

**Numerical Simulations For Active Tectonic Processes:
Increasing Interoperability And Performance**

to

NASA GODDARD SPACE FLIGHT CENTER
Greenbelt, MD 20771

JPL Task Plan No. **83-6791**

Dated: March 7, 2002

Prepared by: _____ Andrea Donnellan, JPL Task Manager _____ Date

Concurred by: _____ Patricia A. Ramirez, JPL Contract Administrator _____ Date

Approved: _____ Diane Evans, Director _____ Date
Earth Science and Technology Directorate, JPL

Approved by: _____ James Fischer/Project Manager _____ Date
GSFC

JET PROPULSION LABORATORY
California Institute of Technology
Pasadena, California 91109

TABLE OF CONTENTS

	PAGE
A. INTRODUCTION	2
B. OBJECTIVE	2
C. MANAGEMENT APPROACH	3
D. TECHNICAL APPROACH	4
E. SCOPE OF WORK	14
F. MISSION ASSURANCE	14
G. SPECIAL REQUIREMENTS	14
H. SCHEDULE	15
I. PERIOD OF PERFORMANCE	19
J. DELIVERABLES/RECEIVABLES	19
ADDENDUM – 1. Additional Requirements	20
– 2. Software Submission Criteria	25
– 3. Additional Contact Information	29

A. INTRODUCTION

JPL will develop a solid earth science framework for creating an understanding of active tectonic and earthquake processes. The sponsor is the NASA Earth Science Enterprise and funding will be received through Goddard Space Flight Center. JPL is the lead technical and management organization for the project. We will construct a fully interoperable system for studying active tectonics and earthquakes, and we will develop simulation and analysis tools to study the physics of earthquakes using state-of-the-art modeling, data manipulation, and pattern recognition technologies.

We will develop clearly defined accessible data formats and code protocols as inputs to the simulations. These codes must be adapted to high-performance computers because the solid earth system is extremely complex and nonlinear resulting in computationally intensive problems with millions of unknowns. Without these tools it will be impossible to construct the more complex models and simulations necessary to develop hazard assessment systems critical for reducing future losses from major earthquakes.

B. OBJECTIVE

JPL's objective is to develop a system to fully model earthquake related data with the following specific components.

1. A database system for handling both real and simulated data.
2. Fully three-dimensional finite element code with adaptive mesh generator capable of running on workstations and supercomputers for carrying out earthquake simulations.
3. Inversion algorithms and assimilation codes for constraining the models and simulations with data.
4. A collaborative portal (object broker) for allowing for seamless communication between codes, reference models, and data.
5. Visualization codes for interpretation of data and models.
6. Pattern recognizers capable of running on workstations and supercomputers for analyzing data and simulations.

The infusion of new data and our current limited understanding of earthquake processes make it an ideal time to develop a high-performance, fully interoperable system for studying active tectonics and earthquake processes. Realistic modeling of the evolution of the Earth's fault systems requires very large-scale computing and data intensive scientific simulations that far exceed the computing and

memory storage capacities of the most advanced desktop computers or even small computer clusters. Effective use of scalable parallel computing systems and related technologies is the only option for geophysicists to achieve a better understanding of the complex evolving behavior of the Earth dynamics.

C. MANAGEMENT APPROACH

Dr. Andrea Donnellan is the PI of this proposal and will oversee all aspects of the work. Many of the team members have worked extensively and effectively together in the past. We recognize that managing such a large team can be difficult. We therefore plan to conduct regular reviews during the course of this work. We will also hold meetings at JPL in which all of the team members can interact and plan work. Team members will be responsible for developing and completing different components of the system as outlined in the following table in which we itemize the component of the system, the responsible members and their background and skills relating to that component of the framework.

Component	Team Members	Task, Background, and Skills
Database system	Dennis McLeod	Database systems
	Lisa Grant	Geology/paleoseismology
	Andrea Donnellan	Tectonics, earthquakes
Finite element code	Greg Lyzenga	Parallel computation/FEM
	Jay Parker	Parallel computation/FEM
	John Lou	Parallel computation/FEM/AMR
Inversion and assimilation codes	Greg Lyzenga	Inversion
	John Rundle	Viscoelastic models
	Terry Tullis	Fault nucleation, fast multipoles
	John Lou	Fast multipoles, data assimilation
Object broker	Geoffrey Fox	Science interoperability
	Dennis McLeod	Science interoperability
Visualization codes	Geoffrey Fox	Visualization (using students shared with FSU visualization group) and integration in portals
	Andrea Donnellan	Geophysical applications
Pattern recognizers	Robert Granat	Hidden Markov Modeling
	John Rundle	Pattern dynamics
Disaster management system	Andrea Donnellan	Earthquakes, data interpretation
	Jay Parker	Data integration
	Greg Lyzenga	Modeling
	Geoffrey Fox	Software Integration
	Karen Yuen	Interfacing with customers
Outreach	Karen Yuen	Outreach to earthquake community

D. TECHNICAL APPROACH

The top-level operational architecture of our proposed solid earth system science framework system shows science users interacting with interface programs as well as modeling, simulation, and analysis tools. These programs generate requests for data in XML/SQL form, issuing them to a distributed object broker constructed in CORBA or Java. This broker dispatches requests to structured databases, managed by a general-purpose database management system, as well as to files and web-based data sources; the latter will be wrapped using mediator technology to look like structured databases. Data integration mechanisms are then employed to fuse data from multiple sources. Information is then returned through the broker to the requestor, in XML form.

Interoperability

One of the most critical aspects of our proposed system is supporting interoperability given the heterogeneous nature of data sources as well as the variety of application programs, tools, and simulation packages that must operate with data from our system. Interoperability will be implemented by using distributed object technology combined with development of object API's that conform with emerging standards. We will define our object API's in XML and dynamically map this specification into the chosen object model. This strategy was successfully used in the Gateway portal, which currently uses a CORBA middle tier but has used a pure Java solution with the same objects.

We use two separate XML interfaces to isolate the middle tier from the client and back end resources. The XML Object API's will be customized to this application but will build on standards set by community organizations. These include the Grid Forum for distributed computing issues and US GEM and international ACES collaborations described below for application specific standards. Fox is co-leader of the Grid Computing Environment (Portal) working group and the ACES computational environment group. We will coordinate our work with emerging activities such as the geology XML specification work at CSIRO. Our general API structure will be consistent with the recent impressive work by the IMS and ADL education community on learning objects.

A framework is beginning to be formed for studying tectonics and earthquakes under a community based effort called General Earthquake Models (GEM). Donnellan, Fox, and Rundle are the leaders of the GEM effort and approximately 125 people have expressed interest in keeping informed of, or involved in, GEM work. We have carried out a series of workshops during the last 3 years to gain an understanding of modeling efforts currently underway, develop lines of

communication between geophysicists and information technology experts, and develop codes and data standards.

This CAN is ideal for accomplishing GEM objectives already established by the community. We will continue to work on establishing requirements for the proposed framework both by working with individuals in the community and through workshops, committees (already established) and meetings. We are well connected with other groups such as the InSAR community and seismology community, both of which are developing XML standards for data products in their respective fields, and will work with them to assure a consistent community framework. The U.S. earthquake simulations community, the GEM group in particular, and this team are actively involved in the APEC Cooperation for Earthquake Simulations. (ACES).

Donnellan is the U.S. Representative to the International Science Board of ACES and several of the team members here are chairs of various ACES working groups, thus further assuring that products produced in this effort will benefit the international as well as the U.S. community. The system enables interoperability and allows new collaborators to easily add new codes and modules to the system. An integrated modeling framework will be further established as new users add various aspects and models to the complicated multi-scale problem of active tectonics and fault interactions.

Database System

The “database system” for this project must manage a variety of types of earthquake science data and information. There are pre-existing collections, with heterogeneous access interfaces; there are also some structured collections managed by general-purpose database management systems. We will also construct at least one new database, characterizing dynamically defined earthquake faults.

Through the GEM effort we have developed XML Document Type Definitions to describe various parameters of earthquake faults and input data. We will develop our earthquake fault databases based on this previous work. The databases will focus on paleoseismic, GPS, InSAR, and seismicity data. We will work with communities that have begun to establish data standards, such as the seismic community (effort led by Berkeley), and the International GPS Service. There has long been a need for establishing a database of faults for seismic hazard analysis (MG95).

Several databases have been constructed for this purpose by the U.S. Geological Survey (USGS), California Division of Mines and Geology (CDMG) and the

Southern California Earthquake Center (SCEC), each with a different format. The primary goal of the existing databases, and current collaborative efforts by USGS, CDMG and SCEC on the "RELM" database, is to provide input for probabilistic assessment of ground motion parameters. Existing databases are not compatible and are not suitably formatted or readily accessible for simulations. For example, much of the focus has been on establishing whether or not certain faults exist or are "active" as defined by the state of California, and how their proposed geometries would affect ground motion estimates.

Most faults in the existing databases have been divided into characteristic segments that are proposed to rupture as a unit. Geologic slip rates are assigned to large segments rather than to the specific locations (i.e. geographic coordinates) where they were measured. These simplifications and assumptions are desirable for seismic hazard analysis, but they introduce a level of geologic interpretation and subjective bias that is inappropriate for simulations of fault behavior. Therefore, we propose to develop an objective database that includes primary geologic and paleoseismic fault parameters (fault location/geometry, slip rate at measured location, measurements of coseismic displacement, dates and locations of previous ruptures) as well as separate interpreted/subjective fault parameters [GL99] such as characteristic segments, average recurrence interval, magnitude of characteristic ruptures etc. Both will be updated as more data is acquired and interpreted through research and the numerical simulations.

To support this earthquake fault database and others, we will acquire and employ a state-of-the-art commercially available general-purpose database management system (DBMS). In particular, we will utilize an extensible relational system. These systems support the definition, storage, access, and control of collections of structured data. Further, we require extensible type definition capabilities in the DBMS (to accommodate application-specific kinds of data), the ability to combine information from multiple databases, and mechanisms to efficiently return XML results from requests. Currently, DBMSs available from Informix, Oracle, and Sybase would provide a good portion of the necessary capabilities.

One key issue that we must address here is the fact that such DBMSs operate on SQL requests, rather than those in some XML-based query language. Query languages for XML are just now emerging; in consequence, we shall initially do XML to SQL translations in our middleware/broker. With time, as XML query language(s) emerge, we will employ them in our system. To provide for the access and manipulation of heterogeneous data sources (datasets, databases), the integration of information from such sources, and the structural organization and data mining of this data, we propose to devise and employ techniques being developed at the USC Integrated Media Systems Center for wrapper-based information fusion to support data source access and integration [MNN99; AM99].

Finite Element Code and Meshing

We have been developing a fully three-dimensional finite element code to model active tectonics and earthquake processes. Because fault systems are so complex, we have also been developing an adaptive mesh generator that constructs a mesh based on geometric and mechanical properties of the crustal structure. The codes make it possible to efficiently model time-dependent deformation of interacting fault systems embedded in a heterogeneous earth structure.

We are in the process of adding new features to the code to increase the realism of the model, such as the ability to model friction on faults, a parameter important to determining how earthquakes occur and faults interact. As realistic features are added to the finite element code, the number of unknowns and time steps required will overwhelm the resources of high-end workstations. We are designing the software to take best advantage of local workstations for mid-range problems as well as high-performance computers such as the Origin 2000 at JPL and the IBM SP2 at the Maui High Performance Supercomputing Center. These machines will be able to solve problems of tens of millions of unknowns for thousands of time steps.

Current workstation capacity can solve for the deformation and stress evolution due to a single-fault rupture, such as the Northridge event, using ~100,000 finite element equations. We seek to analyze the modes of interaction of the entire Southern California system of interacting faults, covering a portion of the crust ~1000 km on a side. Such a simulation would require ~5M equations to determine first-order effects, and certainly higher density for faithful representation of the time-dependent stress field. Current techniques require running such a model through thousands of time steps to attain a stable background stress field and assess the patterns of fault interactions. These considerations motivate tailoring the code toward hundreds of processors to attain solutions in reasonable turnaround time.

Our team has created a meshing tool with mesh generation and adaptation capabilities, while the ESS project has developed a parallel adaptive meshing library (the Pyramid library) for supporting parallel dynamic adaptive meshing of unstructured meshes. Techniques of parallel adaptive meshing, combined with local error estimates, can be effectively used to compute an initial state of the displacement and stress fields, and to significantly reduce the computational load of computing the evolving stress field in the finite element modeling.

We are therefore in an excellent position to evaluate the ESS Adaptive Mesh Refinement (AMR) library with our finite element code, and to combine the strengths of both meshing tools into an integrated adaptive meshing library for a

scalable parallel computing environment. The integrated adaptive meshing library will support the entire process of an unstructured parallel application, including initial mesh generation, dynamic parallel mesh adaptation with efficient mesh quality control, and dynamic load balancing. This will be a valuable tool for improving the computational efficiency of our finite element modeling.

Data Inversion and Assimilation

One of the great challenges in understanding the earthquake process is that an immense variety of data are relevant to the problem. Assimilating and inverting a wide variety of types of data sets that involve both slow and rapid processes, cover a wide range of spatial scales, and include data from both space- and earth-based sensors is an extraordinary task. If we are to construct earthquake models that use this wide variety of data sets in meaningful ways, we need to find ways not only of assimilating all the data, but of inverting it to find the best model parameters.

Model inversion and data assimilation techniques, such as the adjoint model approach, have been applied successfully to atmospheric and ocean general circulation models for model optimization and sensitivity studies. The adjoint model approach is equally applicable to solid earth modeling, and we are investigating and applying the adjoint model to optimize the control variables (physical parameters, initial and boundary conditions) of our simulation models using historical and current observational data, and to achieve a better understanding of the predictions from the existing models. We will look at methods such as those of Geiring and Kaminski [GK97], which are based on code differentiation systems such as the Argonne National Laboratories ADIFOR.

Inversions. In order to do data inversions, it is first necessary to be able to run forward models that predict the data sets and then to find the set of model parameters in the forward problem that best reproduce the data. There are many degrees of sophistication of forward models that can be run, and our goal is to run a wide variety of models with increasing degrees of sophistication, both to get a better understanding of the earthquake process, and to determine what degree of discrimination the available collection of data has between models of different complexity. Ideally the forward models will be physically based and include as much of the physics of the earthquake process as we are able to incorporate.

The most sophisticated of the physically based models use so-called rate and state friction for the constitutive description of the behavior of the fault zone materials [TTE96]. The relevant equations are very non-linear and so running forward models is time consuming. Furthermore, our estimates of some of the parameters in rate and state friction suggest that numerical models that properly represent the behavior of a continuum require quite small element sizes.

For models that represent any significant area, this means that an extremely large number of elements N is required to run the forward problem. In conventional approaches the compute time goes as N^2 . However, as the problem can be treated with Green's functions methods, it is possible to use a Fast Multipole approach in which the compute time goes as $N \log N$ [TSK99]. Because N is very large, we need to be able to make the Fast Multipole methods as fast and efficient as possible in order to acquire science results within a reasonable time frame. Tullis reported initial successes with the fast multipole method at the 2000 ACES workshop in Tokyo; he built on the very successful general package built by Salmon and Warren [SW97].

Data Assimilation. Earthquake data obtained from the historical record as well as geological field studies represent the primary physical signatures of how the earthquake cycle is affected by the frictional properties that exist on the faults. The timing, magnitude and complexity of these historical events are a direct reflection of the values of frictional and other parameters in the model. For simulations in which one or more distinct length scales are chosen for each fault segment (length and width, for example), one must choose these parameters in such a way that the historical record of activity on the fault network is matched as closely as possible. This is the data assimilation problem for which we have developed a simple, but physically motivated, method [RRT00] that we call static data assimilation.

For historical earthquakes, there can be considerable uncertainty about where the event was located [SCH90]. Modern studies [SCH90; GS94; MF98] of earthquakes indicate that slip or seismic moment M_0 is often distributed regionally over a number of faults and sub-faults. Therefore our technique assigns a weighted average of the scalar seismic moments for given events during an observational period to all of the faults in the system. When implemented and compared with data, we find that this method successfully yields average recurrence intervals similar to those found in nature.

Collaborative Portal (Object Broker) and Visualization

There is growing experience in using web-based problem solving environments or portals to integrate the user interface to applications. These integrate information, job submittal, input preparation and visualization. The Grid Forum is currently evaluating and comparing several of these projects including Cactus from Potsdam, the Mississippi web portal, NPACI's HotPage and the NCSA Alliance work at Argonne, Indiana and FSU. We expect this will lead to a better understanding of the issues and we will incorporate lessons from the work in this project.

Our portal will support the object API described earlier and build on the capabilities of the FSU Garnet portal that has been designed to allow powerful collaboration capabilities. Garnet is built around a rather standard multi-tier architecture with back-end objects supported by middle tier brokers, which are accessed by clients that can vary from web clients to personal digital assistants.

For this proposal, we will use the base infrastructure (which mixes commercial event service and audio video modules with other services built at FSU) and extend it to support the special needs of the Earth Science Technology Office/Computational Technologies (ESTO/CT) problem. These needs fall into four major areas.

1. Definition of the object structure in XML for data, simulations, and analysis components.
2. Support of data streams from either real-time sensors or large data banks. Garnet avoids well-known performance difficulties with Java, CORBA and SOAP by manipulating “proxies” in the middle tier. Thereby one has the full power of object technology for control messages, which can be implemented with proxies while one uses native mechanisms where high performance large volume data transfer is needed.
3. Support of the project's visualization needs. We currently have experience with two visualization systems developed as part of DoD's PET program -- DICE from ARL and Visbench from NCSA, both of which use the same general object structure chosen for our approach.
4. Support for the consolidation of multiple simulations, data streams, and databases into a single application

The research issues will not only involve the four specific capabilities above but the implications of these new capabilities on the existing architecture and services. In this proposal, we only address the needed enhancements for these NASA specific applications.

Pattern Recognizers

Earthquakes and other seismic events are the observable product of complex, high-dimensional nonlinear physical systems. However, these underlying systems are themselves not observable. Nevertheless, it is possible to gain considerable information from the patterns of geophysical events in space and time, since these patterns are clearly emergent processes that reflect the structures, dynamics, and properties of the underlying high dimensional system. We propose to develop and use state-of-the-art, high-performance pattern recognition techniques to analyze geophysical data; the resulting tools will be provided for the use of the geophysics research community. These techniques can provide not only understanding of the

physical processes that generate earthquakes, but also the potential for new classes of practical earthquake forecast algorithms.

We will focus our efforts on two different but complementary approaches to earthquake pattern recognition. The first approach is a new pattern dynamics method, which uses the mathematics of pure phase dynamical systems to describe space-time correlations of seismic activity. The observable earthquake activity represents a "wave function," and the underlying dynamics are associated with "hidden variables." This method has the advantage that relative probabilities can be readily defined given the mathematical phase dynamics framework.

We show the results of applying this pattern dynamics type data analysis method to instrumental earthquake data recorded by regional seismic networks since 1932 [RKT00; TRM00]. The data is in the form of a record of approximately 60,000 earthquakes of magnitude greater than 3.0 occurring in California. Retrospective studies of this method using random catalogs and statistical likelihood ratio tests indicates that the pattern dynamics method has considerable forecast skill.

The second pattern recognition approach is based around the use of Hidden Markov models (HMM) [RLR89]. In the HMM framework, the observable data is assumed to have been generated by an underlying stochastic process, which is at any time in one of a set of distinct states. Each state is described by a probability distribution of its observable output, and by the probability of the system being in the same state, or in each of the other states, at the next point in time. Given the observations and a few selected model parameters it is possible to objectively calculate a model for the system that generated the data, as well as to interpret the observed measurements in terms of that model. Furthermore, because the model includes the probability of each state transitioning to each other state at the next time step, it possesses predictive power (i.e., the state of the model at the last observed point in time is known, so the most probable state and output at the next point in time are also known).

Using HMM-based methods to explore data generated by complex physical systems is particularly challenging, as the resulting model must be both consistently reproducible and scientifically meaningful. However recent advances, including large-scale numerical methods, simulated and deterministic annealing [UN98], and automated determination of the number of states [SP00], can be applied to this problem to make HMM-based analysis a valuable tool for analysis of geophysical and other scientific data.

As an example, we show preliminary results of this method applied to a record of approximately 2,000 earthquakes with magnitude greater than 4.0 that occurred in Southern California.

Disaster Management System

The collaborative portal framework used in this project naturally supports disaster management. It provides support for distributed experts to discuss and plan sharing objects such as visualizations of simulations, maps or documents. Basic collaboration capabilities including audio-video conferencing, chat-rooms and whiteboards, support the process by which decisions are made; a command center can communicate with workers in the field.

The Florida State University Garnet collaborative portal builds on the earlier Tango Interactive work by Fox's group at Syracuse University. This system was originally designed for DoD as a web-based command and control battle management system; it was later evolved to support distance education and collaborative computing. We will use this experience in addressing disaster management by adding needed shared data and planning tools to the project's collaborative portal.

We have begun work on a prototype system that takes an earthquake alert as initial input. The interoperable framework will enable real-time analysis and deployment of assets following earthquakes. The analytical tools and codes developed can be used to predict the primary fault responsible for the earthquake, the change in stress field and the locations of potential aftershocks. When an earthquake occurs in southern California a process is spawned that determines whether any of the SCIGN GPS stations may have moved, prioritizes, retrieves, and processes data from the GPS stations nearest the earthquake source first, and calculates station displacements, which are used to invert for fault parameters.

Parallel Systems

Our team has extensive experience developing computational science algorithms and applications on parallel systems.

- Fox has over twenty years experience in parallel computing including building hardware, systems software, and applications. He set up the Caltech Concurrent Computation Program with other Caltech faculty and JPL focusing on parallelizing applications and developing systems software. Fox was involved in the first NASA ESTO/CT Grand Challenges with Peter Lyster (then at JPL) using HPF for Data Assimilation. Currently he is director of the parallel computing laboratory at FSU which is installing a 680 CPU IBM SP system with over 2 teraflop performance.

- Lou has over ten years of high performance computing experience in parallel computational algorithms and code performance optimizations as an ESTO/CT/ESS in-house computational scientist and has collaborated with several ESS grand challenge science teams to achieve ESS performance milestones. Lou has also designed and developed a parallel adaptive meshing library for unstructured meshes, whose development has been funded by the ESTO/CT/ESS project. The library has been ported to the Goddard Cray T3E, Beowulf clusters and Origin2000 and is ready to be integrated in to the VISCO FEM code. The adaptive mesh library has excellent parallel scaling performance on up to 512 processors on Cray T3E. Lou also developed a parallel 3D incompressible flow solver package under ESS project funding. It scales very well on up to 512 processors on the Goddard Cray T3E and is used by several national research labs and universities.
- Parker has developed finite element and linear algebraic solver algorithms for electromagnetic and geodetic applications for 11 years, demonstrating scalable code on the JPL/Caltech Hypercube, Intel Paragon, Cray T3D, HP Exemplar and Sun E10000 computers.
- Lyzenga was among the earliest developers of parallel finite element and finite difference scientific applications for the distributed-memory hypercube architecture, including the precursor to the present VISCO code.
- Granat has developed numerous parallel pattern recognition applications for Cray, Intel, and HP supercomputers, as well as clusters of workstations and PCs. Granat is also the project head of the Scalable Scientific Data mining project at Caltech's Center for Advanced Computing Research.

Our entire FEM code has been ported to the JPL SGI Origin2000 system, running in sequential mode at this moment. A parallel sparse linear system solver library and interface drivers have successfully been built and developed for the parallel FEM code on the system. We expect that the viscoelastic finite element code to be used in this work will demonstrate essentially the same kind of parallel iterative solver for each time step (currently under development) as that for electromagnetic scattering problems. A fixed-sized problem scaling from 16 to 256 processors is 65% (for 166,589 equations), and is higher for the larger problems [CZK00]. The fast multipole library being used by this project runs on a variety of parallel machines.

We have developed estimates of run-time scaling in our Virtual California software, which is an earthquake simulation code in which synthetic histories of earthquakes are computed beginning from initial and boundary conditions. The basic code scales in CPU time as N^2 , similar to an Ising model on the torus, and we

would expect similar scaling rules to apply. We have ported our code to the Maui High Performance Computer Center to a 32 node IBM SP system.

We expect that performance can be considerably optimized by two actions: (1) implementation of fast multipole methods, which should improve the performance from N^2 to roughly $N \log N$ or better, and (2) fully parallelizing the code. In view of the similarities of our model to standard Ising CA-type codes, we expect that optimization of Virtual California will yield a factor of $\sim 30/4$ increase in performance with the Maui Squall machine as compared to a modern workstation.

E. SCOPE OF WORK

JPL will for NASA GSFC, develop high performance interoperable earthquake modeling code. In the performance of this work, JPL will:

- Conduct code baselines, and scaling and performance analysis of existing codes.
- Improve codes for functional enhancement and speedup.
- Attend any ESTO/CT meetings as well as earth science meetings to interface our
- Project with community needs.
- Conduct periodic meetings of the ESTO/CT participants as well as workshops for community users.
- Publish software developed to achieve milestones listed in **Section H. SCHEDULE** on the World Wide Web in accordance with the requirements of **ADDENDUM 2. SOFTWARE SUBMISSION CRITERIA.**
- Prepare a follow-on task plan, if requested.
- Prepare a final report.

F. MISSION ASSURANCE

Mission Assurance shall be performed in accordance with NASA/Caltech Prime Contract number NAS7-1407, Section E-2, Safety and Mission Assurance.

G. SPECIAL REQUIREMENTS

There are no special requirements for the transfer of any ITAR-controlled information to a foreign partner.

H. SCHEDULE

CODE	MILESTONE	DATE
<i>A - Administration</i>	<i>Software engineering plan completed - review board selected.</i>	<i>03/29/02</i>
<i>E - Code Improvement</i>	<p><i>Code Baselines, Scaling Analysis, Performance Analysis completed (generate scaling curves for codes which are already parallel, baseline serial performance for codes which are not already parallel) Documented source code made publicly available via the Web.</i></p> <p>serial code PARK with 15,000 elements for 500 time steps - uses parallel multipole library, but serial main routine.</p> <p>serial code GeoFEST - includes serial iterative solver. 50,000 elements 1000 timesteps - serial implementation</p> <p>serial code Virtual California with N=215 segments for 10,000 time steps, serial implementation on 1 GHz workstation</p>	<i>07/30/02</i>
<i>H - Interoperability</i>	<p><i>Come to agreement on design policy for interoperability and community delivery - Review board approves requirements and a preliminary design for functionality.</i></p> <p>Requirements and preliminary design documents published on the web</p>	<i>07/30/02</i>
<i>B - Administration</i>	<i>First Annual Report delivered.</i>	<i>08/30/02</i>
<i>I - Interoperability</i>	<p><i>Complete prototype described in milestone "H" and test with improved codes. Review board approves.</i></p> <p>Demonstration of interface of Gateway and GeoFEST with simple visualization satisfying preliminary design requirements</p> <p>Mesh generation - Demonstrate ingesting fault geometry and rheology from federated DB, to generate a starting mesh.</p> <p>Functional fault DB and documentation for Southern California</p> <p>Riva: Produce movies of the strain, stress, and displacement data generated from Virtual California and GeoFEST of 1 km resolution for S. California in an integrated way through the grid framework.</p>	<i>02/27/03</i>
OPTIONAL MILESTONE		

CODE	MILESTONE	DATE
<i>F - Code Improvement</i>	<p><i>First code improvement (functional enhancement and speedup) Documented source code made publicly available via the Web.</i></p> <p>PARK on 256 CPU machine with 150,000 elements, 5,000 time steps in the same time as the baseline case</p> <p>GeoFEST - links to PYRAMID and runs on a parallel machine - Produce a plot of scaled speedup that will show that we are maintaining efficiency as the number of processors and problem size increase. Assuming availability of a 64 CPU Beowulf, 1,250,000 elements, 1000 time steps, in the same time as the baseline case.</p>	06/30/03
<i>C - Administration</i>	<i>Second Annual Report delivered.</i>	08/30/03

CODE	MILESTONE	DATE
<p><i>J - Interoperability</i></p> <p><i>OPTIONAL MILESTONE</i></p>	<p><i>Full implementation using improved codes. Review board approves integration into completed framework and updated documentation for:</i></p> <p>Mesh generation - Demonstrate adaptive mesh capability within GeoFEST using a fault stepover geometry wherein the mesh is adapted to accommodate large strains in the stepover as the displacement on the main faults grows.</p> <p>Virtual California SLIDER Phase Dynamical Probability Change Index Data Mining via Karhunen-Loeve Space-Time Pattern Analysis STRESSCO codes as an example of (FLTGRV, FLTGRH, STRGRV, STRGRH) DataMining via Genetic Algorithm Analysis Hidden Markov Model - demonstrate interaction with federated DB through framework Disloc Visco SIMPLEX Final fault DB for California with documentation</p> <p><i>ParVox: Provide an interactive volumetric visualization tool to permit user-controlled view from arbitrary vantage points inside a volume, displaying 3D structure, strain, physical properties, meshes, and seismicity through the grid framework.</i></p> <p><i>PARK on 1024 CPU machine with 400,000 elements, 100,000 time steps</i> <i>GeoFEST (assuming availability of 880 processor machine) 16M elements, 1000 time steps using the Pyramid AMR libraries</i> <i>Virtual California with N=700 segments for 10000 time steps, MPI parallel implementation, running on M-processor machine, speedup of approximately M/10. Investigation of fast multipole method for this code.</i></p> <p><i>Source code for all modules is published on web</i></p>	<p><i>02/27/04</i></p>

CODE	MILESTONE	DATE
<p>G - Code Improvement</p> <p>OPTIONAL MILESTONE</p>	<p>2nd code improvement - further optimization for some codes, pick up others that were neglected in 1st improvement - documented source code made publicly available via the Web.</p> <p>PARK on 1024 CPU machine with 400,000 elements, 50,000 time steps in 5 times the baseline code</p> <p>GeoFEST (assuming availability of 880 processor machine) 16M elements, 1000 time steps in the same time as the baseline code using the Pyramid AMR libraries</p> <p>Virtual California with N=700 segments for 10,000 time steps in 1 hour or less, MPI parallel implementation, running on M-processor machine, with 2 GB of memory per CPU, speedup of approximately M/2 on up to 256 processors. Investigation of fast multipole method for this code.</p> <p><i>PYRAMID: Mesh generation - Demonstrate adaptive mesh capability within GeoFEST using a fault stepover geometry wherein the mesh is adapted to accommodate large strain gradients in the stepover as the displacement on the main faults grows, and coarsening of the mesh in areas wherein the strain field grows smoother.</i></p> <p><i>Source code for all modules is published on web</i></p>	<p>06/30/04</p>
<p>K - Interoperability</p>	<p>Customer delivery - Documented source code made publicly available via the Web</p> <p>Demonstrate integration of one external user application into the framework using the GRID framework wizards</p> <p>Issue testable 5 year earthquake forecast for M>5 for S California</p> <p>Publish the availability of the Portal to the Earthquake community in a peer reviewed periodical such as "Concurrency: Practice and Experience", or "EOS" or an AGU journal.</p>	<p>09/30/04</p>
<p>D - Administration</p>	<p>Final Report delivered</p>	<p>11/30/04</p>

I. PERIOD OF PERFORMANCE

The **contractual** period of performance for this task is from the date of execution of a task order amendment between the National Aeronautics and Space Administration (NASA) and the California Institute of Technology (Caltech) through September 28, 2003.

The **programmatic** period of performance for this task is from the date of execution of a task order amendment between the National Aeronautics and Space Administration (NASA) and the California Institute of Technology (Caltech) through 39 months.

It is recognized, however, that the "contractual" period of performance for the portion of the task performed under Contract NAS7-1407 ends September 28, 2003. This task will be performed in accordance with the programmatic end date contingent upon award of the successor NASA/Caltech contract.

J. DELIVERABLES/RECEIVABLES

JPL will deliver to NASA Computational Technologies Project, in JPL format, reports upon completion of a milestone satisfying the requirements in **ADDENDUM 1. REPORT REQUIREMENTS.**

ADDENDUM

Additional Requirements

1. REPORT REQUIREMENTS

For the purposes of reporting, the required milestones have been grouped into six types. The information to be submitted for each type is listed below. The recipient shall submit the documentation for each milestone by posting to its designated web site on the World Wide Web (WWW). Immediately upon posting to the WWW the recipient shall notify, by e-mail or in writing, the CT Project Manager and Grants Officer of its location. After assessing the releasability of any software submitted as part of this documentation, the Government will subsequently upload this information to the CT Web site and it will become part of the permanent project record. All software provided as part of milestone submissions shall conform to the Software Submission Criteria contained in Additional Term 2 below.

MILESTONE TYPE 1: SOFTWARE ENGINEERING PLAN

Milestone A: Software Engineering Plan Completed

The following documentation shall be provided in fulfillment of this milestone:

- Title of the agreement and agreement number
- Text of the milestone and its due date
- Contact information for the lead software engineer and team members
- A written description of the application being developed
- Preliminary requirements definition for the application.
- Strategy for phased approach to software development for the full lifecycle of the model, linked to the requirements.
- Design elements that enable ease-of-maintenance and robust integration of experimental modules.
- Plans for open source, software reuse, portability, and interoperability with other community efforts.
- Plans for managing the software configuration and controlling multiple versions of the software.
- Plans for ensuring that accepted (and documented) software practices and processes are adopted and followed (quality assurance).
- Plans for community engagement beyond delivery and receipt of comments.
- Plans to collaborate with the CT Evaluation Team's efforts to instrument development codes.
 - Evaluation/audit process
 - Software maintenance plan
 - Validation plan
 - Risk Assessment and mitigation plan

MILESTONE TYPE 2: ANNUAL REPORTS

Milestone B: First Annual Report

Milestone C: Second Annual Report

Annual reports shall describe research accomplished during the report period. These reports shall include at least one representative picture or illustration and provide descriptive text under the following topics:

- Title of the agreement and agreement number
- Period covered by the report
- Objective
- Approach
- Scientific Accomplishments
- Technology Accomplishments (including progress toward milestones)
- Status/Plans
- Point of contact (name, address, email)
- Caption for the graphic
- List (and abstracts) of all publications which cite work performed under this cooperative agreement, identifying which were refereed. If any publication is available on the Web, link from the publication reference to the publication site.
- List of conference presentations resulting from work performed under this cooperative agreement.
- Lists of all other media references in which the research was discussed.
- List of any patents filed or new technology reports resulting from work under this cooperative agreement.
- List of graduate students or post docs trained; list of advanced degrees awarded and thesis titles (if available).

MILESTONE TYPE 3: FINAL REPORT

Milestone D: Final Report

The final report shall summarize the entire set of research accomplished during the duration of the cooperative agreement. This report shall be provided in a format conforming to that of the annual reports described above.

MILESTONE TYPE 4: CODE BASELINING AND ACHIEVEMENT OF CODE IMPROVEMENT

Milestone E: Code Baseline Completed

Milestone F: First Code Improvement Completed

Milestone G: Second Code Improvement Completed

The following documentation shall be provided in fulfillment of these milestones:

- Title of the agreement and agreement number.
- Text of the milestone and its due date.
- A written description of the problem being solved to demonstrate the required improvement.
- A written description of the computer code(s) used to meet the milestone, including descriptions of the algorithms, numerical methods, and parallel implementation.
- If the code is a parallel code, a scaling analysis showing the performance of the code on several numbers of processors including the number used to meet the milestones.
- Documentation as identified in the appropriate milestone.
- The location of an FTP or Web site where NASA may obtain a copy of the computer code(s) in source language form, and any test datasets, makefiles, or other information necessary for NASA to independently verify the achievement of the milestone. This data may also be made available to NASA by noting its location on the file system of a computing system where it can be run. If this system is not a computing system provided by the CT Project, provision must be made for access by CT staff to perform the validation.
- A summary of the scientific or computational significance of achieving the milestone, including graphics if appropriate.

MILESTONE TYPE 5: CODE INTEROPERABILITY

Milestone H: Design Policy For Interoperability And Agreed-Upon Community Delivery

Milestone I: Interoperability Prototype Derived From Milestone “H”, Tested With Improved Codes

Milestone J: Full Interoperability Demonstrated Using Improved Codes

The following documentation shall be provided in fulfillment of these Milestones:

- Title of the agreement and agreement number.
- Text of the milestone and its due date.
- Contact information for the leaders of the teams collaborating on code interoperability as well as their lead software engineers.
- A written description of the application being developed that requires code interoperability.
- Identification of the Software Engineering Plan under which this work is taking place.
- Documentation as identified in the appropriate milestone.
- Description of documented technical achievements toward this goal.

MILESTONE TYPE 6: CUSTOMER DELIVERY

Milestone K: Customer Delivery Accomplished

The following documentation shall be provided in fulfillment of this milestone:

Same as (5) above plus:

- Statements from collaborators not part of the development effort (customers) identifying their successful use of the framework and several key codes; their written commitment to future use is desirable.

2. SOFTWARE SUBMISSION CRITERIA

Recipients may release their software by posting that software to their own World Wide Web addresses. There is no requirement that NASA provide approval of this recipient release prior to its being posted. However, the recipients must make these releases in a manner consistent with this agreement, and are responsible for complying with NASA software release procedures and all export control laws.

Recipients are required to assist NASA in obtaining approval for posting software to a NASA web site.

The recipient shall include a completed NASA Form 1679 (“Disclosure of Invention and New Technology (Including Software)”) with each software submission. All submissions of software under this cooperative agreement shall contain the following legend:

“This software is designated for public release under JPL Task Plan Number 83-6791 and may be publicly released after approval for release has been granted by GSFC Software Release Authority.”

In the event that the GSFC Software Release Authority determines that export control laws may prevent the public release of part or all of the software submitted for release approval, advanced payment for future milestones shall be authorized provided that the parts of the software not subject to export control are publicly released via the web, and that all other requirements of the milestone have been satisfied. After final determination of the export status of the subject software is made by the appropriate government authority, the recipient shall make that software available under conditions appropriate to its export control classification before any further payments are made under this cooperative agreement.

Specific requirements for a software submission in fulfillment of a milestone are as follows:

- I. Software Installation and Delivery Requirements
 1. Recipients shall arrange for a public WEB or FTP repository to be established for archiving software source code and installation files used to achieve milestones. For each milestone completed, the repository shall contain:
 - a. A set of files that represent the distribution version of the source code used to achieve the milestone including makefiles, dependent libraries, etc.
 - i. Source files should be bundled with the standard archive tool for the

- target platform (i.e. TAR, ZIP, etc.) and include all the files needed for an end user of the software to compile and execute the code
- ii. Makefiles should only invoke software included with the distribution and standard software normally delivered with the target system like compilers, loaders and common libraries.
 - iii. Special preprocessors, translators, or libraries must be included in the distribution if they are not available to all users on the target system.
- b. Separate directories for installation files for specific target systems, if the software can be built on more than one system (i.e. commodity Linux cluster, GSFC Compaq system, ARC SGI O3K system, etc.).
 - c. An installation guide that describes the steps needed to install and build the system, including any system or user environment configuration necessary for operation.
 - i. Standard installation packages for the target systems should be provided (i.e. Linux/UNIX Packages, Install Shield, etc.)
 - ii. Installation routines should load all software executables, scripts and data files that are necessary to use the application on the target system.
2. A validation test suite shall be provided for verifying the operational functionality of the software as specified in the milestone. The suite must include any input data files required to execute the test case(s) and copies of expected output data files to validate correct operation
 3. Milestone K delivery must include tutorial examples that illustrate how a member of the user community can customize operational parameters, integrate user developed source code and use the system functionality.

II. System Documentation Requirements

1. A Requirements Document shall be provided which includes sufficient detail to provide requirement traceability appropriate for satisfying the milestone.
2. System Design Documentation shall be provided which includes:
 - a. Sufficient detail to describe the architectural features, module interfaces, APIs, class hierarchies and functionality corresponding to the negotiated milestone.
 - b. A description of the problem class for which the code was designed;
 - c. An enumeration of the equations solved and their boundary conditions;
 - d. A description of the numerical methods used in the code;
 - e. A description of the parallelization techniques used.

- f. References to easily accessible published papers or documents are acceptable as long as the specific sections of those documents which address this particular code are pointed out.
3. A Test Plan shall be provided which includes a suite of test cases that demonstrate that specific requirements for the milestone have been successfully completed. Each test case must:
 - a. Specify all input and output data required to validate the tested functionality.
 - b. Describe the software components and functionality the test case verifies and the specific requirements satisfied.
 - c. Specify the system configuration, libraries, executables and any other test environment requirements necessary to successfully perform the test case.
4. A Users Guide shall be provided with sufficient detail that other members of the Science Team and the user community can (a) scale the problem size or number of processors used up or down, and (b) solve a slightly different problem within the range for which the code was designed (a problem that requires only different initial data and the resetting of existing include-file parameters and flags), i.e., the user is able to compile and run the code for any such problem and be able to understand the results. This information shall include:
 - a. Instructions as to how to modify the source code/include file parameter settings and/or input file data values to change the problem size or number of PEs used to other acceptable settings.
 - b. A detailed description of all code input variables – meanings and valid ranges – sufficient to allow a user to adjust the problem being solved.
 - c. A detailed description of all code output variables sufficient to allow a user to understand the output of the code (including instructions for plotting the output, using standard software, if appropriate).
5. A detailed description of each example tutorial used to satisfy the requirements specified in Section 1, Part 3 of the Software Submission Criteria. A Maintenance Manual or equivalent documentation shall be provided with sufficient detail to allow members of the development team to maintain the source code, implement new features and enhance the functionality of the software system.

III. Source Code Documentation Requirements

1. Sufficient internal documentation of the code shall be provided to ensure that other members of the Science Team and ESTO/CT computer scientists can understand the structure and flow of the code. This documentation shall include:
 - a. An overview of the organization of the code: key routines and data structures, grouping of routines and data structures into functional units or reusable objects/classes.
 - b. Comments describing purpose of each significant data structure/variable used in the code.
 - c. Header comments for each functional routine that conform to the NASA template guidelines.
 - d. Internal comments within each function as required for describing key code segments or explaining subtleties of the algorithm or data representation.

3. ADDITIONAL CONTACT INFORMATION

JPL Task Plan Manager: Andrea Donnellan
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, California 91109-8099
Email: donnellan@jpl.nasa.gov
818-354-4737

CT Project Manager: James R. Fischer
Code 930
NASA, Goddard Space Flight Center
Greenbelt, MD 20771
Email: James.R.Fischer.1@gsfc.nasa.gov
301-286-3465

CT Resources Analyst: Rebecca Knoble
Code 930
NASA, Goddard Space Flight Center
Greenbelt, MD 20771
Email: bknoble@pop500.gsfc.nasa.gov
301-286-8784