Integrating Web 2.0 Technologies with Scientific Simulation Codes for Real-Time Collaboration

Gabrielle Allen[1,2], Frank Löffler[1], Thomas Radke*[3], Erik Schnetter[1,4], Edward Seidel[4,5]

[1] *Center for Computation & Technology, Louisiana State University, Baton Rouge, LA, USA*
[2] *Department of Computer Science, Louisiana State University, Baton Rouge, LA, USA*
[3] *Max-Planck-Institut für Gravitationsphysik, Albert-Einstein-Institut, Golm, Germany*
[4] *Department of Physics & Astronomy, Louisiana State University, Baton Rouge, LA, USA*
[5] *National Science Foundation, Arlington, VA, USA*

***Abstract*—This paper motivates how collaborative tools are needed to support modern computational science, using the case study of a distributed team of numerical relativists developing and running codes to model black holes and other astrophysical objects. We describe a summary of previous tools developed within the collaboration, and how they are integrated and used with their simulation codes which are built using the Cactus Framework.**

We also describe new Cactus tools which use the Twitter and Flickr services. These tools are fundamentally integrated with the code base as Cactus modules and provide reliable, real-time information about simulations that can be easily shared across a collaboration.

I. Introduction

Science is undergoing a rapid transformation at rates unseen in history. In the field of numerical relativity it is now nearly 400 years since Galileo discerned the basic laws of the solar system. For most of the time since then, scientists have worked in small groups, with modest amounts of data collected in notebooks, publishing results in traditional journals. Only in the last several decades has this changed, and the changes are occurring on exponential growth paths. Hawking sketched out the basic theoretical framework for two black holes colliding [1], while Smarr carried out early numerical simulations [2]. Both of these pioneering efforts generated kilobytes to perhaps a megabyte of data, and involved very small teams (e.g. one extraordinary individual!). About 20 years later, in the early 1990's, this same problem was revisited, but now by a team of perhaps 10 people distributed across different sites, generating a thousand times as much data [3]. A few years later, the same problem was studied in full 3D [4], and the team size had grown to perhaps 15, the number of institutions had grown, spanning two continents, while the data generated increased by another factor of thousand.

Problems now being addressed by the same communities, such as the gamma-ray burst problem [5], will require still larger collaborations of groups coming from completely different communities, generating petabytes of data. Hence, after nearly 400 years of slow growth, the last three decades have seen exponential growth in the scale and complexity of the collaborations, the software environments needed to support them, and the amount of data being generated (roughly 12 orders or magnitude increase in data volumes when the next generation petascale facilities are brought online).

At the same time that the theoretical and computational side is undergoing such transformation, likewise experimental activities are also seeing historic growth. Gravitational wave detectors, such as LIGO, GEO, and VIRGO are already generating data at the petabyte level, from which signals from colliding black holes may soon be discovered for the first time. This same trend, illustrated above in gravitational physics, is found in all areas of science and engineering, from astronomy to geosciences to social and economic sciences. The face of science is changing dramatically.

The methodologies of science are naturally changing dramatically as well. This places strong demands on the cyberinfrastructure needed to support the new science. In order for scientific collaborations to work effectively at the this scale, for the data volumes to be handled properly by the collaboration, and for the science results so-generated to be reproducible, new methods of collaboration must be developed, new software environments must be created, and new tools for analysis, visualization, and tracking and recreating the data (and the software codes that generated it) must all be developed.

In this paper we describe new technologies that are being used to address these issues. Web 2.0 technologies are providing new opportunities for cooperation and collaboration in society that have the potential to also transform how we perform scientific research. The literature already describes [6] how Web 2.0 technologies and concepts such as blogs (e.g. [7], [8], [9]), wikis (e.g. Wikipedia, or GRwiki [10] in numerical relativity), and social networks (e.g. LinkedIn, FaceBook, SciLink [11]) are changing how researchers disseminate, publish, and critique scientific information and interact with their peers. In fields involving geospatial concepts (e.g. ocean science, disease modeling) the capabilities provided by Google Maps have been exploited for some years [12]. Web 2.0 is also changing on a global scale how we share publications and data, with new journals and methodologies starting to appear for reviewed digital content [13]. In many cases these new tools are providing an improved model for implementing ideas that in the past have necessitated a more ad-hoc home-grown infrastructure. For example the purely on-line journal Living

*Deceased 2 June 2009

Reviews in Relativity [14] was created in 1998 to provide for open access up-to-date (living) reviews exploiting new content delivery capabilities provided by the web.

While there is some skepticism about the real impact of Web 2.0 on scientific endeavors, and whether this is simply one more buzz word or a new platform for collaboration, this paper describes how new Web 2.0 tools can now easily be integrated into scientific simulation codes to provide capabilities that have been desired and prototyped for at least the last decade. These tools are now being used to support collaboration for researchers involved in large scale scientific computing with the Cactus Framework.

This work builds on a long term search for appropriate collaborative tools to support distributed teams working in computational science based on our experiences since 1991 working with the numerical relativity research teams at the National Center for Supercomputing Applications, the Albert Einstein Institute, Washington University, and Louisiana State University. In Section II we describe our scientific simulation software — the Cactus Framework — and show how Cactus already provides a collaboration-friendly environment which can now exploit Web 2.0. In Section III the numerical relativity use-case is described, highlighting the need for enabling tools. In Section IV we present a summary of past tools and discuss their motivation, successes and shortcomings. Section V highlights issues with and abstract requirements for collaborative tools based on our past experiences, Section VI describes the features of the Web 2.0 technologies we have integrated with Cactus, and Section VII provides details of their implementation and use.

II. Core Scientific Software: The Cactus Framework

The work described in this paper has produced new collaborative components for the Cactus Framework. Cactus ([15], [16]) is a parallel programming environment for scientific computing that was created by the numerical relativity community but is now used in several other scientific domains.

The name Cactus is a metaphor for its design: The *flesh* provides a central set of infrastructure and interfaces upon which programmers can build and integrate components called *thorns*. Thorns can provide code for mesh generation, adaptive mesh refinement, I/O, checkpointing as well as physics modules for fluid dynamics, black hole physics, etc.

The modular structure of Cactus was specifically designed to support collaborative code development. Each thorn for Cactus typically performs a single, well-defined task, for example providing an I/O method for producing jpeg images, or the initial data for a particular science problem. Thorns are described by four configuration files written in the Cactus Configuration Language (CCL) which specify the variables and functions, parameters, scheduling and compilation requirements. Scientists use Cactus by providing a list of thorns needed for their work (a Cactus *ThornList*) which are compiled together to build an executable, where the build system generates the necessary binding code at compile time from the CCL files. The executable is run using a parameter file with key value pairs which also sets the *active thorns* for the simulation. Parameters can be designated by thorn developers as *steerable*, meaning that their values can be changed at run time, providing a mechanism for example to change the amount or frequency of data output, activating new analysis methods, or reacting to the simulation results to change physical details of the run.

The Cactus flesh and a set of core thorns called the *Cactus Computational Toolkit* are developed and distributed via a source code repository hosted at Louisiana State University. Research groups using Cactus typically use the core thorns, develop their own sets of thorns, and additionally take advantage of public or private thorns developed in various other groups and distributed from diverse servers.

Aside from numerical relativity described in more detail in the next section, Cactus is also used as the underlying parallel framework for developing toolkits for other application domains, including coastal modeling, computational fluid dynamics, reservoir simulation, and quantum gravity. The new thorns described in this paper can be used, *without change*, by these applications and others to immediately provide collaborative capabilities. Further, since Cactus and its core computational toolkit is distributed under an open source license (LGPL), the code can also easily be integrated with other stand-alone applications.

III. Motivating Science Case: Numerical Relativity

Since Smarr's seminal work from 1970's, efforts have been underway to use computers to model objects in the universe governed by Einstein's general theory of relativity, a field of research called *numerical relativity*. Einstein's equations govern gravitational waves, black holes, cosmology, neutron stars, supernovae, gamma-ray bursts [17] and other phenomena involving the motion of large, dense masses. The complexity of Einstein's equations has led to advanced computational models requiring the use of cutting edge hardware, and to interdisciplinary researcher teams working together from fields such as astrophysics, computer science and applied mathematics.

In numerical relativity, over 15 research groups have adopted the Cactus Framework and the associated Carpet infrastructure for Adaptive Mesh Refinement as their underlying parallel framework and community toolkit. One such group is found at the Center for Computation & Technology at Louisiana State University. This team of around ten researchers working on evolving black holes and neutron stars typically have simulation codes comprised of around 300 modules. The group works very closely with a large team at the Albert Einstein Institute in Potsdam, Germany, and on different projects with colleagues at Georgia Tech, the Rochester Institute of Technology, University of Maryland, NASA Goddard, and Caltech, among others. The group uses about 20 different supercomputers located at different sites and via state-wide or national resource providers such as LONI [18] and the TeraGrid [19]. In the different projects, collaborative tools that respect the boundaries of the different partnerships are needed to support code development and testing, configuring the physical and computational set ups for computational surveys, analysis and interpretation of results.

The simulations the groups run on these machines typically use between 100 and 2000 cores or processors, and can run for days or weeks in batch systems. Often, due to the limited batch queue length on many supercomputers, simulations need to be *checkpointed* and later *restarted* multiple times until they complete. The runs produce between gigabytes and terabytes of data output, contained in multiple files. Large 3D and 2D data sets are written using a standard binary format, but also ASCII files and 2D slices in jpeg format are created.

IV. Collaborative Technologies in Numerical Relativity

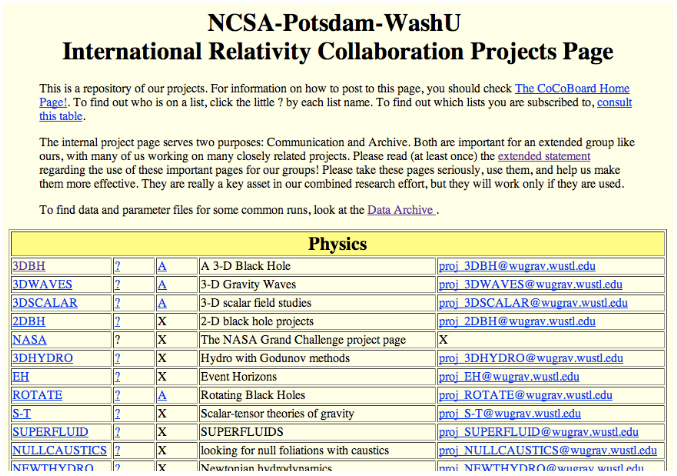
A. *Web-Based Mail Lists*

Tools for collaboration in science were greatly enabled by the arrival of the world wide web in the early 1990's. The first widely used web browser was Mosaic, which originated in 1993 at the National Center for Supercomputing Applications (NCSA) in Champaign-Urbana. At the NCSA, another group led by Seidel was also working at this time in numerical relativity to model colliding black holes, and collaborating closely in this endeavor with a team from Washington University in St. Louis. (This collaboration generated the black hole collision simulations referred to in the introduction [3].) To facilitate collaboration between the groups, a mail list technology called "CoCoBoard" (Collaborative Cork Board) was developed in Seidel's group [20] to provide the researchers with web-based "Project Pages", which provided a set of mail lists for different aspects of the black hole problem, which could be accessed and configured via the web, and to which group members could attach images which would also be displayed via the web. When the NCSA numerical relativity group moved to the Albert Einstein Institute in Germany in 1996 these pages were extensively used for several years (until other mail list technologies provided more features) as the focus of coordination and collaboration. The pages included 33 "projects" divided into areas of Computation, Physics, and Mathematical/Theoretical/Tool development (Figure 1).

Since 2003 the core coordination of group activities uses project based private wikis, where individual researchers in the collaboration post detailed simulation results by hand or using simple scripts (e.g. parameter files, output files, figures) for others to see. This material is an essential aid to accompany weekly phone conferences between the distributed groups.

B. *Thorn HTTPD and Simulation Web Interfaces*

The first collaborative tool fundamentally integrated into the Cactus Framework was a web server module, thorn `http`, that allowed scientists to connect directly to a simulation and monitor the status of the run. The web server module was originally developed by Werner Benger in 1999 while he was visiting the NCSA from Germany to work on visualizations of Cactus simulations that were modeling the first collisions of



NCSA-Potsdam-WashU
International Relativity Collaboration Projects Page

This is a repository of our projects. For information on how to post to this page, you should check The CoCoBoard Home Page!. To find out who is on a list, click the little ? by each list name. To find out which lists you are subscribed to, consult this table.

The internal project page serves two purposes: Communication and Archive. Both are important for an extended group like ours, with many of us working on many closely related projects. Please read (at least once) the extended statement regarding the use of these important pages for our groups! Please take these pages seriously, use them, and help us make them more effective. They are really a key asset in our combined research effort, but they will work only if they are used.

To find data and parameter files for some common runs, look at the Data Archive .

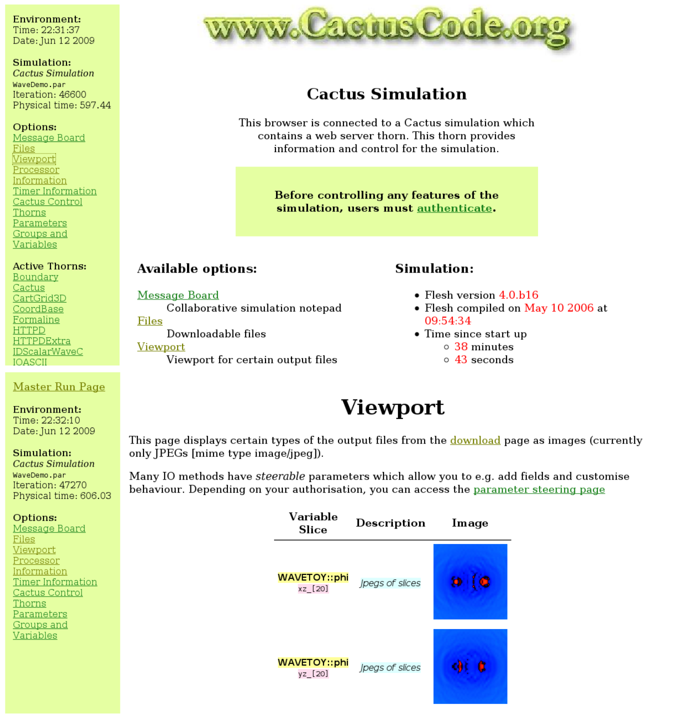
Physics				
3DBH	?	A	A 3-D Black Hole	proj_3DBH@wugrav.wustl.edu
3DWAVES	?	A	3-D Gravity Waves	proj_3DWAVES@wugrav.wustl.edu
3DSCALAR	?	A	3-D scalar field studies	proj_3DSCALAR@wugrav.wustl.edu
2DBH	?	X	2-D black hole projects	proj_2DBH@wugrav.wustl.edu
NASA	?	X	The NASA Grand Challenge project page	X
3DHYDRO	?	X	Hydro with Godunov methods	proj_3DHYDRO@wugrav.wustl.edu
EH	?	X	Event Horizons	proj_EH@wugrav.wustl.edu
ROTATE	?	A	Rotating Black Holes	proj_ROTATE@wugrav.wustl.edu
S-T	?	X	Scalar-tensor theories of gravity	proj_S-T@wugrav.wustl.edu
SUPERFLUID	?	X	SUPERFLUIDS	proj_SUPERFLUID@wugrav.wustl.edu
NULLCAUSTICS	?	X	looking for null foliations with caustics	proj_NULLCAUSTICS@wugrav.wustl.edu
NEWTHYDRO	?	X	Newtonian hydrodynamics	proj_NEWTHYDRO@wugrav.wustl.edu

Fig. 1. Project pages using the CoCoBoard web tool supported an active collaboration in numerical relativity between Washington University and the Albert Einstein Institute between 1997 and 1999.

grazing black holes [4]. The thorn used a socket library and techniques developed for remote visualization by John Shalf and the German TiKSL project [21]. At this time, there was no easy way to find out the status of the simulations running at different locations that may or may not have been successfully producing suitable data for him to use. Such a collaboration with colleagues in Germany was particularly challenging, at that time email was the only real method for contact and the seven hour time difference and lack of Internet in most homes meant that interactions took place on a timescale of days.

This initial web server module was rewritten and released as part of the Cactus Computational Toolkit in September 2000 as thorn `HTTPD`, providing a general Cactus thorn that could be added to any Cactus application to provide a web interface. A mechanism was provided for applications to write their own content to provide through HTTPD, however most applications simply used a second thorn called `HTTPDExtra` which provided standard content, including:

- An overview of the simulation status, including host machine details, simulation time and iteration number, version of the code used, parameter files used, average CPU time taken per timestep, etc. (Figure 2).
- Complete information on all Cactus variables and parameters in the simulation, e.g. their data type, storage, description, names. For parameters their allowed range and actual value are provided.
- Timing information, showing values of the different Cactus timers.
- Authentication to a steering interface for parameter steering, simulation halting and stopping.
- The simulation viewport, which embeds jpeg images of 2D slices through chosen grid variables at the current time, showing a visual status of the simulation (Figure 2).
- A list of all the output files that have been registered with the Cactus I/O layer, allowing easy remote download of the files to a local machine. Using standard MIME



Environment:
Time: 22:31:37
Date: Jun 12 2009

Simulation:
Cactus Simulation
`WaveDemo.par`
Iteration: 46600
Physical time: 597.44

Options:
Message Board
Files
Viewport
Processor Information
Timer Information
Cactus Control
Thorns
Parameters
Groups and Variables

Active Thorns:
Boundary
Cactus
CartGrid3D
CoordBase
Formaline
HTTPD
HTTPDExtra
IDScalarWaveC
IOASCII

Cactus Simulation

This browser is connected to a Cactus simulation which contains a web server thorn. This thorn provides information and control for the simulation.

Before controlling any features of the simulation, users must authenticate.

Available options:

Message Board
    Collaborative simulation notepad
Files
    Downloadable files
Viewport
    Viewport for certain output files

Simulation:

- Flesh version 4.0.b16
- Flesh compiled on May 10 2006 at 09:54:34
- Time since start up
  - 38 minutes
  - 43 seconds

Master Run Page

Environment:
Time: 22:32:10
Date: Jun 12 2009

Simulation:
Cactus Simulation
`WaveDemo.par`
Iteration: 47270
Physical time: 606.03

Options:
Message Board
Files
Viewport
Processor Information
Timer Information
Cactus Control
Thorns
Parameters
Groups and Variables

Viewport

This page displays certain types of the output files from the download page as images (currently only JPEGs [mime type image/jpeg]).

Many IO methods have *steerable* parameters which allow you to e.g. add fields and customise behaviour. Depending on your authorisation, you can access the parameter steering page

Variable Slice	Description	Image
WAVETOY::phi xz_[20]	*Jpegs of slices*	
WAVETOY::phi yz_[20]	*Jpegs of slices*	

Fig. 2. The upper screen-shot in this figure shows the main page of the Cactus thorn HTTPD. General simulation is shown along with a list of options to control the output of HTTPD and the simulation itself. The second screen-shot shows the Viewport embedded in HTTPD displaying 2D-slices of one of the evolved variables of the simulation.

capabilities of browsers it is possible with a single click on a remote file for it to be downloaded to the local machine and immediately visualized with standard scripts installed on the machine. One major problem with this however is that every user needs to configure all of his machines for this, and configuring is different for all operating systems and browsers.

The web pages are public for viewing, but interacting with (modifying or aborting) the simulation requires authentication. User names and password are provided in a parameter file together with other simulation parameters. (These passwords are explicitly filtered out when parameters are displayed.) This mechanism is simple, but is not secure, and requires sharing the password to give access to a group of people.

A new version of `HTTPD`, called `HTTPS`, was developed in the AstroGrid project in 2007 ([22], [23]). `HTTPS` provides essentially the same feature set, but uses the secure `https` communication protocol instead of the insecure `http` protocol. It also uses X.509 certificates [24] instead of a username/password mechanism for authentication. While secure, this mechanism has other drawbacks that we describe in Section V below.

A thorn `Visualisation` accompanies `HTTPS`. This thorn uses the external plotting package `gnuplot` (that must be installed on the system where Cactus executes) to produce graphs of various data sets. Which data are displayed can be chosen and modified at run time. This allows users to quickly grasp the state and health of the simulation, viewing e.g. the speed of the simulation over time, or (in the case of relativistic astrophysics simulations) the time evolution of the masses of black holes or the amplitude of gravitational waves.

Although the web interface module provides an easy way to provide live information about Cactus simulations, and can be straightforwardly extended to provide application-level information about the scientific content of the simulation, it does suffer from some shortcomings.

First, there are technical challenges in configuring systems to use the interface. Typical production simulations are started on compute nodes of supercomputers where the webserver cannot directly be accessed from the internet. Forwarding the corresponding port e.g. via ssh may work, but this has to be set up for each simulation anew. Firewalls on compute nodes which block all "non-essential" network ports can also pose problems. Second, the information is only available while the simulation is actually running. After the simulation completes, the direct path to the information disappears with the web interface. Finally, a more fundamental problem is how to publicize the availability of the simulation and web interface itself. To be able to connect through a web browser to a live simulation, a scientist has to be aware that the simulation is actually running, and also needs to know the URL to connect to. Apart from looking at the direct simulation output, emails have been used to communicate this information, but this was not practical with several simulations running at the same time.

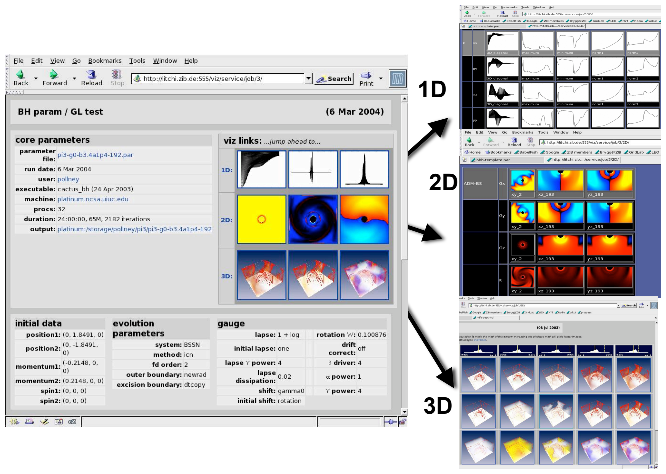
C. Simulation Reports and Email

To address the problem of persistency of information from simulations, and also to provide high level physics information that could be automatically generated from black hole simulations, a thorn was prototyped in 2001 which would interpret results from the simulation and prepare a readable report (e.g. a PDF document). The vision was that individual thorns could contribute different parts of the report, which could be specific to the particular scientific scenario. For example, sections could summarize the initial data, evolution and boundary conditions used, describe the main results including graphs (for the case of black holes including the resulting gravitational waveforms), describe the validation performed during the simulation (e.g. show how closely the constraint equations were satisfied). On the computational side, details such as the machine used, performance attained, best and worst performing thorns, amount of output, etc. could be described.

This effort overlaps with recent initiatives for better data provenance, but was also striving to motivate thorn writers to not just code up equations, but also include interpretation of their software and results.

The reports were generated in the local disk space for the simulation, and as with the other information providing tools provided here, to be truly useful there needs to be a mechanism to publish reports to appropriate collaborators.

One mechanism for publishing information is to use Email. Email has the advantage of being very reliable, since mail is spooled until it can be sent, and can withstand issues seen in



http://litchi.zib.de:555/viz/service/job/3/

BH param / GL test **(6 Mar 2004)**

core parameters

parameter file: pi3-g0-b3.4a1p4-192.par
run date: 6 Mar 2004
user: pollney
executable: cactus_bh (24 Apr 2003)
machine: platinum.ncsa.uiuc.edu
procs: 32
duration: 24:00:00, 65M, 2182 iterations
output: platinum:/storage/pollney/pi3/pi3-g0-b3.4a1p4-192

viz links: *...jump ahead to...*
1D:
2D:
3D:

initial data
position1: (0, 1.8491, 0)
position2: (0, -1.8491, 0)
momentum1: (-0.2148, 0, 0)
momentum2: (0.2148, 0, 0)
spin1: (0, 0, 0)
spin2: (0, 0, 0)

evolution parameters
system: BSSN
method: icn
fd order: 2
outer boundary: newrad
excision boundary: dtcopy

gauge

lapse:	1 + log	**rotation** W:	0.100876
initial lapse:	one	**drift correct:**	off
lapse Y **power:**	4	B **driver:**	4
lapse dissipation:	0.02	α **power:**	1
shift:	gamma0	Y **power:**	4
initial shift:	rotation		

Fig. 3. Visualization report generating a web-based visual summary of a black hole simulation including 1D, 2D and 3D data [Source: Brygg Ullmer].

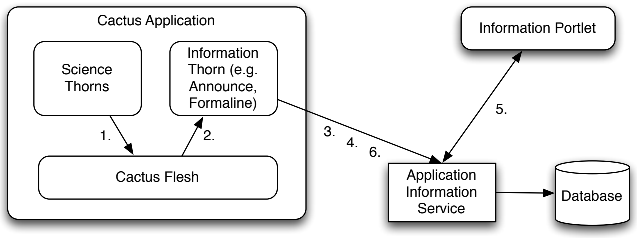


Fig. 4. General architecture for the application-level information system in Cactus used by thorns Announce and Formaline. Application thorns provide content (1.) to a Cactus information thorn (2.) which transports this information to an external application information service with an initial registration (3.), updates (4.), and termination notice (6.). Information portals and other services are able to query the application information service (5.).

other approaches that rely on remote services being available otherwise information is lost. In 2000, a `Mail` thorn was written for Cactus that automatically mailed information about the simulation to a list of email addresses provided in the parameter file. This thorn used a system call to *sendmail* on the resource. Unfortunately, many supercomputers do not allow mail to be sent directly from compute nodes, limiting the applicability of this approach.

A service to provide visualization reports for numerical relativity simulations was developed by Brygg Ullmer in 2004 as part of the European GridLab project. These reports were generated on a visualization server from simulation data and provided web-based reports with thumbnails to provide an 'at-a-glance' visual summary of a simulation (Figure 3). The idea was that users would click on these thumbnails and be taken into an interactive remote visualization session with the full data [25], using video streaming [26].

D. Announcing and Grid Portals

The need for collaborations to have reliable, live information about their simulations became more pressing with the arrival of the concept of Grid Computing. Working with colleagues in the NSF Astrophysics Simulation Collaboratory (ASC) [27], a project which started in 1999, a grid portal was developed to provide a centralized, collaborative interface to submit, monitor and archive simulations [28]. The ASC portal was one of the first implementations of a virtual organization of distributed researchers. The first versions of the ASC portal were built using a 3-tier model-view-controller framework using Java, JSP and Javascript with a back-end database. Later versions of our black hole portals used the GridSphere Portal Framework from the GridLab project [29].

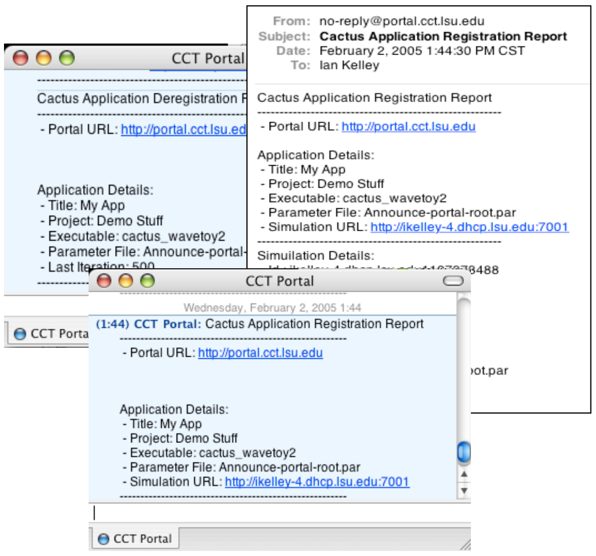
The ASC portal used Java COG [30] directly to submit jobs to the different compute resources in the simulation, and provided basic monitoring of the status of simulations (e.g. Queued, Running). To provide more application specific information, and to allow for tracking of simulations not submitted via the portal (a key requirement for this production virtual organization) a more flexible and ubiquitous mechanism was needed to provide simulation details.

A new Cactus thorn called `Announce` was developed to publish simulation information in 2001, as part of a prototype grid scenario called the *Cactus Worm* [31] that involved a resource-aware simulation self-migrating across European supercomputers. The Announce thorn compiled a message from information available to the Cactus Flesh or provided by the user as parameters, and transported this message using XML-RPC to a socket on a remote machine which matched the corresponding portal service (Figure 4). The published information automatically created included a job id created by Cactus or read from an environment variable, hostname of the compute node and username, executable name, name and contents of the parameter file, start time and iteration number and a list of output files and their location. Information that needed to be provided by user-set parameters included a project name, description of the job, visibility of job on the portal, whether or not notifications should be sent by the portal and a list of users to notify.

Additional parameters set by the user set the URL and port number for the portal, as well as the portal username that the simulation was associated with. Although grid certificates could have been used to provide the unique username on the Grid, these were not used since they were not yet widely used across the group (or the machines used by the group for production simulations).

On the ASC Portal, a service received and catalogued these simulation messages. After authenticating to the portal, a user could view all the received information about his/her own simulations, or those of his/her virtual organization (Figure 6). Portal links were provided for each simulation to take the user directly to the simulation web interface if available.

To notify other members of the collaboration about the different simulations and their status, a notification service was added to the portal. This service used user configuration details entered on the portal to send message to users, originally by email and SMS, and later by instant message (Figure 5). The first use of this required establishing an SMS server at the



Cactus Application Deregistration R

- Portal URL: http://portal.cct.lsu.ed

Application Details:
- Title: My App
- Project: Demo Stuff
- Executable: cactus_wavetoy2
- Parameter File: Announce-portal-

From: no-reply@portal.cct.lsu.edu
Subject: **Cactus Application Registration Report**
Date: February 2, 2005 1:44:30 PM CST
To: Ian Kelley

Cactus Application Registration Report

- Portal URL: http://portal.cct.lsu.edu

Application Details:
- Title: My App
- Project: Demo Stuff
- Executable: cactus_wavetoy2
- Parameter File: Announce-portal-root.par
- Simulation URL: http://ikelley-4.dhcp.lsu.edu:7001

Simuilation Details:

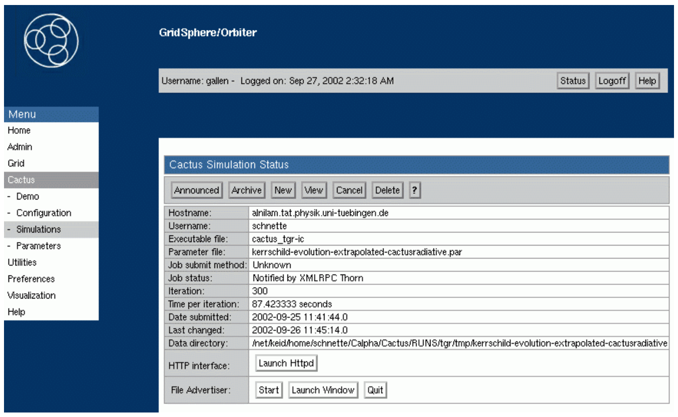
Wednesday, February 2, 2005 1:44

(1:44) CCT Portal: Cactus Application Registration Report

- Portal URL: http://portal.cct.lsu.edu

Application Details:
- Title: My App
- Project: Demo Stuff
- Executable: cactus_wavetoy2
- Parameter File: Announce-portal-root.par
- Simulation URL: http://ikelley-4.dhcp.lsu.edu:7001

Fig. 5. The GridLab Cactus portal includes a notification service that uses instant message, email, and SMS to provide a collaboration about simulation status.



Username: gallen - Logged on: Sep 27, 2002 2:32:18 AM

Cactus Simulation Status

Hostname:	alnilam.tat.physik.uni-tuebingen.de
Username:	schnette
Executable file:	cactus_tgr-ic
Parameter file:	kerrschild-evolution-extrapolated-cactusradiative.par
Job submit method:	Unknown
Job status:	Notified by XMLRPC Thorn
Iteration:	300
Time per iteration:	87.423333 seconds
Date submitted:	2002-09-25 11:41:44.0
Last changed:	2002-09-26 11:45:14.0
Data directory:	/net/keid/home/schnette/Calpha/Cactus/RUNS/tgr/tmp/kerrschild-evolution-extrapolated-cactusradiative
HTTP interface:	Launch Httpd
File Advertiser:	Start Launch Window Quit

Fig. 6. Portlet on the ASC Portal in 2002 showing simulation information published by thorn Announce. Links from this window connect directly to the Cactus web interface for the simulation, or to the File Advertiser is the simulation has completed.

AEI to send messages, but later this capability was provided through different mobile phone providers.

Although notification in this way worked well, it required a lot of development in the portal service to support the configuration of different notification groups, and also required the parameter file of the actual Cactus run to be written to describe the visibility of the simulation and the required set of users to notify.

A new version of thorn `Announce` was developed in 2005, called `Formaline`, with a slightly modified approach. Instead of asking the user to describe the simulation in as much detail as possible in advance, `Formaline` relies on information that it can gather automatically during the build process and a run time; however, describing the simulation is still possible. The gathered information is primarily stored and kept safe at the remote machine to ensure that simulation results are repeatable. At regular intervals, the collected metadata are sent to an information service in the same way as with `Announce`. When a simulation is finished, the collected metadata are archived together with the simulation data.

E. Interactive Access to Ongoing Simulations

In addition to the tools described above, which are concerned with high-level metadata about simulations, it is also necessary to have ways to directly access and interact with a simulation. We are developing tools for this in the Alpaca project ([32], [33]). The Alpaca tools work at the application level, i.e., they have access to and can present the definition and description of all relevant functions, variables, and parameters in the simulation at a high level, corresponding to the physics equations and their discretisation. Existing tools provide access at the source level, examining programmes at the level of C or Fortran variables and statements.

These tools visualise the current state of the physics variables in an ongoing simulation, display performance information, and allow interrupting, single-stepping, or aborting the simulation, and also allow modifying the state of the simulation.

Such direct access is difficult, if not impossible, to provide via web-based user interfaces. Consequently, these tools have their own graphical user interfaces, using e.g. OpenGL for high-speed visualisation of large datasets.

In addition to having to provide their own user interfaces for remote, graphical access to supercomputers, these tools face the same problems as the Web 2.0 based tools described in this paper.

V. Issues with Simulation Information Services

Live services – services directly offered by a simulation as it is running on an HPC system – face a fundamental problem, namely that these systems have not been intended to provide any services to the outside world. They are often hidden behind firewalls, requiring ssh tunnels or similar measures to be accessible, and access problems are difficult to debug. User agreements are often silent about these points, so that it is not clear in how far it is actually legal to offer such services, especially if they are public and use their own authentication mechanism.

Authenticating to a running simulation is complex and problematic in itself. Passwords are insecure if they cannot be changed by the user (and shared passwords more so). They are also cumbersome if they have to be shared within the group. X.509 certificates offer a solution to these issues. A certificate is created by a user, and is then signed by a trusted authority, confirming that the information (name and institution) on the certificate is correct. Together with a public/private key pair mechanism, this allows giving access to specific people without having to handle passwords.

However, certificates come with their own difficulties: many users won't have certificates and will then have to obtain them first; different tools today require different encoding formats for certificates; error messages are often not helpful; certificates have to be set up on every system from which a user wants to access information. Also, most users are simply not familiar with authentication mechanisms more advanced than ssh keys.

X.509 certificates provide not only authentication for individuals, but also describe an organisational hierarchy, thereby defining groups of people. This makes it possible to give access to a simulation e.g. to all colleagues within the same institute. This mechanism is somewhat inflexible and cannot always capture the more fluent and less well defined interactions between researchers at different institutions. It seems that community websites such as Facebook and LinkedIn have solved the issue of describing trust between groups of people in a very different and much more flexible manner, by letting each person decide his/her group of friends or colleagues.

From the experiences described in Section IV, a set of core requirements can be derived for information provision and viewing from scientific simulations:

1) The existence and location of the service must be announced to a known place, such as a portal.
2) The transport mechanism for messages needs to work reliably on production supercomputers which may have internal nodes that are not publicly addressable, often do not provide services such as Mail, and may block ingoing or even outgoing ports.
3) Sending messages from simulations must not impact the efficiency of the code or cause the simulation to terminate.
4) The mechanism must be secure. There should be no possibility for unauthorized access to the simulation or compute resource, tampering with the sent message, or for intruders sending unauthorized messages to the collecting service.
5) The service must be very easy to configure, so that users can "just use" them on every machine – if not, users will simply disable the service.
6) Similarly, the authentication mechanism must be easy to understand and use.
7) Access to the simulation must be possible from a variety of devices such as web browsers, phones, shell scripts, where each user may use several devices at once.
8) There must be a way to identify groups of users with similar privileges, such as a circle of collaborators; if people work with different sets of collaborators on different projects, then defining these access privileges can be complex and will change with time.

VI. Web 2.0 Technologies for Simulation Science

This section describes new Cactus thorns that have been developed to take advantage of the availability and reliability of new Web 2.0 social network services for collaborative, simulation-level real-time messaging and information archiving. The common features of the services that are described here are that they provide reliable, persistent, well-documented and user-configurable features for social interaction.

A. Twitter

Twitter [34] was created in March 2006 and has now grown into a real-time short messaging service. It was initially funded by Obvious, a creative environment in San Francisco, CA. It grew popular very quickly and was moved out of Obvious by the creation of Twitter Incorporated.

Twitter's main service is a message routing system that enables its users to send and read each others' updates, known as *tweets*. Tweets have to be very short (at most 140 characters in length) and can be sent and read via a wide range of devices, e.g. mobile texting, instant message, the web or external applications. The system also features rudimentary social networking features to interconnect users. The person posting the tweets can restrict who can see them, and those receiving the tweets can filter what messages they see and how they receive them.

Twitter provides a Twitter API [35] which allows the integration of Twitter with other web services and applications. The probably most important function is the "statuses/update" API call, which is used to post a new Twitter message from the specified user. Other possible calls can handle e.g. direct messages to other users, account profile editing or friendship management.

B. Flickr

Flickr [36] was launched in 2004 by Ludicorp as an image hosting website targeted at digital photographs. In March 2005, Yahoo acquired Ludicorp and Flickr. Since 2008 users have also been able to upload videos. Flickr can be used for free, however there are limits on the total size of images that can be uploaded (currently 100 MB), the amount of images which can be viewed (currently the 200 most recent images) and other potential services available. A full-featured professional account currently costs $25 a year and allows unrestricted image uploads amongst other removed limitations or raised limits.

Flickr has many features that can be taken advantage of for providing a collaborative repository for Cactus produced images and information. All of them are accessed through a comprehensive web service API for uploading and manipulating images [37].

One important functionality besides the image upload is to be able to group images. Flickr offers a capability to group images into "Sets", and also can group different "Sets" into a "Collection". This provides a hierarchical structure for organizing simulation images. One image can belong to multiple "Sets". "Collections" can also contain other "Collections" (up to five levels).

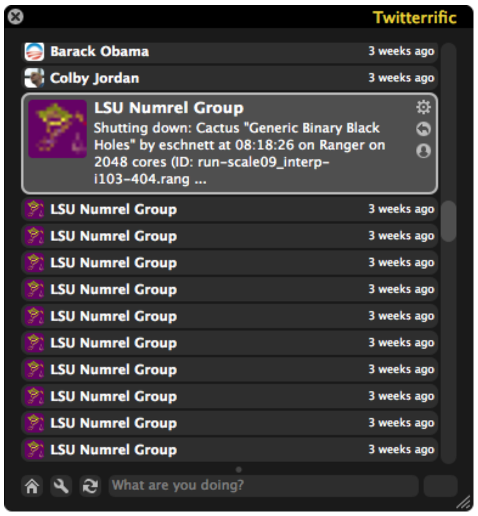
Other interesting features include e.g. metadata which can be applied at different levels. A title and description can be added for each image, and for each set of images. Metadata is displayed for each image from the information embedded into jpeg images, so called "EXIF" headers. Typical example uses are camera settings for photos, or a small thumbnail for fast display without reading and rendering the entire file.

Flickr interfaces allow others to comment on images, where the comments are then displayed. Images can also be tagged with user-defined keywords. Tags are displayed in tag clouds, where the font size indicates the popularity of the tag. A user can mark their own images, or other peoples images, as "interesting" or "favorites".

Not all images posted to Flickr are publicly available. Public and private storage and display of images are provided. Public images can be viewed by anyone, private images can be set to be viewable by family, friends or both categories. One can invite others to be friends, or can request to be a friend.

Flickr images are available as an RSS field through which an individual or group's images can be quickly previewed using RSS aggregators.

Users of Flickr can organize into groups with a pool of group images to which each member of the group has access to. Groups can be open to the public, by invitation only (but still publicly listed, or entirely private.


Twitterrific

Barack Obama 3 weeks ago

Colby Jordan 3 weeks ago

LSU Numrel Group
Shutting down: Cactus "Generic Binary Black Holes" by eschnett at 08:18:26 on Ranger on 2048 cores (ID: run-scale09_interp-i103-404.rang ...

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

LSU Numrel Group 3 weeks ago

What are you doing?

Fig. 7. Twitter message about a numerical relativity simulation shutting down, sent to the Twitter group account 'numrel' by the Cactus thorn Twitter.

VII. Cactus Web 2.0 Thorns

A. Thorn Twitter

The Cactus thorn Twitter interacts directly with the Twitter website using libcurl, a free client-side URL transfer library. This means that either libcurl needs to be already installed on the machine running the simulation, or because that might not be the case especially for supercomputers, there will be a Cactus thorn providing this library.

The Twitter thorn needs the credentials of a user: its Twitter username and password to send messages, which are set as Cactus parameters. It uses one of the Twitter API functions, "statuses/update", to send a message to the web service.

Using this information, Cactus simulations can send messages to any Twitter user account. That does not have to be a personal Twitter account of a single user. By creating a separate Twitter account and sharing its login information, group accounts can be setup and used by collaborators. One example of this is the "numrel" Twitter account which is the default Twitter account the thorn Twitter sends messages to.

The current Twitter thorn sends messages when a simulation starts (which might be several hours or even days after it was submitted to a queue to run), when it finishes the initialization stage and starts the evolution stage, and when the simulation finishes. However, with the Twitter infrastructure in Cactus now in place it is straightforward to send messages about any event during a simulation, such as progress updates at regular intervals or important events the occur during simulation such as the collision of two black holes.

B. Thorn Flickr

The Cactus thorn Flickr can be used to send images from a running simulation to the Flickr service. Unlike the Twitter thorn, it depends on more libraries:

- `flickcurl`: The Flickr thorn itself utilizes the flickcurl library which is a C library for the Flickr API.
- `libcurl`: flickcurl needs the curl library to connect to the Flickr web service.
- `libxml2`: flickcurl needs the xml2 library to handle the xml responses from the Flickr web service.
- `openssl`: The Flickr API requires the use of the MD5 hash function for parts of the API and openssl provides the implementation which is used in the Cactus thorn.

External applications can use Flickr only through its API which requires a more complicated authentication mechanism than simple pair of a username and a password. For every external application an "api key" needs to be created, along with a "shared secret". Both a letter-number combinations of a certain length and identify the application. Both have to be given effectively in clear text to the user of the thorn to avoid that every user has to create its own keys. The thorn cannot access any Flickr account yet, in fact it does not even have the information about which account it should use. To obtain both, it has to create a "login link" which is a web URL, directing the user to the Flickr web pages. This URL consists of the "api key", an identifier for this request (a so called "frob"), the "shared secret", the requested permissions and an MD5 hash of all the previous arguments. The user then has to login using his username and password and can then agree to give the application the requested permissions on the user account. When this is done, the application can access the account and request an "authentication token" which it can use from now on to access the user account.

Because of this complicated procedure, a user of the Flickr thorn cannot right away use it to access his Flickr account. However, after the first authentication of the application on the Flickr website, the thorn outputs the "authentication token", which can then be used for all future uses like a username/password combination.

The thorn only expects the user to set this "authentication token" as parameter during startup for normal operation (otherwise it tries to obtain a new token as described above). There are other parameters which can be set (like another "api key"), but these all have working default values.

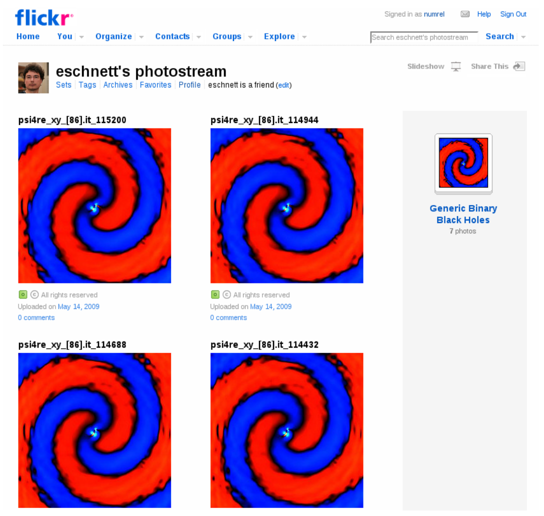


Fig. 8. This screen-shot shows a set of images, taken from a real simulation and and uploaded to a private Flickr account.

The Flickr thorn does not have to handle the image creation itself, it merely handles the upload of any images which are generated by Cactus and advertised via a Cactus-internal API. One example of a thorn which generates and advertises such images is IOJpeg, which generates jpeg images of slices of specific variables. The same thorn and advertising API is also used to provide images for the thorn HTTPDExtra (see IV-B).

Each Cactus simulation sets a simulation title which is used to group all images generated by one simulation into a Flickr set with that name. This is useful because it is not uncommon to have several if not many simulations running at the same time, possibly all submitting images to Flickr.

Similarly to Twitter group accounts, it is possible to use a single Flickr account as a group or collaboration. One example is the *numrel* Flickr account now used by scientists working on numerical simulations of relativistic astrophysics problems.

VIII. Future Work and Possibilities

Much more can now be done to extend the capabilities and use of the existing implementations of the Cactus Twitter and Flickr thorns. A common authentication mechanism would unify access to services and help address issues identified in Section V. Currently we use a shared “numrel” account for publishing information to both services, and alternative mechanisms need to be investigated, either following a model of each user having their own account, or potentially project based accounts. Twitter offers features for filtering of messages, included archived messages, that could be exploited using common tags. The Flickr thorn can be expanded to include tags, richer metadata and simulation reporting.

Flickr is able to not only store images, but also videos. However, their size and length is restricted. While it could nevertheless be useful, there are also other commonly used alternatives available such as YouTube. YouTube is already used in the numerical relativity for sharing videos created from simulation results, which are uploaded manually. Producing such videos is time consuming and not routine for general simulation results. Researchers use YouTube not only to share their work, but to organize and present their movies. Once stored on the YouTube server, researchers can be confident that the movie will play on any machine and through any browser, which can then facilitate their presentations. Cactus already has a prototype thorn for creating movies automatically in MPEG format using 2-D image slices which could through the YouTube API be automatically uploaded.

Other maturing Web 2.0 technologies could provide new capabilities for collaboration. DropBox could be used to synchronize/publish simulation files between a virtual organization. Blogging services such as WordPress could be used for publishing automated reports generated by simulations as described in Section IV-C. Social networking sites such as Facebook which can run third party applications could take the place of grid portals, acting as a focal point for collecting and integrating together different services.

Although not the focus of this paper, Cactus has long been involved in providing and implementing motivating scientific scenarios for Grid computing. Now called cloud computing, the basic motivation for scientific research is the same. The new, simpler APIs provided for cloud computing could be integrated into Cactus and used for past prototyped scenarios, such as writing data to remote locations, spawning asynchronous tasks to more appropriate resources, enabling broad parameter surveys.

IX. Discussion

This paper has described the integration of Web 2.0 services (Flickr, Twitter) in the Cactus Framework and shown how they are being used to enable a collaboration in numerical relativity. Further examples were provided to show how other recent services can potentially provide collaborative capabilities that were previously not possible, and should lead to new opportunities for collaboration in simulation-based science.

While our Cactus-based simulation codes have been able to provide, for many years, a wealth of messages containing status information, performance, physics progress, as well as produce images such as jpeg slices through a 3D space, we have been held back by a lack of reliable delivery, storage and access mechanisms to convey this information, or a collaborative service with rich enough capabilities to provide what is needed.

While Flickr and Twitter provide only partial parts of a full solution, they are providing a mechanism for experimentation to find what the most effective methodologies for collaboration are. Further, they set a target standard for those developing new “cyberinfrastructure” services with their ease-of-use, reliability and third-party applications.

To be truly appropriate for academic scientific computing, we would like to have unified authentication mechanisms for these remote services, have clear policies on ownership and access to data held on these servers as well as mechanisms to easily retrieve data, and have the option of installing site versions of services which can support a single virtual organization. As these services grow and mature, and our experiences in using them help form future requirements, we look forward to the ability to build unified collaborative environments from collections of tools.

The infrastructure that has been prototyped and used in the last decade to support collaboration with Cactus has been very development intensive. It has taken significant effort (and considerable funding) to develop and support the necessary grid portals, SMS servers, Cactus socket thorns and other services. Such development is only realistic for a large, well organized and well funded virtual organization. One huge benefit of the Web 2.0 technologies is that they can now be easily used by small groups or even individuals.

Acknowledgments

We gratefully acknowledge our colleagues who have contributed to this work over the last ten years, including John Baker, Werner Benger, Ruxandra Bondarescu, Alex Clary, Rion Dooley, Tom Goodale, Ian Kelley, Gerd Lanfermann, Michael Russell, John Shalf, Brygg Ullmer, Oliver Wehrens, as well as our patient colleagues in the numerical relativity groups at Louisiana State University, the Albert Einstein Institute, and Washington University.

This work was supported by awards NSF 0721915 (Alpaca), 0701566 (XiRel) and 0701491 (CyberTools). The historical work presented was supported by EC IST-2001-32133 (GridLab), HPRN-CT-2000-00137 (The EU Network), BMBF (D-Grid), NSF 9979985 (ASC), and Microsoft. Binary black hole simulations were performed on Ranger at TACC under the TeraGrid allocation TG-MCA02N014.

References

[1] S. W. Hawking, "Black holes in general relativity," *Comm. Math. Phys.*, vol. 25, p. 152, 1972.

[2] L. Smarr, "Spacetimes generated by computers: Black holes with gravitational radiation," *Ann. N. Y. Acad. Sci.*, vol. 302, pp. 569–604, 1977.

[3] P. Anninos, D. Hobill, E. Seidel, L. Smarr, and W.-M. Suen, "The head-on collision of two equal mass black holes," *Phys. Rev. D*, vol. 52, pp. 2044–2058, 1995.

[4] M. Alcubierre, W. Benger, B. Brügmann, G. Lanfermann, L. Nerger, E. Seidel, and R. Takahashi, "3D Grazing Collision of Two Black Holes," *Phys. Rev. Lett.*, vol. 87, p. 271103, 2001.

[5] E. Schnetter, C. D. Ott, G. Allen, P. Diener, T. Goodale, T. Radke, E. Seidel, and J. Shalf, "Cactus Framework: Black holes to gamma ray bursts," in *Petascale Computing: Algorithms and Applications*, D. A. Bader, Ed. Chapman & Hall/CRC Computational Science Series, 2008, ch. 24. [Online]. Available: http://arxiv.org/abs/0707.1607

[6] T. Hannay, "Web 2.0 in science," *Cyberinfrastructure Technology Watch*, vol. 3, no. 3, 2007. [Online]. Available: http://www.ctwatch.org/quarterly/articles/2007/08/web-20-in-science/

[7] Reed's Ruminations: A Blog by Dan Reed. [Online]. Available: http://www.hpcdan.org/

[8] GridCast: Blogging Behind the Scenes of Grid Computing. [Online]. Available: http://gridtalk-project.blogspot.com/

[9] The Physics arXiv Blog. [Online]. Available: http://www.technologyreview.com/blog/arxiv/

[10] GRwiki: a repository of basic definitions and formulas for gravitational physics. [Online]. Available: http://grwiki.physics.ncsu.edu/wiki/

[11] SciLink. [Online]. Available: http://www.scilink.com

[12] A. Nagelberg, C. Kaiser, H. Kaiser, and G. Allen, "Near realtime visualization of coastal modelling results with wms and google maps," in *MG '08: Proceedings of the 15th ACM Mardi Gras conference*. New York, NY, USA: ACM, 2008, pp. 1–1.

[13] DOI Registration Service for Research Data Sets, German National Library of Science and Technology. [Online]. Available: http://www.tib-hannover.de/en/the-tib/doi-registration-agency/

[14] Living Reviews in Relativity. [Online]. Available: http://relativity.livingreviews.org/

[15] Cactus Computational Toolkit home page. [Online]. Available: http://www.cactuscode.org/

[16] T. Goodale, G. Allen, G. Lanfermann, J. Massó, T. Radke, E. Seidel, and J. Shalf, "The Cactus framework and toolkit: Design and applications," in *Vector and Parallel Processing – VECPAR'2002, 5th International Conference, Lecture Notes in Computer Science*. Berlin: Springer, 2003.

[17] C. W. Misner, K. S. Thorne, and J. A. Wheeler, *Gravitation*. San Francisco: W. H. Freeman, 1973.

[18] LONI: Louisiana Optical Network Initiative. [Online]. Available: http://www.loni.org/

[19] National Science Foundation TeraGrid. [Online]. Available: http://www.teragrid.org/

[20] CoCoBoard: A Collaborative Cork Board for use on the WWW. [Online]. Available: http://jean-luc.aei.mpg.de/Codes/CoCoBoard/index.html

[21] W. Benger, H.-C. Hege, A. Merzky, T. Radke, and E. Seidel, "Efficient distributed file i/o for visualization in grid environments," in *Simulation and Visualization on the Grid*, ser. Lecture Notes in Computational Science and Engineering, B. Engquist, L. Johnsson, M. Hammill, and F. Short, Eds. Springer Verlag, 2000, vol. 13, pp. 1–6.

[22] H. Enke, M. Steinmetz, T. Radke, A. Reiser, T. Röblitz, and M. Högqvist, "Astrogrid-d: Enhancing astronomic science with grid technology," in *Proc. of the German e-Science Conference*, Baden-Baden, Germany, 2007.

[23] AstroGrid-D: German Astronomy Community Grid (GACG). [Online]. Available: http://www.gac-grid.org/project-products/Software/InformationService/InformationProducer/CactusRDFProducer.html

[24] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC 5280 (Proposed Standard), Internet Engineering Task Force, May 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5280.txt

[25] GridLab Report. Deliverable D8.10: Final Release of DM Services. [Online]. Available: http://www.gridlab.org/Resources/Deliverables/D8.10.pdf

[26] A. H. Hans-Christian Hege, Steffen Prohaska, "Towards distributed visualization and analysis of fluid dynamics data," *JSME International Journal, Series B*, vol. 48, no. 2, 2005.

[27] Astrophysics Simulation Collaboratory. [Online]. Available: http://wugrav.wustl.edu/ASC/

[28] R. Bondarescu, G. Allen, G. Daues, I. Kelley, M. Russell, E. Seidel, J. Shalf, and M. Tobias, "The Astrophysics Simulation Collaboratory portal: a framework for effective distributed research," *Future Generation Computer Systems*, vol. 21(2), pp. 259–270, 2005. [Online]. Available: http://www.cactuscode.org/Articles/Cactus_Bondarescu03a.pre.pdf

[29] J. Novotny, M. Russell, and O. Wehrens, "Gridsphere: a portal framework for building collaborations: Research articles," *Concurr. Comput. : Pract. Exper.*, vol. 16, no. 5, pp. 503–513, 2004.

[30] G. von Laszewski, I. T. Foster, J. Gawor, and P. Lane, "A java commodity grid kit." *Concurrency and Computation: Practice and Experience*, vol. 13, no. 8-9, pp. 645–662, 2001.

[31] G. A. et. al., "Early experiences with the egrid testbed," in *IEEE International Symposium on Cluster Computing and the Grid, Brisbane, Australia, May 16-18 2001*, 2001. [Online]. Available: http://www.cactuscode.org/Articles/Cactus_Allen01d.pre.pdf

[32] E. Schnetter, G. Allen, T. Goodale, and M. Tyagi, "Alpaca: Cactus tools for application level performance and correctness analysis," Louisiana State University, Tech. Rep. CCT-TR-2008-2, 2008. [Online]. Available: http://www.cct.lsu.edu/CCT-TR/CCT-TR-2008-2

[33] Alpaca: Cactus tools for Application-Level Profiling and Correctness Analysis. [Online]. Available: http://www.cct.lsu.edu/~eschnett/Alpaca/

[34] Twitter. [Online]. Available: http://www.twitter.com/

[35] Twitter Application Programming Interface. [Online]. Available: http://apiwiki.twitter.com/Twitter-API-Documentation

[36] Flickr. [Online]. Available: http://www.flickr.com/

[37] Flickr API Web page. [Online]. Available: http://www.flickr.com/services/api/