phenomena.

It is obvious that such simulations could not practically be constructed until powerful parallel computer technology became cheap and available. This construction of simulations is often called *experimental mathematics*, except that it is not an experiment in the traditional sense of the particle accelerator or the Petri dish, but rather one that takes place inside a computer's core. It is synthetic mathematics in which analytic simplification is not the ultimate goal, as it is in the traditional field of mathematical analysis. Often these experiments deal with systems that are not available for systematic physical experimentation because of their uniqueness, such as the Earth's ecology or the American economy. In these cases, simulation provides the only avenue for quantitative understanding of observed behavior.

Therefore the ability to synthesize behavior is essential to develop an understanding of complex systems, and the only practical way to achieve this synthesis is through computer simulation. The particular form of computer simulation that is required is a direct consequence of the common architecture shared by most complex systems: they are *large collections of semi-autonomous agents interacting with one another in the context of a shared environment*. The ideal realization of complex systems architecture on a computer would be a parallel object oriented programming language running transparently on a massively parallel computer. One could view such a programming system as a "gas" of computational processes interacting with one another within an "aether" of processors. In fact, we refer to our version of this computational vision as a *Process Gas* simulation system.

2.2 Virtual Reality

Virtual reality (VR) applies advances in computing speed, graphics display, sensor sciences, and programming techniques to immerse a participant within a computer generated space called a *virtual world.* Virtual worlds may or may not be analogues of physical space; virtual environments can be realistic, extra-sensory, telepresent, abstract, or hybrid.

The sensation of immersion is accomplished by peripheral devices such as head-mounted visual and auditory displays with motion trackers. The impression of an environment is generated by a model of a three dimensional world populated by objects obeying their local behavioral rules, within the context of a physics governing the space which embeds both objects and participants. A participant can move around the virtual environment, while interacting with other virtual objects. One can just as easily play virtual tennis with a partner as move around inside an abstract representation of HIV epidemiology data. There are many applications of such virtual worlds, but the most important aspect of the technology is the vastly improved interface between human and computer.

The computational implementation of a VR system is a significant departure from previous approaches to computation. The need to effectively construct and manage a environment populated by a large number of entities and an arbitrary number of human participants puts heretofore intractable demands on an operating system. A VR operating system must operate in parallel on a variety of platforms distributed over networks, while integrating internal information with input generated by human participants consistently and in real-time.

In order to deal with the computational requirements of VR, we have developed a first-generation version of an operating system (the Virtual Environment Operating System, VEOS) that addresses most of the computing environment challenges of virtual reality. In addition to software development, we are actively developing new interface technologies including tools for navigation of virtual spaces, for representation of human physiology in the virtual body, for mapping sensor inputs into arbitrary outputs, for modeling of behaviors, for embedding narrative and dramatic tension into a series of interactions, for rapid construction of virtual environments, for integration of MIDI sound and voice recognition capabilities, and for programming semi-autonomous entities.

As the technology of VR becomes more sophisticated, it becomes conceivable and desirable to use it as a scientific research tool. While current VR research is primarily concerned with constructing worlds that simulate real three-dimensional physical environments, there is no significant technological hurdle to using the technology to explore abstract representational spaces such as might be encountered in complex systems simulations. One could "enter into" a computer simulation, interact with it, modify it, create appropriate tools for the acquisition of data, and then take data as one would in a normal scientific laboratory. The introduction of such a powerful tool could change the basic way that science is done in the 21st century, and do for complex systems research what the microscope did for biology or the telescope for astronomy.

2.3 The ICSS Program

One of the challenges facing the SFI is that it has become impractical in terms of manpower and resource costs to write special purpose software for each simulation task, a fact which makes a goal of a "general simulation engine" based on the Process Gas architecture reasonable and desirable. Perhaps not surprisingly, many different disciplines concerned with the *generation* of complex phenomena are converging on this same computational architecture. In particular, researchers in branches of the computer sciences (such as AI, operating systems, computer networks, databases, computer animation, and virtual reality) share a common vision that computation in the future will be implemented as a parallel, distributed aggregate of computational processes interacting with one another concurrently in the context of a shared environment. Because of this shared vision of the future of computation, there are many ongoing research efforts intended to bring such a vision into reality.

Most of the groups implementing parallel object-oriented systems are focusing too narrowly on specific engineering goals, while not paying enough attention to the requirements for a truly general purpose implementation of the computational architecture we all seek. The SFI has found it necessary to become actively involved in a major research and development effort in order to ensure the development of a Process Gas computational implementation that will be useful for the general scientific investigation of complex systems. Furthermore, the SFI's understanding of the workings of complex systems is of critical importance to the achievement of some of the research and engineering goals that are motivating these different implementation efforts.

The SFI has carefully reviewed the various research efforts aimed at realizing distributed computational architectures, concluding that two research efforts are of direct relevance: *Virtual Reality* and *Distributed Object-Oriented Operating Systems*. The unification of these two topics is best represented by the VEOS Project. The computational architecture of the *Virtual Environment Operating System* (VEOS) is remarkably similar to that of the SFI.

Many of the technical challenges in constructing a general purpose virtual environment are the same problems facing the SFI in constructing a general purpose complex systems simulation engine. A virtual environment might consist of many different agents, all of which have different rules and constraints for interacting with each other, just as a simulated complex system has to allow for many different agents and their interactions. Furthermore, a virtual environment has to be able to be continuously modified "on the fly", since it must accommodate human interaction in real time. To be able to modify a complex simulation in real time would vastly increase the efficiency of exploring possible dynamics; i.e., to experiment with a complex system in a natural and intuitive way. A VR system must also be computationally portable so that worlds can be run on many different types of machines and concurrently accommodate several distributed human participants, just as a simulation engine must be able to make use of a wide variety of available resources and be accessible to multiple participants and analysis programs.

The VEOS system satisfies the requirements of a general simulation engine. VEOS is designed to manage the maintenance and interactions of all of the entities in a virtual world and to allow a human participant to interact with them; its operative metaphor is distributed object oriented computation. Its architecture is very similar to that of the Process Gas, and in fact both ideas can trace their intellectual heritage to early computational problems encountered in constructing artificial ecologies and artificial life.

At the SFI, the emphasis thus far has been on creating accurate simulations rather than on interacting with them. As the simulations become more complex, the issue of interaction changes from a point of convenience to an essential part of the scientific method. Without human interaction, simulations tend to become meaningless since it is impossible to ascertain salient results because of the information overload generated by the simulation. Conventional methods for interacting with computers are beginning to prove inadequate for the task at hand, for presentation of massive amounts of data in a form comprehendible to humans. Many simulations and data sets exist in a large number of dimensions, making it difficult to explore them with a flat screen and a keyboard. Often simulations have a large number of adjustable parameters, and to obtain a qualitative understanding of the phenomena being generated requires "twiddling the knobs", an impractical exercise if a large computer simulation has to be stopped, modified, and recompiled thousands of times. The bandwidth of communication between simulation and participant must be increased, through adding additional sensory modalities for output, through the provision of real-time interaction with the simulation, and by including the participant dynamically and spatially within the simulation.

2.4 The Proposal

The SFI is concerned with *synthesizing* accurate simulations of complex systems and VEOS is concerned with *interacting* with complex generated simulations. The wealth of knowledge at the SFI about the workings of complex systems will aid significantly in the construction of complex virtual realities, while the vastly enhanced capabilities for scientific visualization and interactive exploration of such artificial worlds will greatly enhance the SFI's ability to synthesize and experiment with the dynamics of complex systems.

We therefore propose to unify the two approaches into a powerful cognitive tool with the potential to transform the role of the computer in scientific research. This collaboration, which shall be called the Interactive Complex Systems Simulation (ICSS) program, will have three main research thrusts which will be described in greater detail in the next section of this proposal.

1. The implementation of the Process Gas model using the VEOS architecture.

2. The creation of a General Simulation Engine by applying a Virtual Reality interface to the Process Gas implementation.

3. Application of the General Simulation Engine to outstanding problems in complexity, and in particular to Global Sustainability.

3. Proposals for Research Projects

In this section we present five proposals for research under the ICSS program that span these three areas. The topics of these proposals are:

1. Development of a General Simulation Engine using the Process Gas model, the VEOS architecture, and VR interaction techniques.

2. Installation of the Process Gas on the CM-5 at LANL.

3. Installation of the Process Gas on a KSR.

4. Application of the General Simulation Engine to simulation of Global Information Systems

5. Application of the General Simulation Engine to modeling scenarios of global sustainability and global decision making, in conjunction with the Global Sustainability Program at SFI.

3.1 Research Proposals

Proposal 1: The General Simulation Engine

In order to most effectively pursue the study of complex systems, we need to abstract out their common underlying architecture--a distributed set of autonomous agents interacting in parallel in the context of an environment--and realize it in a general purpose simulation tool, ideally on a parallel computer. We refer to this computational architecture as a *Process Gas*, because it resembles a "gas" consisting of a large collection of computer processes that move around and interact with one another within an "aether" of processors.

A substantive amount of work has been done identifying the computational issues involved in the process gas model. We have implemented, for example, a couple of specific (not general-purpose) simulations on a massively parallel 64,000 processor Connection Machine, built by Thinking Machines Corp. These hard-wired process gas prototypes have helped to identify many of the properties we would like to see in a general purpose version. For instance, one not only needs a general purpose utility for implementing a multi-agent system, but one also needs tools for analyzing the resulting behavior; creating, editing, and saving initial states of such a system; setting and altering parameters of the system interactively; altering the rules for individual agents interactively; displaying the behavior of the system if appropriate, and so forth. Our plan is to construct a General Simulation Engine, containing these tools, that would be generally applicable to any complex-systems modeling project. Such a system would be invaluable to the modeling efforts within the Adaptive Computation program at the SFI, as well as in other programs, such as the Artificial Life program, the Theoretical Biology program, and the Economics program.

There are a growing number of other groups that have recognized the need for such a simulation system and are in the initial stages of developing one. A great deal of effort would be saved by pooling our efforts in the construction of a common simulation utility that would serve the needs of the entire research community. The initial list of interested participants includes

- Apple Computer Corporation's Advanced Technology Group, in particular, the Classroom of Tomorrow project,

- Jean Jouis Denoubourg's group in Brussels, who are working on a simulation system for studying insect colony dynamics,

- Orbital Sciences Corporation, who are working on an operating system to manage computer processes running on, and moving among, several hundred satellites,

- Los Alamos National Laboratory, in particular, the Lattice Gas group in the Theoretical Division, who are working on computational models of fluid dynamics,

- UCLA, in particular, the Artificial Life group of the Computer Science and Biology Departments, who are working on models of insect colonies, population genetics, and ecological dynamics, and

- MIT, in particular, the Media Lab and the Mobile Robot Lab, who are working on managing the collective dynamics of large ensembles of real and simulated mobile robots.

The Virtual Environment Operating System is proposed as the initial model for the General Simulation Engine. The initial version of VEOS was stable in February 1990, with capabilities of managing and displaying multiple participant virtual realities running over a distributed network of UNIX platforms. The current VEOS 2.2 includes a programming interface, real-time interactive editing of the virtual environment, a sensor library, and a virtual body which perceives the world between 20 and 30 frames per second. The initial step in constructing the General Simulation Engine will be to adapt the VEOS system to the Process Gas architecture, with special emphasis on parallel computation.

Virtual Environment Operating System: Design

The VEOS project is the responsibility of Dr. William Bricken and a team of half-a-dozen graduate students led by Geoffrey Coco. VEOS is designed to integrate the diverse components of a virtual environment.

VEOS consists of several software subsystems.

> The *kernel* manages processes, memory, and communication.
>
> The *entity interface* permits modeling objects in the environment, and the environment itself, in a consistent, object-oriented manner.
>
> The *interaction tools* empower a participant within the virtual environment.

As a research vehicle, VEOS emphasizes functionality at the expense of performance. We believe that code is best improved after it is functioning. Since a research prototype must prepare us for the future, VEOS is designed to be as generic as possible; it places very little mechanism in the way of exploring diverse and unexpected design options.

A VR system is far too ambitious an undertaking to begin from scratch. We have conceptualized VEOS as primarily a synthesis of known and understood techniques. The VEOS team has assembled disparate software ideas into a tightly integrated system with new functionality.

VR is characterized by a rapid generation of applications ideas; it is the potential of VR that people find exciting. However, complex VR systems take too much time to reconfigure. VEOS was designed for rapid prototyping, a feature that makes it a natural choice as the operating system for the General Simulation Engine. The VEOS interface is interactive, so that a programmer can enter a new command or world state at the terminal, and on the next frame update, the virtual world display will change.

Any real-time interactive system requires immediate accessibility. VR systems must avoid hardwired configurations, because a participant in the virtual world is free to engage in almost any behavior. For this reason, VEOS is reactive, it permits the world to respond immediately to the participant (and the programmer).

The broad-bandwidth display and the multisensory interaction of VR systems create severe demands for sensor integration. Visual, audio, tactile, and kinesthetic display require the VR database to handle multiple data formats and massive data transactions. Position sensors, voice recognition, and high dimensional input devices overload traditional serial input ports. A VR i/o architecture must incorporate asynchronous communication between dedicated device processors in a distributed computational environment. The database must accommodate updates from multiple processors. In VEOS, we have adopted a communication model which cleanly partitions communication between processes from the computational threads within a process.

The characteristics of the virtual world impose several design considerations and performance requirements on a VR system. The design of the virtual world could readily overwhelm a programmer if the programmer were responsible for all objects and interactions. Virtual worlds are simply too complex for monolithic programming. Entities within the virtual world must be modular and self-contained. The designer should be able to conceive of an entity, say an artificial ant, independent of all other aspects of the world. VEOS is structured so that each entity is designed to be independent and autonomous. The system itself takes care of the lower level details of inter-entity communication, coordination, and data management.

In VEOS, all entities are organizationally identical. Only their structure, or internal detail, differs. This means that a designer needs only one metaphor, the *entity*, for developing all aspects of the world. Changing the graphical image, or the behavioral rules, or even the attached sensors, is a modular activity. Entity modularity is particularly important when one recognizes that hardware sensors, displays, and computational resources are themselves first class entities. The entity model provides integration modularity for any new components to the VR system, whether they are graphical images, added CPUs, or new input devices. Entities can be run independently, as worlds in themselves, or they can be combined into complex worlds. This means that devices and models can be tested and debugged modularly.

Because entities consist of both data and operating system processes, an entity can use other software modules available within the larger operating system. An entity could, for instance, initiate and call a statistical analysis package to analyze the content of its memory for recurrent patterns. The capability of entities to link to other systems software make VEOS particularly appealing as a software testing and integration environment.

Entity autonomy is best implemented by assigning a separate processor to each entity. This approach makes VEOS essentially a distributed operating system. Distributed resources arise naturally in VR, since the virtual environment is a social place, accommodating multiple concurrent participants.

When more than one person inhabits a virtual world, the perspective of each participant is different. This can be reflected by different views on the same graphical database. But in the virtual world, divergent perspectives can be embodied in divergent databases as well as divergent viewpoints. That each participant can occupy a unique, personalized worlds makes VR essentially different than physical reality.

In summary, VEOS is a significant effort to provide transparent low-level database, process, and communications management for arbitrary sensor suites, software resources, and virtual world designs. VEOS is the glue under VR. As such, it provides a strong integration environment for any team wishing to construct, experiment with, and extend VR systems in particular, and simulation engines in general.

Virtual Environment Operating System: Implementation

The VEOS kernel consists of four tightly integrated components:

1. TALK manages interprocess communications.

2. SHELL manages entity initialization and links distributed resources.

3. NANCY manages the distributed parallel database.

4. FERN manages entity processes.

VEOS 2.2 also incorporates several interface utilities, including:

- LISP and C programming languages,
- VOGL, a public domain graphics library,
- MIDI interface and rendering entities,
- Imager interface and light rendering entity,

- Earmager interface and sound rendering entities,
- Sensor library and driver entities,
- Voice recognition driver entity,
- Mouse, spaceball, and other device entities, and
- UM, an entity specification tool.

VEOS is currently being used in several world construction projects. A central advantage of VEOS is that the insertion of a declaration into the database during runtime will change the display environment immediately. VEOS provides real-time interaction with the virtual world, which permits very rapid prototyping of world designs and construction of virtual objects while an application is running.

All operations generated by a VEOS process are implemented by a single computational algorithm: match-and-substitute. Elements which are have a complex syntactic description (such as "3x + 4x") are identified by syntactic pattern matching. Then a simpler expression (7x in the example) is substituted for the complex one. This methodology is commonly called algebraic (substitution of equals for equals).

Algebra provides a particularly appealing mathematical model for a computational engine, for several good reasons:

1) The technique is common and well understood.

2) Equations from any standard textbook can be programmed directly by copying. For example, to implement a concept of force, objects are first assigned mass and acceleration properties. Then the force an object exerts in the direction of its acceleration is identified simply by asserting the equation

$$\text{Force} = \text{Mass} * \text{Acceleration}$$

into the database for that object. By simple pattern matching and substitution, when Mass = 5, the Force equation is constrained by the partial information,

$$\text{Force} = 5 * \text{Acceleration}$$

3) In contrast to logical deduction and data-driven programming, equations are bi-directional and constraint-based. The computational process does not need to wait for the values of every variable to be known in order to make progress. When we have partial knowledge (Force < 25, for example), match-and-substitute will generate

$$25 > 5 * \text{Acceleration}$$

which simplifies by another substitution to

$$Acceleration < 5$$

The partial knowledge automatically generates a constraint on the value of Acceleration, it is never greater than 5.

VEOS must also address multiple data transactions for multiple active entities and participants. To manage the coordination of interprocess communication, VEOS uses a variant of the Linda parallel database model developed by Gelernter. In Linda-like languages, communication and process are treated as independent, relieving the programmer from having to choreograph multiple processes.

Structurally, an entity database consists of a collection of fragments of information, labeled with unique syntactic identifiers. Collections of related data (such as all of the current properties of cube-3) can be rapidly assembled by invoking a parallel pattern match on the syntactic label which identifies the sought after relation (in the example, matching all fragments with the label “cube-3” creates the complete object known as cube-3).

In object-oriented programming, object attributes and inheritance hierarchies commonly must be constructed by the programmer in advance. Efficiency in object-oriented systems requires compiling objects. This means that the programmer must know in advance all the objects in the environment and all their interactions. In effect, the programmer must be a god. Virtual worlds are simply too complex for such monolithic programming. Although object-oriented approaches provide modularity, in large scale applications they result in complex property and method variants, generating hundreds of object classes and forming a complex inheritance web. In many cases, a principled inheritance hierarchy is not available, forcing the programmer to limit the conceptualization of the world. In other cases, the computational interaction between objects is context dependent, requiring attribute structures which have not been preprogrammed. Attributes can be generated interactively and related structures of objects can be identified based on arbitrary pattern structures, such as partial similarities, unbound attribute values (i.e. abstract objects), and ranges of attribute values.

In summary, our mathematical computational model is parallel algebraic pattern-matching and substitution on partitioned databases which support multiple concurrent interactions.

Entities

Entities are the primary organizational structure within VEOS. Entities provide a uniform, singular metaphor and design philosophy for the organization of software resources and displays.

An entity is a collection of resources that can accomplish a specific task. VEOS itself is the prototype entity.

The internal process of an entity consists of a sense-process-act loop. This loop is called a *behavioral cycle*.

The analogy to human behavior is not strong. An entity senses by gathering relevant data from the database. Relevance is the key. Each entity has a filter on the database which limits the amount and the type of data that the entity must process on each behavioral cycle. Filters are patterns which must be matched for a data fragment to be relevant to the entity. For example, an entity configured with a sound play-back capability, can sense any sound bits stored in the database. If it lacks a sound capability, it will ignore all data labeled as sound.

The processes internal to an entity are controlled by two separate processing loops. The React loop reads sensed data and immediately responds by posting modified data to the common database. This cycle handles all real-time interactions and all reactions which do not require additional computation or local storage. The Persist loop runs on a clock local to the specific entity, and is not responsive in real-time. Persist loop computations typically require local memory, function evaluation, and inference over local data. Persist functions can copy data for the shared database and perform local computations in order to gather information, but there is no time constraint on returning results. By decoupling local computation from environmental reactivity, entities can respond in a timely manner while complex responses can still be evaluated whenever computational resources permit.

The entity editor (not yet fully implemented) will provide templates for standard objects and their hierarchies. Entities can be modeled statically in any modeling system which supports standard file formats such as DXF, PICT and Postscript. Of course objects designed for a three dimensional environment require 3D specifications, but 2D objects can easily be incorporated into three dimensional virtual spaces, and three dimensional scenes can be projected into two dimensions for display on screens.

Entity dynamics is achieved by associating behavior functions with sensory input and with internal processes. Functional actions are associated with sensed and internally generated data through rules.

The *space entity* manages field effects and environmental influences. By partitioning a virtual world into objects and spaces, computational load can be carefully balanced. Every VEOS entity has a spatial aspect, so that each entity not only manages its own behavior, it also serves to coordinate interaction between its components. Space entities define measure theory, metrics, coordinate systems, continuity and consistency effects, opacity, fields, and granularity of the immediate environment.

The Process Gas Model and VEOS

The first goal of the project is to use the VEOS system to implement the Process Gas computing architecture. The initial effort will not integrate the VR human interface capabilities of VEOS, but instead utilize VEOS as a specification language for setting up complex simulations on various configurations of machines, from a single workstation to a network of workstations to selected super computers. VEOS is designed to allow rapid prototyping and characterization of entities in a computational environment. We believe that the time gained in setting up a complex simulation quickly more than balances the loss in computing efficiency incurred by not writing special-purpose optimized code.

Potential test applications of the Process Gas include:

-- verification of the laws of thermodynamics by accumulating empirical data from the bottom up, from the behaviors of individual molecules,

-- modeling predator/prey cycles by generating populations of animal models in a virtual environment

The General Simulation Engine

Once several examples of the Process Gas have been created and executed on different hardware configurations, the emphasis will shift to building interface tools for the analysis of and interaction with complex simulations. This will utilize the VR capabilities of the VEOS system. Using VEOS, computational entities can be given visual and audio characteristics, and these will serve as cognitive aids in identifying, analyzing, and modifying the structure of a complex simulation. Moreover, the outputs generated by these complex simulations can be given audiovisual representations, and these can be customized to the nature of the simulation and to the specific needs and analysis requirements of the researcher. Since VEOS supports simultaneous virtual presence of multiple participants, it will allow the construction of simulations that incorporate dynamic computation and multiple human interaction simultaneously and in real time. This then sidesteps the longstanding AI problem of translating human behavioral characteristics into static algorithms by putting real humans "in the loop".

Proposal 2: Installing the Process Gas on a CM-5

In the pursuit of a better theoretical understanding of distributed systems in general, and distributed computation in particular, we propose to implement the VEOS parallel, entity-oriented programming environment on the Connection Machine (CM-5) at LANL. We are calling this implementation a Process Gas by analogy with the Lattice Gas systems developed at LANL for the numerical analysis of fluid dynamics.

In a lattice gas, local rules written from the perspective of fixed lattice-sites dictate the motion of "particles" consisting solely of a simple state-variable, like momentum. This approach works fine if there are very few different kinds of particles and if the particles carry very little internal state information.

There are many interesting physical phenomena that have the same basic architecture — a great many entities moving around in some environment and interacting with each other — but which may involve hundreds or thousands of different types of particles, each of which might have complicated internal states. Such phenomena include ecosystems, economies, epidemics, traffic, immune systems, and so forth. Lattice gas approaches in which the rules are written from the perspective of fixed lattice points will not work for such systems. Such systems are more appropriately modeled by rules written from the perspective of the constituent entities themselves, i.e., from the particle's point of view.

This shift in the point of view from which the rules controlling the system are written is the essential feature underlying the Process Gas model. In a process gas, the frame of reference is shifted from fixed lattice-sites to the moving particles themselves (now more generally termed *processes*). Thus, processes contain rules for their own behavior, including rules for motion, and for interactions with other processes and the environment. The environment has rules for its own behavior and for its effects on, and responses to, the processes moving around within it. Such systems can be viewed as a "gas" of processes.

A Process Gas system is effectively an operating system for managing a large, distributed, communicating, and even changing population of computational processes over a large network of computers or on a parallel computer. It must also include a set of tools for allowing researchers to easily create agents, create environments, set up initial states of simulations, perform real-time analysis and data-collection and on simulation runs, and edit and change the state of the simulation and/or the rules for any of the agents or the environment interactively.

Thus, the primary tasks of the Process Gas operating system will be

1) to manage the execution of all of the agent objects,

2) to manage communication between agent objects, and

3) to manage the interactions between the agent objects and the environment object.

All of this must be accomplished in the context of a large distributed computing network, in which agent-processes themselves may migrate from machine to machine or processor to processor.

This can be most quickly and effectively implemented by making use of the results of the VEOS project. In VEOS, every object is a computational entity, including the contents of objects. Thus, as in real biological organisms, agent-objects may be composed of simpler agent-objects, which in turn are composed of simpler agent-objects, and so forth. Since the environment is also an entity, there is essentially only one kind of entity that the system needs to manage: a computational object.

The simple point to make about a process gas system is to distinguish between a virtual computational object (the process) and a physical computational object (the processor), and to realize that the relation between the two is strictly arbitrary and solely a matter of convenience and efficiency. On a large network of computers, the relationship between the set of virtual computational objects and the set of physical computational objects can be constantly changing. Processes that are producing a lot of communication traffic between themselves can migrate to the same processor. Processes that become inactive for long periods of time can migrate out to virtual memory on disk, to be pulled in again when an attempt is made to communicate with them. This gives rise to a "virtual physics" in which some processes generate an "attractive force", causing their computational distance to diminish, and other processes "settle out of solution" because they are only rarely interacted with.

Proposal 3: Installing the Process Gas on a Kendall Square

The VEOS architecture is inherently parallel in that entities are semi-autonomous processes. We have examined several strategies for coordination of entities and for information exchange between entities. The CM-5 provides many powerful processors which communicate via message passing and provides an excellent architecture of complex interactions between many computationally intensive entities.

We wish to examine a completely different architecture in order to arrive at an understanding of appropriate machine architectures and communication styles for the Process Gas model.

The Kendall Square Research (KSR) platform is a massively parallel shared memory machine. Shared memory architectures are scalable and well-suited for many parallel light-weight computational processes. In particular, the Linda model of parallel database management, upon which VEOS is based, is a shared memory architecture and is very well suited for implementation on the KSR.

We propose to identify classes of complex problem spaces which are well suited for the KSR shared memory architecture, and to contrast these results with similar studies conducted on the CM-5.

Proposal 4: Global Information and Simulation Systems

The amount of information available about the state of our planet with all of its subsystems increases dramatically each year. The data comes both from direct observations as well as from computer simulations and more traditional methods of mathematical modeling. The representation and structuring of this rapidly changing information flood is a challenging and unsolved problem.

From the theory of chaotic dynamics and the study of complex adaptive systems we have a sophisticated mathematical and computational tool-box that can be used to understand global structures that are generated by the interaction of large numbers of simple components, the effect of perturbations on complex systems, and the mapping out of stable, unstable, and sustainable modes of dynamic behavior. These tools have been applied and tested for the analysis and modeling of a number of systems in a large variety of contexts. From a different angle they have been most successfully applied in the virtual realities of educational computer games, such as SimEarth of Maxis Corp. Common to both of these examples is that they deal with a closed environment of theoretical or game pragmatic assumptions and parameters. What has been lacking is some efficient interface to the real world of global dynamics. The technology for such an interface is currently developed as global communication and information systems, high speed computer networks, wide area information servers and other areas of global networks become commonplace and inexpensive.

Modeling global sustainability is made more challenging by the constant interaction of a changing and evolving net of external data sources. An efficient interaction with this net is a vital component of any model of sustainability. The Global Information and Simulation Systems project represents an initial phase of the Global Sustainability Program at SFI. An interactive pilot demonstration called EarthStation allows the user to access media services, construct a simple simulation on a computer screen in a graphic, object-oriented manner, and evaluate the results of the simulation. This represents an important first step in the construction of complex global models which are adaptive, process input from external data sources, and interact with human participants in real time.

Proposal 5: Modeling Global Sustainability using the GSE

It is arguable that the most challenging problem facing science in the coming years is the issue of global sustainability of human civilization. Not only is it extremely important in terms of relevance to all human life on this planet, but it is also a scientific problem of a different nature than previous "grand challenge" problems. The problem of sustainability is inherently complex. The questions involved cannot be reduced to single issues of technology or theory, as in the manned moon landing, the development of the atomic bomb, the discovery of subatomic particles, or the isolation of the human genome. The study of sustainability involves at the very least economics and ecology. Other relevant scientific fields include psychology, game theory, and anthropology. The challenge facing the scientific community is to integrate the observations of these different fields and to create a means of generating predictions about possible future evolutions of the global civilization and the influence of present actions on those outcomes.

The SFI is one of the few scientific institutions in the world where distinguished researchers from all of the above fields assemble and discuss the issue of sustainability amongst themselves. One of the emergent realizations that has come of this "cross-fertilization" of academic disciplines is that it has become useful to consider ecology and economy as not only intimately related, but perhaps as different manifestations of a common underlying dynamics. Both ecology and economy deal with the exchange of matter and information among complex semi-autonomous agents. Both are dissipative systems, i.e., they do not conserve energy but rather generate dynamics from a flow of energy through the system. Both appear to yield their structure more easily to a bottom-up approach, the synthesis of complex behavior from aggregates of relatively simple agents. This suggests a new term for the unified field of ecology and economy: *"Ecolonomic Systems"*

The challenge, then, is to scientifically analyze a unique system that is not available for systematic experimentation. It is a fundamentally different problem than the study of immune responses to disease or the behavior of electrons in magnetic fields, as bacteriological cultures and electrons are available in great abundance for use in systematic experiments. There is only one Earth. Thus the issue is one of accurate modeling, and of generating an ecolonomic model of Earth that is sufficiently complex yet still tractable in terms of computational resources and in terms of conciseness of the results. To find a model that preserves the salient features of the real system while discarding the details that are irrelevant is a tremendously difficult task. It is clear that the model itself must be highly adaptable so that it can be evolved to its most effective configuration, as it is highly unlikely that human intuition will get it right the first time. This evolution will require actions that are both self-programmed and actions that come from human input, as either mode by itself is not capable of a complete and efficient generation of possible scenarios.

To make rigorous scientific progress on the issue of global sustainability therefore requires progress on three fronts:

1) The development of sufficiently complex computational models of the ecolonomic system that are easily configurable and easily modified and that have the capability of modifying themselves in response to external data.

2) The development of information management systems to allow the interaction of the model with a flow of world-wide generated data and to allow the efficient interpretation of the model's output.

3) The development of sophisticated human-interaction metaphors to allow the interaction of human participants with the model in real time so as to avoid the extremely difficult issue of programming human characteristics into the model.

The General Simulation Engine (GSE) that we propose would be the logical next step in bringing computational resources to bear on the problem of global sustainability. The main challenge is no longer the development of adequate computational power, but rather the development of modeling and interaction techniques for the constructing and analyzing of different sustainability scenarios. The GSE will provide a means by which complex simulations can be rapidly constructed, modified, and executed. It will also provide for the interaction of human participants with the model in real time using the virtual reality interface, either for the purpose of passively taking data or for actively generating model dynamics.

The three issues outlined above are connected very strongly to programs already in place at the SFI and collaborating institutions. The program most relevant to this program is the "2050 Project", a joint project between the World Resources Institute, the Brookings Institution, and the Santa Fe Institute. This program has already been funded at the level of $3,000,000 over the next three years by the MacArthur foundation. The following statement of research goals is taken from the synopsis of the 2050 Project as described in the project's grant proposal.

Overview of the 2050 Project

The 2050 Project is designed to explore one of the most complex and difficult problems facing humanity, the achievement of a sustainable existence on this planet.

The project will be a serious first effort to examine the concept of sustainability in an integrated way, exploring both desirable future conditions and the transitions needed to reach them. It will use a combination of policy studies and computer modeling and simulation, taking advantage of developments in the study of complex nonlinear dynamical systems, and making the results known to policymakers and the public as well as expert audiences.

"Sustainability" means creating a world in which the quality of life is improving and in which present threats to the quality of life – and to the environment that sustains life – have been brought under control. These threats include potentially irreversible changes in the biogeophysical environment, such as changes in climate, degradation of fertile soils, destruction of biodiversity, and buildup of toxic materials in the environment. The concept of sustainability employed here also involves social and political factors, including such threats to the quality of life as rapid increases in human population, the persistence of widespread poverty, and the continuing potential for military conflict and political repression. Thus the concept acknowledges the undesirability and likely infeasibility of biogeophysical sustainability achieved at the expense of human freedom or coexisting with a large global underclass cut off from prosperity and the benefits of modern technology.

Under the direction of a project director selected by the three sponsors, substantive work of the project will proceed in two major phases, each lasting two years. In the first phase, an early step will be to provide measurable quantitative content to the concept of sustainability used here. The project will then turn to the main work of the first phase by commissioning a series of "base level" studies to answer these questions:

1. To what extent can the linked challenges of world hunger, world food production, and environmental sustainability in the agricultural sector be met by 2050? What measures and what resources would be needed to achieve these outcomes?

2. To what extent can the interrelated challenges of world energy needs, global climate change, and energy-related environmental pollution be met between now and 2050? What measures and what resources would be needed to achieve these outcomes?

The second phase of the project – the integration and synthesis phase – will draw on the base level studies and other modeling and analytic work in an effort to answer a series of extraordinarily important questions. Will it be possible to achieve by 2050 the improved conditions associated with sustainability, as defined in the first phase of the project? Which conditions appear more easily attainable and which are more difficult? Which are mutually supportive and which are antagonistic? How much time is needed to make smooth transitions from current situations to these conditions? Are we already too late in some cases to achieve certain of these conditions by 2050, or ever?

Modeling will be an essential tool in achieving a synthesis that combines many different areas – from energy and agriculture to toxification and security – and in creating self-consistent scenarios. We have no illusions that our models will be able to predict the future; instead, simple models and conventional types of analysis will enable us to be explicit about the consequences of our assumptions.

Yet we are all too aware that the assumptions embodied in our conventional analysis are very restrictive, even if useful for the analysis to proceed quickly. We know that the world is a complicated system with strong interactions among its parts and that it exhibits highly nonlinear behavior. Such systems cannot be successfully understood merely by studying the various sectors separately and then combining those studies to get a picture of the whole. Moreover, in such systems certain small changes at particular times can trigger very large effects, sometimes with system-wide ramifications.

Fortunately, an emerging set of modeling techniques that encompass nonlinear effects and behavior of complex adaptive systems may lend themselves to the study of the phenomena we wish to understand, even if at a preliminary level.

The project will work through a group of modelers convened by the Santa Fe Institute to develop nonlinear modeling and simulation techniques that explore these features of real-world processes and that utilize the growing body of information about complex adaptive systems. The results of this work will be used in the policy analyses and synthesis. There is some hope, also, that through this modeling some simplification might emerge, so that the gross features of the transition to sustainability would be largely determined by a few key parameters, or at least by a few crucial aspects of the world situation. Though the sponsoring institutions recognize that this effort is the most "experimental" part of the project, they are hopeful that it can contribute to a deeper understanding of the problem of achieving sustainability by offering promising insights and valuable new concepts.

Integration with the ICSS Program

It is the above modeling issues that the ICSS Program intends to address. This project provides the first context through which modeling techniques adequate to the task at hand can be developed. Answering the questions posed by the 2050 Project will require the timely development of Process Gas techniques. Applying these techniques to the questions of global sustainability will provide an important demonstration for the viability of the General Simulation Engine.

Furthermore, the ICSS Program will borrow findings and techniques from two other programs currently being instigated at the SFI. The first and most general of these is the Adaptive Computation (AC) Program. The AC Program is a large-scale long-term program to develop and apply biological metaphors of evolution and adaptation to complex computational processes. These techniques include neural nets, genetic algorithms, simulated annealing, artificial life, game theory, and learning theory. Since it is not clear *a priori* what form a model of global sustainability should take, great computational gains stand to be made by employing techniques in which a model adapts and evolves itself to fit the problem.

The implementation of the General Simulation Engine to explore extremely complex sustainability scenarios is a challenge that goes beyond the bounds of adaptive computation. Adaptive methods are primarily techniques in which a computer program adapts itself to best solve a well-stated problem. Global sustainability is not a well-understood problem, nor are the desired results necessarily rigorously quantifiable beforehand. Sustainability is also an issue in which human factors play an important role in such areas as politics, conflict resolution, subjective quality of life, human freedom, etc. The task of programming a computer to use these ill-defined but important concepts to optimize a scenario is extremely daunting. The artificial-intelligence issues presented by this problem can be sidestepped by putting humans into the loop, which is where virtual reality comes in. An extremely large tree of possible scenarios can be pruned to yield scenarios of merit by immersing a human participant in the evolving model in real time to make branching decisions. This technique has been used to spectacular effect by Karl Sims of Thinking Machines, Inc., to prune LISP constructs so as to generate graphic patterns that are subjectively interesting.