ReGTime - Rent Gigaflops someTimes

Bernd Dreier, Annja Huber, Markus Zahn, Holger Karl, Theo Ungerer

UNIVERSITÄT AUGSBURG

ReGTime - Rent Gigaflops someTimes

Bernd Dreier Annja Huber Markus Zahn
Holger Karl Theo Ungerer

Report 1996-4 Oktober 1996

INSTITUT FÜR INFORMATIK
D-86135 AUGSBURG

ReGTime - Rent Gigaflops someTimes

Bernd Dreier Annja Huber Markus Zahn
University of Augsburg
Institute of Informatics
D-86135 Augsburg, Germany
{dreier,huber,zahn}@Informatik.Uni-Augsburg.DE

Holger Karl Theo Ungerer
University of Karlsruhe
Institute of Computer Design and Fault Tolerance
D-76128 Karlsruhe, Germany
{hkarl,ungerer}@Informatik.Uni-Karlsruhe.DE

October 1996

Abstract

ReGTime[1] (**Re**nt **G**igaflops some**Time**s) is a software package that helps to rent unused computing power. Sites offer unused resources on a "computing power market". Customers specify their requirements using World Wide Web. **ReGTime** creates an offer based on available capacities. If the offer is accepted, **ReGTime** helps to establish a contract, organizes the access, observes the compliance with the contract, and collects data for invoicing. This way e.g. smaller companies may purchase additional computing power without investing in hardware.

1 Introduction

Today computer networking continues to grow in both size and importance. As a result of increasing bandwidth and capacity of computer networks, information can be transferred in reasonable time even across wide area networks. Using high-speed networks, heterogeneous and distributed computer systems offer the possibility to combine a team of computers over large distances to *one virtual*

[1]ReGTime is available through http://www.informatik.uni-augsburg.de/info1/regtime/

machine. By “distributed computing”, tasks can be solved which are too complex for a single computer.

Selling computing power is already widely used in the area of mainframes and today’s parallel computers. There are only a few providers well-known to their customers. The access is done via ftp/telnet or similar means. The (parallel) program is transferred to the remote computer. It is executed on the provider’s machines and after termination all results are sent back to the customer. No distributed computing occurs.

Recently, research in this area are research aims at “metacomputing” [1] and “virtual computing centers” [2], i.e. the cooperative use of several parallel computers by collaborating computer centers. One example is the “Virtuelle Rechenzentrum Südwest”[3], another one the connection of an Intel Paragon XP/S10 of KFA Jülich and an IBM SP2 owned by the GMD using an ATM-connection [4]. A parallel program can be distributed over both parallel machines, however, the remote connection is much slower than internal connections. Here, too, only a few participants exist which are well known to each other.

Workstation clusters — sometimes even several collaborating clusters — are also used as a virtual machine. Spreading work over several workstation clusters requires accounts on each computer of every cluster. Several software packages (e.g. MPI[5], PVM[6] and Linda[7]) or distributed systems (e.g. DCE[8] and CORBA[9]) allow networked computers to appear as a single concurrent computational resource. Up to now, this way of using networked computer systems is mostly found at universities and research institutions. However, both technical and theoretical know-how for parallel and distributed computing will be available in middle and small sized companies soon.

It is the idea of **ReGTime** that managers of networked computers offer their free capacity on a “computing power market”, and customers are able to rent resources on a short-term basis. **ReGTime** negotiates the offers between providers and customers. Small and middle-sized organizations are able to use high computing capacities temporarily without high investments for powerful computer systems. The managers themselves increase the profitability of their machines through this additional income.

ReGTime primarily aims at small and middle-sized companies as customers. As providers, institutions with one or several workstation-clusters are our main focus. Another target group are corporations with computers linked via corporate networks — thereby improving the utilization of their own resources.

Considering the fast progress of networking via internet and intranet, the prerequisites to offer and sell unused computing power on a market are fulfilled. This lays the foundations of the envisioned “computing power market”. In such a market, many potential participants (providers and customers of computing power) exist. There is substantial demand for brokering, since the potential users of distributed systems still miss a system that provides easy access to rentable computing power. These are our reasons for creating **ReGTime**, a software package to offer, broker and rent computing power.

The idea of an internet-based brokering service showed up in several contexts recently: FAST [10] is a project intended to manage the procurement of electronic parts. The BargainFinder [11] searches at several internet-CD-shops for cheap CDs. The MeDoc-project [12] deals with searching for (mostly computer science related) publications in the internet. The "Information Broker" of MeDoc takes requests for some kind of information and answers with references where the information can be found. Except for **ReGTime**, none of today's brokering services is concerned with renting computing power.

2 Functionality of ReGTime

2.1 The basic building blocks

There are two main groups in **ReGTime**: providers and customers. A *customer* needs high computing power and is looking for providers that allow the rental of their underutilized computers. Providers will grant the necessary rights based upon a contract.

A *provider* offers his computers (grouped into one or several clusters) for rental. A computer is characterized by its hard- and software features and by the times at which the machine is available for rental. The provider specifies the computing nodes and the accounts available to the customers. To easily administer these informations, **ReGTime** provides an easy-to-use configuration tool to set parameters like offered nodes, prices, and rental times.

To simplify the negotiations between customer and (several) providers, a *broker* can be used (in resemblance to existing market mechanisms). It mediates requests of customers and contracts between customer and one or several providers. The broker itself does not contract with either party. After publishing his offer (e.g. in the WWW) or registering with some brokers, a provider is part of the "computing power market". This registration can be done automatically, no further human interaction is necessary.

ReGTime is independent of any distributed programming environment (e.g. PVM [6] or MPI [5]) a customer wants to use. Providers specify their available software packages, customers request any particular system explicitly. **ReGTime** could even broker non–distributed software like hardware synthesis tools or numerical packages.

2.2 A session with ReGTime

To give an impression of the dynamic appearance of **ReGTime**, we present a short guided tour of a session with **ReGTime** from request to invoice (see figure 1).

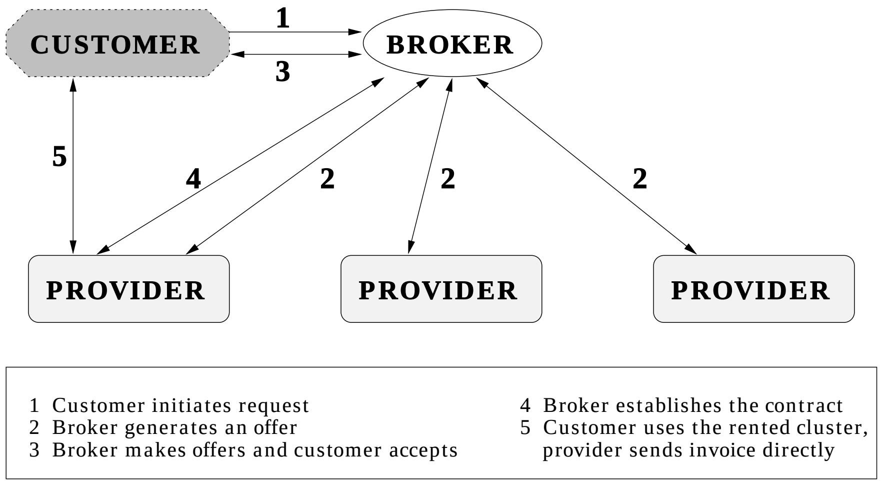
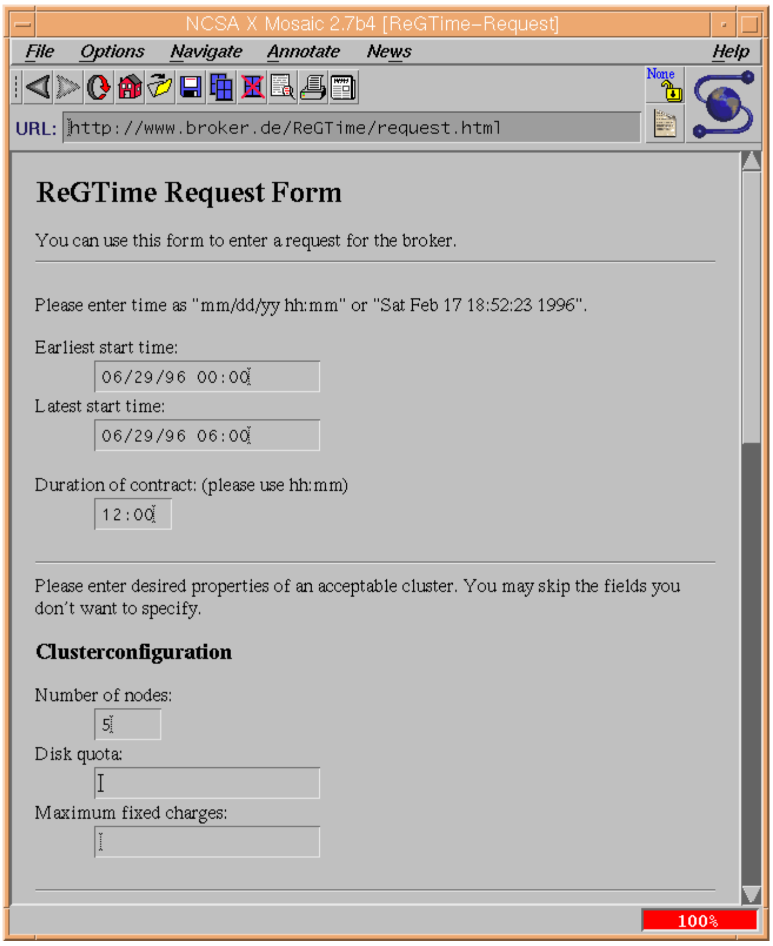

CUSTOMER
BROKER
1
3
5
4
2
2
2
PROVIDER
PROVIDER
PROVIDER
1 Customer initiates request
2 Broker generates an offer
3 Broker makes offers and customer accepts
4 Broker establishes the contract
5 Customer uses the rented cluster, provider sends invoice directly

Figure 1: Flow chart of ReGTime

A customer requests computational power by entering parameters like number and types of acceptable computers, operating system and software prerequisites, and the contract's time frame in a WWW form (see figure 2). The time frame can be specified flexibly by an earliest and latest starting point and a duration. Additionally, price limits can be set for fixed costs, time–dependent costs, cpu–, memory–, and I/O–related costs individually.

Upon receipt of a customer's request, the broker queries all registered providers. If a provider is able to satisfy the request even if only partially, the provider answers the request with an exact description of all available computers, the available times and costs. The broker assembles the replies of the providers to one or several offers for the customer (see figure 3). Usually, the customer receives several offers one of which he may choose. Some of these offers may combine several providers. Combined offers are necessary if a request cannot be satisfied by a single provider. The broker tries to minimize the number of different providers needed to satisfy a request. Using a subsumption relation between offers a heuristic keeps the number of redundant offers small.

The customer can choose from the offers which are presented in a WWW form. After accepting a specific offer the customer has to provide his email address, account name, and the internet name of the computer from which he will use the



http://www.broker.de/ReGTime/request.html

ReGTime Request Form

You can use this form to enter a request for the broker.

Please enter time as "mm/dd/yy hh:mm" or "Sat Feb 17 18:52:23 1996".

Earliest start time: 06/29/96 00:00

Latest start time: 06/29/96 06:00

Duration of contract: (please use hh:mm) 12:00

Please enter desired properties of an acceptable cluster. You may skip the fields you don't want to specify.

Clusterconfiguration

Number of nodes: 5

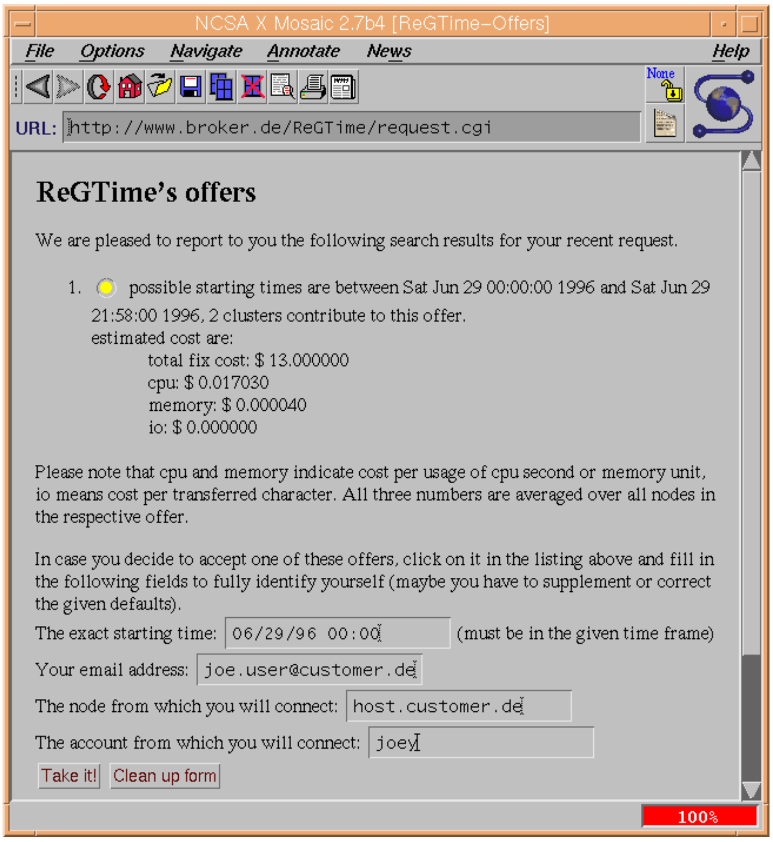
Disk quota:

Maximum fixed charges:

Figure 2: ReGTime's request WWW interface

rented systems. The final steps to make a contract with the providers are initiated by the broker. The participating providers receive the contract. A provider may require the customer to sign the contract (see section 2.3 for a discussion of the security issues involved). Finally, each provider sends a confirmation to the broker, which generates a message for the customer, containing the machine and account names the customer rented.

At the starting time — set in the contract — access to the rented systems is granted to the customer by means of the `.rhosts`-mechanism. Thus access is provided by regular accounts. Shortly before termination of the contract the customer is advised to finish working soon and to transfer back results. All customer processes on the rented machines are notified by sending them a SIGPWR. Upon termination of a contract, all these processes of a customer are terminated (SIGTERM), and after a short grace period, all potentially remaining processes


NCSA X Mosaic 2.7b4 [ReGTime–Offers]
File
Options
Navigate
Annotate
News
Help
URL: http://www.broker.de/ReGTime/request.cgi
Take it!
Clean up form
100%

ReGTime's offers

We are pleased to report to you the following search results for your recent request.

1. possible starting times are between Sat Jun 29 00:00:00 1996 and Sat Jun 29 21:58:00 1996, 2 clusters contribute to this offer.
estimated cost are:
total fix cost: $ 13.000000
cpu: $ 0.017030
memory: $ 0.000040
io: $ 0.000000

Please note that cpu and memory indicate cost per usage of cpu second or memory unit, io means cost per transferred character. All three numbers are averaged over all nodes in the respective offer.

In case you decide to accept one of these offers, click on it in the listing above and fill in the following fields to fully identify yourself (maybe you have to supplement or correct the given defaults).

The exact starting time:	06/29/96 00:00 (must be in the given time frame)
Your email address:	joe.user@customer.de
The node from which you will connect:	host.customer.de
The account from which you will connect:	joey

Figure 3: A sample offer of a contract

on the rented machines are killed (SIGKILL), the account is locked and the home directory is deleted.

During a contract, all customer activities are logged. These accounting informations are used to compute the invoice. CPU-times, average memory consumptions and I/O-activities of the customers' processes may be charged for. The prices for each category are part of the contract. Additionally, fixed costs for a contract and costs based solely upon the duration of a contract are possible.

The invoice is still a bill sent by email. This will change as soon as electronic cash is widely available.

2.3 Security issues

Considering the bad security properties of today's UNIX–systems granting access to an unknown customer is risky. At least, a provider should be certain about the identity of his customers. **ReGTime** approaches this problem by using authentification procedures based upon the public-key mechanism PGP [13].

In the procedure detailed in section 2.2, authentification of the customer is solely based upon node and account names. This information can be manipulated easily and is therefore not acceptable for authentification.

Both provider and customer may demand that contracts must be signed. In this case, a provider sends the complete contract back to a tool executed on the customer host. It passes the contract to PGP for signature and sends the signed contract back to the provider. The provider asks PGP to check the signature against a copy of the customer's public key. If PGP validates this signature, and both the original version of the contract and the signed one are identical, the provider considers the customer to be correctly authenticated and the contract is accepted.

Due to the PGP method it is essential under which prerequisites a provider is willing to add someone's public key to his key ring. If a provider accepts a key received in an email message, the whole authentification is not securer than the email system itself, which is not very secure. The highest security level would be achieved, if keys are only accepted after a personal introduction with a valid identity card. A provider can choose any level of security somewhere along these lines. This additional effort is necessary only once to guarantee the identity of public keys.

The question of how to make legally binding contracts over the internet still remains open.

3 Implementation

ReGTime implements the tasks listed above by a set of three Unix dæmons (see figure 4): `brokerd` acts as the broker i.e. it communicates with the customer by World Wide Web. The `providerd` is working as a provider's manager, it negotiates on behalf of a provider and establishes contracts. To observe compliance with the contract, a third dæmon acts on the provider's side, running on every rentable machine: `guardiand` mainly controls access, terminates processes at the end of a contract and computes invoices. For portability reasons most of the guardian's functionalities are implemented as Perl scripts (cp. [14]).

ReGTime's customer interface is designed as a WWW form. Customers' input is forwarded (via sockets) to the associated broker (`brokerd`) by CGI scripts. Thus, the broker may run on a different host as the http-dæmon does. Every request is handled by a separate process forked by the broker. To provide a secure signature, a program called `sign-it` establishes an interface between **ReGTime** and PGP and the customer's site.

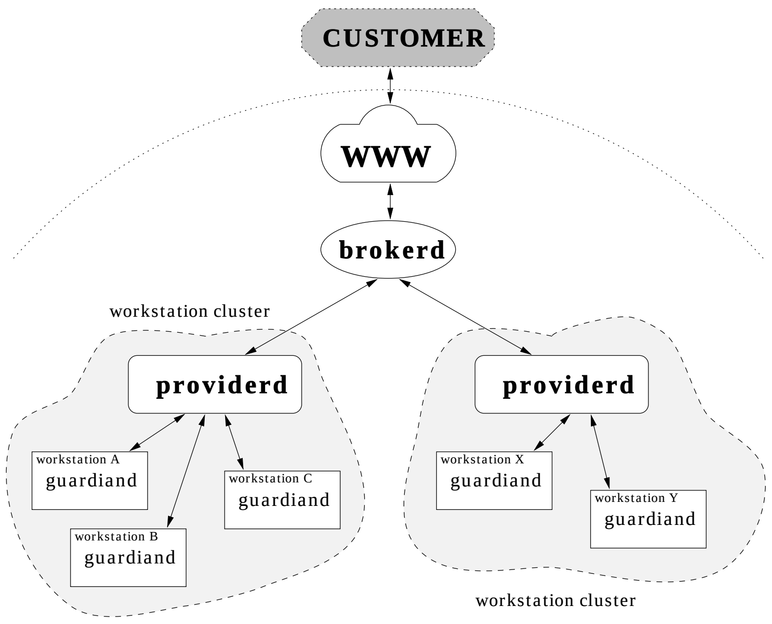


Figure 4: Implementation of ReGTime

Accounting information is gathered by the operating system of the rented machines. Today, almost all wide-spread UNIX-systems offer a `pacct`-like accounting system. Using this system, the kernel generates information for every terminating process, containing its run-time and resource consumption (like cpu time, memory and I/O). This data is used by the `guardiand` to assemble the invoice. The invoice is sent to the customer via email.

4 Installation

Prerequisites for a successful installation are:

- UNIX operating system
- ANSI-C compiler (e.g. gcc)
- Perl (Version 4.036)
- `pacct`–accounting system (optional)
- PGP (Version 2.6.3 or higher) for electronic signatures (optional)
- Tcl/Tk (Version 7.3/3.6 or higher) to use graphical configuration tools (optional)

At first the file `config.data` in the `sources` directory has to be modified. `config.data` includes a number of variables that have to be set according to the local system configuration. Then, `make config`, `make` and `make install` builds the complete system, i.e. the `providerd`, `guardiand` and `brokerd`. The accounts used for renting must be set up separately. The `brokerd` can also be installed independently from the provider's software. Finally, the graphical configuration tools of **ReGTime** allow the provider to specify the node and cluster information (like prices and availability) in a user-friendly manner.

If a customer doesn't want to use signatures, no special software for **ReGTime** is needed by a customer. Otherwise, a customer only needs an ANSI-C compiler and PGP and has to install `sign-it`.

5 Results and experiences

ReGTime was tested on SUN SparcStations running Solaris 2.4 and IBM RS/6000 workstation with operating system AIX 4.1. The demonstration system consisted of two independent workstation clusters at the universities of Augsburg (up to 20 IBM workstations) and Karlsruhe (four SparcStation5). **ReGTime** turned out to be easy to use, access to computing power is provided fast and comfortable. We have successfully tried several existing distributed applications (based upon PVM) in conjunction with **ReGTime**.

A first prototype was presented at CeBIT'96 in Hannover. The described clusters were accessed from the exhibition, an additional IBM RS/6000 on the fair ground acted as a third cluster. The audience widely accepted **ReGTime** and its idea of a "computing power market", although doubts concerning provider's security risks were raised. Therefore, **ReGTime** puts a strong emphasis on authentification of customers. If a provider is able to restrict the range of customers to trustworthy persons (like members of the same corporate network, or the same company or university), the security problems are cut down. From the customer's point of view, its software is unlikely to be abused as it is (as a rule) only a small part of a distributed application. With distributed systems like DCE, data transfer between different components of a distributed application can also be encrypted.

Further experiments were done to test the potential loss of communication speed when parallel programs are executed distributed over several, remotely coupled workstation clusters. The latencies incurred by small messages between two workstations clusters, one in Karlsruhe and one in Augsburg, were surprisingly small, sometimes even within one order of magnitude to the latencies in a local ethernet–based cluster. A master–worker–paradigm turned out to be nearly as efficient when running on two remote clusters as when running only locally.

6 Conclusion

ReGTime is a software package for an envisioned “computing power market”. It consists of three main parts. At first, **ReGTime** helps customers to search for providers who allow rental of their workstation clusters. Second, it manages the rental of disposable machines on provider’s side and contracts with the customer. Third, it is responsible for granting access to rented systems and compliance with established contracts. Furthermore it supports secure authentification between business partners by PGP signatures.

Testing of **ReGTime** is successfully finished. During our presentation of **ReGTime** at CeBIT’96 in Hannover visitors showed great interest in using and providing space in the “computing power market”. Our next step is to test **ReGTime** in industrial projects.

References

[1] Larry Smarr and Charles E. Catlett. Metacomputing. *Communications of the ACM*, 35:45–52, June 1992.

[2] Friedhelm Ramme. Building a virtual machine-room — a focal point in metacomputing. *Future Generation Computer Systems*, 11:477–489, 1995.

[3] Rechenzentrum Karlsruhe. Virtuelles Rechenzentrum Karlsruhe. http://www.uni-karlsruhe.de/Uni/RZ/VRZ/, 1996.

[4] M. Weber and E. Nagel. Metacomputing zwischen KFA und GMD: Ein verteilter massiv-paralleler Rechner. *PARS-Mitteilungen*, 14:9–18, December 1995.

[5] Message Passing Interface Forum. *MPI: A Message Passing Interface Standard*, June 1995.

[6] V.S. Sunderam, A. Geist, J. Dongarra, and R. Mancheck. The PVM concurrent computing system: Evolution, experiences, and trends. *Parallel Computing*, 20:531–546, April 1994.

[7] N. Carriero, D. Gelernter, T.G. Mattson, and A.H. Sherman. The Linda alternative to message-passing systems. *Parallel Computing*, 20:633–655, April 1994.

[8] Jr. Harold and W. Lockhart. *OSF DCE Guide to Developing Distributed Applications.* McGraw-Hill, Inc., 1994.

[9] Object Management Group. *The Common Object Request Broker: Architecture and Specification*, July 1995.

[10] Craig Milo Rogers, Anna-Lena Neches, and Paul Postel. Transitioning to the web. In *Online Procurement at ISI.* World Wide Web Fall Conference, 1994.

[11] Andersen Consulting. BargainFinder Agent. http://bf.cstar.ac.com/bf/, 1996.

[12] A. Brüggemann-Klein, G. Cyranek, and A. Endres. Die fachlichen Informations- und Publikationsdienste der Zukunft — eine Initiative der Gesellschaft für Informatik. pages 2–12. GISI, 1995.

[13] Simson Garfinkel. *PGP: Pretty Good Privacy.* O'Reilly & Associates, Sebastopol, CA, March 1995.

[14] Larry Wall and Randal L. Schwartz. *Programming Perl.* O'Reilly & Associates, Inc., Sebastopol, CA, 1991.