



Technical Report GriPhyN-2001-xx  
[www.griphyn.org](http://www.griphyn.org)

## GriPhyN Overall Project Plan

Version 14  
22 December 2001

Developed by members of the GriPhyN Project Team

Submit changes and material to:

Mike Wilde, editor  
[wilde@mcs.anl.gov](mailto:wilde@mcs.anl.gov)

## Table of Contents

<b>1</b>	<b>Introduction: Managing GriPhyN .....</b>	<b>3</b>
<b>2</b>	<b>The GriPhyN Vision .....</b>	<b>6</b>
<b>3</b>	<b>The GriPhyN Computer Science Research Program .....</b>	<b>8</b>
<b>4</b>	<b>Research Milestones .....</b>	<b>9</b>
4.1	Virtual Data .....	9
4.2	Request Planning .....	10
4.3	Request Execution .....	11
<b>5</b>	<b>VDT Milestones .....</b>	<b>12</b>
<b>6</b>	<b>Infrastructure (testbed) construction.....</b>	<b>14</b>
<b>7</b>	<b>Project Process Flow .....</b>	<b>16</b>
7.1	Application analysis .....	17
7.2	Challenge Problem Identification .....	19
<b>8</b>	<b>Coordination Between Grid Projects.....</b>	<b>20</b>
8.1	Coordination Regarding Virtual Data Toolkit .....	22
<b>9</b>	<b>Project Logistics.....</b>	<b>22</b>
9.1	Coordination Meetings .....	22
9.2	Communications.....	22
9.3	Planning.....	23
9.4	Reporting.....	23
9.5	Project Personnel .....	23
9.6	Faculty .....	24
9.7	Project Team Structure .....	25
<b>10</b>	<b>Education and Outreach .....</b>	<b>25</b>
10.1	Web page for GriPhyN E/O.....	25
10.2	Research Experience for Undergraduates (REU) supplement .....	26
10.3	Grid-enable the UT Brownsville Linux cluster .....	26
10.4	Involving other minority serving institutions .....	26
10.5	Leveraging on-going existing E/O programs.....	26
10.6	Course development .....	27
10.7	Workshops and tutorials .....	27
10.8	Other activities.....	27

# 1 Introduction: Managing GriPhyN

The goal of GriPhyN is to increase the scientific productivity of large-scale data intensive scientific experiments through these Grid-based approaches:

- Apply a methodical, organized, and disciplined approach to scientific data management using the concept of virtual data to enable more precisely automated data production and reproduction.
- Bring the power of the grid to bear on the *scale* and *productivity* of science data processing and management.

Virtual data is to the Grid what object orientation is to design and programming. In the same way that object orientation binds method to data, the virtual data paradigm binds the data products closely to the transformation or derivation tools that produce data products. We expect that the virtual data paradigm will bring to the processing tasks of data intensive science the same rigor that scientific method brings to the core science processes: a highly structured, finely controlled, precisely tracked mechanism that is cost effective and practical to use.

Similarly for the Grid paradigm: we see grids as the network OS for large-scale IT projects. Just as today the four GriPhyN experiments use an off-the-shelf operating system (Linux, mainly), in the future, our goal is that similar projects will use the GriPhyN VDT to implement their Grid. The road to this level of popularization and de facto standardization of GriPhyN results needs to take place outside of and continue beyond the time frame GriPhyN; hence the partnerships that we will create between GriPhyN and other worldwide Grid projects are of vital importance.

One of GriPhyN's most important challenges is to strike the right balance in our plans between research – inventing cool stuff – and the daunting task of deploying that “stuff” into some of the most complex scientific and engineering endeavors being undertaken today. While one of our goals is to create tools so compelling that our customers will beat down our doors to integrate the technology themselves (as they do with UNIX, C, Perl, Java, Python, etc), success will also require that we work closely with our experiments to ensure that our results do make the difference we seek.

We will be successful if enough results of GriPhyN flow into the 4 experiments to make a difference for them, and demonstrate the value and future of continuing on this path to make a high-value contribution to enhance the ability of science to deal with high data volumes efficiently, cost effectively, and thus open new doors.

Achieving these goals requires diligent and painstaking analysis of highly complex processes (scientific, technical, and social); creative, innovative, but carefully focused research; the production of well-packaged, reliable software components written in clean, modular, supportable code and delivered on an announced schedule with dependable commitment; the forging of detailed integration plans with the experiments; and the support, evaluation, and continued improvement and re-honing of our deployed software.

While GriPhyN is a research project, its scale and importance and its complex relationships with other projects makes it important to identify clear goals, milestones, and schedules, both for internal planning and for use by our external collaborators. This document, which we revise periodically over the course of the project, provides the highest level view of this information, and serves as a master-plan for the project. Its scope includes all activities that are common to all four of the participating science experiments. To supplement the master plan, all activities that are specific to the GriPhyN interaction with each experiment are described in a planning document for that experiment. These planning documents each cover one project year, running from October 1 to September 30.

Some of the work to be undertaken by GriPhyN is being performed in collaboration with participants in PPDG, EU Data Grid, and DOE SciDAC, as well as, of course, the four physics experiments with whom GriPhyN is partnered. We indicate what components or services we expect to obtain from these projects, and appropriate contingency plans if these deliverables are not forthcoming. We are also working in partnership with the NMI GRIDS Center to provide support for our VDT, and with the iVDGL project to create and operate testbeds.

The GriPhyN methodology for making this difference involves the integration of five overlapping top-level processes: research, development, integration, support, and evaluation.

**Research** – exploring current knowledge and results; proposing new paradigms and techniques; documenting new architectures; building prototypes of new frameworks and architectures; evaluation of the prototypes; research also includes the “market analysis” – the detailed study of our customers, the four physics experiments.

We need to strike a balance between focusing research to solve specific problems of the experiments, and letting research come up with new techniques beyond those that the experiments can even envision today.

**Development** – taking results from research and turning them into packaged software components that can be readily delivered to and installed by customers. Our plans in this area are described as successive releases of the GriPhyN VDT, as described in Section 5, VDT Milestones.

**Integration** – the process of planning and executing the enhancement of experiment data processing by integrating GriPhyN components into the experiments IT. We are working closely with the experiments to execute this step, as described in the individual experiment plans.

**Support** – in order for our tools to be used in such serious scientific projects, they need to be highly reliable, supportable, and *supported*. Since GriPhyN clearly has limited resources, we need to forge relationships whereby support comes largely from Grid support projects like the NMI GRIDS Center, and from the customers themselves. This comes back to the requirement to create tools that are reliable and require minimal support; to document tool usage clearly but at lowest possible cost; and to leverage and enlist the support of the user community itself to support and contribute to the toolkit.

**Evaluation** – in order to succeed, we must continually, diligently, and critically evaluate our software tools, with a critical eye to their deficiencies even as we promote them based on their strengths and benefits. Support and evaluation processes are closely coupled, as we can learn the most about our tools weaknesses when we work closely with the integrators and users of the tools. If we lose sight of this while focusing on ongoing research, we will not succeed. Thus, our project plan includes regular “challenge problems” (the first of which have already been completed) to support evaluation and feedback back into our research and development processes. Some deficiencies will be the result of development shortcomings, while others will dictate that we go back to the research drawing board and look for better ways to solve customer problems.

We view this process as a pipeline (albeit with numerous feedback paths). Not all research results make it into live use, but all are evaluated at some scale.

The critical computer research breakthroughs we are pursuing to achieve these goals are in the areas of:

- The virtual data paradigm, and its supporting catalog structures and integration languages

- Policy and condition-sensitive execution planning and scheduling algorithms and architectures
- Ubiquitous, globally accessible, highly available cataloging systems
- The Petabyte range scalability of data storage and transport and cataloging systems
- Interfaces and levels of automation that make a worldwide grid as easy to use as a workstation

Of equal importance are the critical engineering and project management capabilities we need:

- The ability to turn research results into robust and supported tools that can gain widespread adoption
- The ability to design tools that empower scientific software developers and capture their imagination
- The ability to thoroughly analyze scientific data processing paradigms and uncover and simplify data dependencies
- The ability to understand and overcome the social and organizational issues that often block the adoption of off-the-shelf software by large, complex projects

In the rest of this document, and in a set of four subsidiary “Application Plan” documents, we provide both background information on GriPhyN and detailed technical roadmaps for the various components of the project: computer science research, virtual data toolkit development, and application integration. Because the technical landscape in which we operate is so complex, and the interrelationships between GriPhyN and other activities (ATLAS, CMS, LIGO, SDSS, NVO, EU DataGrid, PPDG, iVDGL, TeraGrid, etc.) are so critical to planning, we focus our planning efforts on identifying:

- The technology development and application integration tasks to be undertaken during the next 12 months, for which we provide detailed plans; and
- The research priorities for CS research that will address what are seen as likely stumbling blocks in out years.

We do not attempt to provide detailed task lists for more than 12 months out, but instead work constantly to update our 12-year-out plans in the light of progress to date and our assessment of the evolving external situation. However, we can express in general terms how we expect GriPhyN to progress over years 2-5 of the project.

In Year 2, as described in considerable detail elsewhere in this document and in the four Experiment Plans, we will deploy the VDT with non-distributed virtual data support; deploy first planner and policy language; integrate virtual data into real efforts in each experiment; start research foci in planning, fault tolerance, and knowledge representation; and demonstrate scaling to hundreds of processors and O(100 TB) of data.

In Year 3, we will introduce distributed, scalable, and fault tolerant virtual data catalog services; deploy scalable and fault tolerant execution services; start research foci in knowledge representation for virtual data; execute substantial challenge problems in each experiment; and demonstrate scaling to thousands of processors and O(1 PB) data.

In Years 4 and 5, we will undertake first field tests and then deployments of knowledge representation techniques and undertake substantial international challenge problems (and, we would expect, production computations). During this time, we would also conduct considerable tuning and evaluation, move forward to new versions of planning and catalog structures, work to

deploy VDT and its various components widely, including to non-GriPhyN experiments, and provide support for VDT users.

## 2 The GriPhyN Vision

The science projects that we target share the common need to harness large-scale distributed resources through data grid technologies. We state an approach to doing this by describing what the four GriPhyN experiments should look like when the fruits of the project in place.

This vision has a direct bearing on our project planning effort. If the scenarios described below depict the end goals of this project, we must create a year-by-year plan that clearly identifies how we'll develop the specified capabilities. This will demand a lot of inter-related and inter-working technologies and components, and will require that we solve research problems in a manner that creates solutions for the missing pieces of this puzzle.

Each step in our plan needs to fit clearly into building the type of solutions that we have described in the following scenarios.

### **Scientists can seamlessly harness powerful grid resources across multiple organizations with little knowledge of the complexities of resource allocation and distributed computing.**

*Example: a CMS physicist can look in a catalog for simulation results. Some of the desired results might be already at their site; others may be at other sites and can be fetched quickly. Still others existed at one time and can be re-derived; the remote network to yet another set of results is going to be congested with a major transfer for the next 8 hours, so a new computation is kicked off to re-derive some of these results, which will finish in 1 hour. The new computation uses 75% local resources, the remaining resources are from remote sites with available cycles on uncongested network paths.*

*The analysis job that needs to run in these results is scheduled and initiated when all data dependencies have been located or materialized. This job runs at 4 different sites, and the final result is emailed to the scientist in the morning. The scientist can check status of the computation at any point, can stop or pause the job; sometimes even steer it.*

*Despite the scale and the complexity of the resources used here, to the physicist this task was no more difficult than if all the work was done on a laptop – the Grid was as easy to use as a PC.*

### **Experiment data is tracked in a uniform manner, clearly identifying how most data objects were derived.**

*Example: A scientist questioning the validity of an analysis can look in the catalog, find that the analysis was based on 1000 event reconstructions, and can check which version(s) or reconstruction code was used to create each of the 1000 events. She discovers that 15 events were reconstructed using outdated code, and she initiates a new reconstruction for these events, keeping the new data in a private store. She then notifies her data administrator of the problem, pointing him to the new events; the data administrator then replaces the outdated reconstructions, and interrogates the virtual data catalog to look for similar events that require upgrading – anywhere in the collaboration, anywhere in the world. With a simple change to a data derivation specification, the results are recomputed, much like a complex program is rebuilt with a simple invocation of a “make” command.*

### **Resource allocations are controlled, measured and tracked by resource administrators who set policies to achieve and arbitrate the overall goals of both the experiment's virtual organization and the resource owners.**

*These policies are not excessively complex to express and maintain, and they control the way in which the grid machinery executes user requests. Resource policies are used to control the use of storage, computing, and network resources by users, groups, and virtual organizations.*

**Enormous resources can be brought to bear with relative ease, but in a controlled fashion.**

*Example: When the ATLAS data administrator at CERN has received the final approval from the developers of a 5 stage pipeline of reconstruction software (who work in Italy, Scotland, France, Moscow, and California), he initiates the rebuilding of the Event Summary Data of **4 petabytes** of raw data. This process takes 5 weeks to complete, and uses resources in 25 centers of 8 countries. 4 of these centers were not even part of the ATLAS collaboration when the reconstruction run started.*

*As the run proceeds, most failure scenarios are handled automatically and transparently. Failing compute nodes are brought out of service and reported for repair through grid and cluster monitoring and management frameworks. Failing jobs are restarted automatically, many from checkpointed results. The progress of the run is continuously monitored and reported back to the group of the submitting administrator. The success of the run is independent of not only the client computer from which it was submitted, but in fact from the failure of the entire site of the submission.*

**Scientists use end-user-oriented tools and express their jobs in science terms rather than in CS terms.** *Scientists can say: I want to run code X on data Y, or even more simply: I want data object Z. The automated grid “planner” mechanism decides where to get the data from, where to run the code, when to run the code, and tells the user the expected completion time. Then the planner decides where to place the resulting data. The planner can slow down, pause, or stop existing work if the new work has sufficiently high priority. The planner (if asked) can explain its decisions to the user, much like today’s SQL query optimizers can diagram a query plan and trace how various query alternatives were selected. The execution planning decisions factor in both site-specific policies and dynamic resource utilization levels.*

**Scientists request data and/or computations using powerful job description languages, with a high degree of interoperability with visual tools. The right balance of visual and textual tool usage is made possible through well-designed architectures. These languages help expose parallelism, fault recovery, and data dependencies and derivations and enhance the location-independence of job specifications. Scientists can use simple but powerful job control languages, embedded in familiar scripting languages, to request data products and transformations.**

*For example, an astrophysicist in Moscow uses a graphical user interface to simulate the gravitational waves produced by a collapsing black hole, running a multi-stage pipeline to filter information in both the time and frequency domain. She then asks the interactive science portal to emit the job control code for the pipeline, and using a simple Python script creates the virtual data definitions in the project’s centralized data catalog to define the output of 5000 such analysis pipelines on a large set of input files from the gravitational wave observatory. Colleagues at Caltech and MIT then explore this virtual data, each requesting different 3000-element subsets of the virtual datasets. These overlapping products result in 4000 pipelines, executing at 5 sites around the world, totally transparently to any of the scientific collaborators.*

**Users are given performance predictions for their jobs before they submit them, with alternatives spelled out for them so that they can make prudent cost-delay-benefit decisions.** *In cases where extremely complex jobs are being scheduled, potentially taking days or weeks to fully complete, the users responsible for these jobs can experiment with policy alternatives to guide data sources or data placement, or priorities for computing resources, to achieve the*

necessary goals. In addition, users can select tradeoffs in execution plan based on quality of input sources and/or transformation algorithms (e.g., fast and accurate vs. slower but lower resolution simulations).

Such long running jobs are amazingly robust and fault tolerant in their execution, working around downed hardware, unavailable data, space shortages, and network outages, and re-adjusting the execution plan as needed to continue with little or no manual intervention and updated completion estimates.

The GriPhyN goal is to create unified, software components and toolkit solutions that are used *in common* across all four experiments, and that these results are used by future experiments in other disciplines with relative ease, changing and enhancing the way data-intensive science will be done in the future.

### 3 The GriPhyN Computer Science Research Program

GriPhyN's success depends on achieving two fundamental breakthroughs in computer science research. Our premises are:

- 1) that we can develop a paradigm, framework, and representation for the complex dependencies between science data objects, and can efficiently and with high integrity capture and accurately replay with high fidelity the steps needed to re-derive data.
- 2) that we can abstract and automate the highly complex and policy-driven decision making processes to automatically schedule work in a complex grid of loosely coordinated resource pools without central ownership.

The computer science research program that we have created to achieve these capabilities is organized as follows:

<b>Institution</b>	<b>Research Topics</b>
University of California Berkeley and LBL	Database query processing research; Request management; Tertiary storage management; simulation
University of California San Diego	Fault tolerance; Metadata management; Storage resource management architecture (SRB)
University of Chicago	Virtual data cataloging and processing; domain knowledge representation; policy research; distributed catalog architectures; simulation of replica placement for computational proximity
Indiana University	User management; Science Portals; portal-to-grid interface language; Grid Operations software
Northwestern University	Performance data collection, analysis and predication; Infrastructures for monitoring and measurement
University of Southern California Information Sciences Institute	Virtual data cataloging and processing; Execution planner algorithms; Request management components; Fault Tolerance; distributed catalog architectures
University of Wisconsin	Job description language and scheduling algorithms and frameworks (Condor, ClassAds, Matchmaking, DAGMan); storage appliance and integration of data movement into job descriptions and processing; recovery of data transfer (NeST)

	and Kangaroo)
--	---------------

Cross-institution collaborations currently taking place within the teams above include:

Virtual Data cataloging and processing (UC & ISI)  
Robust distributed scalable cataloging service architectures (UC & ISI)  
Fault Tolerance (UCSD, UW, and ISI)  
Virtual Data Grid Simulation: (UC and LBL)

As can be seen from this list, GriPhyN will provide an extremely rich and fertile ground for the selection of a large number of doctoral-level thesis topics.

## 4 Research Milestones

The following list, organized by technology area, describes the major milestones in the project for addressing the key technology deliverables that will then be packaged into the VDT and made available for integration into the experiments.

The details of the integration and deployment efforts are described in the detailed project plans being developed by GriPhyN sub-teams working directly with the experiments. (The year-2 plans for these efforts are now available in separate documents temporarily posted at [www.mcs.anl.gov/~wilde/griphyn](http://www.mcs.anl.gov/~wilde/griphyn)).

### 4.1 Virtual Data

#### *Year 1*

- Develop basic information model to represent data elements, the relationships between different data types and the characteristics of data elements. Develop protocols for storing, discovering and retrieving these models. Design and develop tools for creating, accessing and manipulating these models by interactive tools, and planning and scheduling tools.

*Status:* accomplished; demonstrated at SC2001

- Deploy centralized metadata and replica catalog services. Develop tools for managing catalogs.

*Status:* All four experiments have progressed with their own metadata databases. ATLAS and CMS (via the EUDG) have deployed the Globus replica catalog in their testbed. GDMP, used within that testbed, is based on this catalog implementation.

#### *Year 2*

- Develop techniques for representing data transformations, and integrate these techniques into the information model. Develop methods and catalogs for categorizing and curating code elements.

*Status:* Virtual Data Catalog version 0, and a corresponding manipulation language, Virtual Data Language 0, was developed and demonstrated at SC2001. The design for VDC/VDL1, a candidate for VDT release 2.0, is scheduled to begin in Jan 2002.

- Extend catalog services to support distributed and replicated catalogs. Develop techniques for failure detection and fail-over in the situation of catalog failure.

*Status:* A design for a robust and scalable distributed catalog (the Replica Location Service) has been designed and documented, and is circulating for review among the GriPhyN (Globus) and EUDG teams. Prototyping of that design has begun.

#### *Year 3*

- Extend information model to support multiple versions of both data dependencies and data transformation components. Also extend catalogs to support interfaces to request planning and request execution modules.
- Develop distributed algorithms for discovery of information across distributed virtual data catalogs.

#### *Year 4*

- Augment the information model to include information about alternative implementation of data transforms with alternative performance characteristics.
- Develop methods for collecting historical performance information and incorporate into the catalogs.

#### *Year 5*

- Augment information model to incorporate local and global policy constraints.

## **4.2 Request Planning**

#### *Year 1*

- Develop generic models for representing execution plans. Define a set of API and tools for constructing, traversing, and manipulating plan data structures. Develop protocols and formats for storing and exchanging execution plans.

*Status:* DAGman – the directed acyclic graph manager, has provided the project with a unifying language to manipulate and communicate grid execution plans.

- Develop uniform policy representation for code, data and resource access. Develop a set of global and local policy scenarios that reflect the requirements of the user communities of the four physics experiments.

*Status:* The Community authorization server has been developed and is scheduled for incorporation into the VDT 2.0 release. Research into policy language has begun at the U of Chicago.

- Develop simple optimization heuristics. Initial thrust will be on data movement only and focus on the use of alternative, or branching plans to compensate for both resource failure and changes in resource performance. Implement planning heuristics in prototype planning module. Evaluate performance of alternatives with simulation and model based studies, as well as execution on GriPhyn testbed.

*Status:* Basic planning manipulation, and the transformation of abstract to concrete execution plans has been prototyped and demonstrated for both CMS and LIGO at SC2001.

#### *Year 2*

- Design a basic planning API, to facilitate access to remote planning services from high-level tools without dependence on underlying planning heuristics or planning methods. Define and implement planning toolkit, providing access to catalogs as well as remote planning servers.

Status: DAGman has provided a start in this area.

- Extend planning toolkit to incorporate global and local policy considerations into policy construction. Initial focus will be on the application of matchmaking as a means method for the introduction of policy.

Status: research in progress on this at U of Chicago.

- Extend optimization heuristics to include computational resources and data transformations (i.e., code). Evaluate the use of alternative plans to meet optimization goals.

#### *Year 3*

- Extend request planning APIs and toolkit to support incremental plan generation and dynamic replanning. This extended interface will be used to couple the request planner with the request execution services.
- Extend the range of planning algorithms to incorporate alternative optimization heuristics, for example including factors such cost.
- Investigate the hierarchal and distributed planning algorithms and evaluate their impact on scalability, reliability, and the ability to share plans across multiple, independent requests.

#### *Year 4*

- Develop algorithms that incorporate policy constraints into request planning process. These algorithms just examine the constraints applied to each element of the request being planned, and respect the constraints for each local resource as well as for the entire request, with respect to global policies. Initial focus will be on static, non-incremental planning.

#### *Year 5*

- Extend policy sensitive optimization algorithms to incorporate incremental planning. Develop hybrid strategies that combine static and incremental planning. Evaluate performance of new planning algorithms both in simulations and on testbed.

### **4.3 Request Execution**

#### *Year 1*

- Develop and evaluate a task control language capable of capturing the requirements, preferences and dependencies of a PVDG request. Implement prototype of an interpreter to a basic subset of the language.
- Develop a protocol for information exchange between the execution and planning agents.

Status: DAGman graphs (“abstract” and “concrete”) have been adopted for both of these purposes and used in prototype demos.

#### *Year 2*

- Develop an execution agent capable of receiving a simple plan from the planner and interacting with the PVDG services and resources in order to carryout the plan.

Status: The integration of DAGman, Condor, and the Condor-G agent has been demonstrated to accomplish this.

- Develop a protocol for the exchange of co-allocation information (availability, policy, statistics, ...) between the planner and the co-allocation agents.

- Develop a basic portable and configurable event and trigger manager.

*Status:* being explored by the joint PPDG-GriPhyN Monitoring working group.

*MDS2*

- Develop a framework for gathering statistics on the resource consumption profile of completed and in-progress requests and the availability of resources.

*Status:* being explored by the joint PPDG-GriPhyN Monitoring working group.

*Sudharshan, MDS2 , GridFTP perf eval*

### *Year 3*

- Develop a fault-tolerant version of the execution agent.

*Status:* research in progress at UCSD.

- Develop a basic recoverable co-allocation agent. The agent will support basic reservation services.

- Add fault-tolerance to the Condor master process (Keith Marzullo at SDSC) and reliability to basic propagation protocols.

Note: DAGman stuff has fault tolerance

- Develop a fault-tolerant and persistent repository of PVDG statistics

### *Year 4*

- Develop a distributed (mobile) version of the execution agent and enhance the ability of the agent to adapt to changes in the availability, location and capabilities of the grid resources.

- Interface co-allocation agents with planning agents.

- Develop reliable, efficient and secure event propagation and notification protocols

- Develop and implement dynamic and incremental execution algorithms

### *Year 5*

- Evaluate the performance of different execution policies.

- Evaluate co-allocation and reservation policies.

- Add real-time services to the event and trigger system.

- Evaluate impact of incremental and dynamic planning on request execution.

## **5 VDT Milestones**

The following release plan summarizes our initial vision of the main features delivered each year. We will produce one major release of the VDT every project year. VDT features will be enhanced and refined from experience and user feedback, with enhancements introduced throughout each project year through point releases. Note that this integration and deployment activity tracks the principal features listed earlier in the CS research program description. *Also note that this schedule of feature rollouts is still subject to change, based on the availability of*

*underlying technology and the changing needs of experiment deployments and integration.*

The first release of VDT as a integrated whole will be packaged and deployed in testbed efforts in PY2-Q2. All of the components of VDT-1 are already in use in the USCMS and USATLAS testbeds.

VDT 2 will be the first release that contains a virtual data catalog. This will be a version based on the VDC that was demonstrated at SC2001, with features from the LIGO and CMS prototypes consolidated, and significantly enhanced based on early experience and further CS research design effort. We expect to deliver VDT 2 first release in the PY2-Q3 with several point releases of 2.0 before end of PY2.

Once VDT 2 is released, subsequent releases will be scheduled for the second quarter (Jan-Mar) of the remaining 3 project years. The rationale for this is to synchronize the yearly project cycle with the annual Supercomputing conference, which takes place each fall at the beginning of the GriPhyN project year. This conference is an excellent forum in which to showcase GriPhyN results, and from which to take back a lot of useful hands-on experience from building prototypes and demos that can guide the release of subsequent VDT components.

The overall VDT plan is as follows:

**VDT 1.0 (Basic Grid Services)** provides an initial set of grid enabling services and tools, including security, information, metadata, CPU scheduling, and data transport. VDT-1 will support efficient operation on O(10 TB) datasets, O(100) CPUs, and O(100 MB/s) wide area networks and will build extensively on existing technology.

ClassAd toolkit (library)

Condor 6.3.1

DAGMan

Globus 2.0

GDMP 2.0

VDT 1.0 will also include: configuration scripts for replica catalog, GridFTP, GDMP, and MDS that are specific to the GriPhyN / iVDGL test grids. It is tentative slated to use the PACMan installer to provide easy and uniform installation across all GriPhyN testbeds.

Tentative release date: PY2-Q2

**VDT 2.0 (Centralized Virtual Data Services)** provides a *first set of virtual data services and tools*, including support for a centralized virtual data catalog, centralized request estimation, centralized request planning, network caching, and a simple suite of distributed execution mechanisms. Representation and exchange of local policies will be supported for network caches.

Potentially includes:

Virtual Data Catalog structures and VDL engine

VDL and rudimentary centralized planner / executor

Community Authorization Server

Initial Grid Policy Language

User login management tools

Tentative release date: PY2-Q3

**VDT 3.0 (Distributed Virtual Data Services)** supports *decentralized and fault tolerant execution and management* of virtual data grid operation, via integration of distributed execution mechanisms able to select alternatives in the event of faults, agent-based estimation and monitoring mechanisms, and iterative request planning methods. This will be a major release,

and will depend on new functionality from Globus, Condor, and other sources. This version will support O(100) TB datasets, O(10 TB) network caches, O(1000) CPUs, and O(400 MB/s) networks.

Potentially includes:

- Reliable File Transfer service
- Striped GridFTP service
- Managed storage elements (NeST-based)
- Policy-based planner
- Enhanced monitoring and measurement infrastructure
- Distributed high-capacity catalogs (based on Replica Location Service)
- Virtual data generation semantics
- Basic fault tolerance (master-worker fault tolerance for Condor)
- Metadata database integration (from SRB)

Tentative release date: PY3-Q2

**VDT 4.0 (Scalable Virtual Data Services)** *scales virtual data grid operation to realistic magnitudes*, supporting applications involving widely distributed O(1 PB) datasets, O(100 TB) network caches, and O(10,000) CPUs.

Potentially includes:

- Science Portal interfaces
- Distributed planner
- Performance measurement and prediction framework
- "Fuzzy" virtual data description mechanisms
- Catalog and Transport scalability enhancements
- Advanced fault recovery

Tentative release date: PY4-Q2

**VDT-5 (Enhanced Services)** enhances VDT functionality and performance as a result of application experiences.

The final project software deliverable. Provides an enhanced and stable VDT base and is packaged for general use outside GriPhyN, in particular for use by other scientific disciplines.

Potentially includes:

- Knowledge representation framework (ontology-based) for the virtual data catalog
- Object tracking for OO and relational databases
- Additional planner transparency and sophistication
- Further enhancements in fault tolerance, robustness, and transparent recovery

Tentative release date: PY5-Q2

## 6 Infrastructure (testbed) construction

The computing infrastructure of GriPhyN will involve three levels of Grid resources:

*Research testbeds*: small grids where software or application research can be conducted and tools developed. These testbeds will consist of Linux platforms of diverse releases, representing what is currently needed to support experiment application code. This grid, however, will be made available to the entire GriPhyN community.

*Experiment testbeds:* larger grids, owned by each experiment, where challenge problem solutions can be developed and their feasibility measured and demonstrated. These grids exist today, in the form of the US ATLAS test grid, the prototypical US CMS and LIGO grids that were used at SC2001.

*Production resources:* where challenge problem solutions can be placed in live (production) use by the experiments. These grids are being planned and constructed by the iVDGL and the LHC Computing Grid projects.

The current vision is that these resources will be unified into a single “GriPhyN Grid” to provide uniform access to and control of these resources as needed by the project. Currently, we expect this grid to contain some mix of research, challenge problem, and a limited amount of production testbeds within it. (We defined a “testbed” as a set of nodes within the Grid).

The grids listed here will:

- run successive releases of the VDT
- be available to all or selected GriPhyN project members for research, development, and challenge problem work
- utilize a single Certification Authority
- have well-maintained Grid Information Services
- contain an agreed-upon mix of job execution facilities
- contain other shared infrastructure such as test and production replica location services and data transfer services

The experiment-maintained testbeds (for example, the ATLAS testbed) would be used mainly in the challenge-problem development and demonstration phases of the project, and the construction and management of those testbeds could be handled by the experiments and related projects (such as PPDG in the case of ATLAS and CMS).

Most of the final stage of GriPhyN solution development – live deployment – is expected to take place on the experiment’s production resources, but we expect that there may be cases where some types of live deployment can take place on GriPhyN grids (for example, running preliminary analyses where the experiment does not require complete control of the execution environment).

Compatibility issues: if challenge problems are developed using a specific VDT toolset, it’s important that deployment take place on an identical or compatible base. Due to the complexity of the tools (both grid and application) involved, its very difficult to ensure that GriPhyN-developed solutions will run correctly if the target environment is not precisely matched to the development environment.

The grid testbed construction tasks will include:

- Identification of resources (hosts, storage servers, networks)
- Design of login administration mechanism and certification mechanism
- Installation of application software
- Installation of VDT
- Creation of VO’s
- Establishment of CAS’s and policies (policy design a major task)

- types of work
- types of user groups
- priorities of access to resources: computing, storage, network
- Design and setup of Catalog architectures
- Namespace management

Once the basic infrastructure is in place for both production and research, the level of infrastructure effort should diminish somewhat, and involved mainly the installation of new releases of the VDT.

In our process diagram (figure 1), infrastructure deployment is shown keyed to the availability of new toolkit releases; note that it could also be triggered by the availability of new hardware resources that could be integrated into the GriPhyN testbed.

## 7 Project Process Flow

Although the complex worlds of the four GriPhyN experiments defy common and uniform solutions, we nonetheless propose to base the GriPhyN activities within each experiment on a similar planning approach, based on 1-year cycles. This section describes this common, and how the experiment activities interplay with CS research and toolkit development.

This yearly goal-oriented approach will fit well with cyclic events such as project reviews and the demo-driving Supercomputing conference; it may require adjustments, however, in order to accomplish integrations of deliverables into experiment plans that are each driven by a project-specific calendar.

The figure below describes an idealized one-year activity plan for the overall project; for each remaining year we would presumably follow a similar pattern (at least, at this point in our planning). CS activities are shaded dark, experiment activities light. The activities nearer the top of the figure feed the activities lower down, with the challenge problem solutions representing the ultimate GriPhyN goals. A very important aspect of our coordination plan is that the CS activities *span across experiments*, striving to conduct research and create tools that fit the needs of *all* of the experiments.

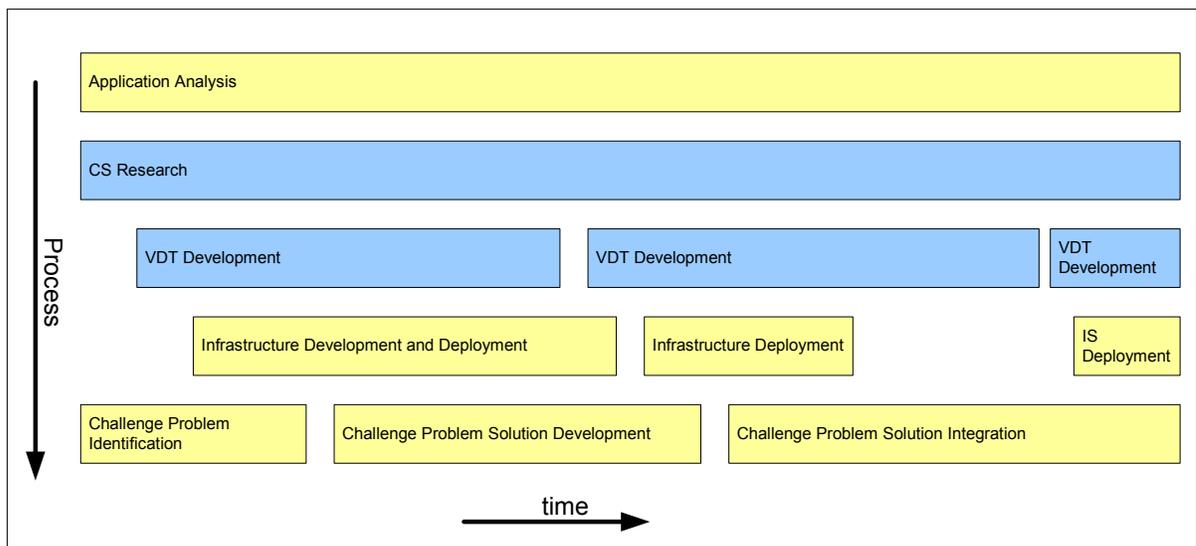


Figure 1: Common Yearly Plan for Experiment Activities.

The yearly plan predicts about 2-3 VDT point releases per year (based on a VDT major-release plan described in Section 3), continuous CS research and application analysis (the latter at a steady but less intensive rate), and one cycle of challenge problem identification, development, and integration. As appropriate, the challenge problem cycle can be repeated several times per year, possibly in an overlapped fashion, depending on the nature of the chosen problems and available integration opportunities that are driven by experiment needs and schedules. This plan reflects the project interaction model that was described in the original proposal, shown in figure 2, below.

The main activities of this common planning approach are outlined in the following sections.

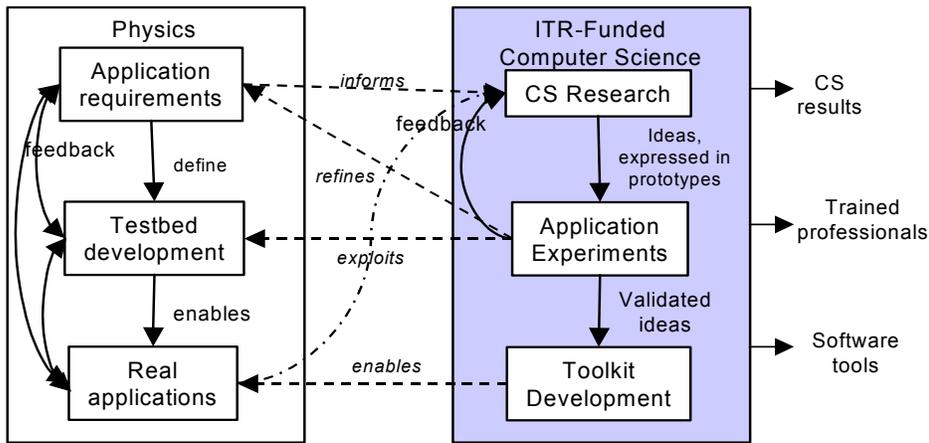


Figure 2: Process and information flow within the GriPhyN project.

Describe seed planting paradigm: vdl; measurements;

## 7.1 Application analysis

The purpose of application analysis is to determine what processes in each experiment could benefit from the type of results that GriPhyN seeks to produce, to refine the requirements for GriPhyN research, development, and deliverables, and to figure out how to apply the results back into the experiment. *This activity is critical to the relevance and utility of GriPhyN results to the experiments.*

Unfortunately, this activity has also proved to be difficult to conduct, and has not *yet* yielded the necessary information back into the project. Difficulties include: much of the information is not yet known, and needs to be extrapolated from past experience; experiment architectures, applications, and decisions are still being formed; the information involved is heavily distributed and the project documents and information sources in which requirements are embedded are usually voluminous. On the other hand, the LHC experiments have been very successful at modeling the expected processing flows within their applications, so there is a positive basis for hope here. We propose to re-assess the process and to find an approach that works and delivers the expected benefits to the project.

Another goal of application analysis is to compare requirements of the four experiments, and then identify common needs that can be met with common tools, architectures, paradigms, testbeds, and infrastructure. If the mechanisms that GriPhyN seeks to create need to be heavily customized for each experiment, then our work is likely to be of less value to other scientific efforts than if we can demonstrate that our results have proven themselves in the four participating experiments.

Stated in an over-simplified manner, the key question we need to analyze is this: how do the experiment's scientists process data? The information we need to capture includes:

- dataset types stored (both types of files and types/classes of object collections)
- definition of jobs and job types, what their control parameters are, and how they are expressed and invoked
- the grouping of jobs into processing pipelines that may be internally amenable to virtualization
- derivation dependencies for each dataset type (the sequence of transformations to create each dataset type)
- the frequency and priority of each data object, program, and information process
- with what mechanisms and frequency will jobs look up and locate data copies to use
- the location of the storage of each dataset type in a virtual organization (e.g., what tier / location the data resides at)
- the likely replication patterns of the datasets (where to, for how long, how/if replicas are eventually disposed of)

To begin with, it will be useful to create a high-level, abstract model of experiment data flow, to use as a guide to the data flow analysis. For example, it seems that at a very high level, most of our experiments follow a model that is something like:

*Capture* and/or simulate

*Refine* the raw or simulated data into a more manipulatable form

*Re-factor* and/or reorganize the data, sometimes changing the dimensions upon which it's based

*Index* the refined data, gradually building knowledge about the science phenomena inherent in the data

*Distribute* the data within the virtual organization for local processing

*Analyze* the data, typically through search, filtering, and statistical correlation techniques

*Reprocess* previous steps, backtracking as necessary, as algorithms, indexing, and search criteria are refined

We proposed to develop a common format for describing these processes, data flows and dependencies. We need to look for new ways to describe the changing rates of data production and consumption within each experiment, and to describe the manner in which data items depend on and are derived from each other. A good example of the beginnings of such an effort are represented in the documents that Koen Holtman has produced to describe the CMS experiment [references].

We see the analysis effort proceeding in 2 phases: Step 1 is to locate the best possible source information and make it broadly available to all GriPhyN by maintaining reference web pages that link to appropriate experiment activity pages. Step 2 is to analyze that information, and reduce it, and then extract the information from those sources into a common GriPhyN format.

Ideally, we will identify common information about each experiment in a common format using a common vocabulary. We need to determine the aspects of each experiment's data processing processes that are most relevant to GriPhyN's mission.

The following knowledge bases are proposed, to document the analysis. This information should be maintained as a continually updated GriPhyN document set:

- Data and Application Map – a chart showing data types, application tools, and their dependencies at a glance. By “data types” we mean: file types, object classes within persistent object-oriented databases, and tables within relational databases.
- Data dictionary – a detailed description of each data type, down to the level where computer scientists can see and understand the access patterns and the derivation dependencies of that data.
- Tool dictionary – a detailed description of each relevant application that will be part of Grid jobs and/or data “transformations”. We need to, in particular, understand these applications from a data transformation and data dependency point of view. What data objects are searched or read by the application, and what data objects are produced or transformed by the application? We also need to understand in detail how parameters and input/output specifications are passed to applications, and how some applications dynamically navigate around a massive information base.
- Data requirements spreadsheet – a summary of quantitative data storage and transfer requirements, detailing a time-varying birds-eye view over multi-year periods of how data will be produced, consumed, and replicated throughout the multiple sites and tiers of each experiment’s virtual organization. This data will be used primarily to determine the scaling needs for data transport and cataloging mechanisms, in terms of storage capacity, catalog capacity, and transaction rates.

Much of this knowledge base should consist of concise, reference-manual-style documentation that details specific data formats, tools, and science-driven IT processes. We want to collect and tabulate a useful reference base of information, rather than a lot of words. Where possible, this reference base should consist of pointers to the experiments’ document repositories.

An excellent start towards this analysis process can be seen in two documents from the GriPhyN-CMS collaboration:

CMS Data Grid System and Requirements (edited by Koen Holtman)

[http://www.griphyn.org/documents/document\\_server/technical\\_report/2001/1/cmsreqs.pdf](http://www.griphyn.org/documents/document_server/technical_report/2001/1/cmsreqs.pdf)

Introduction to CMS from a CS viewpoint (written by Koen Holman)

[http://www.griphyn.org/documents/document\\_server/technical\\_report/2001/4/koen\\_intro\\_cms.pdf](http://www.griphyn.org/documents/document_server/technical_report/2001/4/koen_intro_cms.pdf)

## 7.2 Challenge Problem Identification

We propose to conduct all integration of GriPhyN results into the experiments through the vehicle of *challenge problems*. This phrase is appropriate, in that we view this integration as the *most challenging* aspect of the entire GriPhyN program. Our partner experiments are large and complex: scientifically, technically, logistically, and organizationally. Challenge problems serve as a focal point of our efforts. They give the plan a concrete grounding, help identify integration points within experiments’ processes, and provide demonstrable results of clear value.

Challenge problem solutions involve integrating VDT components with application code and tools to yield working solutions that *are suitable for* live experiment usage. Examples of challenge problems and CP sequences that are created from the GriPhyN feature sets include:

- CP-1 Virtualize an application pipeline
- CP-2 High speed data transfer to replicate results
- CP-3 Automated planning
- CP-4 Mixed replication and re-materialization at high speeds
- CP-5 Abstract generator functions added to virtualization

CP-6 Jobs submitted from high-level tools/UIs (e.g., GRAPPA)

CP-7 Intelligent job management: Transparency, Fault Tolerance, Advanced policy and scheduling

CP-8 Monitoring and information synthesis

It is not easy for an orthogonal research program to insert its results into the mainstream of independent experiments, with independent schedules and, in many cases, funding and oversight. This is further complicated by the GriPhyN mandate to find common solutions across the experiments which runs counter to the need typically felt by each experiment for precisely tailored custom solutions to their complex software problems. The need for commonality is dictated in part by limited staff resources, but primarily by the need to produce results which can benefit numerous disciplines.

We will apply an “intercept” strategy to challenge problem design: we need to determine where the experiments will be when the GriPhyN results are expected to be ready for live usage; otherwise, the results will be irrelevant to the experiments. This will require that we identify integration points (both functionally and in the experiment’s schedule), negotiate the willingness of the experiments to accept and perform integrations, achieve timely deliverables, and track the experiments and their commitments to GriPhyN, so that we can adjust the GriPhyN plans to accommodate any changes that occur in the experiment’s plans.

In designing challenge problems, we need to clearly document the value proposition that the GriPhyN research results would bring to each experiment. In some cases, we will need to make a tradeoff between value to the experiment, difficulty of the challenge, and risk to the experiment for integrating a GriPhyN result. We need to be keenly aware of the quality assurance processes of the experiments if we are to propose integrating changes into mainstream tools upon which the experiments are critically dependent.

As part of the challenge problem identification, we need to develop a plan for how the solution to the problem will (or can) be ultimately integrated back into the experiments standard science processes.

## 8 Coordination Between Grid Projects

This section presents a high-level list of the anticipated relationships with external projects and organizations. Many of these are already in progress, and most have been discussed between several organizations. The details of these relationships will be planned and managed in many different documents and electronic forums.

### **PPDG**

*Expect from PPDG:* GDMP; Scalability feedback and testing of distributed catalog solutions and high-end databases for catalogs; research into mechanisms for distributed physics analysis (Conrad Steenberg); research into fault tolerance (Takako Hickey)

*Provides to PPDG:* VDT, Data Grid Architecture document (for PPDG review and feedback); research results in such areas as scalable catalog solutions, data replication strategies, fault tolerance techniques.

*Shares with PPDG:* analysis of ATLAS and CMS data management requirements; development of US testbeds; application experiments.

### **iVDGL**

*Expect from iVDGL:* acquisition and setup of hardware for testbeds; uniform installation of VDT; technology for operations of testbeds; feedback on VDT based on large-scale experiments; trace information for CS research into replication strategies, etc.

*Provides to iVDGL:* VDT; maybe components (beyond VDT) for construction of iGOC (open issue).

*Shares with iVDGL:* Hardening of VDT; conduct of experiments.

### **EUDG**

*Expect from EUDG:* GDMP (via PPDG); Scalability feedback and testing of distributed catalog solutions and high-end databases for catalogs. Currently being discussed is whether we may receive (or co-develop) some additional components, e.g., schedulers.

*Provides to EUDG:* VDT.

*Shares with EUDG:* design and development effort for data grid components (RLS, RFT, RRT); conventions for toolkit installation; conduct of large-scale transatlantic experiments. We may jointly participate in Education and Outreach activities related to the EUDG-IE's work by Gianfranco Mascari, Emanuela Piervitali.

### **GRIDS Center**

*Expect from GRIDS Center:* basic packaging of Globus and Condor to serve as base for VDT; improved hardening, installation, and documentation of these components; training; some support services to GriPhyN user community for VDT; outreach to additional user communities.

*Provides to GRIDS Center:* VDT.

*Shares with GRIDS Center:* Outreach to common user communities.

### **HICB/HIJTB**

*Expect from JTB:* Coordination with international Data Grid communities.

*Provides to JTB:* Input on recommended technologies, testbed requirements and plans.

*Shares with JTB:* Development of standard technologies, development of testbeds, international experiments.

### **GGF**

*Expect from GGF:* Best practices and standards in such areas as GridFTP, security.

*Provides to GGF:* Requirements analyses, best practices, and proposed standards.

*Shares with GGF:* Development of standards.

### **Globus Project™**

*Expect from Globus:* All services of Globus 2.0 (GSI, MDS, GRAM, Replica catalog and management; GridFTP); advanced security technologies (CAS); RLS; RFT (to be developed with funding from DOE SciDAC and PPDG);

*Provides to Globus:* Requirements and requirement analyses; simulation results; monitoring tools

*Shares with Globus:* Requirements, techniques, and/or tools for eventual incorporation into and support by Globus, in such areas as monitoring and security.

### **DTF**

*Expect from DTF:* Detailed specification of environment: Chips, compilers, OS, and data storage and transfer architecture; job execution and scheduling environment.

*Provides to DTF:* GriPhyN needs to arrange with experiments the porting and certification of some portion of their applications to the specific IA64 chip and platform needed to run the in

DTF. Potentially, an NMI-compatible version of the VDT to augment the base Grid software on DTF nodes.

*Shares with DTF:* Outreach to user communities, development of requirements for testbeds and tools.

## 8.1 Coordination Regarding Virtual Data Toolkit

This work will be done in collaboration with the GRIDS Center and the NCSA Alliance, and will integrate software from other DOE-and NSF-funded Globus and Condor development activities. We envision this work providing the base software installation for the iVDGL.

The GriPhyN contributions to this work will be as follows:

- Addition of various components, over time (e.g., in the first year, DAGMan)
- Definition of a standard software release, consistent with the Data Grid Reference Architecture (DGRA).
- In collaboration with the GRIDS Center, packaging, integration, and documentation to produce an easily installable, documented binary version of this software release.
- In collaboration with the GRIDS Center, provide support for the software.

We assume that the iVDGL will handle:

- Operation of central services for GriPhyN experiments that wish to use this.
- Deployment of software on resources to construct testbeds.
- Development of push mechanism for automatic distribution of updates.

## 9 Project Logistics

In this section we described project plans for Reporting, Communications, and Meetings.

### 9.1 Coordination Meetings

We hold the following types of meetings:

- All-hands meeting: twice a year. Plans are presented here, people report on progress, etc.
- Periodic applications-software meetings to discuss software development and iVDGL deployment: twice a year (half-way in-between the two all-hands meetings). These provide input to the next round of planning.
- Periodic application-CS one-on-one meetings: structured as an “all-hands” on each side. Ideally these are held twice a year, also.
- Occasional CS research meetings: as needed, on specific topics.

### 9.2 Communications

We maintain:

- Various email lists, all archived.
- A web site, [www.griphyn.org](http://www.griphyn.org), with news etc., a calendar of upcoming events, and an archive of material from past events.

- Analysis and design documents for each experiment, and for elements of the VDT architecture.
- The VDT and associated documents.
- A Data Grid Pubs web site with relevant publications.
- Status reports, collated and made available on the GriPhyN web site.

### 9.3 Planning

We maintain a set of GriPhyN Project Plan documents, revised on a six-monthly basis prior to each all-hands meeting, and presented at the all-hands meeting. The documents consist of this master plan document, plus one yearly planning document per experiment. Eventually we will probably break the CS Research, VDT, and Testbed activities into standalone planning documents.

We also plan to integrate the use of a planning tool (such as MS Project) into our planning process to track detailed task and milestone lists.

### 9.4 Reporting

We provide NSF with a yearly report once each year, in June. The June 2001 report is posted as:

*GriPhyN Annual Report for 2000-2001, NSF Grant 0086044*

[http://www.griphyn.org/documents/document\\_server/status\\_report/2001/1/griphyn\\_report2001\\_final.pdf](http://www.griphyn.org/documents/document_server/status_report/2001/1/griphyn_report2001_final.pdf)

### 9.5 Project Personnel

*Note: The following table is now out of date and will be revised shortly. It still, however, correctly reflects the approximate division of resources across the project's institutions.*

Table 1 shows the allocation of graduate students, postdocs, and staff to the various elements of the project. In the initial budget allocation, the institutions with larger allocations have some ramp up in the first year; some CS institutions do not receive funding for graduate students in the fifth year. We emphasize that while we show a scenario for approximately constant level of effort at each institution, we do not expect that we will stick to a static budget for a five-year project. The management structure outlined in the proposal will work to optimize the use of resources during the project period.

We anticipate funded personnel being mapped to activities roughly as follows:

- CS research will involve 9 graduate students, and 4 postdoctoral associates.
- Virtual Data Grid Toolkit development will involve 3 graduate students, and 4 technical staff.
- Discipline applications will involve 2 graduate students and 6 postdocs.
- Outreach and education involves one dedicated staff member.

**Table 1: Personnel per project and institution, categorized as Graduate students, Postdoc, or Staff.**

Inst.	CS Research			VDT			Physics Experiments											
							ATLAS			CMS			LIGO			SDSS		
	G	P	S	G	P	S	G	P	S	G	P	S	G	P	S	G	P	S
Caltech											1	0.5			0.5			
IU							2	1										
JHU																1	1	
NWU		1																
UC	1	1	*	1		2*												
UCB <sup>1</sup>	3		*															
UCSD	2		*															
UF <sup>2</sup>		1									1							
USC		1	*	1														
UTB														1 <sup>3</sup>				
UW-Mad	3		*	1		2*												
UW-Mil															1 <sup>4</sup>			
<b>Total</b>	<b>9</b>	<b>4</b>		<b>3</b>		<b>4</b>	<b>2</b>	<b>1</b>			<b>2</b>	<b>0.5</b>		<b>1</b>	<b>1.5</b>	<b>1</b>	<b>1</b>	

This is based on the original proposal; we need to update to reflect final budgets. Note that some of the larger budgets were somewhat reduced in the first year. A \* denotes summer salary for faculty.

## 9.6 Faculty

The following is a list of the faculty involved in the project:

- CS research: Arpacci-Duseau, Foster, Franklin, Kesselman, Livny, Marzullo, Moore, Otoo, Schopf, Taylor
- VDT development: Foster, Kesselman, Livny
- ATLAS: Gardner, Huth
- CMS: Avery, Bunn, Newman
- LIGO: Allen, Lazzarini, Campanelli, Romano
- SDSS: Szalay

<sup>1</sup> UCB graduate students are only funded for 4 years.

<sup>2</sup> UF also has cost sharing for a project coordinator and part of an administrative assistant

<sup>3</sup> The UT Brownsville LIGO staff person is Manuela Campenella, who is in fact focused primarily on Education and Outreach.

<sup>4</sup> The UW Milwaukee staff member, Scott Koranda, is funded 1/3 by NCSA in support of GriPhyN.

## 9.7 Project Team Structure

We structure our research project as three distinct but tightly integrated activities:

- **IT research:** groundbreaking IT research motivated and guided by challenges encountered by domain scientists in meeting their computational and data management needs.
- **Application experiments:** prototyping new information technology and interfacing it with scientific applications in real-life test-bed environments defined by CMS, ATLAS, LIGO, and SDSS requirements.
- **Tool development:** turn “winning” prototypes into production quality tools to be used by the scientific community.

The 26 researchers—12 computer scientists and 14 experimental physicists—that we have brought together to attack these problems have been intrigued by the research challenges, the possible impact, and the scope of this project. These researchers represent a wide inter- *and* interdisciplinary spectrum of research interests, talent, and experience. The computer scientists are organized in seven groups—Berkeley, Chicago, Indiana, NWU, San Diego, USC, and UW Madison—each managed by local GriPhyN personnel and each contributing a unique capability to the project and responsible for one major research activity. The physicists are distributed across a comparable number of institutions, chosen with a view to IT expertise and connections with the physics experiments.

The rich collection of technology and software already developed by project participants means that we can start all three of the phases just listed in parallel. The result will be a steady stream of IT research results, application experiences, and production quality software.

We plan to coordinate these diverse efforts via the definition and frequent review of a *PVDG architecture* that defines the interfaces between key components, and the scheduling of frequent *integration events* in which components developed by different groups are brought together for interoperability testing.”

## 10 Education and Outreach

The Education and Outreach (E/O) program of the GriPhyN project is designed primarily to expose faculty and students at other U.S. universities and institutions to GriPhyN research. In particular, it intends to promote learning and inclusion via the integration of a diverse set of minority and under-represented institutions into the scientific program of participating physics and computer science experiments. This program will engage all GriPhyN senior personnel, with each committing to lecturing and mentoring activities at these institutions.

In order to facilitate and coordinate these activities, The University of Texas at Brownsville recently hired a full-time faculty member, Manuela Campanelli, to serve as E/O coordinator for GriPhyN, with a start date of Fall 2001. Our plans for E/O are described below.

### 10.1 Web page for GriPhyN E/O

The E/O web page will grow to contain basic educational material about data grids and the participating physics experiments. In addition, it will provide basic technical support information (e.g., documentation, user manuals, how-to guides, etc.) for the GriPhyN virtual data toolkits. We also plan to extend a web-site that Alex Szalay (at Johns Hopkins University) and Jim Gray (at Microsoft) are developing for accessing SDSS data, to illustrate the concept of virtual data. One

idea is to produce on-the-fly custom data sets representing images of the sky as viewed in different frequency bands.

The evolving GriPhyN Education and Outreach Web page is available now, at

<http://www.aei-potsdam.mpg.de/~manuela/GridWeb/main.html>

and is linked to from the main GriPhyN web page at [www.griphyn.org](http://www.griphyn.org).

## **10.2 Research Experience for Undergraduates (REU) supplement**

In Fall 2001, Campanelli plans to submit a proposal to NSF requesting an REU supplement to support students doing grid-related research. If funded, students would be able to work on research projects during the summer months at participating GriPhyN (or iVDGL) institutions. At the end of the year, the students would present posters or give talks at a conference specifically designed to showcase their work. (Note: Philip Dukes, another faculty member at UT Brownsville, who has experience in physics education, may serve as co-PI on this proposal.)

## **10.3 Grid-enable the UT Brownsville Linux cluster**

UT Brownsville has nearly completed constructing a 96-node Linux-cluster. Although the cluster will be used primarily for LIGO data analysis, it can also serve as a testbed for GriPhyN software and be used to introduce minority students at UT Brownsville to distributed computing and grid-related technology. If the iVDGL proposal and the REU supplement get funded, we will be able to support an additional two or three undergraduate students to learn how to install and run grid software, like Globus or Condor, on the completed cluster.

## **10.4 Involving other minority serving institutions**

If the iVDGL proposal is funded, two other minority-serving institutions (Hampton University and Salish Kootenai College) will receive funds for hardware and personnel to construct small clusters (i.e., Tier3 centers), thus bringing a large number of additional minority students directly into contact with large-scale grid research. Campanelli will coordinate the E/O effort for the iVDGL project as well, providing technical support for the Tier3 centers and functioning as an interface with the GriPhyN research community. Moreover, Keith Baker, who would be the lead person for the Hampton University Tier3 center, is a principal investigator for QuarkNet.

## **10.5 Leveraging on-going existing E/O programs**

There are a number of on-going education and outreach programs that we plan to utilize in the forthcoming months:

- Indiana University and The University of Florida are participating GriPhyN institutions and active QuarkNet centers. GriPhyN researchers at these institutions will be encouraged to provide a grid-related component to the already existing QuarkNet activities.
- Valerie Taylor (a GriPhyN senior investigator at Northwestern University) is a PI for the Coalition to Diversify Computing project within EOT-PACI---the education, outreach, and training partnership for advanced computing resources. Campanelli plans to work with Taylor on ways to link the E/O activities of GriPhyN with the EOT-PACI program.
- We have already begun talks with ThinkQuest to develop special challenge projects based on the application sciences and grid technology. GriPhyN (or the iVDGL project) would provide resources in the form of interesting data sets (e.g., SDSS images or LIGO data) and/or "sandbox" CPUs that students could use when creating innovative web-based educational tools for ThinkQuest competitions.

## **10.6 Course development**

All GriPhyN senior investigators will be encouraged to include grid concepts in their physics or IT courses. In addition, each GriPhyN senior investigator is committed to lecturing and mentoring activities at other (in particular minority serving) institutions. These activities are important as they are specifically targeted to promote and improve the education of students regarding grid-related activities.

## **10.7 Workshops and tutorials**

Campanelli, together with several other GriPhyN senior investigators, will organize several 'how to' workshops and tutorials at the various Tier3 minority serving institutions. In order to facilitate the participation of all senior investigators and given the limited E/O budget, we propose to do this in conjunction with at least one of the all-hands GriPhyN meetings. This would allow a more direct and larger participation of minority students, without needing additional travel money for students. UT Brownsville has volunteered to hold such a meeting in Spring 2003.

## **10.8 Other activities**

An interesting possibility for GriPhyN E/O is the creation of an educational documentary about GriPhyN computational data grids. Because this project would require significant budgetary and human resources that are not presently available in our E/O program, we are considering the possibility of working together with other national or international grid partners, like the European DataGrid, which has expressed an explicit interest in this direction. Such a documentary would certainly produce a worldwide impact on the general public, and could also be used as a powerful media to reach high-school students and teachers.

Additional Specific Education/Outreach (E/O) goals for Oct. 2001 – Oct. 2002:

1. Since the E/O program of the GriPhyN and iVDGL projects is designed to expose faculty and students at other U.S. universities and institutions to grid-related research, it is extremely important that all GriPhyN and iVDGL senior personnel be committed to lecturing and mentoring activities at other institutions. We will keep a record and make available a list with materials of talks given about the grid and GriPhyN.
2. Our second near term goal is to give undergraduate students the opportunity to participate in grid-related research at GriPhyN/iVDGL institutions by taking advantage of the NSF Research Experience for Undergraduate (REU) Program. We will submit a proposal for an REU supplement early in 2002.

We will seek funds to support 10 to 20 undergraduate students to do GriPhyN related research at various GriPhyN institutions. Students themselves will then apply and choose the mentoring institutions on the basis of the research projects proposed by each institution. This will already provide some pre-selection process. Further selection will be done by the mentor based on the individual skills required for the chosen project.