Final Report and Assessment SIRS projects

Maarten van Steen

February 2002

Contents

1 Introduction

SIRS-3 is the third subproject carried as part of the *Scalable Internet Resource Server* project, which started in 1999. This final report has two purposes. First, it discusses the results of SIRS-3 by taking a look at the original plans as described in the SIRS-3 project proposal. Second, it provides a brief assessment of the entire SIRS project. This assessment also serves as the basis for an outlook on future activities.

We start with giving some background information on SIRS in Section 2, followed by a presentation of the SIRS-3 achievements in Section 3. An assessment of the entire SIRS project is given in Section 4.

2 Background information

In this section, we briefly describe the ideas underlying SIRS and the relation between SIRS and the Globe research project carried out at the Vrije Universiteit.

2.1 The SIRS projects

The SIRS project has concentrated on the development of a service that allows Internet resources to be widely distributed and replicated across the Internet in a scalable way. The key observation underlying SIRS is that scalability can be obtained only if we provide support for resource-specific solutions to distributing and replicating resources.

Consider, for example, a file that can be accessed by remote clients. If the file is highly popular, but hardly updated, scalability can be achieved by pushing it to servers close to where its clients are. A popular file that is also regularly updated may benefit from a replication strategy in which clients poll the server to see whether their local copy is still up-to-date. Likewise, for unpopular files it may be best not to apply replication at all. Other examples of replication scenarios and how these effect performance are discussed in [5].

To allow for resource-specific distribution and replication, resources are embedded in Globe distributed shared objects. In contrast to traditional distributed objects, the state of a Globe object can be distributed and replicated across multiple machines, and in a way that is determined by the object. In other words, a Globe object encapsulates its own distribution and replication scenario.

For SIRS, we decided to concentrate on the distribution and replication of free-software packages. The original plan was to develop a service that offers functionality comparable to FTP servers, but that supports automatic distribution of its files across multiple sites. In SIRS, we assume that sites are located across the entire Internet. Clients are transparently redirected to the nearest copy of the software package they want to download. In addition, SIRS offers secure uploading and downloading of packages.

SIRS has been organized into three subprojects, named SIRS-1, SIRS-2, and SIRS-3. In SIRS-1, emphasis was put on developing client-side software that would allow transparent Web access to replicated software packages. In SIRS-2, we concentrated on enhancing and further development of our object servers to support large-scale distribution of software packages. In addition, effort was put into developing tools for uploading files. In SIRS-3, we further concentrated on making SIRS a usable system by incorporating security and fault tolerance, and providing support for easy installment. We return to SIRS-3 below. Project descriptions and evaluations of the first two subprojects can be found at `http://www.nlnet.nl/projects/sirs/index.html`.

2.2 SIRS and Globe

SIRS is carried out as part of the Globe research project [6]. In Globe, we address the fundamental problem of how to achieve scalability for systems that need to support one billion users, each user having on average 1000 objects. The specific problem addressed in SIRS (i.e., the distribution of free software), is organized as a subproject within Globe, called the Globe Distribution Network (GDN). GDN has the same goals as SIRS, but also contains the research efforts needed to come to scalable worldwide distribution of software packages. The SIRS projects are devoted to developing the software needed with GDN. The GDN project has substantial research component and only those ideas that have been worked out in considerable detail are eligible for further development as part of SIRS. However, it is easier to think of SIRS and GDN being the same project. An early description of GDN has been presented in the USENIX Freenix track [1].

3 Achievements

SIRS-3 was set up to develop support in three different areas: security, fault tolerance, and management tools.

3.1 Security

For security, we wanted to design and implement an architecture that would allow producers of illicit content to be traced. In addition, several authorization mechanisms needed to be provided. The following three subgoals were formulated:

1. The maintainer of a software package should be traceable.
2. Only the maintainer can modify a package.
3. Only the maintainer can issue requests for creating new replicas.

These choices were mainly based on the observation that SIRS should provide guarantees with respect to the secure *distribution* of software packages. An alternative that we only briefly considered, was to provide the means to check for the distribution of illicit *content*. In general, content verification by technical means only is virtually impossible.

In the current solution, we have set up the means to trace exactly where an uploaded package came from. In essence, when uploading a package to SIRS, the client is requested to digitally sign that package with a key that has been provided by a trusted *Access Granting Organization* (AGO). The AGO does not verify the content of a package. However, by signing an uploaded package, a downloading client can always check where that package came from. Downloading unsigned packages is at the client's risk.

It is not the goal of this report to describe the security infrastructure for SIRS in any detail. These can be found in [2] and has been added as an appendix to this report. Practical issues concerning security in SIRS (such as key management, signing, and verification) can be found in the *Globe Operations Guide* (GOG), which is provided as part of the GDN distribution.

3.2 Fault Tolerance

The software resulting from SIRS-1 and SIRS-2 had only minimal support for tolerating faults. We wanted to add the following functionality:

1. A simple scheme to allow object servers to recover from crashes.
2. Recovery from a communication error.

An object server provides fault tolerance by periodically storing the complete in-memory state related to a package object to disk. If the server crashes during operation, the most recently saved state is restored, effectively recovering packages that were being accessed at the time of the crash. No checkpoint is made if the state has not been altered since the last checkpoint.

We decided to implement periodic checkpointing for performance reasons. The alternative is to checkpoint the state at each update operation, but this was felt too expensive as it requires a synchronous disk operation. Further research is needed to see whether and how we can improve this situation. In particular, as part of our long-term research, we intend to explore the promising combination of object-specific fault tolerance and replication for performance. Some initial work on this matter is described in [3].

When recovering from a server crash, the object server normally fetches the new state from a master replica or other source that is guaranteed to have a fresh copy. This synchronization is necessary to ensure consistency. Before doing so, the server first checks whether any updates have occurred during the crash period to avoid needless state transfer across the network.

Recovery from a communication error deals with avoiding that we need to restart the uploading or downloading of a package from the beginning. We have implemented a simple client-side scheme that handles interrupted downloads. Downloads occurs in blocks of data that are first stored at the client's side to be later assembled into a package. In this way, when a connection between a client and an object server breaks, the downloading client can later continue where it had left off. This mechanism is transparent to the user. There is no support for recovery from a communication error during uploads. The problem was considered less important as a client already uploads individual files instead of complete packages.

Fault tolerance is described in detail in an upcoming dissertation by Arno Bakker.

3.3 Management tools

One of the problems with the SIRS-2 variant was the considerable effort it took to install a site before it could become fully operational. We planned to develop the following tools and mechanisms to alleviate these problems:

1. Tools to start and shutdown an entire site in a user-friendly way.
2. Tools to monitor the resource usage within a site, and that allow a user to inspect which objects it is currently running.
3. User-friendly installation and configuration scripts.
4. A tool to remove a replica.
5. Global system management tools.
6. Adaptation of globify, a tool that generates scripts by which entire Web sites can be embedded inside a Globe object.

The management tools have only been partly realized. An important extension to SIRS is a special directory service, called GIDS, that stores all the necessary configuration information for a site. GIDS stands for the *Globe Infrastructure Directory Service*. The service can be queried regarding configuration settings and is built to be enhanced for other purposes such as finding a suitable object server to host a replica. A description of GIDS and its integration into GDN and Globe can be found in [4], which has been added as an appendix to this report.

A special utility, called grunt, has been developed to assist in bringing up, and shutting down a site. Grunt keeps track of dependencies between different servers and ensures that these servers are started in the correct order, thus making it easier for a user to start a site.

Finally, our globify tool has been enhanced so that it can now be successfully applied to various organizations of Web sites. Globify assists in converting a collection of files into either a GDN package or a GlobeDoc. The latter is a persistent object designed to store a collection of related Web files into a Globe object, known as a Globe Web document.

We have not implemented monitor tools, nor special tools for removing replicas. System management is mostly handled by the combination of GIDS and grunt, and seems to be sufficient. The Globe Operations Guide (gog) has been updated in such a way that it should be easier to bring up a Globe site.

It is yet unclear whether the lack of monitor tools or those for removing replicas is a serious omission to GDN. The main reason for not implementing these tools is lack of time, caused by having spent more effort in disseminating the results of the project, as well as the integration of GIDS and GDN. I return to dissemination of results below. Integration of GIDS and GDN was more difficult than we expected. In hindsight, we were simply too optimistic. When formulating the SIRS-3 proposal, we had a prototype version of GIDS running that we felt was almost completed. However, this version needed to be adapted to handle the site configuration information stored in the site.cfg file, but should also allow a fall-back to that file if so required (e.g., for overruling the configuration information stored in GIDS).

At this point, we feel more practical and daily experience with GDN is needed before deciding on what the appropriate management tools should be. However, we do intend to continue putting effort in some simple monitoring tools, and include these tools in future releases. I return to these issues below.

3.4 Current status

At present, GDN has seen the release of version 1.0, a month later than the originally planned date of 1 January 2002.

The software for release 1.0 has been tested in various ways. Tests have been conducted with the installment and replication of more than 20 Gbytes of the Sourceforge database. These tests have lead to the identification and repair of numerous faults. In addition, we have asked several non-Globe users to bring up a site giving them only the release 1.0 software and the operations guide. Practice now shows that installment can be done in less than a few hours by an experienced UNIX user, giving us confidence that management of a site has indeed been improved. We do not expect that a site can be brought up easily by an average Internet user.

It should also be noted that the Globe Web site as well as various home pages of its developers, have been hosted by SIRS/GDN software for many months now. The software distribution is, by default, downloaded from the GDN system thereby redirecting a client to the nearest replica. This other form of testing has revealed several smaller errors and has lead to a number of functional improvements.

4 Assessment of SIRS

In the previous sections, we have mainly concentrated on the results for SIRS-3. Formally, SIRS-3 is the last subproject to be completed as part of the much larger SIRS project. In this section, we reconsider the SIRS project as a whole and see what has been achieved with respect to the original plans and where further improvements are needed. In particular, we take a look at the dissemination of the results and future plans

4.1 SIRS in hindsight

One of the original goals of SIRS was to come to a solution for offloading popular FTP sites, notably the NLUUG software archive, which acts as a mirror for many software packages. The brute-force approach by effectively improving network access, storage capacity, and possibly processing power, will generally work (and which has now been successfully applied to the NLUUG site). However, such an approach is not immediately interesting from a scientific and long-term perspective.

The development efforts put into the SIRS projects amount to almost six person-years of work. I estimate that the accompanying research amounts to another four person-years. In addition, there are approximately two person-years of development efforts done researchers that have lead to software that is now integrated into SIRS (notably GIDS and security software), leading to a grand total of 12 person-years. Research and development concerning the Globe location service (at least another eight person-years) have not been included in this calculation.

The scientific output related to SIRS (and again excluding the Globe location service) is quite large: we have published approximately seven papers in conferences and journals. At least two papers are planned for production this year. The combination of the research results and development efforts have improved the visibility of the Globe project as a whole.

In conclusion, the SIRS project from a scientific and development point of view seems to be reasonably successful. However, when evaluating the usability of the results and the impact the project has, there is clearly room for improvement. In the following two sections, we concentrate on two important issues related to making SIRS successful: the dissemination of the results and the future work.

4.2 Dissemination of results

One of the issues for SIRS that has not been dealt with in a satisfactory manner, is the dissemination of the results. It is taking quite some effort to push SIRS/GDN to usage by users other than those that directly or indirectly linked to the Globe project. Let us first take a closer look at what we have done with respect to "marketing" SIRS/GDN.

Source code availability. We decided to make the source code for SIRS/GDN available at the end of the SIRS-2 project. At that point, it was possible to set up a simple GDN system that we felt was stable enough for experimentation. It took us quite some effort to reach this point, mainly because the amount of "baseline" software (object server, name server, client) was quite substantial. Since December 2001, we have made a new release of SIRS/GDN every 1-2 months.

All the code and the gog have been available from the FTP site at the VU. Also, we have made the package available through SourceForge since April

2001. The total number of downloads and groups that have voluntarily started SIRS/GDN is effectively restricted to our own group. We have had mail from a few developers, but it is obvious that the Globe community is essentially only the group at the VU along with a small circle of friends.

Demonstrations. During the course of the SIRS-3 project, it was felt that we should more pro-actively attempt to disseminate the software to interested parties. It was decided to do so by means of a number of demonstrators. At present, GDN is hosting software for Linux (2.4 kernels and the 7.2 updates), Amoeba, and Minix. We are also hosting various Web sites. Experiments have begun to host a large portion of the SourceForge database, replicating it across San Diego (CAIDA), Redwood City (Vixie Enterprises), and Amsterdam (VU). These experiments still need to be completed and require additional development work at the VU.

SIRS/GDN is now running on multiple sites across the Internet. There are various “local” sites hosted in The Netherlands; the main international sites are (in alphabetical order):

- Amerongen, The Netherlands (NLnet)
- Amsterdam, The Netherlands (Vrije Universiteit)
- Haifa, Israel (Technion)
- Rocquencourt, France (INRIA)
- San Diego, California, USA (CAIDA)
- Sao Paulo, Brasil (University of Sao Paulo)

Meanwhile, we are continuing our efforts to host content. For example, attempts are being made to see whether SIRS/GDN can act as a replicated mirror of `www.faqs.org`. Also, we have contacted people that may be able to provide us access to Internet2 nodes to expand the SourceForge experiment across more nodes.

Publications. Being a group of researchers, we are keen on publishing scientific papers in which SIRS/GDN can play a role. As mentioned above, we now have a handful of papers (and accompanying presentations) on SIRS/GDN. A next presentation is planned for SANE 2002, including an accompanying paper describing the system from the perspective of its users; a paper describing SIRS/GDN is also planned for submission to Wiley’s Software Practice & Experience.

Despite these efforts so far, it turns out that disseminating SIRS/GDN to the community requires considerable effort. Although Globe, by now, has established a good reputation, more effort is clearly needed to bring it to the public. It is not obvious what the best strategy is that we can follow. However, we have learned from the current efforts that pro-actively advertising SIRS/GDN is better than just making the package available.

The strategy we are currently following (seeking for sites that are willing to host SIRS/GDN, and letting us manage those nodes) appears to be the best we can do when it comes to demonstrating that SIRS/GDN is for real. We know from colleagues that organizing wide-area experiments often requires a considerable amount of time. However, the results that can be reported at conferences and such tend to help people get convinced that something serious is happening.

In addition, we want to continue our search for hosting content. Unlike the SourceForge experiment in which we focus on large amounts of data, it is presumably more effective for the dissemination of results to concentrate on many different data sources. In this sense, turning SIRS/GDN into a “cheap” content delivery network is perhaps the best way to draw attention. Again, we expect that it may take a relatively long time before SIRS/GDN will prove its value to a larger community, but we see no other way to reach this point.

4.3 Future work

Although SIRS-3 is formally finished, the work on GDN continues. First, as already indicated, we will actively seek for content that can be hosted by GDN. In doing so, we hope to attract attention from content providers seeking for a solution for distributing their content.

Second, and related to hosting real content, we plan to conduct a series of experiments with SIRS/GDN so that we can convince third parties of the technological advances our software has to offer. In addition, experimentation should reveal important faults (bugs), making it possible to improve the robustness of our software. It is already encouraging to see that the performance of our system is comparable to that of standard-configured Apache-based systems.

However, we expect that SIRS/GDN will attract more attention as we continue to enhance its functionality based on our future research. At present, our group is actively conducting research in the following areas:

Adaptive replication: At present, we have one full-time senior researcher looking into the dynamic replication of Web documents. So far, this research has provided us insight in how and when to evaluate access traces in order to decide what a best replication strategy is. Starting in August 2002, two PhD students will be added to the project. One student will concentrate on the selection of the best location to place a replica. The other student will take a look at dynamically switching the consistency protocol so as to optimize network resources and access delays. A separate scientific programmer will be hired to implement the results. Embedding these results in SIRS/GDN is an obvious choice.

Systems management: We are currently looking at management issues related to the Globe location service. One postdoc is considering the problem how we can automatically bring up an entire, worldwide-spanning tree without initially having to physically distribute that tree across the Internet as well. This research will certainly influence the (semi-)automatic distribution of GDN across multiple sites.

In addition, a PhD student has recently started to concentrate on more general systems management issues related to distributed systems such as GDN. Again, the results of this result are planned to be embedded in SIRS/GDN, partly to demonstrate the feasibility, but also to make it easier to deploy SIRS/GDN.

In conclusion, we feel that we have just made a start with SIRS/GDN when it comes it its deployment. However, we are continuing the research that has been set out as part of SIRS/GDN, exemplified by the two subprojects mentioned above. Recognizing that deploying SIRS/GDN may be a matter of much time, it is important that our research is set out along the lines of such a deployment.

5 Agreement on this report

With respect to this report, NLnet Foundation and the Globe team at the Vrije Universiteit have agreed on the following:

1. The deliverables specified in the original SIRS-3 plan have been formally provided, except with respect to a number of management tools as described above.
2. The efforts that have taken place during the SIRS-3 project contract period to disseminate SIRS/GDN are such that no more could have been reasonably expected.
3. The assessment of the SIRS/GDN project as given in this report is sufficient.
4. Both parties explicitly express their commitment to further disseminate SIRS/GDN, and to pro-actively distribute current and new results to the interested community.

6 Financial Report

A financial statement has been added separately to this report.

References

[1] A. Bakker, E. Amade, G. Ballintijn, I. Kuz, P. Verkaik, I. van der Wijk, M. van Steen, and A. Tanenbaum. “The Globe Distribution Network.” In *Proc. Annual Technical Conference (FREENIX Track)*, pp. 141–152, San Diego, CA, June 2000. USENIX.

[2] A. Bakker, M. van Steen, and A. Tanenbaum. “A Law-Abiding Peer-to-Peer Network for Free-Software Distribution.” In *Proc. International Symposium on Network Computing and Applications,* Cambridge, MA, Feb. 2002. IEEE.

[3] J. Ketema. “Fault-Tolerant Master-Slave Replication and Recovery in Globe.” Master’s thesis, Vrije Universiteit Amsterdam, Mar. 2001.

[4] I. Kuz, M. van Steen, and H. J. Sips. “The Globe Infrastructure Directory Service.” *Computer Communications*, 25, 2002. *Scheduled for publication*.

[5] G. Pierre, M. van Steen, and A. Tanenbaum. “Dynamically Selecting Optimal Distribution Strategies for Web Documents.” *IEEE Transactions on Computers*, 51(6), June 2002. *Scheduled for publication*.

[6] M. van Steen, P. Homburg, and A. Tanenbaum. “Globe: A Wide-Area Distributed System.” *IEEE Concurrency*, 7(1):70–78, Jan. 1999.