# GriPhyN Project Plan

## Draft

## Version 5 – 26 Sep 2003

# 1 Plan Overview – Project Years 2 through 5

## 1.1 Review of GriPhyN Vision and Goals

The goal of GriPhyN is to increase the scientific productivity of large-scale data intensive experiments. We state an approach to doing this by describing what the four GriPhyN experiments should look like when the results of GriPhyN are in place:

Scientists can harness significant grid resources 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 results they want 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 fina 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.*

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 ech 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 She discovers that 15 events were reconstructed using outdated code, and she initiates a new reconstruction for these events, keeping the new data in a private store. She then notifies her data administrator of the problem, pointing him to the new events; the DA then replaces the outdated reconstructions. He also interrogates the catalog to look for similar events that require upgrading.*

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.*

Scientists use end-user-oriented tools and express their jobs more in science terms than in CS terms. *Scientists should say: I want to run code X on data Y. The grid planner should decide where to get the data from, where to run the code, when to run the code, and tell the user the expected completion time. Then decide where to place the resulting data. The planner should slow down or pause/stop existing work if the new work has sufficiently high priority. The planner should be able to explain its decisions and recommendations to the users.*

Users are given performance predications for their jobs before they submit them, with alternatives spelled out for them so that they can make prudent cost-delay-benefit decisions.

Unified, off-the-shelf component/toolkit solutions are created in common across all 4 experiments; the results are usable by future experiments with relative ease, and change the way science will be done in the future.

What does this goal statement have to do with project planning? If the scenarios represent the end goals of this project, we must create a year-by-year plan that clearly identifies how we'll create all this. This will demand a lot of inter-related and interworking technologies and components, and will have to 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 scenarios above.

## 1.2 Challenge Problems

Challenge problems are employed to 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. These need more refinement. A first-pass list of general problem categories are:

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 a high-level tools/UIs (e.g., GRAPPA)

Challenge problem solutions involve integrating VDT components with application code and tools to yield working solutions that *could* be deployed for live experiment usage. Solving these problems has high value even if they are *not* deployed. Achieving production deployment has complex planning, technical, and managerial dependencies.

## 1.3 VDT Releases

VDT-1 Initial release
VDT-2 Includes virtual data catalog structures, and rudimentary centralized planner / executor, + CAS & RFT
VDT-3 Includes managed storage element (NeST-based?); policy based planner
VDT-4 Distributed planner; higher performance data transfer and distributed high-capacity catalogs
VDT-5 Enhanced and stable base for use by other sciences

## 1.4 Goals, approximately by Project Year

Year 2: Grid structures in place; CP-1 solution in development
Year 3: CP-1 solution in production; CP-2 solution in development
Year 4: CP-2 solution in production; CP-3 solution in development
Year 5: CP-3 solution in production; CP-4 solution in development

Transparency, Fault Tolerance, Advanced policy and scheduling?

## 1.5 Metrics

(to be developed)

Non-raw Data volume derivation-cataloged: 2: 5 3: 25 4: 50 5:75

Applications (by jobs submitted) covered by GriPhyN processes

New applications developed grid-aware from scratch:

%jobs run on Grid

%jobs run under GriPhyN planner

## 1.6 Seeking Common Solutions

Wherever possible, we strive to implement common solutions across experiments.

Unifying threads: common languages across experiments: eg xsil, dagman.

If we diverge along multiple lines to solve the same problem (for research purposes) we do it with a plan to converge before the end of the project – we don't want to litter this field with multiple solutions that do the same thing.

## 1.7 "Managing" Research

Research tie-ins – different ranges of research, but all clearly identified

Need to clearly tie results to GriPhyN funding; document conventions for Attribution to griphyn funding (already done somewhere – could not find email).

## 1.8 Funding, reporting, and accountability

# 2 Project Year 2: October 1, 2001 to September 30, 2002

## 2.1 Overall Goals

Our primary goals in Year 2 are:

- Release DGRA and RVD document revisions that resolve open issues in catalog structure, planner architecture, monitoring and measurement, policy implementations, and the languages for job and data description techniques that will be used throughout the data grid.
- Release VDT 1.0
- Establish a GriPhyN testbed in each of the 4 experiments, and maintain it with successive VDT deliveries.
- In at least two experiments, do production computing using VDT 1.0, using resources in multiple sites (at least four sites where possible).
- Identify and solve Year 2 challenge problem(s)
- Create Data & Process Map and Dictionary / Use Case Digest: describes in detail the nature of jobs, data, and requests that each experiment will generate, with quantifications of scale, frequency, importance to the experiment, and feasibility of managing that data and/or process using GriPhyN technology.
- To the extent possible, develop a roadmap for challenge problems for project years 3-5.
- Release VDT 2.0 with a solid set of virtual data software, addressing materialization transparency.
- Achieve solid results from research projects established in the first year, with plans for transitioning these results into VDT 3.0.
- Create a GriPhyN Result Integration Plan (GRIP) for each experiment – describes the technical and project management mechanisms that need to be achieved in order to achieve successful result integration.
- Achieve solid results from the Education and Outreach program: Facilities, Programs, Seminars, Papers, Activities
- Create a VDT Canonical Application that captures key elements of virtual data concepts, and utilize that application to facilitate more rapid, parallel research.
- Align research plans with the concrete goals of the GriPhyN project, and address the highest priority research goals: ojb description languages; scheduling and planning; distributed catalog architecture.

Year 2 challenge problems are:

- ATLAS: Virtualization of test beam capture and analysis process. Ideally, virtualize a simulation process that uses the new Athena framework.
- CMS: Virtualization of Monte Carlo production in the MOP high-throughput framework. Actual production, at least at Fermilab, will use the virtualized MOP framework. Other sites will be able to both replicate and materialize data products produced under Fermilab control. Sites will be able to replicate early initial files in the simulation pipeline and materialize the final files in the pipeline. Some amount of intelligence for dealing with Objectivity object sharing of simulation results will be implemented. Late in the project year, an automated planner will make decisions about where to execute simulation runs. CONTINGENCY: there is a possibility that after further analysis, a better challenge problem will be identified. If this happens, we will need to consider changing the plan to switch to the alternative challenge.
- LIGO: capture in a sentence or two.
  Would it be possible to utilize Grid tools (GridFTP, replica catalog) for LIGO-VIRGO data exchange? Or is this data exchange of too low volume to be interesting?

- SDSS: TBD: The previous release of this document mentioned galaxy cluster finding application as a challenge problem. Support for public data access has also been mentioned – does this include application software execution, and, if so, does it call for a datagrid-based solution?

## 2.2 Project Year 2 Milestone Summary By Quarter

<u>Y2 Q1: Oct-Dec 2001:</u>

VDT 1.0 Release
CMS Virtual Data Demo
LIGO Virtual Data Demo
SDSS Challenge Problem Plan
ATLAS Challenge Problem Plan
ATLAS testbed upgraded to VDT 1.0
CMS testbed upgraded to VDT 1.0

<u>Y2 Q2: Jan-Mar 2002:</u>

VDT 2.0 Release
LIGO-CMS virtual data catalog unification
VDT virtual catalog component development
Virtual Data Catalog integration into CMS simulation production framework (MOP)
SDSS Grid testbed deployed (using VT 1.0)

<u>Y2 Q3: Apr-Jun 2002:</u>

CMS MOP component
CMS starts identification and design of Challenge Problem 3 (or 4), moving from the simulation process to reconstruction and analysis processes.

<u>Y2 Q4: Jul-Sep 2002:</u>

CMS request planning demo
ATLAS request planning demo

## 2.3 Research Objectives

### 2.3.1 <u>Overall</u>

[UC-Yong] Define "canonical virtual data problem" for use in development activities

[UC,ISI] (Overall) In collaboration with application scientists, complete Data Grid Reference Architecture v2 [Draft by July 15, final version following review by September 1].

### 2.3.2 <u>Virtual data</u>

Develop techniques for representing data transformations, and integrate these techniques into the information model. Develop methods and catalogs for categorizing and curating code elements. Develop information structures for identifying the control parameters of executable programs in a uniform manner that is amenable to automation of execution plans.

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

[UC,ISI,UW] Complete design of first version of virtual data catalog.

[UC,ISI] 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.

Develop tools for managing catalogs.

[?] Object / Relational issues: how to deal with data in databases instead of files.

[ UCB] relationshiop of db query planning to grid query planning

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

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

Koen:...add something about collaborating on the definition of a common job description language based on DAGs? Is this a CS goal?

Koen:...add monitoring?

Koen:...need to add something that integrates the experiments more with the scheduling CS research in GriPhyN? What are their plans anyway? I would love to see a CS-written grid scheduler take a location-independent DAG representing a CMS production job and map it to locations after which it is executed, however this is likely to be too ambitious for year 2?

#### 2.3.3 Storage Management

Define NeST/Globus/DRM integration with the goal of producing a managed storage element architecture.

Design how data can be striped for faster transfer. This will be essential to achieving the scalability goals that were described in the proposal ("peta-scale").

Design how data can be clustered, and events re-clustered and moved around for fast transfer. (Eg. Move 10M events (= 10 TB) from one Objy fed to another at ultra high speeds. Do same problem using both CMS and Objy and ATLAS/Athena. Can we saturate a 2.5GB link doing this? (This by the way would make a great SC challenge problem…)

[UW, UC] Develop plan for GridFTP/NeST integration, focusing in particular on space management.

#### 2.3.4 Request Planning

[UW] Complete work enhancing the ClassAd language to support events and triggers.

[UW] 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

[UW] 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. A key aspect of this language is that it must be capable of representing data derivation dependencies, so that the virtual data catalog can be populated and maintained through the interpretation of this language.

[UW] Enhance the "Gang Matching" capabilities of the ClassAd language and add these enhancements to the run-time support library.

[UW] Develop a protocol for information exchange between the execution and planning agents.

[NWU] item – Work on performance expectation modeling - Prophesy

[UCB] item – DB query stuff

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.

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 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.

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.

Develop an execution agent capable of receiving a simple plan form the planner and interacting with the PVDG services and resources in order to carryout the plan. 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. Develop a framework for gathering statistics on the resource consumption profile of completed and in-progress requests and the availability of resources.

[UW] 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.

[UC - Kavitha] (*We place simulation under planning because it can be used as an aid to developing better plans and to understanding the execution process better.*) Complete prototype of data grid simulator, with documentation. Initial paper evaluating alternative data replication strategies.

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.

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.

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

### 2.3.5 Request Execution

[NWU] item – Data collection and logging mechanism (object/event reference traces; Node execution traces (CPU load; net load; disk load); actual time vs predicted time. Job nature: cpu, I/O, network resources used.

[ANL-Jenny?] (Request execution) Develop a set of global and local policy scenarios that reflect the requirements of the user communities of the four physics experiments. The nature of these policies should specify what user and project jobs and data can utilize what storage, network, and computation resources, for what purposes, on what basis, what priority, what limits, etc. We need to understand how the policy model can be implemented, how the model affects and fits into the design of the CAS, and how it interacts with planning/scheduling decisions.

[ISI:Laura; UC:Von, ?] (Request execution) Develop uniform policy representation for code, data and resource access.

From Koen: ...need to add something that integrates the experiments more with the scheduling CS research in GriPhyN? What are their plans anyway? I would love to see a CS-written grid scheduler take a location-independent DAG representing a CMS production job and map it to locations after which it is executed, however this is likely to be too ambitious for year 2?

Enhance the "Gang Matching" capabilities of the ClassAd language and add these enhancements to the run-time support library.

Explore ways to enhance the ClassAd language to support events and triggers.

Resolve architectural differences between ClassAds and GRAM RSL.

Develop a protocol for information exchange between the execution and planning agents.

### 2.3.6 Fault tolerance

[UC: Anda, Matei; ISI: Ann, Ewa] Examine the issues of "consistency" in the data replication services and virtual data with respect to: (1) replicated metadata; (2) replicated data (being a grid file service, there is less trust one can place in repositories); (3) dependency tracking - being able to trace the effects of an error introduced by an application or filter.

[UCSD: Keith?] (Fault tolerance) Produce a paper detailing fault-tolerance issues in GriPhyN. Jenny Schopf will be involved, also perhaps Miron Livny. This model could/should deal with issues like job failure detection, cleanup, and restart. How failures should affect the planner and executor stages; how fault recovery should interact with the virtual data catalog.

[UCSD: Keith?] (Fault tolerance) Define fault tolerance model for replica management utilities. (& implement?)

## 2.4 Virtual Data Toolkit Development

**VDT-1 (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.

VDT 1.0 will provide three distinct software packages:

- Server code to be installed on a data grid node: (GSI, MDS, GRAM, GridFTP, Condor)
- Client side programs and libraries for use in client scripts and applications: DAGMan, Condor-G; client-side MDS, GRAM (?), GridFTP; replica catalog, replica management
- Standalone services: replica catalog

**VDT-1 (to be delivered by end of November 2001), consists of:**

*Instructions or script for installing a specific recent Condor release (including Condor-G and DAGMan).* The Condor part will be a "standard" release prepared and supported by the Condor Team. The DAGMan version may be a "non-standard" release that meets the special needs of GriPhyN and is maintained and supported by the VDT team.

*Instructions or script for installing a specific Globus 2.0-based release (including replica catalog and GridFTP).* The Globus (including GridFTP) part will be a "standard" release prepared and supported by the Globus Team. The Replica Catalog package within this release may be a "non-standard" release that meets the special needs of GriPhyN and is maintained and supported by the VDT team.

*Instructions or script for installing GDMP based on gridftp but not requiring Objectivity*. The VDT team will maintain and support this version if it requires changes for data grid use that are not yet available in the standard release.

*Configuration scripts for replica catalog, GridFTP, GDMP, and MDS that are specific to the GriPhyn/iVDGL test grid.*

Specific milestones:

Definition of VDT v1.0 components and negotiation of schedule: July 20
Creation of VDT campaign plan to obtain DSL resources (if needed).
Packaging and documentation of server-side functions (provided by GiB + GridFTP): Aug 1
Packaging and documentation of replica catalog server installation: Sep 1
Establish VDT support system based on the resources and capabilities of the GRIDS Center: Sep 1
Release of VDT 1.0: Sep 1
Dependency: need stable Globus Toolkit 2.0 (GT2)

**VDT-2 (Centralized Virtual Data Services) (to be delivered by Feb 1, 2002)**

VDT 2.0 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.

VDT 2.0 will add: CAS; GDMP support for Objectivity; RFT;
Initial Metadata catalog
Integration with Globus Replica Catalog 2.0
Virtual Data Catalog and Virtual Data Language interpreter
Initial (rudimentary) support for performance monitoring and logging
Condor enhancements: *identify*

VDT 2.1, etc: The remainder of the releases in project year 2 will consist of maintenance releases and improvements to the catalog data structures and the VDML virtual data manipulation language.

## 2.5 Common Planning Approach for All Experiments

This section describes the approach to be taken for planning each *experiment's* activities. A summary of this common approach is depicted and then described below. The figure below describes a one-year activity plan; each remaining year would follow a similar pattern. CS tasks are in blue, experiment tasks in yellow. The pattern shows about 2-3 VDT release per year, continuous CS research and application analysis (the latter at a steady but less intensive rate), and one cycle of challenge problem planning, development, and integration per year. As appropriate, the challenge problem cycle can be repeated several times per yar, possibly in an overlapped fashion, depending on the nature of the chosen problems. Note that the activities nearer the top of the figure feed the activities lower down, with the challenge problem solutions representing the ultimate GriPhyN goals.

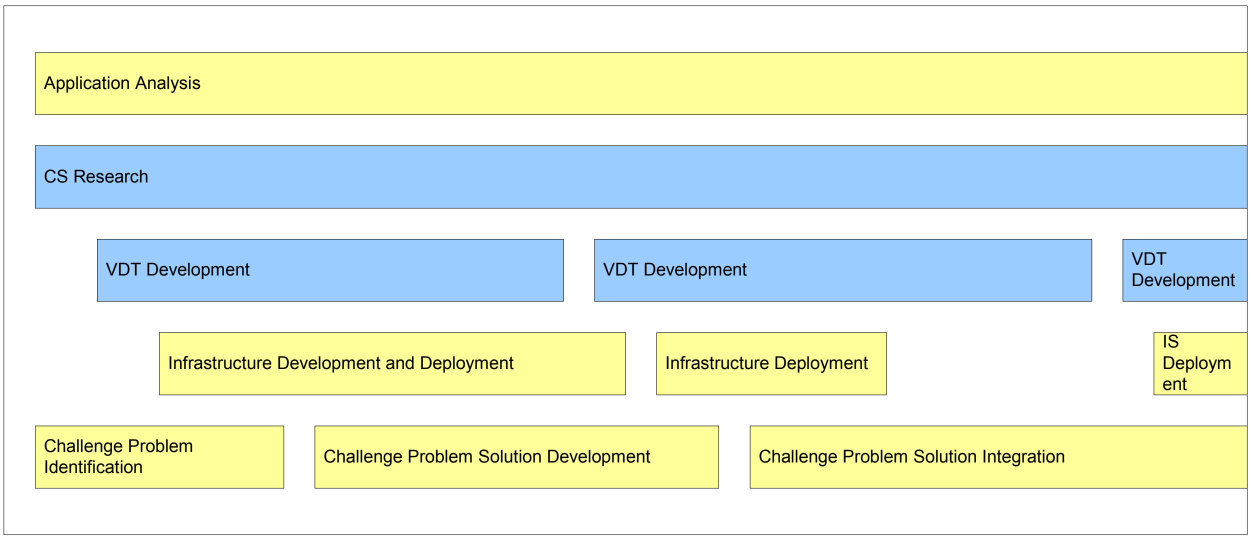


*Figure 1: Common Yearly Plan for Experiment Activities. (*Time --> *)*

### 2.5.1 <u>Application analysis</u>

This activity is critical to the relevance and utility to the experiments of GriPhyN results. We need to identify common information about each experiment in a common format using a common vocabulary. We need to determine the aspects if each experiment's data processing processes that are most relevant to GriPhyN's mission. Stated in a possibly over-simplified manner, the key question is: how do the experiment scientists process data? The information we need to capture is:

- data types stored
- definition of jobs and job types, what their control parameters are, and how they are expressed and invoked
- derivation dependencies for each data type
- a format for describing the process flow and dependencies
- the frequency and priority of each data object, program, and information process

We need to define the documents that will contain the analysis, and continually update them as part of the yearly process:

- Data and Application Map – a chart showing data types, application tools, and their dependencies at a glance
- Data dictionary – a description of each data file type
- Tool dictionary – a description of each relevant application
- Data requirements spreadsheet – a summary of quantitative data storage and transfer requirements

The documents should consist mainly of "man-page" style documentation that describes the specific details of specific data formats, tools, and science-driven IT processes. This style of documentation will be the easiest to write (or, in many cases, to simply gather from existing documentation) and grow, and will enable

We need to look for new ways to describe the rate of data production and consumption, and to describe the manner in which data items depend on and are derived from each other.

This activity will be carried out continuously and in parallel with other GriPhyN activities for each experiment.

### 2.5.2 Infrastructure building

In Year 2, there will be a larger focus on infrastructure identification and construction. We need to define the infrastructure that will be used for research, for challenge problem demonstration, and for GriPhyN result integration. 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.

The infrastructure tasks will typically include:

Identification of resources (hosts,…)

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
Catalog architectures
Namespace management

Its important to note that most of the CS research, VDT development, and Challenge Problem solution development activities will not require significant testbed resources. Resources for these GriPhyN activities can be maintained separately from testbeds actually managed by the experiments (for example, by using resources at UW, UFL, UTB, and ANL). The experiment-maintained testbeds (for example, the ATLAS testbed) would be used mainly in the challenge-problem integration and production 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).

### 2.5.3 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. 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. Its further complicated by the GriPhyN need to find common solutions across the experiments, (dictated by limited staff resources), which runs counter to the need typically felt by each experiment for precisely tailored custom solutions to their complex software problems.

We will apply an "intercept" approach 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.

(insert value proposition list for virtual data here – good list is in some earlier notes)

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.

#### 2.5.4 Challenge problem solution

Demonstration/proof-of-concept phase

Being able to create a compelling demonstration of the application of GriPhyN research is perhaps the single most important phase of the entire program. This is a good thing, because for most of us its also the most exciting and rewarding phase: proving that we can apply our research to the solution of practical problems.

#### 2.5.5 Challenge Solution Integration

QA cycles; test processes; resources (people, machines, and data) to execute tests.

Define the transition of support from GriPhyN staff to the staff of the experiment.

#### 2.5.6 Requirement Differences between Projects

The following factors may dictate differences between the 4 experiments' plans and require deviations from the common experiment activity template shown above:

Timelines; availability of and development schedules for new tools.
Technology bases (database and data storage technology; languages and compilers; application frameworks)
Object model differences
Science-driven data processing differences
Inter-grid-project dependency differences (eg, ATLAS and CMS influenced by both PPDG and EDG)
Coordination w/ other science projects:
  LIGO-VIRGO
  SDSS-NVO, (LSSC?)

### 2.6 ATLAS

#### 2.6.1 Goals

- Establish an ATLAS testbed linking resources with basic Grid services at N sites. Purpose: obtain expertise; provide a basis for experimental work.
- Conduct preliminary integration of the Athena analysis framework with initial virtual data services (replication only) and demonstrate effective operation on a testbed data challenge problem. Purpose: evaluate effectiveness of replica catalog tools in ATLAS context; obtain expertise.
- Complete design documents detailing ATLAS requirements for virtual data technologies. (These will be part of the Challenge Problem descriptions)

#### 2.6.2 People Involved

Randy Bramley and IU Active Notebook researchers (RB)
Rob Gardner (RG)
David Malon (DM)
Ed May (EM)
Jennifer Schopf and ANL/DSL students (JS)
Shava Smallen (SS)
Alex Undrus (AU)
Valerie Taylor and NU monitoring group (VT)
Saul Youssef (SY)

This list is from the 8/2-3 UC meeting, but may be to broad to manage under GriPhyN.

#### 2.6.3 Application analysis

[IU,ANL,NWU] Complete Version 2 of ATLAS Virtual Data Requirements document

Develop several revisions of :

  Data and Application Map
  Data dictionary

Tool dictionary
Data requirements spreadsheet

### 2.6.4 Infrastructure development and deployment

[?] Complete a document specifying in detail the testbed configuration, and which projects and people are responsible for creating it.

[IU] Build basic services for 1-2 prototype Tier 2 centers. [Need more info: what are "basic services"? What are the prototype centers? How does this relate to proposed ATLAS testbed? Is the AT a GriPhyN, PPDG, or ATLAS activity?]

[IU, ??] Deploy VDT services on a small number of machines at a small number of sites, identifying a skilled person at each site who is responsible for making this happen.

[IU] Develop a Condor-G interface to the ATLAS testbed

Bring all development nodes up to the level of VDT 1.0

Install VDT 2.0 on all development nodes

Bring all nodes for Challenge Problem production up to VDT 2.0

Implement Grid testbed between BNL (tier 1), ANL, LBNL(tier 2) prototype centers.

Items below are from the 8/2-3 UC Meeting. Tasks in this section that are not directly (or immediately) needed by the GriPhyN research plan should, potentially, be left to ATLAS to manage outside of GriPhyN.

ATLAS-capable testbed subset (ANL, BU, BNL)

- GridFTP must be installed on testbed nodes (EM?)
- "pacman –install atlas" (SY)
- mds-2 installation (JS)
- objectivity installed (EM?, DM?)
- GRIPE for accounts (RG)
- Add in Indiana to testbed
- Full testbed (add UTA, NERSC, UMich, OU) with install above?
- GDMP on Testbed nodes

GRAPPA using Condor-G (GRAM) submission [RB]

Functional 3 node testbed – ANL, BNL, BU [EM, JS, SY]

DBYA Using GridFTP [TW]

DBYA running on 2 testbed nodes [TW]

Grappa on 3 node testbed (running on one site, can submit a grappa job from any of the testbed nodes) [RB]

Check completion date estimate for setting up a grappa submission and then have the output files registered through DBYA (shared naming convention) [JS]

Register data outputs with Globus replica catalog [TW]

Moving data to an archival place [TW]

Application performance data logged [VT]

Check-in meeting in Boston (10/12)

GriPhyN All-hands meeting in CA (10/15)

Grappa and Athena on 1 node of testbed [SS, RB]

Submit jobs using condor-g/dagman (maybe through GRAPPA) [RB]

Monitoring and logging messages related to file usage, part of GRAPPA [VT]

Test gridftp on sites

Install Objectivity at BU

### 2.6.5 Challenge-problem Identification

Create ATLAS Challenge Problem 4-year roadmap

Challenge Problem 1 description document: software to be produced, experiments to be conducted

*Potential challenge problems are:*

*CP1:* Testing of basic file replication, transport, and virtual data tracking using 500 GB testbeam data sets.
*CP2:* Implementation of PPDG/GriPhyN tools into ATHENA (Atlas analysis framework) for Monte Carlo simulation.
*CP3:* Using multi-site replication and caching services for test beam data analysis and simulation at scale of 2-10TB.

A challenge problem sketch that was created at the 8/2-3 UC Meeting was:

Have different generator sets, run through simulation, generate reconstruction, run it through analysis
Monitor performance, harvest information
Use replica catalog and replica selection
Data resident at tier 1, any of the tier 2's need to be able to access it through the tier 2's

### 2.6.6 Challenge-problem Solution Development

(Requirement: needs VDT 2.0)

[IU] Develop and demonstrate use of Condor-G interface to the ATLAS testbed and for computationally intensive analysis of ATLAS data, using ATLAS testbed.

[IU] Complete testing of basic file replication and transport using 500 GB testbeam data sets. (Are these the Tile Calorimeter test beam datasets that Ed May is working on?)

[IU] Connect the Athena analysis framework to a set of prototype virtual data services. Begin with the Globus replica catalog service; incorporate Athena EventSelector service to replica catalog (data reading); add Athena Replica catalog update service (data upload).

### 2.6.7 Challenge-problem Solution Integration

Document the solution

Install and integrate the solution

Package the solution and transition its support to the experiment

[IU?] What about GRAPPA work? -> Move to research section? Becomes part of a challenge problem?

### 2.6.8 Dependencies

TBD:

- ATLAS Challenge Problems may require VDT 2.0

## 2.7 CMS

### 2.7.1 Goals

*Y1:* Build basic services and 1-2 prototype Tier 2 centers.

*Y1:* Complete High Level Trigger milestones and perform studies with ORCA, the CMS object-oriented reconstruction and analysis software

*Y2:* Initial Grid system working with Tier 1 center.

Work with the Condor team to develop DAGMan further, in particular its expressiveness in terms of error recovery, with the goal of applying it to the CMS production system. [UF taking lead]

Work further on exploring the impact of end-user physics analysis workloads on the grid system, by prototyping distributed end user analysis tools, demos, and pilot facilities which allow end-user physicists without specific grid training to accomplish basic physics data manipulation tasks using the grid. Show user collection creation and transport over the grid, driven by an easy-to-understand grid interface. [CIT talking lead]

Show the utility of Grid technology as a basis for enhancing the robustness and reproducibility of distributed computing, by integrating grid components more deeply into the CMS production software, with the results of the integration being used either in real production or in challenge demos. [UF taking lead, heavy interaction with PPDG, US production, FNAL]

File replication service.

Multi-site cached file service

#### 2.7.2 People Involved

James Amundson
Lothar Baurdick
Rick Cavanaugh
Greg Graham
Koen Holtman
Rajesh Rajamar
Jorge Rodriquez

#### 2.7.3 Application analysis

Develop several revisions of :

Data and Application Map
Data dictionary
Tool dictionary
Data requirements spreadsheet

Work with the Globus team and the EU DataGrid to develop a set of long-term scalability requirements for the file replica catalog service.

Do further work on the issue of reconciling the object and object-collection nature of the CMS data model with the file nature of the low-level data grid services.

Actively participate in the development of a Grid architecture by reviewing architectural documents created in the Grid projects, and by communicating architectural lessons learned in CMS production to the Grid projects.

#### 2.7.4 Infrastructure development and deployment

[CIT, UF, ??] Deploy VDT services on a small number of machines at a small number of sites, identifying a skilled person at each site who is responsible for making this happen.

[CIT,UF] Build basic services at 1-2 prototype Tier 2 centers. [Need more info: what are "basic services"? What are the prototype centers? How does this relate to testbed?]

#### 2.7.5 Challenge-problem Identification

Create CMS Challenge Problem 4-year roadmap

Challenge Problem 1 description document: software to be produced, experiments to be conducted

*Potential challenge problems are:*

Demonstrate the robustness gains of the use of DAGMan in a realistic CMS production setup by doing a challenge demo which includes at least 3 sites. In this demo, certain times system crashes will be injected to show the capability of the system to auto-recover without human intervention.

Demonstrate virtual location of data at a file level in a realistic CMS production setup by doing a challenge demo which includes at least 3 sites.

The CMS challenge problem for this year is to integrate virtual data and request planning concepts into the CMS Monte Carlo Production system (MOP).

Request estimation?

Request estimates stored in VDC?

[CIT,UF] Complete High Level Trigger milestones and perform studies with ORCA, the CMS object-oriented reconstruction and analysis software. [More details! How does this use VDT services? Need to make clear how this relates to GriPhyN.] MW: is this software simulation of the hardware HLT? If so, how does it relate to the MC production that's part of MOP? Same, similar, or very different?

#### 2.7.6 Challenge-problem Solution Development

#### 2.7.7 Dependencies

Ability and willingness for CMS to let us use MOP as a challenge problem and bring modified versions of it into production.

Requires VDT 2.0

(Koen)...MOP, PPDG, testbeds

(Koen)...could construct some dependencies here based on CMS goals above...

## 2.8 LIGO

Goals:

The challenge problem for this year is to grid-enable the scientific analysis of gravitational lensing application codes.

Integration with HPSS.

Unification of the catalog schemes used by CMS and LIGO in Y2Q1 – basing it on a common VDT 2.0 release.

Establishment of a test grid spanning CIT, UW-Madison, and UW-Milwaukee.

Integration of XSIL and LDAS concepts into the VDT; Grid-enabling of LDAS

LIGO goals during Year 1, and some of Year 2, are to:

- Advance virtual data concepts by defining a virtual data language and architecture. Purpose: explore virtual data concepts, develop understanding of LIGO VDT requirements.
- Implement a LigoVista Web display. Purpose: why?
- Apply replication concepts by developing a real-time international mirror, and a fault-tolerance replica at UW-Mil. Purpose: develop expertise, develop software.
- Explore scheduling issues by performing a large parallel pulsar search. Purpose: evaluate software.

Specific milestones are as follows:

L.1.a. [CIT] Q1/01 - Q1/02: Extract code from LDAS environment and "grid-enable" it. [Jul 1] [Done?]

L.1.b. [CIT,USC,UW-Mil] Complete a document detailing the work to be finished during Year 1, indicating the testbed configuration, software to be produced, experiments to be conducted [Aug 1].

L.1.c. [CIT,UW-Mil] Build basic services at 1-2 prototype Tier 2 centers. [Need more info: what are "basic services"? What are the prototype centers? How does this relate to testbed?]

L.1.d. [CIT] Implement simple filters and math transformations. [Need details.] [Oct 1]

L.1.e. [CIT] Define Web service and prototype without data. [Need details.] [Oct 1]

L.1.f. [CIT] Implement replication LIGO-VIRGO with GridFTP. Document performance. [Oct 1]

L.1.g. [CIT] Design interface for reliable mirroring, implement GridFTP at CACR. [Oct 1]

L.1.h. [CIT] Start work on implementing checksum/backup with GridFTP

Known dependencies:

- Note that the "implement checksum/backup with GridFTP", scheduled for Q1/02, may be done by ANL group?
- More? Not very VDT-specific??

Milestones:

Work on LIGO continues with work started in the last period. Specific milestones are as follows:

L.2.a. [CIT] Use LIGO virtual data to drive display. [April 1, 2002]

L.2.b. [CIT] Complement implementation of checksum/backup with GridFTP [April 1, 2002]

L.2.c. [CIT] Implement replica catalog and conduct full-scale replication [July 1, 2002].

L.2.d. [CIT] Start work on runs and validation of pulsar search code with Condor farms.

L.2.e. [CIT] Continue runs and code validation Condor farms [Jan 1, 2002]

L.2.f. [CIT] Full-scale search with LIGO science data. [Apr 1, 2002]

??? also: Perform LIGO-VIRGO data transfer using grid tools?

Dependencies:

*Y1: LIGO:* Develop a cataloging approach for data access methods and data location (metadata definition, design).

*Y1:* Develop an access and use model for LIGO data across the GriPhyN system.

*Y2: LIGO:* Demonstrate efficient access to LIGO data via GriPhyN caches.

## 2.9 SDSS

Milestones:

Goals:

- 

Milestones:

- Install VDT toolkit on …

Grid-enabled galaxy cluster finding code. (*Need to define what Grid enabled means… Catalogs? Job defns? GRAM? Etc?)*

- Test replication of SDSS databases using existing grid infrastructure.
- Install first generation hardware.
- Integrate with SDSS early data release.

- Install second generation iVDGL software.
- Tests of code migration to other iVDGL sites.
- Grid-enable gravitational lensing application code.
- Integrate first SDSS data release.

Dependencies: TBD: We need to say what is expected, who will produce it, what we do if it doesn't arrive.

*Y1: SDSS:* Build a prototype distributed analysis system

*Y2: SDSS:* Bring the system into production, open it to project

## 2.10 Education and Outreach (to be addressed in detail in next revision; see Manuela's mail)

Specific milestones are as follows:

O.1.a. [UTB] Install VDT software on UTB cluster (Oct 1).

O.1.b. More …: Facilities, Programs, Seminars, Papers, Activities, Internships/Co-Ops

Q4-FY00: Begin search for E/O coordinator
Begin benchmarking tests for UTB linux cluster

Q1-FY01: Search for E/O coordinator continues

Benchmarking continues; Order equipment

Search completed (M. Campanelli hired)

Q2-FY01: Begin construction of UTB linux cluster

E/O coordinator makes contacts with European data grid projects, other E/O projects (EOT-PACI, QuarkNet, ThinkQuest), and possible tier3 centers for the iVDGL proposal

Construction of UTB cluster completed

Q3-FY01: Begin installing condor, globus on UTB cluster

Q4-FY01: E/O coordinator starts at UTB

Prepare and submit proposal for REU supplement for iVDGL (if

funded)

Continue grid-enabling UTB cluster

Design simple (passive) web-page for GriPhyN E/O activities

## 2.11 Coordination

We will hold the following meetings:

- All-hands meeting, LA, Oct 15-17, 2001.
- CS research meeting(s): to be defined.
- Application/VDT integration meeting: TBD.
- Jan 2002 at UF meeting?
- All-hands meeting, Chicago, April 8-10, 2002.

# 3 Year 3: October 1, 2002 to September 30, 2003

## 3.1 Overall Goals

## 3.2 Research Accomplishments

***Year 3***

*Virtual data:* 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.

*Request Planning:* 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.

Request Execution: Develop a fault-tolerant version of the execution agent.

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

Add fault-tolerance to the manager and reliability to basic propagation protocols.

Develop a fault-tolerant and persistent repository of PVDG statistics

## 3.3 Virtual Data Toolkit Development

**VDT-3 (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 version will support O(100) TB datasets, O(10 TB) network caches, O(1000) CPUs, and O(400 MB/s) networks.

## 3.4 ATLAS

*ATLAS:* Implement Grid testbed between CERN (tier 0), BNL(tier 1), ANL, LBNL, BU, IA(tier 2)

Begin multi-site distributed simulations to generate 100TB Monte Carlo for Physics studies and Mock Data Challenges.

## 3.5 CMS

*CMS:* Second set of Tier 2 centers.

CMS data challenges.

Software and Computing Technical Design Report (TDR)

## 3.6 LIGO

*LIGO:* Develop a robust distributed computing model for using GriPhyN to process continuous wave gravitational wave searches across a grid of computing resources.

## 3.7 SDSS

Begin integration of National Virtual Observatory infrastructure with iVDGL technology. Large scale production runs on core science using half SDSS dataset. Update to second generation hardware. Integrate third SDSS data release.

*SDSS:* Design prototype distributed public analysis system, full production use internally

## 3.8 Education and Outreach

Q1-FY02: Begin extension of web-interface to SDSS data to include concept of virtual data

Q2-FY02: Continue work on SDSS web-interface

# 4 Year 4: October 1, 2003 to September 30, 2004

## 4.1 Overall Goals

## 4.2 Virtual Data Toolkit Development

**VDT-4 (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.

## 4.3 Research

***Year 4***

*Virtual data:* 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.

*Request Planning:* 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.

*Request Execution:* 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

## 4.4 ATLAS

*ATLAS:* Mock Data Challenge 2: Distributed Data Analysis Using 00TB Monte Carlo sample based on ATHENA Framework using GriPhyN tools between CERN/BNL/ANL/LBNL/UM/BU/IU testbed grid.

## 4.5 CMS

*CMS:* Tier 2 centers at last set of sites.

5%-scale data challenge.

Physics TDR; production Data Grid test.

## 4.6 LIGO

[CIT] Full-scale search with LIGO science data. [Apr 1]

*LIGO:* Develop query tools to search (mine) event databases derived from search algorithms which pre-process LIGO data.

## 4.7 SDSS

Implement grid-enabled tools for statistical calculations on large scale structure. Begin full scale tests with additional iVDGL sites. Integrate second SDSS data release.

*SDSS:* Start using public system, monitor usage

## 4.8 Coordination

We will hold the following meetings:

- CS research meeting(s): to be defined.
- Application/VDT integration meeting: TBD.
- All-hands meeting, Chicago, April ??-??, 2003.

# 5 Year 5: October 1, 2004 to September 30, 2005

## 5.1 Overall Goals

## 5.2 Virtual Data Toolkit

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

## 5.3 Research

*Virtual data*: Augment information model to incorporate local and global policy constraints.

*Request Planning*: 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.

*Request Execution*: 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.

## 5.4 ATLAS

*ATLAS:* Build production quality offline distributed data analysis for ATLAS Grid using GriPhyN tools. Use in Mock Data Challenge 3.

## 5.5 CMS

*CMS:* Production-quality Grid system.

20% production CMS mock data challenge.

## 5.6 LIGO

*LIGO:* Optimization of Grid-based techniques to process time series data to develop event databases.

## 5.7 SDSS

Large scale production runs on core science using full SDSS dataset. Conduct initial joint SDSS/NVO analyses. Integrate final SDSS data release.

*SDSS:* Full production mode of public analysis system

## 5.8 Coordination

We will hold the following meetings:

- CS research meeting(s): to be defined.
- Application/VDT integration meeting: TBD.
- All-hands meeting, Chicago, April ??-??, 2004.

**Plan To Dos**

Create a list of specific research projects and their relevances and plan for integrating their results into the overall G plan.

Which projects need HPSS access? How many phases (proto vs production)? How to integrate into the plan? How related to PPDG and the ASCI DRM work? (Who was doing the basic integration of GSI-based HPSS first?)

Define what system resources each organization needs to meet the plan, I particular, the test grirds that each exp wil need to set up: details of who what where when why.

List deliverables that GriPhyN will draw from SciDAC Security and Mware

**Open Issues**

How much of each project's infrastructure do we really want to get into planning here? I propose that we do the *minimal* infrastructure effort that we need in order to achieve our goals. I would saythat our focus should be on the research and challenge activities – those are the things that seem to be the essence of GriPhyN.

How to coordinate the establishment of test grids with other projects: PPDG, EDG, etc? (Mainly PPDG). Whose resources? Who sets up the grids? How does resource allocation get divvied up (CPU time, disk space, netwk bw?)

Where will petascale performance come from? The proposal clearly identified this as a goal, but we still need to identify the work tasks that are needed to enable petascale computing. The sources of these technologies need to be clearly identified, especially in the context of the VDT. When we talk about, say, O(10TB), O(1PB), we need to state how this will be done. In HEP, a PB is 1G events. We just need to state how we will make 1PB of disk and tape interact and be shared. Where will n PBs exist in the multi-tiered grid?

What is Virtual Sky Project and how does it fit into the GP project? (Was this the outreach project in Joe's task list?)

What project plan items come out of the "Cost Sharing" resources?

How do we show in the project plan how research "scouts ahead" of VDT and challenge problems to create solutions?

**Hard Issues**

Scheduling: degrees of freedom; policy interaction. How/where in this plan do we want to clarify our goals in this area.