U.S. DEPARTMENT OF ENERGY
OAKLAND OPERATIONS OFFICE
OFFICE OF PUBLIC AFFAIRS
1301 CLAY STREET, 700N
OAKLAND, CA 94612-5208

UNCLASSIFIED FAX MESSAGE

TO: Rachel Claus FAX #: (650) 926-5360

SLAC DATE: 9/5/00

Legal Counsel

Message Consists of this Cover Sheet and 6 Pages

FROM: RoseAnn Pelzner-Goodwin
FOIA Officer, OPA, DOE/OAK

MESSAGE:

As we Discussed. Thank You!

OFFICE OF PUBLIC AFFAIRS FAX # (510) 637-2015 VER: (510) 637-3195

FROM: RoseAnn Pelzner-Goodwin
FOIA Officer, OPA, DOE/OAK

MESSAGE:

As we Discussed. Thank You!

OFFICE OF PUBLIC AFFAIRS **FAX # (510) 637-2015** **VER: (510) 637-3195**

REASON FOR ERROR
E-1) HANG UP OR LINE FAIL
E-2) BUSY
E-3) NO ANSWER
E-4) NO FACSIMILE CONNECTION

FILE MODE	OPTION	ADDRESS (GROUP)	RESULT	PAGE
675 MEMORY TX		916509265360	OK	P. 7/7

* * * COMMUNICATION RESULT REPORT (SEP. 5.2000 3:57PM) * * *

TTI DOE/OAK/OPA/COS

P. 1

FOIA # 2000-01K-71

STANFORD LINEAR ACCELERATOR CENTER

FACSIMILE TRANSMITTAL SHEET

TO: Rose Ann Pelzner Goodwin	FROM: Rachel Claus
COMPANY: DOE	DATE: 9-1-00
FAX NUMBER: (510) 637-2015	TOTAL NO. OF PAGES INCLUDING COVER: 2
PHONE NUMBER: (510) 637-3195	SENDER'S REFERENCE NUMBER: (650) 926-5360 BSD
RE:	YOUR REFERENCE NUMBER: (650) 926-4343

☐ URGENT ☐ FOR REVIEW ☐ PLEASE COMMENT ☐ PLEASE REPLY ☐ PLEASE RECYCLE

NOTES/COMMENTS:

CONFIDENTIALITY NOTICE

The documents accompanying this facsimile transmission may contain confidential information belonging to the sender. Such information may be legally privileged. The information is intended only for the use of the individual or entity named above. If you are not the intended recipient, you are hereby notified that any disclosure, copying, distribution or the taking of any action in reliance on the contents of this facsimile information is strictly prohibited. If you have received this facsimile in error, please immediately notify us by telephone to arrange for the return of the original documents to us.

U.S. DEPARTMENT OF ENERGY
OAKLAND OPERATIONS OFFICE

FREEDOM OF INFORMATION ACT OFFICE

DATE: September 20, 2000

TO: Rachel Claus, SLAC

PHONE: (650) 926-4343

FAX #: (650) 926-5360

FROM: RoseAnn Pelzner Goodwin

PHONE: (510) 637-3195

Number of pages to follow: 1

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Thank you for your assistance on this FOIA.

Rick Vergas was going to give you a call concerning releasing the documents to our office rather than directly to the requester and that the charges should be payable to the U.S. Treasurer. Hope he already did so.

FAX (510) 637-2015
1301 CLAY STREET
OAKLAND, CA 94612-5208

THIS MESSAGE CONTAINS ONLY CONFIDENTIAL INFORMATION AND IS LEGALLY PRIVILEGED. THE INFORMATION IS INTENDED ONLY FOR THE USE OF THE INDIVIDUAL OR ENTITY NAMED ABOVE.

* * * COMMUNICATION RESULT REPORT (SEP.20.2000 10:43AM) * * *

TTI DOE/OAK/OPA/COS

FILE MODE	OPTION	ADDRESS (GROUP)	RESULT	PAGE
936 MEMORY TX		916509265360	OK	P. 2/2

REASON FOR ERROR
E-1) HANG UP OR LINE FAIL
E-3) NO ANSWER
E-2) BUSY
E-4) NO FACSIMILE CONNECTION

PHONE: (510) 637-3195

Number of pages to follow: 1

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Thank you for your assistance on this FOIA.

Rick Vergas was going to give you a call concerning releasing the documents to our office rather than directly to the requester and that the charges should be payable to the U.S. Treasurer. Hope he already did so.

FAX (510) 637-2015
1301 CLAY STREET
OAKLAND, CA 94612-5208

THIS MESSAGE CONTAINS ONLY CONFIDENTIAL INFORMATION AND IS LEGALLY PRIVILEGED. THE INFORMATION IS INTENDED ONLY FOR THE USE OF THE INDIVIDUAL OR ENTITY NAMED ABOVE.

ATTN: Mr. J. David Hadden

Please submit a new check made out to "Treasurer of the U.S." and send to * address below. Also send documents attached to this packet. The other letter and check may be discarded.

Pelzner, Roseann

Thank you!
Rose Ann Pelzner

From: Dea, Daniel
Sent: Tuesday, January 02, 2001 12:54 PM
To: Pelzner, Roseann
Subject: Have remitter send check to ...

F U.S. DOE
AFSC
PO BOX 5400
ATTN: Grace Crook
ALBUQUERQUE, NM 87185

Dan Dea

U.S. DOE
Oakland Operations Office
1301 Clay St
Oakland, CA 94612

dan.dea@oak.doe.gov
Phone- 510.637.1581 (Desk)
510.637.2006 (Fax)

Department of Energy
Oakland Operations Office
1301 Clay Street
Oakland, California 94612-5208

FILE COPY

September 20, 2000

CERTIFIED MAIL
RETURN RECEIPT REQUESTED

Mr. J. David Hadden, Esq.
McCutchen, Doyle, Brown & Enersen
3150 Porter Drive
Palo Alto, California 94304-1212

Subject: Freedom of Information Act Request - #2000-OK-71

Dear Mr. Hadden:

This office has been advised by Rachel Claus, Legal Counselor, of Stanford Linear Accelerator Center (SLAC), that she has provided you responsive documents to your Freedom of Information Act request of August 30, 2000.

Although the requested records concern SLAC and are kept at their facility, they are considered to be the property of the U.S. Government. This is to inform you, therefore, that the funds should be remitted to the U.S. Department of Energy and made payable to the "Treasurer of the United States". An invoice in the amount of $534.35 will follow from our Finance Division.

If you have any questions, please contact me at (510) 637-3195.

Sincerely,

RoseAnn Pelzner
FOIA Officer

STANFORD LINEAR ACCELERATOR CENTER
Operated for the U.S. Department of Energy by Stanford University

Business Services Division
2575 Sand Hill Road, MS 02
Menlo Park, CA 94025

rachel@slac.stanford.edu (e-mail)
(650) 926-4343 (phone)
(650) 926-5360 (fax)

September 7, 2000

David Hadden, Esq.
McCutchen, Doyle, Brown & Enersen
3150 Porter Drive
Palo Alto, California 94304-1212

re: FOIA Request No. 2000-OK-71
SLAC Archives & History of the WEB

Dear Mr. Hadden:

Enclosed please find the response from our DOE FOIA Officer, approving the provision of 1302 copies (651 pages) of documents from SLAC Archives & History Office. These were researched, copied, and assembled in 15 staff-hours. The FOIA provides for repayment for this on a cost-recovery basis. The hourly staff charge is $29.18 an hour for the Archivist (10 hours, $291.80) and $42.00 an hour for supervision, direction, and research by Louise Addis (5 hours, $210.00); the per page copy charge is $0.05 (651 pages, $32.55). Please remit $534.35, payable to the Stanford Linear Accelerator Center.

Each copy has been marked on the back to indicate the location of the original document in the accession.

Only the title pages of the SLAC numbered publications were photocopied because the full-text is available online at *http://www.slac.stanford.edu/spires/hep/*. Similarly only title or first pages of other web-accessible documents are copied to indicate their location in the collection.

Sincerely,

Rachel Claus

Rachel Claus
SLAC Legal Counsel

RC:nw

Enclosure: a/s

cc: ✓ RoseAnn Pelzner, FOIA Officer, DOE/Oak (w/enc.)
Tyndal Lindler, Contracting Officer, DOE/Oak (w/o enc.)

[g:\foia\HaddenD.McCutchen.9-00.rlc]

STANFORD LINEAR ACCELERATOR CENTER
Operated for the U.S. Department of Energy by Stanford University

Business Services Division
2575 Sand Hill Road, MS 02
Menlo Park, CA 94025

rachel@slac.stanford.edu (e-mail)
(650) 926-4343 (phone)
(650) 926-5360 (fax)

September 7, 2000

Ms. RoseAnn Pelzner
Office of Public Affairs, Floor 8
Oakland Operations Office
U.S. Department of Energy
1301 Clay Street, Room 700N
Oakland, California 94612-5208

re: FOIA Request No. 2000-OK-71

Dear Ms. Pelzner:

Provided herewith are copies of the public domain documents we have sent directly to Mr. David Hadden under the above-referenced FOIA request.

Sincerely,

Rachel Claus

Rachel Claus
SLAC Legal Counsel

RC:nw

Enclosures: a/s

cc: David Hadden, McCutchen
Tyndal Lindler, DOE/OAK/CAPD

[z:\foia\SLAC Archives.Web History.Pelzner.9-7-00.rc]

DOE F 1325.8
(12-84)

U. S. Department of Energy
Oakland Operations Office

DATE: 9/5/00

REPLY TO: RoseAnn Pelzner, DOE/OAK FOIA OFFICER
SUBJECT: **REQUEST FOR ASSISTANCE ON FOIA CASE #:** 2000-OK-71
TO: Rachel Claus, SLAC, Legal Counsel

Your assistance is needed to search for records responsive to the attached Freedom of Information Act (FOIA) Request. All responsive records are to be sent to me, the DOE/OAK FOIA Officer, for review prior to release. Questions regarding the request are to be directed to me and I will thereafter contact the requester if the need exists.

The responsive records must be provided to me by: as soon as possible
If this date cannot be met advise me as soon as possible so that I may inform the requester of the delay and the reason why we are unable to respond to the request within the 20 days required by law. If the request is too broad or is not specific enough please notify me immediately.

The requester has agreed to pay $ yes! for records responsive to this request. If the search and copying costs for this request will exceed this amount, please contact me immediately. [optional]

In order to assure that fees are properly assessed for this request please provide the following information with the responsive records:
Time spent searching for and retrieving records: # 15 hours
Hourly salary of searching/retrieving employee: $ 10@$29.18/hour and 5@$42.00/hour.
Number of pages provided in response to Request: # 651 pages

If you have any questions regarding this matter please call me at (510) 637-3195. My fax number is (510) 637-2015. Thank you for your prompt attention to this matter. I remain looking forward to hearing from you by the deadline.

RoseAnn Pelzner
RoseAnn Pelzner
FOIA Officer

MCCUTCHEN
McCUTCHEN, DOYLE, BROWN & ENERSEN, LLP

08-31-00P03:30 RCV

August 30, 2000

Direct: (650) 849-4898
jdhadden@mdbe.com

Rachel Claus, Esq.
Stanford Linear Accelerator Center
2575 Sand Hill Road
Mail Stop 02
Menlo Park, CA 94025

Freedom of Information Request:
Stanford Linear Accelerator Center ("SLAC")

Dear Rachel:

Pursuant to the Freedom of Information Act (FOIA), 5 U.S.C. 552, we request copies of all records, documents, source code, software, memos, notes, emails, or correspondence, whether electronic or in hard copy, that constitute, discuss, relate to, or mention the materials and topics listed on Attachment A hereto. We believe that knowledgeable people at SLAC include Louise Addis, Jean Marie Deken, George Crane, Tony Johnson, Paul Kunz, Mark Barnett, Terry Hung, Patricia Kreitz, Les Cotrell, and Bebo White. We would be happy to reimburse you for any copying fees incurred in complying with our request. Thank you in advance for your assistance. Please call me if you would like to discuss these requests.

Sincerely yours,

J. David Hadden

J. David Hadden

3150 Porter Drive
Palo Alto, California 94304-1212
Tel. (650) 849-4400 Fax (650) 849-4800
www.mccutchen.com

San Francisco
Los Angeles
Walnut Creek

Palo Alto
Taipei

DOE F 1325.8
(12-84)

U. S. Department of Energy
Oakland Operations Office

DATE: 9/5/00

REPLY TO: RoseAnn Pelzner, DOE/OAK FOIA OFFICER
SUBJECT: **REQUEST FOR ASSISTANCE ON FOIA CASE #:** 2000-OK-71
TO: Rachel Claus, SLAC, Legal Counsel

Your assistance is needed to search for records responsive to the attached Freedom of Information Act (FOIA) Request. All responsive records are to be sent to me, the DOE/OAK FOIA Officer, for review prior to release. Questions regarding the request are to be directed to me and I will thereafter contact the requester if the need exists.

The responsive records must be provided to me by: as soon as possible
If this date cannot be met advise me as soon as possible so that I may inform the requester of the delay and the reason why we are unable to respond to the request within the 20 days required by law. If the request is too broad or is not specific enough please notify me immediately.

The requester has agreed to pay $ yes! for records responsive to this request. If the search and copying costs for this request will exceed this amount, please contact me immediately. **[optional]**

In order to assure that fees are properly assessed for this request please provide the following information with the responsive records:
Time spent searching for and retrieving records: # _____ hours
Hourly salary of searching/retrieving employee: $ _____
Number of pages provided in response to Request: # _____ pages

If you have any questions regarding this matter please call me at (510) 637-3195. My fax number is (510) 637-2015. Thank you for your prompt attention to this matter. I remain looking forward to hearing from you by the deadline.

RoseAnn Pelzner
FOIA Officer

STANFORD LINEAR ACCELERATOR CENTER

Operated for the U.S. Department of Energy by Stanford University

Business Services Division
2575 Sand Hill Road, MS 02
Menlo Park, CA 94025

rachel@slac.stanford.edu (e-mail)
(650) 926-4343 (phone)
(650) 926-5360 (fax)

September 1, 2000

Ms. RoseAnn Pelzner Goodwin
Office of Public Affairs, Floor 8
Oakland Operations Office
U.S. Department of Energy
1301 Clay Street, Room 700N
Oakland, California 94612-5208

re: FOIA Request
Web Development Materials

Dear Ms. Pelzner Goodwin:

We forward the attached to you for review. I see no impediment to the release of the requested documents; they are in the public domain, albeit some may be difficult to access without knowledgeable guidance from the named SLAC staff. Please let me know at your earliest convenience if we may proceed.

Sincerely,

Rachel Claus

Rachel Claus
SLAC Legal Counsel

Enclosure: a/s

cc: ✓ Dave Hadden, McCutchen

[z:\foia\WebDevelop.Mats.Pelzner-Goodwin.9-1-00.rc]

STANFORD LINEAR ACCELERATOR CENTER
Operated for the U.S. Department of Energy by Stanford University

Business Services Division
2575 Sand Hill Road, MS 02
Menlo Park, CA 94025

rachel@slac.stanford.edu (e-mail)
(650) 926-4343 (phone)
(650) 926-5360 (fax)

September 1, 2000

Ms. RoseAnn Pelzner Goodwin
Office of Public Affairs, Floor 8
Oakland Operations Office
U.S. Department of Energy
1301 Clay Street, Room 700N
Oakland, California 94612-5208

re: FOIA Request
Web Development Materials

Dear Ms. Pelzner Goodwin:

We forward the attached to you for review. I see no impediment to the release of the requested documents; they are in the public domain, albeit some may be difficult to access without knowledgeable guidance from the named SLAC staff. Please let me know at your earliest convenience if we may proceed.

Sincerely,

Rachel Claus

Rachel Claus
SLAC Legal Counsel

Enclosure: a/s

cc: Dave Hadden, McCutchen

RECEIVED
SEP 5 2000
OPA/OAK

[z:\foia\WebDevelop.Mats.Pelzner-Goodwin.9-1-00.rc]

Attachment A
To SLAC Request

1. Any and all materials collected, documented or identified by SLAC's Archives and History Office, and/or by SLAC's Library, in the effort (described, at least in part, in SLAC-PUB-7636, April 1998 (copy attached as Exhibit A)) to gather into the archives evidence and documentation of the early days (including 1991 and 1992) of WWW usage or implementation at SLAC.

2. All materials referred to in the following excerpt from Jean Marie Dekin's paper, SLAC-PUB-7636 (copy attached as Exhibit A), to the extent such materials pertain to the period prior to January 8, 1993:

   a. "the Library has been able to provide the Archives with documentation of the Wizards' activity and of the HEP Pre-Prints site. Some documentation has also been preserved by individual WWW Wizards, and all of the computer backup tapes from the founding era of 1991 - 1994 still exist (although not on the current main-frame platform). Individual Wizards have conveyed to the Archives paper documentation that they had retained for "historical" purposes in their own files, including paper copies of e-mail messages that were exchanged about setting up the first server, getting it to work properly, and negotiating changes and additions to the SLAC home page. Wizards and members of the Web Coordinating Committee have also provided hard copies of written papers and presentations on the WWW and SPIRES-HEP, and, those papers usually have included examples of then-current SLAC Web pages."

3. The following materials referred to in SLAC-PUB-7636 (Exhibit A):

   a. The "21 September 1994 Memorandum, To: Appendix to WWW Wizards Report to C. Dickens, From: WWW Wizards Committee, Subject: Brief background info on Web at SLAC (SLAC Archives, unprocessed papers of the WWW Wizards.)," referred to in endnote 9 of SLAC-PUB-7636;

   b. The "A Brief and Biased History of Preprint and Database Activities at the SLAC Library 1962-1994,"p. [26] Appendix to: Addis, Galic, Kreitz and Johnson, "The Virtual Library in Action." Presented at the American Chemical Society (ACS) National Meeting "Chemical Information Symposium"(CINF) "The Library of the Future," Anaheim California, 4 April 1995 (unpublished manuscript, SLAC Archives, unprocessed papers of the WWW Wizards.)," referred to in endnote 8 of SLAC-PUB-7636; and

c. The "Paul F. Kunz, e-mail to Jean Deken 30 March 1998, Re: History of WWW at SLAC," referred to in endnote 7 of SLAC-PUB-7636.

4. Any and all e-mails sent (on any date) to or from the SLAC Library or Archives and History Office, pertaining to any WWW pages, servers, clients, or browsers during the period prior to January 8, 1993.

5. The events referred to by Paul Kunz in Slides 12-14 of his "Bringing the World Wide Web to America" presentation attached as Exhibit B.

6. The events referred to by Louise Addis in the following excerpts from her "First Monday" interview (attached as Exhibit C):

   a. "I can still remember the day Paul Kunz appeared in my office after his trip to CERN. He showed me the Web and we started moving right away. I scurried around and got some accounts that were needed. Paul and Terry Hung set up the server on our mainframe. George Crane was able to easily extend remote SPIRES to talk to the new interface. And I was able to make the SPIRES-HEP database write HTML.";

   b. "And as soon as we were on the Web, we were able to start linking the records in our database to the TeX source at Los Alamos National Laboratory (http://xxx.lanl.gov) where, in August 1991, Paul Ginsparg had started the first e-print server."; and

   c. "When we notified our registered users about how to get a free browser from Tim at CERN, many did right away."

7. Any and all materials of the WWWizards (including, without limitation, Louise Addis, Mark Barnett, George Crane, Tony Johnson, Joan Winters, and Bebo White) pertaining in any way to the WWW and to the period prior to January 8, 1993.

8. For any and all SLAC computer back-up tapes for the period between December 11, 1992 and January 9, 1993, any and all files which have been moved or copied from those tapes which pertain in any way to the WWW.

9. Any and all WWW page(s), server(s), browser(s), or client(s) installed or used at or by SLAC prior to January 8, 1993.

10. Any remote access, or means for remote accessing, of the High Energy Physics (HEP) Database at SLAC using the WWW or HTTP, prior to January 8, 1993.

11. Any and all interfaces or gateways between a WWW server and a SLAC SPIRES-HEP database prior to January 8, 1993.

12. Any and all installations or uses of Wide Area Information Servers (WAIS) servers or clients by or at SLAC prior to January 8, 1993.

13. Any and all installations or uses of the Z39.50 protocol by or at SLAC prior to January 8, 1993.

14. Any and all remote access, or means for remote accessing, of the High Energy Physics ("HEP") Database at SLAC from the Superconducting Super Collider Laboratory ("SSCL") prior to January 8, 1993.

15. The "X-Windows interface" project referred to by Louise Addis in the following excerpt from her "First Monday" interview (attached as Exhibit C:

    a. "Well, the real motivating issue was a project that we been working on earlier. In 1989-1990, I was helping set up a library at the Superconducting Supercollider (SSC) project in Texas. Pat Kreitz, the SCC librarian, wanted to use the SPIRES-HEP database and wanted to present it to the SSC community in a seamless way within a graphical user interface. SSC had funded the development of an X-Windows interface to do this, but as soon as I saw the Web in action, I knew that it was a fast, cheap solution."

16. Memorandum Purchase Order from Superconducting Super Collider Laboratory ("SSCL") to LBL, GFAA-5SG-4302, Award # SSC91WO9978, entitled "Remote Database Object Project HEP" (referred to on page 1 of the May 14, 1992, Petition for Waiver (copy attached as Exhibit D)), and work performed under such award or purchase order.

17. The project referred to in the following excerpt from page 3 of the attached May 14, 1992, Petition for Waiver (copy attached as Exhibit D):

    a. "The scope of the work for SSCL was to provide a working prototype of the Remote Database Object to SSCL configured to support their usage of the High Energy Physics database at the Stanford Linear Accelerator Center."

18. Any and all communications, prior to January 8, 1993, between Patricia Kreitz or other personnel of the Superconducting Super Collider Laboratory ("SSCL") Library and Louise Addis or other personnel of the SLAC Library regarding, in whole or in part, remote access to the SPIRES-HEP database.

19. Any and all remote access, or means for remote accessing, of a SPIRES database at SLAC from Lawrence Berkeley Laboratory ("LBL") prior to January 8, 1993.

20. Any and all communications between personnel of LBL's Computing Science Research and Development (CSRD) department (including Information Planning and

Services, part of CSRD) and SLAC, prior to January 8, 1993, concerning, in whole or in part, SPIRES or HEP database accessing.

21. Any and all SPIRES Consortium Advisory Board meetings during the period from January 1, 1988 to January 8, 1993 including, without limitation, those which were attended by Allan M. Konrad of LBL.

22. Any and all SPIRES workshops during the period from January 8, 1985 to January 8, 1993, including, without limitation, those which were attended by Allan M. Konrad of LBL.

23. Any and all communications between Allan M. Konrad and any SPIRES Consortium staff or member prior to January 8, 1993.

24. The "Remote Object Client Program" source code attached as Exhibit E (note references to SLAC personnel Addis and Gex at pages 66 and 73) and any variations thereof.

SLAC–PUB–7636
April 1998

First in the Web, But Where Are the Pieces?*

Jean Marie Deken

Stanford Linear Accelerator Center
Stanford University
Stanford, California 94309 USA

Invited talk presented at the Society of American Archivists Annual Meeting,
8/27/97—8/31/97, Chicago, IL, USA

* Work supported by Department of Energy contract DE–AC03–76SF00515.

SLAC–PUB–7636
April 1998

First in the Web, But Where Are the Pieces?*

Jean Marie Deken

Stanford Linear Accelerator Center
Stanford University
Stanford, California 94309 USA

Invited talk presented at the Society of American Archivists Annual Meeting,
8/27/97—8/31/97, Chicago, IL, USA

* Work supported by Department of Energy contract DE–AC03–76SF00515.

Georgia Row
(650) 926-4391
ligmr@slac.stanford.edu

Sharon Thayer
(650) 926-4394
sthayer@slac.stanford.edu

Acquisitions Specialist & Monographic Cataloger

Shyue-Rong (Victoria) Sha
(650) 926-4386
vsha@slac.stanford.edu

Archives

Jean Marie Deken
Archivist
Laura O'Hara
Archives Assistant
(650) 926-3091
slacarc@slac.stanford.edu

HEP Copy Cataloger and Serials Assistant

Denise Van Sandt
(650) 926-4393
vansandt@slac.stanford.edu

UNIX Systems Manager

Richard Dominiak
(650) 926-2339
dominiak@slac.stanford.edu

Circulation Assistant

Stephanie Laszlo
(650) 926-2411
libcirc@slac.stanford.edu

Go Back

Last Updated: September 2, 1999

The task for the coming months, therefore, will be to develop and gain approval for a disposition schedule for SLAC Web documents based on this automated gathering approach. We will also need to investigate the mechanics of retiring permanent Web documents to NARA, as well as to one of the previously mentioned independent repositories, if such a transfer is deemed to be worthwhile. As we go about these tasks, we will also maintain a keen interest in developments in thinking about World Wide Web archival and records management issues coming from other projects currently underway, particularly the Syracuse University project to investigate the status of material posted on state agency World Wide Web Sites [29].

NOTES

1. SLAC Welcome Page, http://www.slac.stanford.edu/

2. Hobbes Internet Timeline, http://info.isoc.org/guest/zakon/Internet/History/HIT.html; also The World Wide Web Consortium's "A Little History of the World Wide Web, http://www.w3.org/History.html.

3. WWW History Project, http://www.webhistory.org/project/book.html:

   "...Our goal is to establish a permanent process for recording and disseminating the ongoing history of the Web and networked information, in as close to real time as possible. Much of our work on the Project has been clearing up the six years worth of myths and half-truths which have accumulated around the Web's origins, precisely because there was no definitive history. We want to make it easier next time!"

4. Fermi-News August 16, 1996, p. 1 and ff. (It should be reiterated here that this Fermi-News article did not, in fact, claim that FermiLab was the first U.S. WWW site.)

5. http://www.w3.org/pub/WWW/People/Berners-Lee/.

6. http://www.w3.org/pub/WWW/History.html

7. ibid. Work on installing the server at SLAC was undertaken by both Terry Hung and Paul Kunz. (Paul F. Kunz, e-mail to Jean Deken 30 March 1998, Re: History of WWW at SLAC:

   "The time between September 1 1991 when I brought a Web browser to SLAC and source code for the server and December when [the] server came up is very long. The explanati[o]n is that Terry Hung .... did not know the SLACVM system very well so dropped the project...Finally in December, I put myself to finishing the server installation since I knew how to handle VM and the interface to SLAC Spires...")

8. Addis, Louise.

   "A Brief and Biased History of Preprint and Database Activities at the SLAC Library 1962-1994," p. [26] Appendix to: Addis, Galic, Kreitz and Johnson, "The Virtual Library in Action." Presented at the American Chemical Society (ACS) National Meeting "Chemical Information Symposium" (CINF) "The Library of the Future," Anaheim California, 4 April 1995 (unpublished manuscript, SLAC Archives, unprocessed papers of the WWW Wizards.)

9. 21 September 1994 Memorandum, To: Appendix to WWW Wizards Report to C. Dickens, From: WWW Wizards Committee, Subject: Brief background info on Web at SLAC (SLAC Archives, unprocessed papers of the WWW Wizards.)

10. 21 Sept 1994 Memorandum, To: Appendix to WWW Wizards Report to C. Dickens, From: WWW Wizards Committee, Subject: Brief Background info on Web at SLAC (SLAC Archives, unprocessed papers of the WWWizards; also Addis, Galic, Kreitz and Johnson, "The Virtual Library in Action" p. 5.)

11. http://www.slac.stanford.edu/slac/www/wwwtech/wwwtech.html

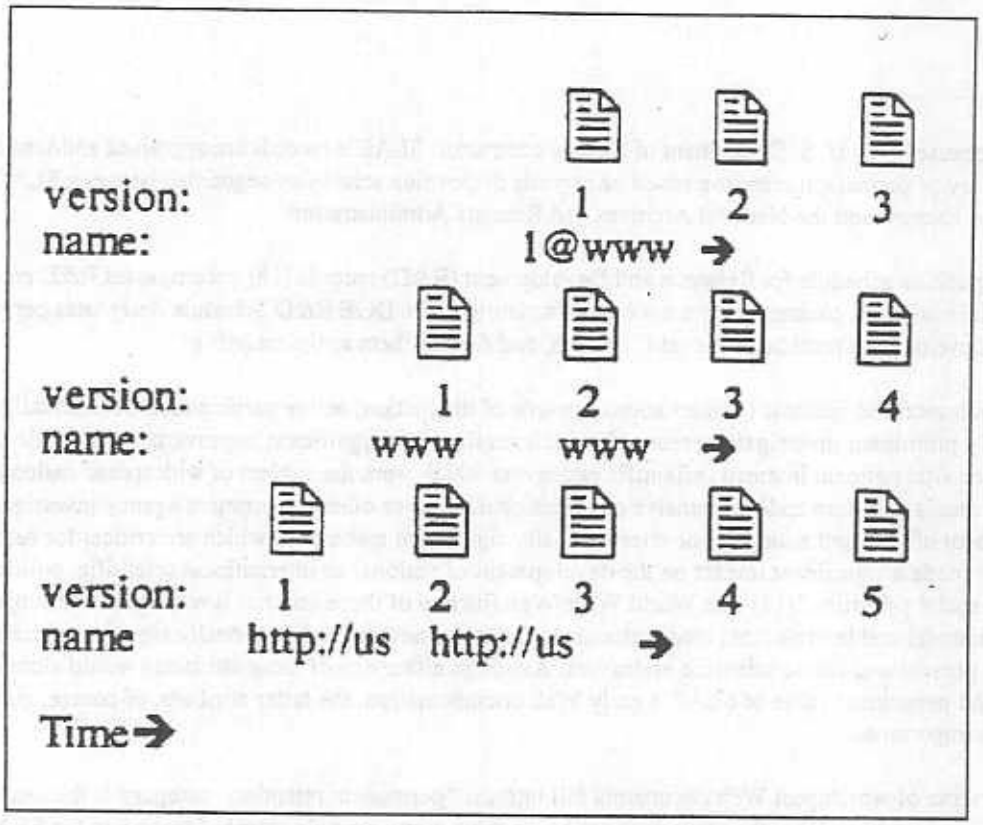


Figure 2

At SLAC, Web page changes have been automatically handled by the backup procedure on the HEP web site by means of a cascading name change that allows a limited number of iterations of a particular web page to be backed up and saved at any given time. In this process, the URL name travels to the newest iteration of the page; and backup name 1 is assigned to the immediately superseded iteration. At the time of the next backup, name 2 is assigned to the first iteration; name 1 is assigned to the second iteration, and the URL name is placed on the most current, or third, iteration. After the upper limit of backup documents of a single page have been stored (at SLAC, this number is six) the next subsequent backup cycle triggers the disposal of the oldest iteration in storage. Using file lists of the backup tapes, one can determine when a particular Web document was created ONLY IF a tape was taken out of the backup cycle, and therefore not overwritten, before an original Web document reached its seventh iteration. In fact, the earliest Web pages at SLAC were set aside through precisely such interventions. In the effort to locate SLAC's first web page on the "archived" backups, a combination of personal recollections of WWW Wizards and information from other electronic records (e-mail) was used to establish a rough estimation of the date of the page. "Archived" backups from that time period were then reviewed for the earliest-dated version of the document name. Using this approach, a digital copy of SLAC's first web page was located on the backups. This same approach will now have to be used to locate the other early web pages which will be appraised as permanent at SLAC, including the first web pages for each experimental group and, most likely, the first web pages for each department. Another historic page that will be sought in the early SLAC web backups is the code for the first World Wide Web page in China, which was set up with assistance from SLAC staff, in support of a collaboration with the Institute of High Energy Physics in Beijing. [22]

THE "DIMENSIONALITY" PROBLEM

Once Web documents have been appraised, and the electronic version of each permanent page has been located, the next issue becomes one of preservation. How should the Web documents be preserved, and who should preserve them? As a U. S. Department of Energy contractor, SLAC has an obligation to retire its permanent records to the

WHEN

The very early days of the World-Wide Web at SLAC date from 1991 to 1994.[10] This is the period that the SLAC Archives and History Office is attempting to reach back and capture, while a separate effort is being made to appraise and arrange for the orderly transfer of any permanently-valuable present and future Web documents. In the beginning, World Wide Web efforts at SLAC had no official status in the organization. This changed some time in 1994, when an ad hoc group called the SLAC WWW Technical Committee was formed to discuss and address technical, "as opposed to stylistic, aesthetic, content or policy" issues. [11] This was followed by the formation of a "SLAC WWW Users Group." Then, in 1995, a World Wide Web Coordinating Committee was established by the Associate Directors' Committee on Computing, "to provide SLAC web authors, groups and committees with policy, guidance, standards, and further support." [12]

WHAT

One of the first Web events at SLAC was a revolution in the delivery of bibliographic information. Since 1974, the SLAC Library had participated in providing SPIRES-HEP, a 300,000 record bibliographic database, to the world particle physics community via the Internet and through clone sites in Europe and Japan. The 1991 introduction of the e-print archives at Los Alamos National Lab, coupled with the World-Wide-Web from CERN, suddenly made it possible for the Library to expand its service and to provide easy linkage between bibliographic database records and the actual full-text of papers.[13] A further improvement was realized when SLAC physicist Tony Johnson introduced "MidasWWW," the first graphical Web browser that could handle compressed files in PostScript, a page description language favored by high-energy physicists because of its ability to describe images in a device-independent manner.[14]

According to Berners-Lee (the Web's creator), the mounting of SPIRES-HEP on the World-Wide Web was a vitally-important factor in the rapid acceptance and utilization of the Web in the international high energy physics community [15]. Driven by increasing pressure to disseminate results as quickly as possible and to as wide a group as possible, the high energy physics community has embraced the World Wide Web because it a significant improvement in the communication of time-sensitive, much-sought-after information. While some of the records mounted on the Web at SLAC are also generated in paper form, many — in fact, less and less as time goes by — exist only in their Web incarnation. The authors see no need to produce the document or information in two formats, and the interactive electronic format of the Web is much preferred

WHERE AND HOW

Because the WWW Wizards group met under the sponsorship of the SLAC Library, and because the maintenance and delivery of the Web HEP Pre-Prints interface was spearheaded by Louise Addis, SLAC Associate Head Librarian, the Library has been able to provide the Archives with documentation of the Wizards' activity and of the HEP Pre-Prints site. Some documentation has also been preserved by individual WWW Wizards, and all of the computer backup tapes from the founding era of 1991 - 1994 still exist (although not on the current main-frame platform). Individual Wizards have conveyed to the Archives paper documentation that they had retained for "historical" purposes in their own files, including paper copies of e-mail messages that were exchanged about setting up the first server, getting it to work properly, and negotiating changes and additions to the SLAC home page. Wizards and members of the Web Coordinating Committee have also provided hard copies of written papers and presentations on the WWW and SPIRES-HEP, and, those papers usually have included examples of then-current SLAC Web pages

APPRAISAL

One thing IS certain at this point, however, and that is that SLAC World Wide Web pages are official records. They fit the statutory definition of records [16] because they are documentary materials, in machine readable format, made or received in the transaction of public business and appropriate for preservation "as evidence of the organization, functions, policies, decisions, procedures, operations, or other activities of SLAC." [17] As official records, SLAC's early World-Wide Web pages, which were created by elements of the Research Division, need to be appraised and

In association with

[redacted] **Luncheon Talk**

Home
Agenda
Tutorial
Travel
Registration
Attendees
Discussion

Bringing the Web to America

Speaker: Paul F. Kunz

Organization: Stanford Linear Accelerator Center (SLAC)

Abstract: On 12 December 1991, Paul Kunz installed the first Web server outside of Europe at SLAC. Today, if you do not have access to the Web you are considered disadvantaged.

Before it made sense for Tim Berners-Lee to invent the Web at CERN, there had to a number of ingredients in place. Paul will present a history of how these ingredients were developed and the role the academic research community had in producing them. In particular, he will address the role played by big science including high energy physics in giving us the World Wide Web and the Internet as we know it today.

Bio: Paul Kunz received his Ph. D. from Princeton University in 1968 and first went to CERN that year to do an experiment as a member of the Saclay group. In 1971 he went on to Michigan State and worked on one of the first experiments at Fermilab. He joined SLAC in 1974 where he has been ever since.

In the late 1970s, Dr. Kunz invented the 168/E emulators and the concept of event processing via parallel processor farms. Dr. Kunz has been a pioneer amongst physics colleagues in adopting new computer technologies. Examples include his move to UNIX and object oriented programming over ten years ago. Lately, he has been giving a course, "C++ for Particle Physicists," a course that has now been held fifty-one times all over the world for more than 1700 students.

Slides: PDF

Video: 62 5.0 (Video Player Information)

Chair: Bebo White

Bringing the World Wide Web to America

Paul F. Kunz

<Paul_Kunz@SLAC.Stanford.edu>

Stanford Linear Accelerator Center

On Dec. 12, 1991 the first Web server in America was installed, by me, on SLAC's IBM mainframe.

Today, if you don't have access to the Web, you are probably considered disadvantaged

- How did it happen that research in High Energy Physics invented the Web?
- What role did academic research, and HEP in particular, play in developing the ingredients needed for this?

This is the story on how it happened from my perspective

UNIX and the ARPANet

They grew up together in the '70s and early '80s

- Both were computer science research funded by Advanced Projects Research Agency of the DoD
- when academics had needs, they invented new protocols
  - e-mail
  - ftp
  - telnet
- UNIX easily allowed this, or was it that UNIX was made to allow it?
- the Open Source culture developed
- during this time, HEP thought UNIX was a four letter word

Networks of broader scope

- BITNET: network of academic computing centers using IBM protocols
  - started May 1981 between CUNY and Yale
  - open to entire university including users non-IBM computers (e.g. DEC VMS)
  - SLAC joined in June 1983, 25th site and the rest of HEP followed
  - no central funding agency
  - self governing by volunteers
  - but limited capability compared to ARPANET or DECNet

- EARN: BITNET extension to Europe
  - funded by IBM
  - had significant impact on international networking and HEP played important role

The PTTs were beaten

Slowly, the PTT's grip on networking was lost...

- US to Italy link got established in February 1984
- Italy to Israel link got established next. Theoretical HEP physicist Haim Harari played crucial role
- links spread to Spain and southern France
- the Swiss, following the German lead, refused to allow CERN to connect
- finally, the German PTT allows temporary links, until their X.25 infrastructure was in place early in '85
- for many months Northern and Southern Europe are connected only by separate links to the States
- the Swiss allows link between CERN and Italy
- the US HEP DECNet links to CERN as well the Italian HEP DECNet; the Swiss PTT allows it after much discussion
- the ARPANET links Scandinavia to the States
- by the time PTTs had X.25 in place, traffic on temporary networks was too high for it to handle

The Internet becomes dominant

The National Science Foundation made it happen…

- Al Gore, Jr. *did* play a role
- Congress mandated that the NSF build supercomputer centers for American researchers
- NSF solicited proposals for region networks
  - use ARPANET technology
  - self governing
  - must be available to the entire university research community, like BITNET
  - NSF supplied backbone as network of networks, thus the word *internet*
- NSFNet saturates and was upgraded to 1.544Mbps
- still academic network, except for companies with joint research projects with academics
- HP asks NSF if it can support customers via the net
- NSF stops funding of backbone, internet becomes self supporting commercial network
- The use of backbone remains free and ARPANET culture persists

The Web is Invented

The ingredients are in place…

- international, free and open internet
- excellent application development environment geared to desktop publishing

Tim Berners-Lee, working at CERN, was trying to solve a problem…

- easy document creation and viewing
- document cross reference via hyper links
- world wide document distribution

What were his solutions…

- buy a NeXT computer (Sept. 1990)
- write a hypertext application
- extend hypertext to remote computers by adding a new protocol to the internet: HTTP

He had a demonstrable application by Christmas

My first view of the Web

What Tim and I did in his office…

- Tim demonstrates Web browser on his NeXT which was the first web server as well
- *Oh hum, I didn't really care about documentation systems*
- Tim demonstrates the Web interface to CERN's IBM mainframe by sending a query to the help system database
- *Now that is interesting, and I immediately had an application in mind. but will it work across the Internet?*
- We upload my NeXT at SLAC with browser software and ran it there with windows sent back to CERN.
- *Worked well, remarkably well*

I told Tim I was going to put SLAC's SPIRES database on the Web, as soon as I got home.

What happened after that

The SPIRES-Web application at SLAC was the killer app for the Web…

- The SLAC SPIRES database had 200,000 records that physicists really wanted to search; 1000s of users in 40 countries
- Before the Web, it was hard to access and had an awkward command line interface
- The Web interface was easy to use and could be accessed from any computer on the internet
- enthusiasm for the Web within HEP grew enormously, even at CERN, because of SLAC server
- growing use of the Web by HEP was seen by other academic centers such as NCSA where Marc Andressen developed the Mosaic browser

first monday
PEER-REVIEWED JOURNAL ON THE INTERNET

FM Interviews

Louise Addis

I was so excited that I almost dropped the book.

I was reading *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*, by Tim Berners-Lee. The paragraph that caused such a reaction described how Paul Kunz, a California physicist, had returned from a visit to CERN (Conseil Européen pour la Recherche Nucléaire; now officially named the Organisation européenne pour la recherche nucléaire or European Organization for Nuclear Research) in Switzerland where he had seen an interesting technology developed by Berners-Lee. Kunz came home to the Stanford Linear Accelerator Center (SLAC) and told Louise Addis, a SLAC librarian, about this new invention, the World Wide Web.

Berners-Lee states that Louise saw the Web "as a godsend - a way to make SLAC's substantial internal catalogue of online documents available to physicists worldwide." [1] He further states that "under Louise's encouragement SLAC started the first Web server outside of Europe." [2]

The first Web server created in the United States - set up anywhere outside of its birthplace at CERN - was established at the encouragement of a librarian!

As someone working in both the Web world and the library community, I decided then that I had to meet, or at least talk to, Louise Addis. I knew that I could learn something from the librarian who saw the potential in this revolutionary technology before the majority of the world had even heard of the Internet.

I tracked Louise down using the Web (of course!) and e-mail. When we first spoke on the phone, I told her how excited I was to meet the person who was responsible for setting up the first Web server in the U.S. Louise laughed and disclaimed total responsibility for bringing the Web to America. She told me about her colleagues, Paul Kunz and Tony Johnson and others, who were critical to the launch of the first Web server at SLAC. At one point in the conversation, she noted that the history of the Web technology is rather like *Rashomon*, Akira Kurosawa's 1951 movie in which the same story is told from widely diverging perspectives. It appears to be true that there are a number of stories about how the Web came to America. This is one of them. - Melissa Henderson.

■ ———————————

First Monday (FM): Could you first tell me a bit about your academic background? Where

did you study for your undergraduate degree?

Louise Addis (LA): I studied International Relations at Stanford University. I always had an interest in the sciences, however.

FM: And from Stanford?

LA: I initially went to work for a publisher of the "Dick and Jane series" in the claims department and as a sales correspondent. I then moved to the Stanford Research Institute Radio Science Lab where part of my job was handling classified documents. I was thinking about going to library school when a colleague encouraged me to apply for a position in the soon-to-be-opened library for Project M.

FM: Project M?

LA: "Project Monster" now known as the Stanford Linear Accelerator Center. At that time, it was still on the drawing board and had no funding. We started our library in a warehouse on the Stanford campus. It was the hottest place on campus; they ran water on roof to make it more bearable!

FM: How long were you at SLAC?

LA: From before the beginning! It was very exciting to build a library. I started as the assistant to the librarian in 1962 and retired in 1994 as Associate Head Librarian. Since retirement, I have continued to work off and on for the SPIRES-HEP database project at SLAC and most recently on Y2K conversion.

FM: What were you working on in your early years at SLAC?

LA: From about 1969 on, I was working with the SPIRES-HEP database of particle physics documents. SPIRES (initially, Stanford Physics Information Retrieval System, then Stanford Public Information Retrieval System) database management system was the product of brilliant programmers at Stanford. SLAC was a guinea pig for SPIRES, which was originally designed for an IBM mainframe environment. SLAC sponsored development of the Unix version, which is now in use.

FM: Did anything in your education or background prepare you for the higher level computing work you were doing with SPIRES-HEP or the Web?

LA: I got into programming and database development by picking up information on my own. If I needed to know something, I asked someone to show me how to do a particular task. Then I went back to the Library and tried it on my own.

When the SLAC Library was started, the Stanford University computing center was a little building with keypunch machines and a line printer. At the Stanford Research Institute, I had seen an "advanced serials handling system" which meant that each serial had its own punch card. This was cutting edge! So when we built the Library at SLAC, we put the serials on punch cards. I remember walking across campus with my stack of punch cards. So

advanced!

FM: What was so important about SPIRES-HEP database of particle physics documents?

LA: There weren't any library database management tools available at the time and SLAC needed something to meet the unique needs of particle physics researchers. Research papers produced by the particle physics community often have hundreds of authors. The record is 1,200 authors! We were committed to listing all authors or researchers and needed a tool that would accommodate us.

Research papers produced by the particle physics community often have hundreds of authors. The record is 1,200 authors.

The SPIRES-HEP database covers the full literature of particle physics and is a collaborative project between a number of particle physics labs, primarily SLAC and DESY in Hamburg. It's so very flexible, so it's constantly developing. We've always been able to easily manipulate the database to do whatever we needed to do.

Of course, there are two sides to any database - the technology side and the content side. The content side is really what ultimately determines the value of any database. Bob Gex, then chief librarian at SLAC, was responsible for the content side, along with our long-time and remarkable preprint librarian, Rita Taylor. And from the beginning, there have been major contributions of content from our DESY partners.

FM: When did the Web first arrive at SLAC?

LA: On December 12, 1991, the first Web server in the U.S. was established at SLAC to help improve access to the SPIRES-HEP database. For a number of years, SLAC had been providing remote access to the SPIRES-HEP database. In fact, by the time we launched our first Web server, the SPIRES-HEP database had nearly 5,000 registered users in 40 countries.

George Crane had developed a remote interface, initially using BITNET. Researchers could send a search query to SPIRES-HEP via an interactive messaging tool, which was sort of like the instant messaging tools of today. Or they could request information via e-mail. When Internet access came along, the interactive messaging was no longer available; researchers had to use e-mail. Researchers were less satisfied with the e-mail interface; they liked the instant response system better.

On December 12, 1991, the first Web server in the U.S. was established at SLAC to help improve access to the SPIRES-HEP database.

FM: Was this a motivating issue when you first thought of setting up a Web server?

LA: Well, the real motivating issue was a project that we been working on earlier. In 1989-1990, I was helping set up a library at the Superconducting Supercollider (SSC) project in Texas. Pat Kreitz, the SCC librarian, wanted to use the SPIRES-HEP database and wanted to present it to the SSC community in a seamless way within a graphical user interface. SSC had funded the development of an X-Windows interface to do this, but as soon as I saw the Web in action, I knew that it was a fast, cheap solution.

I can still remember the day Paul Kunz appeared in my office after his trip to CERN. He showed me the Web and we started moving right away. I scurried around and got some accounts that were needed. Paul and Terry Hung set up the server on our mainframe. George Crane was able to easily extend remote SPIRES to talk to the new interface. And I was able to make the SPIRES-HEP database write HTML.

I've always felt deeply indebted to Paul for spotting the potential of the Web for our situation and for helping develop the first Web server at SLAC, which proved that potential.

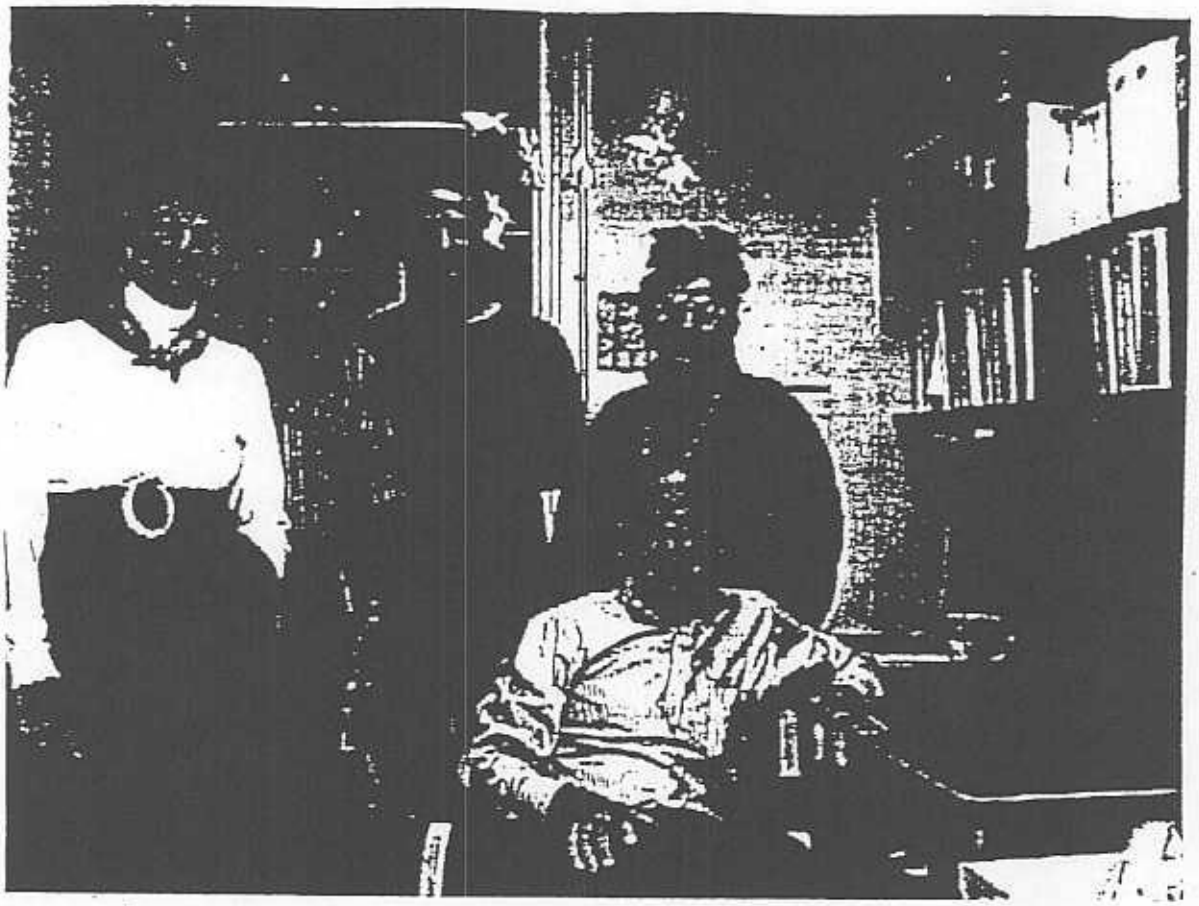
FM: This sounds like it was really a team effort.

LA: It was! While I was working on this project - and for most of my other projects at SLAC - I was constantly seeking input and advice from colleagues. There was almost always someone around who understood the most arcane, technical issues and was willing to help or to give advice. Sometimes conflicting advice!

FM: Did this teamwork model work?

LA: Yes, it was the only way any of these projects could continue. None of us got raises or payment for this work. We did some of it on our own time. We knew that there was value in the work - and it was fun! (I hope my colleagues would agree.)

By February 1992, I had twisted enough arms to start an ad hoc Web development and support group, the WWWizards. The initial group was made up of Mark Barnett, George Crane, Tony Johnson, Joan Winters, Bebo White, all of whom brought important skills to the project.



The original WWW Wizards at SLAC visit Paul Kunz. Left to right: Louise Addis, George Crane, Tony Johnson, Joan Winters, Paul Kunz, and a NeXt computer (missing: Bebo White and Mark Barnett). Photograph courtesy of the Stanford Linear Accelerator Center, Archives & History Office, Stanford, Calif.

Again, I have to mention that in the beginning this was not part of our regular jobs. Luckily, the SLAC Computing Center was quite supportive. Les Cottrell, the assistant director of computing, could certainly have discouraged the computing staff from participating. Instead, they were able to work on the WWWizards team and to support this unsupported software. And, they helped run the server on their computer!

FM: Did you have any idea then of what the Web would become?

LA: No! I was just hoping it would survive at SLAC. For a long time, the Web suffered from the stigma of being unsupported software.

My goal was simply to provide better community access to the particle physics literature via SPIRES-HEP.

FM: Was SLAC a good environment for developing new tools, such as SPIRES-HEP or the Web?

LA: Yes! We always had the ability to respond quickly and be flexible. This was particularly instrumental in the development of the Web. When Paul Kunz came back from Europe and said, "Let's do this," the Library could move ahead.

And as soon as we were on the Web, we were able to start linking the records in our database to the TeX source at Los Alamos National Laboratory (LANL) where, in August 1991, Paul Ginsparg had started the first e-print server. Paul, by the way, almost single-handedly made another kind of revolution in the way scientific literature is handled ... but that's a whole other story.

As soon as we were on the Web, we were able to start linking the records in our database to the TeX source at Los Alamos National Laboratory.

FM: How were you able to just jump right into this new technology?

LA: First of all, our faculty were used to using unsupported software. We didn't have manuals or training classes, but we did have all kinds of skills and knowledge available for something interesting. The Web was done entirely with volunteer labor at first.

Generally speaking, the Lab encourages experimentation, sometimes officially and sometimes not. The Web server project was something that the SLAC Library could jump right into because of our track record with SPIRES-HEP and our environment. All that had gone before made it easier to get management approval. Plus, the chief librarian, Bob Gex, was terribly supportive. I was lucky in that regard.

MH: Do you feel that development of resources such as SPIRES-HEP has gone faster with the implementation of Web technology?

LA: Oh yes. In fact, this was one of the things that really sold me on the web - it was easy! The underlying Web server and Web browser programming is more technically complex, but creating the resources is much easier. For example, the x-windows project relied on programmers who could manipulate a more complex system. On the other hand, the Web allows for easy development of resources; you can make the page look exactly the way you want without deep programming.

FM: The particle physics world seems tailor-made for the World Wide Web - or should I say that the Web was tailor-made for the physics world?

LA: Absolutely! There is the issue of the need for rapid communication in a field that has a slow publication schedule. Additionally, the particle physics community is made up of very large groups of scientists all working on the same project, but from remote, diverse sites.

Also, these folks are used to trying something new. When we notified our registered users about how to get a free browser from Tim at CERN, many did right away. They wanted to get to our database, so they were really motivated to get that browser and get on the Web. This helped spread use of the Web in the particle physics community and it also helped people learn about the archives at LANL.

FM: What about the democratizing of the publishing process?

LA: Gray literature is a very important part of the particle physics world. One of the great things about the Web is that it democratized access to these resources. Before electronic communication people who were further away from population centers or technology centers were really behind. They had no way to find out quickly about research.

Gray literature is a very important part of the particle physics world. One of the great things about the Web is that it democratized access to these resources.

Before machine-readable, full-text e-prints were available, paper preprints were mailed by the author or the institution only to major institutions. And researchers at smaller institutions didn't receive the preprints. The only way to find out about these articles was through a preprint list such as the one SLAC published weekly. The researcher then had to request the actual article through the mail. This was too slow.

FM: And the interactive nature of the Web?

LA: In large particle physics collaborations, the experiment may take years from conception through publication. And there may be hundreds of physicists around the world working on these projects. They are all producing materials that need input from others.

Many papers are almost dialogues. Papers go through revisions. Discussion is a big part of the process. Physicists really needed that ability to interact quickly.

The Web was a revolution!

FM: Back to your career development, did you feel more affiliated with the library community or the particle physics community?

LA: Much more affiliated with the particle physics community. Our issues were so different from what others in most libraries were experiencing. In a special library, you have to know your community and listen to them. They're the people that need the library.

FM: Doesn't this apply to all libraries?

LA: More so to special libraries, which are vulnerable and can be more easily closed.

FM: What do you think is the response to this vulnerability?

LA: We need to apply other skills and expertise in order to allow the library to succeed. You almost never have all the skills you need. You have to find them elsewhere. For example, at the SLAC Library we wanted to make PostScript files and graphics available via the Web, rather than just linking to the TeX sources. So, when Tony Johnson developed Midas, the first GUI browser that was bug-free enough for us to use, he added capability for reading and displaying postscript. This was the kind of development done on someone's personal initiative. This is skill and interest that you cannot buy. Also, Tony had the breadth of knowledge of a particle physicist's needs and a willingness to tinker.

FM: So, we're back to an environment of experimentation ...

LA: Yes, a lot of environments don't make it easy to go outside or take risks and that's what is required to be visionary. You can't pay attention to things like job descriptions.

FM: Is there anything that concerns you about the development of the Web?

LA: I'm concerned about commercialization. The Web has been a democratizing place. Now, we're moving toward large entities controlling most content - or the content that most people see. You can get to other sites, but these services shape your experience. AOL tells you the story of the day and offers you its preferred links.

On the one hand, anyone with a little curiosity can find things they'd never dream of having access to. But portals do control what a lot of us see - or have time to see! But this has always been true of the media; the Web is just another instance of this.

FM: What about the issue of the digital divide? Or does this problem not exist in the academic community?

LA: Actually, this has been a continuing problem for some of the remote or less prosperous regions of the world. In the U.S., a small physics department in a small school may have a slow line or slow equipment.

When I was at SLAC, we always created pages in formats that were much simpler than what was technically possible. Harv (Hrovje) Galic, the HEP database manager at that time, originated much of our early Web interface. Harv was adamant about trying to have pages that could be easily read in some of the less prosperous parts of the particle physics community. Harv was from Zagreb and had a real awareness of the constraints faced by many of his colleagues.

In fact, we were occasionally ridiculed because our pages didn't have graphics; but this was done purposely. The main reason was because many of our users didn't have the capability to see graphics - or graphics were an impediment on a slow system.

The digital divide is real. It's less of a problem in academic communities where the issue is

fast or slow connection. Or good network support and stability. In society as a whole, the issue is having any access at all.

FM: Finally, do you think many folks will be surprised to learn that the first Web server in the U.S. was installed to help support a library project?

LA: Not at all! To me, it's really significant that our own early success story on the Web was driven by the need to bring a large body of carefully organized bibliographic records, SPIRES-HEP, to its worldwide audience!

The Web has certainly proven to be a shiny but challenging new tool for librarians to use in their crucial role as collectors, organizers, preservers, and presenters of information. Librarians will continue to play a critical role in helping to organize and provide access to information. Perhaps an even larger and more important role now that we're in the "wild, wild west" era of Web content development.

Notes

1. Tim Berners-Lee, 1999. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor.* New York: Harper SanFranciso, p. 45.

2. Berners-Lee, p. 46.

Editorial history

Paper received 13 April 2000; accepted for publication 17 April 2000.

Contents Index

Copyright ©2000, First Monday

FM Interviews: Louise Addis by Melissa Henderson
First Monday, volume 5, number 5 (May 2000),
URL: http://firstmonday.org/issues/issue5_5/addis/index.html

UNITED STATES

DEPARTMENT OF ENERGY

PETITION FOR WAIVER OF DOMESTIC AND FOREIGN RIGHTS TO AN IDENTIFIED INVENTION

DOE WAIVER NO._____
(Supplied by DOE)

DOE Case NO. *S-73,985 (RL-11508)*

Title of Contract/Contract No.:[1]

(A) *Contract Between the Regents of the University of California and the United States of America, to operate Lawrence Berkeley Laboratory ("LBL") from October 1987 to September 1992, DE-AC03-76F00098. (Note: This contract is identified on the Invention Disclosure by its precursor designation: W-7405-ENG-48).*

(B) *Memorandum Purchase Order from Superconducting Super Collider Laboratory[2] ("SSCL") to LBL, GFAA-SSG-4302, Award # SSC91W09978, entitled "Remote Database Object Project HEP".*

Title of Subject Invention:

REMOTE OBJECTS, REMOTE DATABASE OBJECT

Petitioner's Identification No.: _____

Petitioner's Name/Address:

Allan M. Konrad, P.O. Box 4023, Berkeley, CA 94704.

Petitioner does hereby petition the Secretary of the United States Department of Energy for waiver of domestic and foreign rights of the United States of America to the Subject Invention which was made in the performance of work under the above-identified contract. It is understood that any waiver of rights shall be subject to the government license, march-in rights, and preference for U.S. industry set forth in 35 U.S.C. 202, 203, and 204, regardless of whether the Petitioner is a small business or nonprofit organization.

---

[1] *The Subject Invention was not funded directly by the Department of Energy, but was funded by two sources which can be indirectly traced to Department of Energy funding, labelled herein as (A) and (B). Where an answer to a question is different for each source of funding, the question is answered in a part (A) and a part (B), corresponding to the two sources of funding.*

[2] *The Superconducting Super Collider Laboratory is a DOE National Laboratory.*

Is Petitioner a small business as defined by 1-1.701 FPR? _____

Yes.[3]

In support of this petition, answers to the following questions are submitted:

1. Identify the specific rights which the Petitioner desires to obtain in Subject Invention. (Field of use, geographic area, exclusivity, term, etc.)

Petitioner desires to obtain the right to patent the Subject Invention which was developed by him at LBL using overhead funding from the DOE. By this petition for wavier, Petitioner petitions for the right to apply for U.S. and foreign patents covering the Subject Invention, prosecute such patent applications, have patents issued from such applications in due course with the Petitioner as the sole assignee.

Implied in such a waiver is a waiver of any objections to the making, using, or selling of the Subject Invention by the Petitioner or his assigns, and objections to the assertion of patent rights against third parties, subject of course to the rights reserved by the U.S. Government referred to above.

The field of use of these rights would be determined by the eventual uses of the Subject Invention, which cannot be fully contemplated at this time, however Petitioner envisions the use of the Subject Invention is a wide variety of remote information services applications. Since the Subject Invention implies geographically disperse computing, delineation of a particular geographic area is not meaningful in this instance. As for exclusivity, Petitioner plans for obtain exclusivity through patent grants. The term of the rights would therefore also be determined by the patent laws.

2. Give a brief description of the Subject Invention. Specify any known or potential weapons fields of use or weapons applications or naval nuclear propulsion use or application of the Subject Invention. To your knowledge, are any fields of use or applications of the invention classified or controlled or sensitive under Section 148 of the Atomic Energy Act (dealing with unclassified, sensitive nuclear information)?

The Subject Invention is summarized herein, and a fuller description of the invention is found in the attached "Disclosure and Record of Invention Form" previously submitted by the Petitioner in accordance with DOE and LBL regulations. The Subject Invention relates to a process and apparatus for establishing connectivity between a human user at a local host machine

---

[3] *1-1.701 FPR has been superceded by 48 CFR §19.000 et seq. (esp. §19.001 and §19.102). According to the definitions of small businesses therein, Petitioner qualifies as a small business, since Petitioner has annual sales and employee counts of less than the limit for small businesses in the field of either SIC 7371 (computer programming services) or SIC 7372 (pre-packaged software), and Petitioner is not dominant in his field.*

and a remote host machine to provide information services or other utilities from the remote host to the local host.

The Subject Invention has no known weapon-specific or naval-specific applications, although it might be used as a tool by computer users doing weapons or naval work. An apt analogy is to desktop computers, which are not defense specific, but can be used in support of defense applications and projects.

Petitioner is not aware of any applications of the Subject Invention which are classified, controlled or sensitive under §148 of the Atomic Energy Act (as codified at 42 USC §2168). Since the dissemination and patenting of the Subject Invention would not have any effect on the likelihood of illegal production of nuclear weapons or theft, diversion, or sabotage of nuclear materials, equipment, or facilities, §148 of the Atomic Energy Act is inapplicable to this petition.

3. Give a brief description of the scope of work of the above contract. Specifically describe the source of funding, including the name of the specific project under which the Subject Invention was made and the name of the cognizant DOE program director (Laboratory operators furnish B&R code, if available).

(A) The Subject Invention, Remote Objects, was developed by Petitioner in the course of work within the Information and Computing Sciences Division ("ICSD"), which is a support division of LBL. The division is funded primarily by overhead charged to funded research groups and provides services such as computer maintenance and support to the funded research groups throughout LBL. The scope of work relating to the Subject Invention was defined informally as a small scale prototype development project with the possibility that an application might be developed in support of Laboratory requirements. Consequently, there is no identified cognizant DOE program director. Petitioner reported progress and status information relating to this project to Dr. Stu Loken, Division Director, Information and Computing Sciences Division, at LBL.

(B) The scope of the work for SSCL was to provide a working prototype of the Remote Database Object to SSCL configured to support their usage of the High Energy Physics database at the Stanford Linear Accelerator Center. This was not research funding, but rather routine "Work For Others" (WFO) for technical services for the SSCL Library. Consequently, there is no cognizant DOE Program Director. No B&R code was assigned to this Memorandum, since it is WFO.

4. What is the dollar amount and period of performance of this contract and the amount specifically expended in making the Subject Invention? Is there further Governmental funding anticipated in the development of the invention? Describe any continuing Government funding of the development of the invention (including investigation of materials or processes for use therewith), from whatever Government source whether direct or indirect, and, to the extent known by petitioner, any anticipated future Government funding to further develop the invention.

(A) No amount was budgeted or attributable to develop the Subject Invention. The Subject Invention was a byproduct of support assignments in ICSD and work in conjunction with other LBL assignments. The period of performance was indefinite since this was an informal project.

Petitioner estimates that LBL overhead funding allowed for least 2 person-years of effort to be applied to the development of the Subject Invention. Additionally, summer student and part time undergraduate students assisted as part of the ICSD's outreach program.

(B) An amount of $48,000 was expended on a Purchase Order Memorandum by SSCL, which covered a period of performance from August 5, 1991 to September 30, 1991, and extended to December 31, 1991.

No further Government funding for the development of the Subject Invention is anticipated, and no continuing funding exists. Primarily due to present budgetary constraints at LBL ICSD for funding Remote Objects, and prior expression of disinterest of DOE in funding Remote Objects explicitly, Petitioner believes that neither ICSD, LBL, the University of California, nor SSCL intends to pursue any further development of the Subject Invention.

SSCL has declined to make any commitment for further funding for development of the Subject Invention, or to invite a proposal for funding. Furthermore, even if such interest arose, the ICSD management has indicated that such funding would not be accepted without review.

5. Briefly describe the Petitioner's technical competence in the field of technology in which the Subject Invention lies or to which it relates in terms of prior experience, know-how and patent position. (Attach exhibits to substantiate your technical competence, e.g., patents, technical publications, etc. If these are voluminous a representative sample is sufficient.)

Petitioner is a Staff Scientist II (Computer Science) in the Information and Computing Sciences Division at LBL. He has worked on assignments in information science for fifteen years at LBL, and has served as a Principal Investigator on the Joint Laboratory for Information Research and Technology Project for the past two years. Related assignments have included technical support for network-based remote tele-commuting, strategic information planning, and support for network-based access to on-line library catalogs and other bibliographic resources.

Petitioner's experience includes numerous other assignments in database and digital network applications, including design, implementation, and support. Petitioner's other experience includes responsibility for regulatory and compliance assignments relating to DOE Automated Data Processing Equipment (ADPE), management of hardware and software maintenance contracts, hardware procurement management, software licensing, administration of computer related software license agreements and nondisclosure agreements.

Consequently, Petitioner has working knowledge and familiarity with the primary technical fields which encompass the Subject Invention, Remote Objects: information technology, database applications, digital computer network deployment and network policy.

Attached are representative exhibits to substantiate Petitioner's technical competence, in the form of Field Task Proposals which were prepared by Petitioner for submission to DOE and the Joint Laboratory for Information Research and Technology in the course of seeking further

funding for development of the Subject Invention.[4] *However, the best evidence of Petitioner's competence in the field is performance. Petitioner has sucessfully assembled and coordinated the efforts of a multi-institutional team which did produce a working prototype of a Remote Object.*

In addition to Petitioner's wide experience in the field, he has a Master of Science degree in Management of Research and Development.

6. Briefly describe the Petitioner's established non-governmental commercial position in the field of technology in which the Subject Invention lies or to which it relates. (Discuss in terms of selling goods or providing services in such field outside of sales to the U.S. Government. Attach exhibits to substantiate your commercial position, e.g., sales brochures, etc. If these are voluminous a representative sample is sufficient.)

Petitioner has delayed the establishment of a non-governmental commercial position pending the grant of a waiver to the Subject Invention in response to this petition. The commercial position to be developed is that of a software supplier and consultant to commercial and noncommercial remote computing service providers. Petitioner expects that much of the commercialization will be to non-governmental entities such as publishers and information providers.

Petitioner has established considerable personal and professional contacts in the commercial remote computing, publishing, and information services industries. At the appropriate stage in the marketing of the Subject Invention, once intellectual property rights are secured by the filing of patent applications, Petitioner will call on these resources in order to promote Remote Objects vigorously.

7. What is the financial and technological investment that has been made by the Petitioner with regard to the field of technology in which the invention lies or to which it relates and with regard to the making and developing of the Subject Invention?

Petitioner has invested $10,000 of his own money to secure the rights to the Subject Invention and to develop a business plan for a business which will provide Remote Objects software and consulting services.

Petitioner's technological investment includes nearly 15 years of career work in the field of computer science, database design, implementation, and support, and network deployment and policy.

8. To what extent will the Petitioner make a substantial investment of financial resources or technology which will directly assist the further development and promote the commercial utilization of the Subject Invention? Summarize

---

[4] *These proposals are no longer pending, as the DOE has declined to fund either Field Task Proposal.*

petitioner's plans, intentions and ability in developing the Subject Invention to a point of commercialization. This should include a description of the technical steps required, funds necessary therefor and associated time periods necessary to develop the invention to the point of readiness for commercialization, and a statement by petitioner that either it or its present or prospective licensee will expend the necessary capital and resources.

Petitioner is prepared to invest substantially in Remote Object technology because he has strong confidence that this technology is a better way of accessing remote information and service resources than any other. It's extreme ease-of-use and maintenance will consequently appeal to a wide consumer market.

Petitioner is committed to Remote Objects, and as such, is prepared to use personal resources to fund the development of Remote Objects, however, Petitioner expects that with the granting of a waiver and the subsequent patent protection of Remote Objects, Petitioner will be able to fund further development of customized versions of Remote Objects from the sales of Petitioner's services to various customers who will need custom software and consulting.

Initially, Petitioner expects to commercialize the Subject Invention actively, instead of relying on licensees, and Petitioner plans to expend the necessary capital and resources and put in the necessary time and effort to fully commercialize the Subject Invention.

<u>Plans and Intentions</u>: Petitioner is prepared to devote personal savings and equity toward establishment of an early round of "flagship" customers, the revenue from which will enable further promotion of Remote Objects.

These flagship customers will be selected partly on their willingness to permit exposure of their use of Remote Objects to the public, and to enter into co-development agreements for implementations of Remote Objects on specific platforms. For example, a commercial database vendor may be a Remote Object customer for access to their particular database.

<u>Financial Ability</u>: Petitioner has the financial ability to perform the tasks above, as the costs are minimal since Petitioner's business strategy requires no capital outlay for physical (computing) plant, since the Subject Invention will be developed for the most part through Petitioner's consulting and using the consulting client's physical plant. Since the Subject Invention will be commercialized by software customization and consulting, the majority of the necessary investment to develop the Subject Invention will be time investment rather than monetary investment.

<u>Entrepreneurial Ability</u>: Petitioner has a demonstrated entrepreneurial ability in attracting internal and external funding to LBL for Remote Objects and related projects over the past three years, thus is in a unique position to promote Remote Objects to commercial customers.

The presentation of this petition is a testament to Petitioner's entrepreneurial initiative. As mentioned above, Petitioner's demonstrated ability to form a team and develop a prototype is further evidence of Petitioner's entrepreneurial ability.

<u>Management Ability</u>: Petitioner has management experience in the Office of Computing Resources at LBL, managing hardware and software maintenance contracts and software licensing. Petitioner also holds graduate degree in Management of Research and Development.

Petitioner's experience, in management of research and development, is sufficient that he appreciates the importance of attracting a different kind of managerial talent than Petitioner's, namely a professional management team with financial contacts and commercial experience. It is Petitioner's intent to recruit such a team at the appropriate stage in the product's development.

Technical Steps Needed to Commercialize the Subject Invention:

1. Identification of interested customers

2. Contracting with interested customers for a customized version of Remote Objects software and consulting

3. Tailoring Remote Object product to customer's network

4. Tailoring components to customer's operating environment

5. Tailoring Remote Object Client components to customer's target database or target service

6. Establishment of system monitoring and maintenance program tailored to customer's environment

7. User beta testing with customer's internal staff and beta user.

Funds necessary to perform the required technical steps:

As is standard in the field of software customization, the costs of technical steps 2-7 will be borne largely by the customers. The costs of technical step 1 has for the most part been expended by Petitioner in identifying likely customers. The initial funds will be provided by Petitioners own investment, and are expected to cover the following costs: developing and executing a software agreement with a prospective customer including legal fees and patent application costs ($5,000), establishing a business entity and office location ($9,400) and consultation for tailoring and installation of one Remote Object ($15,000).

Associated time period necessary to develop Subject Invention:

Identification of at least one customer: 3 mos.

Development of an operational, custom Remote Object: 3-6 mos.

Petitioner will expend the necessary capital and resources to bring Remote Objects to commercialization.

9. Why will the grant of the above-requested waiver more effectively promote the development and the commercial utilization of the Subject Invention?

Currently the DOE and LBL are not interested in funding further development of the Subject Invention, nor is DOE or LBL interested in pursuing a patent application on the Subject Invention. All Remote Object work has been discontinued by ICSD. Therefore, if the above-requested waiver is not granted, NO development or commercial utilization of the Subject Invention will occur.

After receiving the DOE waiver, Petitioner plans to pursue a strategy of contracting with a small number of prominent flagship information providers to produce customized versions of Remote Objects to demonstrate the value of Remote Objects and provide public exposure to the technology.

Specifically identify any potential licensees or joint ventures interested in further development of the invention to commercialization. If the invention will be commercialized by a licensee, summarize the licensee's plans and intentions to develop the invention to a point of commercialization.

Petitioner cannot solicit "interest in further development of the invention to commercialization" without prejudicing patent rights. Consequently, Petitioner has refrained from disclosing the invention until proprietary rights can be secured. Such rights will be promptly secured once the petition is granted. However, the fact that Petitioner has filed this petition indicates his willingness to bring the product to market, and since the DOE and LBL do not plan to fund further development of the Subject Invention, the only means for the Subject Invention to reach the public is for the DOE to grant a waiver to Petitioner, thereby allowing Petitioner to obtain patent protection to protect his investment, an investment which must be made in order that the Subject Invention can be commercialized.

10. What will be the effect on competition and market concentration if the above-requested waiver is granted? Describe any competitive technologies or other factors which would reduce any anticompetitive efforts of granting the waiver. Would the acquisition of the waiver rights requested be likely to place the Petitioner in a preferred or dominant position in this field? Give reasons for your conclusions.

The industries which would be affected by the granting of the waiver would be the publishing and digital information service industries. Since the Subject Invention, Remote Objects, is but one of many means of remote access, no anticompetitive effects of a waiver are anticipated. For example, many software programs exist in the market for remote access, such as terminal emulators and network access protocols. Petitioner is not in a position, even if the waiver is granted, to reduce the competitive efforts of the many large companies which provide remote connection systems.

The granting of a waiver would put the Petitioner in a preferred position only with respect to patent rights to Remote Objects obtained by the Petitioner after a waiver is granted. Such rights would be no more than any other patent owner, but would provide the necessary protection to the Petitioner to justify the investments needed to bring the product to market.

11. Under what other contracts has the Petitioner worked or what other contracts has the Petitioner had with any Branch or Agency of the U.S. Government which include all or a part of the scope of work covered by this contract or which deal with the field of technology in which the Subject Invention lies or to which it relates.

None.

12. If the Government has already prepared a patent application, is petitioner willing to reimburse the Government for its expended costs, if any, in the preparation, filing, and prosecution of patent application(s) on the Subject Invention?

The Government has not prepared any patent applications on the Subject Invention, and indeed the DOE has indicated an unwillingness to pursue patent protection for the Subject Invention. Therefore, this question is not relevant for this particular petition.

13. If applicable, provide a statement of reasons why the petition was not timely filed or why a request for an extension of time to file the petition was not filed in a timely manner.

This petition is timely filed. While a timely filed "Disclosure and Record of Invention Form" was filed by Petitioner in September, 1990, only recently has both the DOE and LBL determined that no further Government interest in the Subject Invention exists.

Petitioner did receive a letter from the Livermore office of the DOE on October 15, 1991 indicating that the Petitioner should submit a petition for waiver of U.S. Government rights in the Subject Invention, however a petition at that time would have been improper. At that time, and at least until January, 1992, the Petitioner had funding proposals pending at LBL which would have provided further government resources for further development of the Subject Invention. Only when these funding proposals were rejected, could Petitioner properly file this petition.

The work under Contract (B) was extended to December 31, 1991, so independent of other activity, a petition would have been improper before that time. In fact, only in February of 1992 did Petitioner learn that no further funding of the project would occur.

14. Is the Petitioner aware of any governmental regulations which require or which might require use of the subject matter of the Subject Invention by the general public or a segment thereof? (If yes, explain.)

No.

15. Does the Subject Invention lie in or relate to a field of technology which concerns the public health, safety or welfare; (for example, the development of drugs, medical or safety instruments, anti-pollution devices or such other products that may have a bearing on health, safety or welfare of the general public). (If yes, explain.)

No.

16. If the Petitioner is a nonprofit educational institution, what is the technology transfer capability and program of the Petitioner?

Not Applicable. Petitioner is not a nonprofit educational institution.

17. Give any other facts the Petitioner believes will establish that the interests of the United States and the general public will best be served by the granting of this waiver. Sufficient information is required so that the Secretary can consider specifically each of the areas and objectives covered in Sec. 9(c), 9(d) and (e) of P.L. 93-577.

Under §§9(c)-(e) of P.L. 93-577 (as codified at 42 USC §5908(c)-(e), the Secretary of the DOE is charged with promoting the commercial utilization of the Subject Invention. If a waiver is not granted, no commercialization or further development of the Remote Objects technology will occur. This has been made clear by the DOE, as evidenced by the attached letter stating that the U.S. Government is not interested in proceeding with the filing of patent applications to protect the Subject Invention.

The University of California and LBL have also declined to pursue patent protection on the Subject Invention.

In contrast, if the waiver is granted, Petitioner can then begin patenting and marketing the Subject Invention. In attempting to get further funding from the U.S. Government, Petitioner has shown a willingness to foster the Subject Invention into a marketable product, and is now prepared to continue his efforts outside the auspices of the DOE.

If the invention was made in the course of or under a contract or subcontract of the Naval Nuclear Propulsion Program or the weapons programs or other atomic energy defense activities of the Department of Energy, the considerations of paragraph (b) of Section 3131 of the 1987 Defense Authorization Act *[as codified at 42 USC §7261a(b)]* shall also be addressed. These are as follows:

(a) whether national security will be compromised;

Given the nature of the Subject Invention, the granting of this waiver will not affect, much less compromise, the national security.

(b) whether sensitive technical information (whether classified or unclassified) under the Naval Nuclear Propulsion Program or the nuclear weapons programs or other atomic energy defense activities of the Department of Energy for which dissemination is controlled under Federal statutes and regulations will be released to unauthorized persons;

The Subject Invention does not involve sensitive technical information, classified or unclassified, under either ~~the~~ Naval Nuclear Propulsion Program, nuclear weapons programs, or atomic energy defense activities of the Department of Energy.

(c) whether an organizational conflict of interest contemplated by Federal statutes and regulations will result; and

Petitioner does not foresee any organizational conflicts of interest in the granting of this waiver.

(d) whether waiving such rights will adversely affect the operation of the Naval Nuclear Propulsion Program or the nuclear weapons programs or other atomic energy defense activities of the Department of Energy.

The granting of this waiver is expected to have no effect on the operation of the Naval Nuclear Propulsion Program, or the nuclear weapons programs or other atomic energy defense activities of the Department of Energy.

18. State below the name, address and telephone number of the person to whom correspondence is to be directed.

Philip H. Albert
Townsend and Townsend
One Market Plaza, Steuart Tower
San Francisco, CA 94105

The facts set forth in this request for waiver are within the knowledge of Petitioner and are submitted with the intention that the Secretary or his designee rely on them in reaching the waiver determination.

Respectfully submitted,

Allan M. Konrad

(Signature)

Allan M. Konrad, Petitioner

Date submitted to DOE:

14 MAY 92

hepmain.c

```
#pragma runopts(isa(4K),isainc(4K))
/*
#pragma runopts(report)
*/
/*
 *  hepmain.c
 *  main program
 *
 */

/* Motif include files */
#include <Xm/Xm.h>
#include <X11/Shell.h>
#include <Xm/Dialogs.h>

/* other includes */
#include <stdio.h>
#include "hludge.h"
#include "xmapire.h"

/* defines */
#define BX_APP_NAME "lbiBdo"
#define BX_APP_CLASS "LbiBdo"

/* global Widget variables */
Widget Shell000;
Widget Shell001;
Widget Shell002;
Widget Shell003;
Widget BulletinBoard;
Widget BulletinBoard1;
extern Widget ResultText;
extern Widget numrec;
Widget CreateBulletinBoard();
Widget CreateBulletinBoard1();

extern int ExactMatch;
extern int SSCOnly;
extern int SortBy;
extern int DispForm;

/* result  output buffer */
extern char line_buf[];

extern void init_cursor();

extern void ReturnAndPopup();

static XtActionsRec my_actions[] = {
    {"ReturnAndPopup", ReturnAndPopup},
};

/* my main loop -- replaces XtAppMainLoop */
/*
void
MyAppMainLoop(app)
XtAppContext app;
{
    XEvent event;

    for (;;) {
        XtAppNextEvent(app, &event);
        XtDispatchEvent(&event);
        if(quitflag)
            return;
    }
}
*/

/* MAIN PROGRAM */
void
main(argc, argv)
int argc;
char **argv;
{
    Display *display;
    Arg         args[256];
    int  argcnt, i;
    XtAppContext context;
    SSCOnly = 0;
    ExactMatch = 1;
    SortBy = DispForm = 0;
    fprintf(stderr, "hepmain main()\n");

    printf(" before spires init\n");
    system("progmap");
    system("mempem");
    if(spires_startup() != 0){
        printf("error connecting to SPIRES\n");
        return;
    }

    spires_command("select hep;;");
    spires_command("define area xdo_out(500,8)) on hlca(1,1);;");

    printf(" before Xt init\n");
    system("progmap");
    system("mempem");
    XtToolkitInitialize();
    context = XtCreateApplicationContext();
    display = XtOpenDisplay(context, 0, BX_APP_NAME, BX_APP_CLASS,
        0, 0, &argc, (char **)argv);
    if(display == NULL){
        fprintf(stderr, "%s:  Can't open display\n", argv[0]);
        exit(1);
    }
    XmRegisterConverters();
    argcnt = 0;
/*
    XtSetArg(args[argcnt], XmNx, 40); argcnt++;
    XtSetArg(args[argcnt], XmNy, 40); argcnt++;
*/
    Shell000 = XtAppCreateShell( BX_APP_NAME, BX_APP_CLASS,
        applicationShellWidgetClass, display, args, argcnt);

    argcnt = 0;
/*
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 10); argcnt++;
*/
    Shell001 = XtAppCreateShell( BX_APP_NAME, BX_APP_CLASS,
        applicationShellWidgetClass, display, args, argcnt);

    argcnt = 0;
    Shell002 = XtAppCreateShell( BX_APP_NAME, BX_APP_CLASS,
        applicationShellWidgetClass, display, args, argcnt);
```

```
Shell001 = XtAppCreateShell( BX_APP_NAME, BX_APP_CLASS,
    applicationShellWidgetClass, display, args, argcnt);

BulletinBoard = CreatebulletinBoard(Shell000);
XtManageChild(BulletinBoard);
BulletinBoard1 = CreatebulletinBoard1(Shell001);
XtAppAddActions(context, my_actions, XtNumber(my_actions));
XtRealizeWidget(Shell000);
splines_command("set format default;;");

printf(" before XtMainLoop\n");
system("progmap");
system("mapmem");
XtAppMainLoop(context);
}
```

hepres.c

```c
/*
 * hepres.c
 * format and display SPIRES query results
 */

#include <Xm/Xm.h>
#include <X11/Shell.h>
#include <Xm/DialogS.h>

#include <X11/cursorfont.h>

#include <stdio.h>
#include "kludge.h"
#include "xmspires.h"

extern Widget Shell000;
extern Widget Shell001;
extern Widget BulletinBoard;
extern Widget BulletinBoard1;
extern Widget ResultText;
extern Widget numrec;

#define MAXCHARS 80*1000

char line_buf[MAXCHARS];
char line[80];

Cursor waitcursor;
Cursor maincursor;

void
init_cursor()
{
    maincursor = XCreateFontCursor(XtDisplay(Shell000), XC_draft_large);
    waitcursor = XCreateFontCursor(XtDisplay(Shell000), XC_watch);
}

void
set_cursor()
{
    Display *d;
    XDefineCursor(d = XtDisplay(Shell000), XtWindow(Shell000),
        waitcursor);
    XDefineCursor(d, XtWindow(BulletinBoard),
        waitcursor);
    if(XtIsManaged(Shell001)){
        XDefineCursor(d, XtWindow(Shell001),
            waitcursor);
    }
    if(XtIsManaged(BulletinBoard1)){
        XDefineCursor(d, XtWindow(BulletinBoard1),
            waitcursor);
    }
}

void
reset_cursor()
{
    Display *d;
    XDefineCursor(d =XtDisplay(Shell000), XtWindow(Shell000),
        maincursor);
    XDefineCursor(d, XtWindow(Shell001),
        maincursor);
    XDefineCursor(d, XtWindow(BulletinBoard),
        maincursor);
    XDefineCursor(d, XtWindow(BulletinBoard1),
        maincursor);
}

void
read_active_file()
{
    FILE *fp;
    int i = 74;

    bzero(line_buf, (int)MAXCHARS);
    if((fp = fopen("ACTIVE FILE A", "r")) == NULL){
        perror("opening active file");
        exit(1);
    }
    do{
        line[0] = '\0';
        fgets(line, MAXCHARS, fp);
        strcat(line_buf, line);
    } while ( line[0] != '\0');
}

void
expose_result_window(numres)
int numres;
{
    char buf[80];
    /* raise result_window ? */
    /* bring up result window if not managed */
    if(!XtIsManaged(BulletinBoard1)){
        XtManageChild(BulletinBoard1);
        XtRealizeWidget(Shell001);
    }
    sprintf(buf, "Number of records in current search result : %d",
        numres);
    XtVaSetValues(numrec, XmNvalue, buf, NULL);
}

/* draws a window for search results
 * raises window if window is already managed
 */
draw_result_window(numres, result)
int numres;
char *result;
{
    /* draw result in text field */
    if(numres){
        XtVaSetValues(ResultText, XmNvalue,
                      line_buf,
                      NULL);
    }
    else {
        XtVaSetValues(ResultText, XmNvalue,
                      "Null search result",
                      NULL);
    }
}
```

hlis.c

```c
#define VM
#include <stdio.h>
#include <stdlib.h>

#define HLI_AREA_SIZE 500*80
/*#define HLI_AREA_SIZE 24*80 */
#define SET_HLI_BIT(addr) ((int) addr | 0x80000000)

/* The following are the external routines for calling SPIRES */

extern int spinet();
extern int spiwit();
extern int spires();

/* Global variables used by the SPIRES routines */

char hlca_area[HLI_AREA_SIZE];
char hlca_data[HLI_AREA_SIZE];
int hlca_length = HLI_AREA_SIZE;
int user_1 = 1;
int spires_response[15];

/*
 * These routines interface to the SPIRES routines, this
 * localizes the code which would need to changed if you
 * need special OS linkage routines.
 */

int spires_startup()
{
   int rc;

   rc = spinet(&user_1, SET_HLI_BIT(&spires_response));
   if (rc != 0)
      {
         printf("Return code of %d from SPINET\n",rc);
      }
   return rc;
}

int spires_command(char *cmd)
{
   int rc;                    /* Return Code */

   rc = spires(&user_1, cmd, hlca_area, SET_HLI_BIT(&hlca_length));
   if (rc != 0)
      {
         printf("Return code of %d from SPIRES\n",rc);
      }
   return rc;
}

int spires_exit()
{
   int rc;

   rc = spiwit();
   if (rc != 0)
      {
         printf("Return code of %d from SPIWIT\n",rc);
      }
   return rc;
}
```

```
#define VM
/* ftp module for xspire */

/* Motif includes */
#include <Xm/Xm.h>
#include <X11/Shell.h>
#include <Xm/ArrowB.h>
#include <Xm/ArrowBG.h>
#include <Xm/BulletinB.h>
#include <Xm/CascadeB.h>
#include <Xm/CascadeBG.h>
#include <Xm/Command.h>
#include <Xm/CutPaste.h>
#include <Xm/DialogS.h>
#include <Xm/DrawingA.h>
#include <Xm/DrawnB.h>
#include <Xm/FileSB.h>
#include <Xm/Form.h>
#include <Xm/Frame.h>
#include <Xm/Label.h>
#include <Xm/LabelG.h>
#include <Xm/List.h>
#include <Xm/MainW.h>
#include <Xm/MenuShell.h>
#include <Xm/MessageB.h>
#include <Xm/PanedW.h>
#include <Xm/PushB.h>
#include <Xm/PushBG.h>
#include <Xm/RowColumn.h>
#include <Xm/Scale.h>
#include <Xm/ScrollBar.h>
#include <Xm/ScrolledW.h>
#include <Xm/SelectionB.h>
#include <Xm/SeparatoG.h>
#include <Xm/Separator.h>
#include <Xm/Text.h>
#include <Xm/ToggleB.h>
#include <Xm/ToggleBG.h>


#include <stdio.h>
#include "xmspire.h"

void
write_file()
{
    FILE *fp;
    system("erase ftp temp a");
    fp = fopen("ftp temp a", "w");
    fprintf(fp, "%s\n" ,hlca_data );
}


void
do_put(username, passwd, filename, machine)
char *username, *passwd,  *filename, *machine;
{
    char s[100];

    sprintf(s, "EXEC ITPCMD %s %s %s", username, passwd, filename);
    system(s);
}
```

hepcall.c

```
/*
 * hepcall.c
 *
 * callbacks for Motif interface code
 * HEP database version
 * *
 */
#include <Xm/Xm.h>
#include <X11/Shell.h>
#include <Xm/DialogS.h>

#include <X11/cursorfont.h>

#include <stdio.h>
#include <ctype.h>
#include "kludge.h"
#include "awspire.h"
/* #include "awce.h" */
#include "almage.h"

extern Widget BulletinBoard1;
extern Widget Shell001;
extern Widget Shell002;
extern Widget Shell003;
extern Widget syntax;
extern Widget ResultText;
extern Widget diagnostics;
extern  Widget pbbox, mvbox, aabox, helpbox;
extern int ISCOnly;
extern int ExactMatch;
extern int SortBy;
extern int DispForm;
extern char line[];
extern char line_buf[];
extern void read_active_file();
extern void write_file();
extern void set_cursor();
extern void reset_cursor();
void set_format();
void set_sort();
extern void draw_result_window();
extern void expose_result_window();

extern char query_syntax[];

/* action for carriage return in text1 widget */
void
ReturnAndPopup(w, event)
Widget w;
XKeyPressedEvent *event;
{

    XtManageChild(pbbox);

}

void
date_none(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
}
```

```
void
date_eq(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
}


void
date_le(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
}

void
date_ge(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
}


void
not_implemented(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget dialog = (Widget) client;
    XmAnyCallbackStruct *ace=(XmAnyCallbackStruct*)call;
    XtManageChild(dialog);
}

void
popup_diagnostic(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget dialog = (Widget) client;
    XmAnyCallbackStruct *ace=(XmAnyCallbackStruct*)call;
    XtManageChild(dialog);
}

void
format_wlt_result()
{
    int i;

    for(i = 0; i < HLI_AREA_SIZE; )
        strncat(hlca_data, &hlca_area[i], 8);
        i += 8;
        hlca_data[i] = '\n';
    }
}

/* pull up a dialog */
void
accounting bill(w, client, call)
Widget w;
caddr_t client;
```

```
caddr_t call;
{
    Widget dialog = (Widget) client;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    XtManageChild(dialog);
}

/* puts up a dialog */
void
general_info(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget dialog = (Widget) client;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    XtManageChild(dialog);
}

/* puts up dialog */
void
get_copyright_perm(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget dialog = (Widget) client;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    XtManageChild(dialog);
}

/* draws a help dialog */
void
help(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget dialog = (Widget) client;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;

    /* popup help dialog */
    if(!XtIsManaged(sebox)){
        XtManageChild(helpbox);
        XtRealizeWidget(Shell00);
    }
}

/* puts up a special services dialog */
void
special_services(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget dialog = (Widget) client;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;

    /* popup help dialog */
    XtManageChild(dialog);
}

void
set_level(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
}

/* execute show elements, put up result in a dialog */
show_elements(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    /* popup dialog */
    if(!XtIsManaged(sebox)){
        XtManageChild(sebox);
        XtRealizeWidget(Shell002);
    }
}

/* init connection to another host ? */
void
start_new_session(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
}

/* quit the whole application; close all windows */
void
terminate_session(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    /* quit db session here */
    spires_exit();
    /* return control to alvp program */
    exit(0);
}
/*
void
browse_noname()
{
    spires_command("for subfile;;");
    spires_command("/skip $skip;;");
    spires_command("in rdo_out clr display 20;;");
    spires_command("endfor;;");
}
*/

/* do browse name in SPIRES
 * takes name argument from "text1" / prints syntax in "syntax"
 */
void
show_topic_values(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
}
```

```
    Widget t = (Widget) client;
    String buf, setscan, upbuf;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    int i;
    buf = (char *) malloc(100);
    upbuf = (char *) malloc(100);
    setscan = (char *) malloc(100);
    upbuf[0] = buf[0] = '\0';

    spires_command("select pre-desykeys;;");
    spires_command("set format aspires;;");
    spires_command("for tree;;");

    XtVaGetValues(t, XmNvalue, &buf, NULL);
    if(!isspace(buf[0]) && (buf[0] != '\0'))
      for(i = 0; i < strlen(buf); i++)
          upbuf[i] = toupper((int)buf[i]);
      }
    }
    else {
       sprintf(upbuf, "A");
    }
    sprintf(setscan, "set scan start %s;;", upbuf);

    spires_command(setscan);
    spires_command("skip 1;;");
    spires_command("set scan back;;");
    spires_command("skip 7;;");
    spires_command("set scan forward;;");
    spires_command("in active clean clear display 100;;");
    sprintf(query_syntax, "select pre-desykeys;;\nset format aspires;;\n\
for tree;;\n%s\nskip 1;;\nset scan back;;\nskip 7;;\n\
set scan forward;;\nin active clean clear display 100;;\n", setscan);
    read_active_file();
    expose_result_window(100);
    draw_result_window(100, line_buf);
    spires_command("select hep;;");
}

void
show_title_values(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget t = (Widget) client;
    String buf, setscan, upbuf;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    int i;
    buf = (char *) malloc(100);
    upbuf = (char *) malloc(100);
    setscan = (char *) malloc(100);
    upbuf[0] = buf[0] = '\0';

    spires_command("select pre-titles;;");
    spires_command("set format aspires;;");
    spires_command("for tree;;");

    XtVaGetValues(t, XmNvalue, &buf, NULL);
    if(!isspace(buf[0]) && (buf[0] != '\0'))
      for(i = 0; i < strlen(buf); i++)
          upbuf[i] = toupper((int)buf[i]);
      }
    }
    else {
       sprintf(upbuf, "A");
    }
    sprintf(setscan, "set scan start %s;;", upbuf);

    spires_command(setscan);
    spires_command("skip 1;;");
    spires_command("set scan back;;");
    spires_command("skip 7;;");
    spires_command("set scan forward;;");
    spires_command("in active clean clear display 100;;");
    sprintf(query_syntax, "select pre-titles;;\nset format aspires;;\n\
for tree;;\n%s\nskip 1;;\nset scan back;;\nskip 7;;\n\
set scan forward;;\nin active clean clear display 100;;\n", setscan);
    read_active_file();
    expose_result_window(100);
    draw_result_window(100, line_buf);
    spires_command("select hep;;");
}

void
show_author_values(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget t = (Widget) client;
    String buf, setscan, upbuf;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    int i;
    buf = (char *) malloc(100);
    upbuf = (char *) malloc(100);
    setscan = (char *) malloc(100);
    upbuf[0] = buf[0] = '\0';

    spires_command("select pre-authors;;");
    spires_command("set format aspires;;");
    spires_command("for tree;;");

    XtVaGetValues(t, XmNvalue, &buf, NULL);
    if(!isspace(buf[0]) && (buf[0] != '\0'))
      for(i = 0; i < strlen(buf); i++)
          upbuf[i] = toupper((int)buf[i]);
      }
    }
    else {
       sprintf(upbuf, "A");
    }
    sprintf(setscan, "set scan start %s;;", upbuf);

    spires_command(setscan);
    spires_command("skip 1;;");
    spires_command("set scan back;;");
    spires_command("skip 7;;");
    spires_command("set scan forward;;");
    spires_command("in active clean clear display 100;;");
    sprintf(query_syntax, "select pre-authors;;\nset format aspires;;\n\
for tree;;\n%s\nskip 1;;\nset scan back;;\nskip 7;;\n\
set scan forward;;\nin active clean clear display 100;;\n", setscan);
    read_active_file();
    expose_result_window(100);
    draw_result_window(100, line_buf);
    spires_command("select hep;;");
}
```

```
/* do find title in SPIRES
 * takes title argument from "text1"
 */
void
find_title(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget t = (Widget) client;
    String querybuf, s;
    int i,j,num;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    querybuf = (char *) malloc(100);
    querybuf[0] = '\0';
    s = (char *) malloc(100);

    set_cursor();
    /* build up query command */
    XtVaGetValues(t, XmNvalue, &querybuf, NULL);

    set_format();
    if(!isspace(querybuf[0]) && (querybuf[0] != '\0'))
        strcpy(s, "find title ");
        strcat(s, querybuf);
        if(!ExactMatch)
            strcat(s, "*");
        if(SSCOnly)
            strcat(s, " and ssc ssc");
        }
        strcat(s, ";;");
        strcpy(query_syntax, s);
        printf("search string  = %s\n", s);
        spires_command(s);
/*      spires_command("clr stack;;"); */
        spires_command("in rdo_out clr show eval $result;;");
        hlca_area[200] = '\0';
        printf("hlca_area = %s\n", hlca_area);
        num = atoi(hlca_area);
        expose_result_window(num);
printf("after expose\n");
        if(num != 0)
            spires_command("for result;;");
            set_sort();
            spires_command("in act cln clr type;;");
        }
    }
    else {
    /* write to diagnostics window */
        XtManageChild(evbox);
        return;
    }

    read_active_file();
    draw_result_window(num, line_buf);
}

void
find_topic(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget t = (Widget) client;
    String querybuf, s;
    int i,j,num;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    querybuf = (char *) malloc(100);
    querybuf[0] = '\0';
    s = (char *) malloc(100);

    set_cursor();
    /* build up query command */
    XtVaGetValues(t, XmNvalue, &querybuf, NULL);

    set_format();
    if(!isspace(querybuf[0]) && (querybuf[0] != '\0'))
        strcpy(s, "find topic ");
        strcat(s, querybuf);
        if(!ExactMatch)
            strcat(s, "*");
        if(SSCOnly)
            strcat(s, " and ssc ssc");
        }
        strcat(s, ";;");
        printf("search string  = %s\n", s);
        strcpy(query_syntax, s);
        spires_command(s);
/* if no result found , then skip sorting */
/* change when we sort */
        spires_command("clr stack;;");
        spires_command("in rdo_out clr show eval $result;;");
        hlca_area[200] = '\0';
        printf("hlca_area = %s\n", hlca_area);
        num = atoi(hlca_area);
        expose_result_window(num);
        if(num != 0)
            spires_command("for result;;");
            set_sort();
            spires_command("in act cln clr type;;");
        }
    }
    else {
    /* write to diagnostics window */
        XtManageChild(evbox);
        return;
    }

    read_active_file();
    draw_result_window(num, line_buf);
}


/* do find author in SPIRES
 * takes author argument from "text1"
 */
void
find_author(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget t = (Widget) client;
    String querybuf, s;
    int i,j,num;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    querybuf = (char *) malloc(100);
    querybuf[0] = '\0';
```

```
    s = (char *) malloc(100);

    set_current();
    /* build up query command */
    XtVaGetValues(t, XmNvalue, &querybuf, NULL);

    set_format();
    if(!isspace(querybuf[0]) && (querybuf[0] != '\0')){
        strcpy(s, "find author ");
        strcat(s, querybuf);
        if(!ExactMatch)
            strcat(s, "*");
        if($SCOnly){
            strcat(s, " and sec sec");
        }
        strcat(s, ";;");
        printf("search string  =_%s\n", s);
        strcpy(query_syntax, s);
        spires_command(s);
/* if no result found , then skip sorting */
/* change when we sort */
        spires_command("clr stack;;");
        spires_command("ln rdo_out clr show eval $result;;");
        hlca_area[200] = '\0';
        num = atoi(hlca_area);
        expose_result_window(num);
        if(num != 0){
            set_sort();
            spires_command("for result;;");
            spires_command("ln act cln clr type;;");
        }
    }
    else {
    /* write to diagnostics window */
        XtManageChild(evbox);
        return;
    }

    read_active_file();
    draw_result_window(num, line_buf);
}

void
show_session_history(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
}

/* ftp */
void
send_search_result(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{

    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
/*  write_file(); */
}

/* closes the search result window */
/* should this func turn the result text field? i.e. call close_result() */
void
close_result_window(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
/*
    XtUnrealizeWidget(Shell001);
    XtUnmanageChild(BulletinBoard5);
*/
}


void
close_sebox(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{

    XtUnrealizeWidget(Shell002);
    XtUnmanageChild(XtParent(w));

}

void
close_helpbox(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{

    XtUnrealizeWidget(Shell003);
    XtUnmanageChild(XtParent(w));

}

/* clears out the search result text field (text5) */
void
clear_result(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
    Widget t = (Widget) client;
    String buf;
    XmAnyCallbackStruct *acs=(XmAnyCallbackStruct*)call;
    buf = (String) malloc(4);
    strcpy((char *)buf, "");
    XtVaSetValues(t, XmNvalue, (String)buf, NULL);
}

/* cancel in mid-search */
void
interrupt_query(w, client, call)
Widget w;
caddr_t client;
caddr_t call;
{
/* signal SIGINT */
    not implemented(w, client, call);
}
```

```
/* toggle button callbacks */
void
tog_syntax(w, client, call)
Widget w;
caddr_t client, call;
{
    Boolean b;
    XtVaGetValues(w, XmNset, &b, NULL);
    if(b)
        printf("toggle is set\n");
    }

    else printf("toggle is unset\n");
}

void
show_image(w, client, call)
Widget w;
caddr_t client, call;
{
/*  ximage_main(); */
}

/* show syntax of previous search */
void
show_search(w, client, call)
Widget w;
caddr_t client, call;
{
    XmString xmstr[128];
    XtManageChild(syntax);
    XtVaSetValues(syntax, XmNmessageString,
        xmstr[0]=XmStringCreateLtoR(
        query_syntax,
        XmSTRING_DEFAULT_CHARSET), NULL);
    XmStringFree(xmstr[0]);
}

/* toggle "exact match" for <find> commands */
void
set_match(w, client, call)
Widget w;
caddr_t client, call;
{
    ExactMatch = 1;
    printf("exact match ON\n");
}

/* toggle "exact match" for <find> commands */
void
unset_match(w, client, call)
Widget w;
caddr_t client, call;
{
    ExactMatch = 0;
    printf("exact match OFF\n");
}

void
set_filter(w, client, call)
Widget w;
caddr_t client, call;
{
    SSCOnly = 1;
    printf("ssconly =1\n");
}

void
unset_filter(w, client, call)
Widget w;
caddr_t client, call;
{
    SSCOnly = 0;
    printf("ssconly =0\n");
}

void
sortrepnum(w, client, call)
Widget w;
caddr_t client, call;
{
    SortBy = 2;
    printf("sortby = %d\n", SortBy);
}

void
sortnonum(w, client, call)
Widget w;
caddr_t client, call;
{
    SortBy = 0;
    printf("sortby = %d\n", SortBy);
}

void
sortfirstau(w, client, call)
Widget w;
caddr_t client, call;
{
    SortBy = 1;
    printf("sortby = %d\n", SortBy);
}

void
sortdate(w, client, call)
Widget w;
caddr_t client, call;
{
    SortBy = 3;
    printf("sortby = %d\n", SortBy);
}


void
set_sort()
{
printf("in set_sort()\n");
    switch(SortBy)
        case 0:
            /* no sort */
            break;

        case 1:
            printf("srq by first au\n");
            spires command("srq Author;;");
            break;
```

```
}
}
        case 2:
            printf("seq by report num\n");
            spires_command("seq rs;;");
            break;

        case 3:
            printf("seq by date\n");
            spires_command("seq ds;;");
            break;
    }
}


void
set_format()
{
    switch(Dispform)
        case 0:
            spires_command("set format espires.default;;");
printf("format = default\n");
            break;
        case 1:
            spires_command("set format espires.full;;");
printf("format = full\n");
            break;
        case 2:
            spires_command("set format espires.brief;;");
printf("format = brief\n");
            break;
        case 3:
            spires_command("set format espires.tex;;");
printf("format = tex\n");
            break;

    }
}


void
set_full(w, client, call)
Widget w;
caddr_t client, call;
{
    Dispform = 1;
}


void
set_brief(w, client, call)
Widget w;
caddr_t client, call;
{
    Dispform = 2;
}

void
set_tex(w, client, call)
Widget w;
caddr_t client, call;
{
    Dispform = 3;
}
```

```
void
set_def(w, client, call)
Widget w;
caddr_t client, call;
{
    Dispform = 0;
}
```

```
/*
 * hepcrea.c
 *
 * creates widgets and defines their layout
 *
 */

/*
 * Motif includes
 */
#include <Xm/Xm.h>
#include <X11/Shell.h>
#include <Xm/ArrowB.h>
#include <Xm/ArrowBG.h>
#include <Xm/BulletinB.h>
#include <Xm/CascadeB.h>
#include <Xm/CascadeBG.h>
#include <Xm/Command.h>
#include <Xm/CutPaste.h>
#include <Xm/Dialogs.h>
#include <Xm/DrawingA.h>
#include <Xm/DrawnB.h>
#include <Xm/FileSb.h>
#include <Xm/Form.h>
#include <Xm/Frame.h>
#include <Xm/Label.h>
#include <Xm/LabelG.h>
#include <Xm/List.h>
#include <Xm/MainW.h>
#include <Xm/MenuShell.h>
#include <Xm/MessageB.h>
#include <Xm/PanedW.h>
#include <Xm/PushB.h>
#include <Xm/PushBG.h>
#include <Xm/RowColumn.h>
#include <Xm/Scale.h>
#include <Xm/ScrollBar.h>
#include <Xm/ScrolledW.h>
#include <Xm/SelectionB.h>
#include <Xm/SeparatoG.h>
#include <Xm/Separator.h>
#include <Xm/Text.h>
#include <Xm/ToggleB.h>
#include <Xm/ToggleBG.h>

#include "kludge.h"
#include "xmsplit.h"

extern Widget Shell00;
extern Widget Shell01;
Widget  syntax;
Widget ResultText;
Widget scrolledWindow;
Widget diagnostics;
Widget numerr;
Widget ntbox, dtbox, ptbox, evbox, sebox, helpbox;
Widget wtbox;
Widget brsvat;

int InsertMatch;
int [illegible];
int Sorting;
[illegible];
```

```
Widget CreateHelp();
Widget CreateShowTree();

char query_syntax[100];
extern char line_buf[];
extern char line[];

static char defaultTranslations[] = "#override\n\
        <Key>Return: ReturnAndPopup() \n\
        <Key>Linefeed: ReturnAndPopup()"


XtTranslations mytrans;

/*
 * convenience routines
 * written by the interface builder, not by me!
 */
#ifndef IGNORE_COLOR
static unsigned long
COLOR(w, name)
Widget w;
char *name;
{
XrmValue        fromVal, toVal;
unsigned long *pixel;

    fromVal.size = sizeof(char*);
    fromVal.addr = name;

    XtConvert(w, XmRString, &fromVal, XmRPixel, &toVal);
    pixel = (unsigned long*)toVal.addr;
    if( pixel == NULL )
    {
        fromVal.addr = XtDefaultBackground;
        XtConvert(w, XmRString, &fromVal, XmRPixel, &toVal);
        pixel = (unsigned long*)toVal.addr;
    }
    return(*pixel);
}
#endif
#ifndef IGNORE_FONT
static XmFontList
FONT_LIST(w, name)
Widget  w;
char    *name;
{
XrmValue        fromVal, toVal;
XmFontList      *fontListPtr;

    fromVal.size = sizeof(char*);
    fromVal.addr = name;

    XtConvert(w, XmRString, &fromVal, XmRFontList, &toVal);
    fontListPtr = (XmFontList*)toVal.addr;
    if(fontListPtr == NULL)
    {
        fromVal.addr = "fixed";
        XtConvert(w, XmRString, &fromVal, XmRFontList, &toVal);
        fontListPtr = (XmFontList*)toVal.addr;
    }
    return(*fontListPtr);
}
```

```
#endif
#ifndef IGNORE_MENU_POST
static void
MENU_POST(p, m, e)
Widget  p;
Widget  m;
XButtonEvent *e;
{
    Arg args[2];
    int argcnt;
    int button;

    argcnt = 0;
    XtSetArg(args[argcnt], XmNwhichButton, &button);
    argcnt++;
    XtGetValues(m, args, argcnt);
    if( e->button != button) return;
    XmMenuPosition(m, e);
    XtManageChild(m);
}
#endif
#ifndef IGNORE_STRING_TABLE
#include <myvarargs.h>

static XmString*
STRING_TABLE(va_alist)
int va_alist;
{
    va_list     ap;
    int         count;
    XmString    *array;
    int         i;
    int dummy;  /* for compatibility w/myvarargs.h */
    va_start(ap, dummy);
    count = va_arg(ap, int);
    array = (XmString*)XtMalloc((count + 1) * sizeof(XmString*));
    for(i = 0;  i < count; i++ )
    {
        array[i] = XmStringCreateLtoR(va_arg(ap, char*),
            XmSTRING_DEFAULT_CHARSET);
    }
    array[count] = (XmString)0;
    va_end(ap);
    return(array);
}

static void
FREE_STRING_TABLE(table)
XmString        *table;
{
    int         i;

    for( i = 0; table[i] != 0 ; i++)
    {
        XmStringFree(table[i]);
    }
    XtFree(table);
}
#endif

/*
 * callback declarations
 */
extern void accounting_bill();
```

```
extern void general_info();
extern void get_copyright_perm();
extern void help();
extern void special_services();
extern void set_level();
extern void show_elements();
extern void start_new_session();
extern void terminate_session();
extern void find_title();
extern void find_author();
extern void find_topic();
extern void show_topic_values();
extern void show_author_values();
extern void show_title_values();
extern void show_session_history();
extern void send_search_result();
extern void close_result_window();
extern void close_sebox();
extern void close_helpbox();
extern void clear_result();
extern void interrupt_query();
extern void toq_syntax();
extern void show_image();
extern void set_match();
extern void unset_match();
extern void set_filter();
extern void unset_filter();
extern void sortauthor();
extern void sortinstitution();
extern void sortrepnum();
extern void sortfirstau();
extern void sortnone();
extern void sortdate();
extern void sortauinstl();
extern void set_full();
extern void set_def();
extern void set_brief();
extern void set_tex();
extern void set_format();
extern void not_implemented();
extern void date_none();
extern void date_eq();
extern void date_le();
extern void date_ge();
extern void show_search();

Widget
CreateBulletinBoard(parent)
Widget parent;
{
    Arg         args[512];
    int         argcnt;
    Widget      retval;
    XmString    xmstr[32];
    XmString    *xmstrTable[8];
    Widget      bulletinBoard;
    Widget separator, separator1, separator2, separator3;
    Widget      separator4;
    Widget label, label1, label2, label3, label4, label5, label6;
    Widget label7, label8, label9, label10, label11, label12;
    Widget scale, rowColumn, rowColumn1, rowColumn2;
    Widget pushButton, pushButton1, pushButton2, pushButton3;
    Widget pushButton4, pushButton5, pushButton6, pushButton7;
    Widget pushButton8, pushButton9, pushButton10, pushButton11;
```

```
    Widget pushButton12, pushButton13, pushButton14;
    Widget radioBox, radioBox1, radioBox2, radioBox3;
    Widget radioBox4;
    Widget      toggleButton;
    Widget toggleButton1, toggleButton2, toggleButton3;
    Widget toggleButton4, toggleButton5, toggleButton6;
    Widget toggleButton7, toggleButton8, toggleButton9;
    Widget toggleButton10, toggleButton11, toggleButton12;
    Widget toggleButton13, toggleButton14, toggleButton15;
    Widget toggleButton16, toggleButton17, toggleButton18;
    Widget text1, text2, scrolledWindow;
    Widget temp;
    Widget accbox, spacbox, glbox, gcbox;
    XmRegisterConverters();

    argcnt = 0;
    XtSetArg(args[argcnt], XmResizePolicy, XmRESIZE_NONE); argcnt++;
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 500); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 500); argcnt++;
    bulletinBoard = XtCreateWidget("bulletinBoard",
        xmBulletinBoardWidgetClass,
        parent,
        args,
        argcnt);
    retval = bulletinBoard;

 /* draw "not implemented" message dialog */
    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
      "function not implemented",
       XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    nibox = XmCreateMessageDialog(bulletinBoard, "sdfadfsd",
                                  args, argcnt);
    temp = XmMessageBoxGetChild(nibox, XmDIALOG_CANCEL_BUTTON);
    XtUnmanageChild(temp);
    temp = XmMessageBoxGetChild(nibox, XmDIALOG_HELP_BUTTON);
    XtUnmanageChild(temp);

/* draw "syntax" message dialog */
    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
      "syntax",
       XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    syntax = XmCreateMessageDialog(bulletinBoard, "sdfadfsd",
                                   args, argcnt);
    temp = XmMessageBoxGetChild(syntax, XmDIALOG_CANCEL_BUTTON);
    XtUnmanageChild(temp);
    temp = XmMessageBoxGetChild(syntax, XmDIALOG_HELP_BUTTON);
    XtUnmanageChild(temp);

 /* draw "diagnostics" message dialog */
    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
      "diagnostics",
       XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    dibox = XmCreateMessageDialog(bulletinBoard, "sdfadfsd",
                                  args, argcnt);
    temp = XmMessageBoxGetChild(dibox, XmDIALOG_CANCEL_BUTTON);
    XtUnmanageChild(temp);
    temp = XmMessageBoxGetChild(dibox, XmDIALOG_HELP_BUTTON);
    XtUnmanageChild(temp);

 /* draw "press button" message dialog */
    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
      "press a <show values> or <find> button",
       XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    pbbox = XmCreateMessageDialog(bulletinBoard, "sdfadfsd",
                                  args, argcnt);
    temp = XmMessageBoxGetChild(pbbox, XmDIALOG_CANCEL_BUTTON);
    XtUnmanageChild(temp);
    temp = XmMessageBoxGetChild(pbbox, XmDIALOG_HELP_BUTTON);
    XtUnmanageChild(temp);

 /* draw "enter value" message dialog */
    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
      "enter a search value before pressing <find> button",
       XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    evbox = XmCreateMessageDialog(bulletinBoard, "sdfadfsd",
                                  args, argcnt);
    temp = XmMessageBoxGetChild(evbox, XmDIALOG_CANCEL_BUTTON);
    XtUnmanageChild(temp);
    temp = XmMessageBoxGetChild(evbox, XmDIALOG_HELP_BUTTON);
    XtUnmanageChild(temp);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNscrollingPolicy, XmAPPLICATION_DEFINED); argcnt++;
    XtSetArg(args[argcnt], XmNx, 20); argcnt++;
    XtSetArg(args[argcnt], XmNy, 510); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 470); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 68); argcnt++;
    scrolledWindow = XtCreateWidget("scrolledWindow",
        xmScrolledWindowWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XtManageChild(scrolledWindow);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
      "Interrupt \n\
query",
       XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 370); argcnt++;
    XtSetArg(args[argcnt], XmNy, 260); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 85); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton = XtCreateWidget("pushButton",
        xmPushButtonWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XtAddCallback(pushButton, XmNarmCallback, not_implemented, (caddr_t) nibox);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNmarginWidth, (Dimension)0); argcnt++;
    XtSetArg(args[argcnt], XmNmarginHeight, (Dimension)0); argcnt++;
```

```
XtSetArg(args[argcnt], XmNspacing, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNx, 10); argcnt++;
XtSetArg(args[argcnt], XmNy, 380); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 90); argcnt++;
XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
XtSetArg(args[argcnt], XmNradioBehavior, True); argcnt++;
XtSetArg(args[argcnt], XmNradioAlwaysOne, True); argcnt++;
radioBox = XtCreateWidget("radioBox",
    xmRowColumnWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XtManageChild(radioBox);

argcnt = 0;
XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
        "filter by:",
        XmSTRING_DEFAULT_CHARSET)); argcnt++;
XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
XtSetArg(args[argcnt], XmNx, 100); argcnt++;
XtSetArg(args[argcnt], XmNy, 340); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 90); argcnt++;
XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
label2 = XtCreateWidget("label2",
    xmLabelWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtManageChild(label2);

argcnt = 0;
XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
        "sort by:",
        XmSTRING_DEFAULT_CHARSET)); argcnt++;
XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
XtSetArg(args[argcnt], XmNx, 265); argcnt++;
XtSetArg(args[argcnt], XmNy, 360); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 80); argcnt++;
XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
label = XtCreateWidget("label",
    xmLabelWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtManageChild(label);

argcnt = 0;
XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
        "search for:",
        XmSTRING_DEFAULT_CHARSET)); argcnt++;
XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
XtSetArg(args[argcnt], XmNx, 10); argcnt++;
XtSetArg(args[argcnt], XmNy, 360); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 90); argcnt++;
XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
label3 = XtCreateWidget("label3",
    xmLabelWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtManageChild(label3);
```

```
argcnt = 0;
XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
        "display format:",
        XmSTRING_DEFAULT_CHARSET)); argcnt++;
XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
XtSetArg(args[argcnt], XmNx, 365); argcnt++;
XtSetArg(args[argcnt], XmNy, 340); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 90); argcnt++;
XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
label12 = XtCreateWidget("label12",
    xmLabelWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtManageChild(label12);

mytrans = XtParseTranslationTable(defaultTranslations);
argcnt = 0;
XtSetArg(args[argcnt], XmNeditMode, XmMULTI_LINE_EDIT); argcnt++;
XtSetArg(args[argcnt], XmNcolumns, (short)30); argcnt++;
XtSetArg(args[argcnt], XmNrows, (short)3); argcnt++;
XtSetArg(args[argcnt], XmNx, 130); argcnt++;
XtSetArg(args[argcnt], XmNy, 250); argcnt++;
text1 = XtCreateWidget("text1",
    xmTextWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XtOverrideTranslations(text1, mytrans);
XtManageChild(text1);

argcnt = 0;
XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
    "enter value and push a button",
    XmSTRING_DEFAULT_CHARSET)); argcnt++;
XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
XtSetArg(args[argcnt], XmNx, 130); argcnt++;
XtSetArg(args[argcnt], XmNy, 280); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 200); argcnt++;
XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
label13 = XtCreateWidget("label13",
    xmLabelWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtManageChild(label13);

argcnt = 0;
XtSetArg(args[argcnt], XmNmarginWidth, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNmarginHeight, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNspacing, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNx, 365); argcnt++;
XtSetArg(args[argcnt], XmNy, 380); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 130); argcnt++;
XtSetArg(args[argcnt], XmNheight, 74); argcnt++;
XtSetArg(args[argcnt], XmNradioBehavior, True); argcnt++;
XtSetArg(args[argcnt], XmNradioAlwaysOne, True); argcnt++;
radioBox4 = XtCreateWidget("radioBox4",
    xmRowColumnWidgetClass,
    bulletinBoard,
    args,
```

```
    argcnt);
XtManageChild(radioBox4);

argcnt = 0;
XtSetArg(args[argcnt], XmNmarginWidth, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNmarginHeight, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNspacing, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNx, 275); argcnt++;
XtSetArg(args[argcnt], XmNy, 380); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 120); argcnt++;
XtSetArg(args[argcnt], XmNheight, 96); argcnt++;
XtSetArg(args[argcnt], XmNradioBehavior, True); argcnt++;
XtSetArg(args[argcnt], XmNradioAlwaysOne, True); argcnt++;
radioBox3 = XtCreateWidget("radioBox3",
                           xmRowColumnWidgetClass,
                           bulletinBoard,
                           args,
                           argcnt);
XtManageChild(radioBox3);

argcnt = 0;
XtSetArg(args[argcnt], XmNmarginWidth, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNmarginHeight, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNspacing, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNx, 100); argcnt++;
XtSetArg(args[argcnt], XmNy, 380); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 145); argcnt++;
XtSetArg(args[argcnt], XmNheight, 34); argcnt++;
XtSetArg(args[argcnt], XmNradioBehavior, True); argcnt++;
XtSetArg(args[argcnt], XmNradioAlwaysOne, True); argcnt++;
radioBox1 = XtCreateWidget("radioBox1",
    xmRowColumnWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XtManageChild(radioBox1);

argcnt = 0;
XtSetArg(args[argcnt], XmNmarginWidth, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNmarginHeight, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNspacing, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNx, 200); argcnt++;
XtSetArg(args[argcnt], XmNy, 380); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 145); argcnt++;
XtSetArg(args[argcnt], XmNheight, 34); argcnt++;
XtSetArg(args[argcnt], XmNradioBehavior, True); argcnt++;
XtSetArg(args[argcnt], XmNradioAlwaysOne, True); argcnt++;
radioBox2 = XtCreateWidget("radioBox2",
    xmRowColumnWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XtManageChild(radioBox2);

argcnt = 0;
XtSetArg(args[argcnt], XmNborderWidth, (Dimension)0); argcnt++;
XtSetArg(args[argcnt], XmNorientation, XmHORIZONTAL); argcnt++;
XtSetArg(args[argcnt], XmNspacing, (Dimension)3); argcnt++;
XtSetArg(args[argcnt], XmNentryAlignment, XmALIGNMENT_CENTER); argcnt++;
XtSetArg(args[argcnt], XmNpacking, XmPACK_NONE); argcnt++;
XtSetArg(args[argcnt], XmNnumColumns, (short)4); argcnt++;
XtSetArg(args[argcnt], XmNradioAlwaysOne, False); argcnt++;
XtSetArg(args[argcnt], XmNisHomogeneous, False); argcnt++;
XtSetArg(args[argcnt], XmNx, 10); argcnt++;
XtSetArg(args[argcnt], XmNy, 305); argcnt++;
XtSetArg(args[argcnt], XmNwidth, 480); argcnt++;
XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
rowColumn1 = XtCreateWidget("rowColumn1",
    xmRowColumnWidgetClass,
    bulletinBoard,
    args,
    argcnt);
XtManageChild(rowColumn1);
/*
    argcnt = 0;
    XtSetArg(args[argcnt], XmNborderWidth, (Dimension)0); argcnt++;
    XtSetArg(args[argcnt], XmNorientation, XmHORIZONTAL); argcnt++;
    XtSetArg(args[argcnt], XmNspacing, (Dimension)3); argcnt++;
    XtSetArg(args[argcnt], XmNentryAlignment, XmALIGNMENT_CENTER); argcnt++;
    XtSetArg(args[argcnt], XmNpacking, XmPACK_NONE); argcnt++;
    XtSetArg(args[argcnt], XmNnumColumns, (short)4); argcnt++;
    XtSetArg(args[argcnt], XmNradioAlwaysOne, False); argcnt++;
    XtSetArg(args[argcnt], XmNisHomogeneous, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 345); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 480); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    rowColumn2 = XtCreateWidget("rowColumn2",
        xmRowColumnWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XtManageChild(rowColumn2);
*/
    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
            "find\ntitle",
            XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 275); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 65); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton1 = XtCreateWidget("pushButton1",
        xmPushButtonWidgetClass,
        rowColumn1,
        args,
        argcnt);
    XtAddCallback(pushButton1, XmNarmCallback, find_title,
        (caddr_t)text1);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton1);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
            "find\nauthor",
            XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 345); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 65); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton11 = XtCreateWidget("pushButton11",
        xmPushButtonWidgetClass,
        rowColumn1,
        args,
        argcnt);
    XtAddCallback(pushButton11, XmNarmCallback, find_author,
        (caddr_t)text1);
    XmStringFree( xmstr[0] );
```

```
    XtManageChild(pushButton11);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
            "find\ntopic",
            XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 415); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 65); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton14 = XtCreateWidget("pushButton14",
        xmPushButtonWidgetClass,
        rowColumn1,
        args,
        argcnt);
    XtAddCallback(pushButton14, XmNarmCallback, find_topic,
        (caddr_t)text1);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton14);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "show sample\n\
topic values",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 85); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton7 = XtCreateWidget("pushButton7",
        xmPushButtonWidgetClass,
        rowColumn1,
        args,
        argcnt);
    XtAddCallback(pushButton7, XmNarmCallback, show_topic_values,
        (caddr_t)text1);
    XmStringFree( xmstr[0] );

    XtManageChild(pushButton7);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "show sample\n\
author values",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 90); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 85); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton12 = XtCreateWidget("pushButton12",
        xmPushButtonWidgetClass,
        rowColumn1,
        args,
        argcnt);
    XtAddCallback(pushButton12, XmNarmCallback,show_author_values,
        (caddr_t)text1);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton12);


    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "show sample\n\
title values",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 180); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 85); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton13 = XtCreateWidget("pushButton13",
        xmPushButtonWidgetClass,
        rowColumn1,
        args,
        argcnt);
    XtAddCallback(pushButton13, XmNarmCallback, show_title_values,
        (caddr_t)text1);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton13);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNfontList, FONT_LIST(parent, "-adobe-helvetica-bold-o-n
--12-120-75-75-p-69-iso8859-1")); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "Perform Query",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNalignment,XmALIGNMENT_BEGINNING); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 245); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 130); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    label6 = XtCreateWidget("label6",
        xmLabelWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(label6);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 215); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 480); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    separator2 = XtCreateWidget("separator2",
        xmSeparatorWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XtManageChild(separator2);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNborderWidth, (Dimension)0); argcnt++;
    XtSetArg(args[argcnt], XmNorientation, XmHORIZONTAL); argcnt++;
    XtSetArg(args[argcnt], XmNspacing, (Dimension)3); argcnt++;
    XtSetArg(args[argcnt], XmNentryAlignment, XmALIGNMENT_CENTER); argcnt++;
    XtSetArg(args[argcnt], XmNpacking, XmPACK_NONE); argcnt++;
    XtSetArg(args[argcnt], XmNnumColumns, (short)5); argcnt++;
    XtSetArg(args[argcnt], XmNradioAlwaysOne, False); argcnt++;
    XtSetArg(args[argcnt], XmNisHomogeneous, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 185); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 604); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    rowColumn = XtCreateWidget("rowColumn",
        xmRowColumnWidgetClass,
        bulletinBoard,
        args,
```

```
        argcnt);
    XtManageChild(rowColumn);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
"Innocent                    expert",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 230); argcnt++;
    XtSetArg(args[argcnt], XmNy, 120); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 260); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
    label7 = XtCreateWidget("label7",
        xmLabelWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(label7);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNminimum, (int)1); argcnt++;
    XtSetArg(args[argcnt], XmNmaximum, (int)10); argcnt++;
    XtSetArg(args[argcnt], XmNorientation, XmHORIZONTAL); argcnt++;
    XtSetArg(args[argcnt], XmNprocessingDirection, XmMAX_ON_BOTTOM); argcnt++;
    XtSetArg(args[argcnt], XmNtitleString, (xmstr[0]=XmStringCreateLtoR(
            " ",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNshowValue, True); argcnt++;
    XtSetArg(args[argcnt], XmNdecimalPoints, (short)0); argcnt++;
    XtSetArg(args[argcnt], XmNx, 245); argcnt++;
    XtSetArg(args[argcnt], XmNy, 140); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 290); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 50); argcnt++;
    scale = XtCreateWidget("scale",
        xmScaleWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XmStringFree( xmstr[0] );

    XtAddCallback(scale, XmNvalueChangedCallback, set_level, (caddr_t)0);
    XtManageChild(scale);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
        "Specify level of expertise",
        XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 250); argcnt++;
    XtSetArg(args[argcnt], XmNy, 100); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 190); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
    label8 = XtCreateWidget("label8",
        xmLabelWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(label8);

/* get spices subtitle size for display */
    spices_command("set active active file a;");
    spices_command("in set clear show subtitle size;");
    system("cms copyfile active file a - - - (recfm f lrecl 200 rep");
    read_active_file();

    argcnt = 0;
    XtSetArg(args[argcnt], XmNvalue, line_buf); argcnt++;
    XtSetArg(args[argcnt], XmNautoShowCursorPosition, False); argcnt++;
    XtSetArg(args[argcnt], XmNeditable, False); argcnt++;
    XtSetArg(args[argcnt], XmNcolumns, (short)33); argcnt++;
    XtSetArg(args[argcnt], XmNrows, (short)1); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 140); argcnt++;
    text2 = XtCreateWidget("text2",
        xmTextWidgetClass,
        bulletinBoard,
        args,
        argcnt);

    XtManageChild(text2);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 80); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 480); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
    separator3 = XtCreateWidget("separator3",
        xmSeparatorWidgetClass,
        bulletinBoard,
        args,
        argcnt);

    XtManageChild(separator3);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNfontList, FONT_LIST(parent, "-adobe-helvetica-bold-n-normal
--12-120-75-75-p-69-iso8859-1")); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "Getting Acquainted",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNalignment,XmALIGNMENT_BEGINNING); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 100); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 180); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    label9 = XtCreateWidget("label9",
        xmLabelWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(label9);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNfontList, FONT_LIST(parent, "-adobe-helvetica-bold-n-normal
--8-80-75-75-p-50-iso8859-1")); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
  "| Remote Database Object # Copyright 199], Regents of the\
University of California. |",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNalignment, XmALIGNMENT_BEGINNING);argcnt++;
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNy, 101); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 640); argcnt++;
```

hepcrea.c

```
    XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
    label10 = XtCreateWidget("label10",
        xmLabelWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(label10);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNfontList, FONT_LIST(parent, "-adobe-helvetica-bold-o-norma
l--14-140-75-75-p-82-iso8859-1")); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
  "SLAC High Energy Physics (HEP)",
            XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 40); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 480); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
    label11 = XtCreateWidget("label11",
        xmLabelWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(label11);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
        "start new session",
        XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 240); argcnt++;
    XtSetArg(args[argcnt], XmNy, 10); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 230); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
    pushButton3 = XtCreateWidget("pushButton3",
        xmPushButtonWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtAddCallback(pushButton3, XmNarmCallback, not_implemented, (caddr_t)inibox);
    XtManageChild(pushButton3);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 10); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 225); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 20); argcnt++;
    pushButton4 = XtCreateWidget("terminate session",
        xmPushButtonWidgetClass,
        bulletinBoard,
        args,
        argcnt);
    XtAddCallback(pushButton4, XmNarmCallback, terminate_session, (caddr_t)0);
    XtManageChild(pushButton4);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNwidth, (Dimension)250); argcnt++;
    XtSetArg(args[argcnt], XmNheight, (Dimension)50); argcnt++;
    XtSetValues(scale, args, argcnt);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNalignment, XmALIGNMENT_CENTER); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
      "help",
      XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 50); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    pushButton5 = XtCreateWidget("pushButton5",
        xmPushButtonWidgetClass,
        rowColumn,
        args,
        argcnt);
    XmStringFree( xmstr[0] );

 /* draw a message dialog */
    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
      "We at SLAC are happy to answer questions about search\n\
strategies, etc. Phone L. Addis or R. Gee, x2411 or send\n\
email to ADDIS@SLACVM or LIBTG@SLACVM.\n\n\
Send bug reports to Cynthia Hertzer, chertzer@usl.3bl.gov",
        XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    helpbox = CreateHelp(pushButton5);
    XtAddCallback(pushButton5, XmNarmCallback, help,
      (caddr_t) helpbox);
    XtManageChild(pushButton5);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNalignment, XmALIGNMENT_CENTER); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "accounting\n\
billing",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 60); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 80); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    pushButton6 = XtCreateWidget("pushButton6",
                xmPushButtonWidgetClass,
                rowColumn,
                args,
                argcnt);
    XmStringFree( xmstr[0] );

    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
      "Accounting + billing info",  XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    accbox = XmCreateMessageDialog(pushButton6, "accounting billing",
                                   args, argcnt);
    temp = XmMessageBoxGetChild(accbox, XmDIALOG_HELP_BUTTON);
    XtUnmanageChild(temp);
    temp = XmMessageBoxGetChild(accbox, XmDIALOG_CANCEL_BUTTON);
    XtUnmanageChild(temp);

    XtAddCallback(pushButton6, XmNarmCallback, accounting_bill,
                  (caddr_t)accbox);
    XtManageChild(pushButton6);
```

```
    argcnt = 0;
    XtSetArg(args[argcnt], XmNalignment, XmALIGNMENT_CENTER); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "show\n\
elements",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 150); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 70); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    pushButton7 = XtCreateWidget("pushButton7",
        xmPushButtonWidgetClass,
        rowColumn,
        args,
        argcnt);
    XmStringFree( xmstr[0] );

    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
      hlca_data, XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    xbox = CreateShowElem(pushButton7);
    XtAddCallback(pushButton7, XmNarmCallback, show_elements,
        (caddr_t)xbox);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton7);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNalignment, XmALIGNMENT_CENTER); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "general\n\
information",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 230); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 80); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    pushButton8 = XtCreateWidget("pushButton8",
                xmPushButtonWidgetClass,
                rowColumn,
                args,
                argcnt);
    argcnt = 0;
    XtSetArg(args[argcnt], XmNmessageString, (xmstr[0]=XmStringCreateLtoR(
"General Information\n\n\
HEP is a joint project of the SLAC and DESY Libraries.\n\
As of March 1990 it included more than 200,000 records from both\n\
sources, dated from 1974 to the present. HEP is updated daily with new\n\
preprints received in the SLAC library and biweekly with new journal\n\
articles and conference proceedings papers indexed at the DESY\n\
library. Input is also received via BITNET from CERN, Fermilab,\n\
KEK, Kyoto U., RIFP, and SSC. \n\
\n\
HEP includes all SLAC Library preprint and report holdings from\n\
1974 (all SLAC items from 1962+) as well as journal articles,\n\
conference papers, theses, etc. from the DESY High Energy Physics\n\
Index, a comprehensive bibliography produced at our sister\n\
laboratory in Hamburg, Germany.",
        XmSTRING_DEFAULT_CHARSET)));
    argcnt++;
    qtbox = XmCreateMessageDialog(pushButton8, "general information",
                                        args, argcnt);
    temp = XmMessageBoxGetChild(qtbox, XmDIALOG_HELP_BUTTON);
    XtUnmanageChild(temp);
    temp = XmMessageBoxGetChild(qtbox, XmDIALOG_CANCEL_BUTTON);
    XtUnmanageChild(temp);

    XtAddCallback(pushButton8, XmNarmCallback, general_info,
                (caddr_t)qtbox);
    XmStringFree( xmstr[0] );

    XtManageChild(pushButton8);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNalignment, XmALIGNMENT_CENTER); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "get copyright\n\
permission",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 320); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 85); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    pushButton9 = XtCreateWidget("pushButton9",
                xmPushButtonWidgetClass,
                rowColumn,
                args,
                argcnt);
    argcnt = 0;
    XtSetArg(args[argcnt],XmNmessageString,(xmstr[0]=XmStringCreateLtoR(\
    "Copyright 1991 by the Regents of the University of California\n\
\n\
Permission to use, copy, modify, and distribute this software and its\n\
documentation for any purpose and without fee is hereby granted,\n\
provided that the above copyright notice appear in all copies and that\n\
both that copyright notice and this permission notice appear in\n\
supporting documentation, and that the name of the University of\n\
California, Lawrence Berkeley Laboratory, or the U.S. Department of\n\
Energy not be used in advertising or publicity pertaining to\n\
distribution of the software without specific, written, prior\n\
permission. The University of California makes no representations\n\
about the suitability of this software for any purpose. It is\n\
provided \"as is\" without express or implied warranty. The Remote\n\
Database Object is a Trademark of the University of California.",  \
XmSTRING_DEFAULT_CHARSET))); argcnt++;

    qcbox = XmCreateMessageDialog(pushButton9, "get copyright permission",
                                        args, argcnt);
    temp = XmMessageBoxGetChild(qcbox, XmDIALOG_HELP_BUTTON);
    XtUnmanageChild(temp);
    temp = XmMessageBoxGetChild(qcbox, XmDIALOG_CANCEL_BUTTON);
    XtUnmanageChild(temp);

    XtAddCallback(pushButton9, XmNarmCallback, get_copyright_perm,
                (caddr_t)qcbox);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton9);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "special\n\
services",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
```

```
    XtSetArg(args[argcnt], XmWx, 415); argcnt++;
    XtSetArg(args[argcnt], XmWy, 0); argcnt++;
    XtSetArg(args[argcnt], XmWwidth, 70); argcnt++;
    XtSetArg(args[argcnt], XmWheight, 30); argcnt++;
    pushButton10 = XtCreateWidget("pushButton10",
                xmPushButtonWidgetClass,
                rowColumn,
                args,
                argcnt);
    XmStringFree( xmstr[0] );

    argcnt = 0;
    XtSetArg(args[argcnt], XmWmessageString, (xmstr[0]=XmStringCreateLtoR(
      "Special Services",  XmSTRING_DEFAULT_CHARSET)));
    argcnt++;

    specbox = XmCreateMessageDialog(pushButton10, "special services",
                                    args, argcnt);
    temp = XmMessageBoxGetChild(specbox, XmDIALOG_HELP_BUTTON);
    XtVaSetValues(temp,
                XmWlabelString, (xmstr[0]=XmStringCreateLtoR(
                "Current Awareness\n\
Service",
                XmSTRING_DEFAULT_CHARSET)),
                NULL);
    temp = XmMessageBoxGetChild(specbox, XmDIALOG_CANCEL_BUTTON);
    XtAddCallback(pushButton10, XmNarmCallback, special_services,
                (caddr_t)specbox);
    XmStringFree(xmstr[0]);
    XtManageChild(pushButton10);


    argcnt = 0;
    XtSetArg(args[argcnt], XmWlabelString, (xmstr[0]=XmStringCreateLtoR(
  "all papers",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmWset, True); argcnt++;
    XtSetArg(args[argcnt], XmWspacing, (short)0); argcnt++;
    XtSetArg(args[argcnt], XmWindicatorOn, True); argcnt++;
    XtSetArg(args[argcnt], XmWx, 0); argcnt++;
    XtSetArg(args[argcnt], XmWy, 0); argcnt++;
    toggleButton1 = XtCreateWidget("toggleButton1",
        xmToggleButtonWidgetClass,
        radioBox1,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtAddCallback(toggleButton1, XmNarmCallback, unset_filter,
        (caddr_t) 0);
    XtManageChild(toggleButton1);

    argcnt = 0;
    XtSetArg(args[argcnt], XmWlabelString, (xmstr[0]=XmStringCreateLtoR(
        "SSC papers\n\
only",
        XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmWspacing, (short)0); argcnt++;
    XtSetArg(args[argcnt], XmWx, 0); argcnt++;
    XtSetArg(args[argcnt], XmWy, 17); argcnt++;
    toggleButton2 = XtCreateWidget("toggleButton2",
        xmToggleButtonWidgetClass,
        radioBox1,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtAddCallback(toggleButton2, XmNarmCallback, set_filter,
        (caddr_t) 0);
    XtManageChild(toggleButton2);

    argcnt = 0;
    XtSetArg(args[argcnt], XmWlabelString, (xmstr[0]=XmStringCreateLtoR(
  "none",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmWset, True); argcnt++;
    XtSetArg(args[argcnt], XmWspacing, (short)0); argcnt++;
    XtSetArg(args[argcnt], XmWx, 3); argcnt++;
    XtSetArg(args[argcnt], XmWy, 3); argcnt++;
    toggleButton3 = XtCreateWidget("toggleButton3",
        xmToggleButtonWidgetClass,
        radioBox1,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtAddCallback(toggleButton3, XmNarmCallback, sortnone,
        (caddr_t) 0);
    XtManageChild(toggleButton3);

    argcnt = 0;
    XtSetArg(args[argcnt], XmWlabelString, (xmstr[0]=XmStringCreateLtoR(
        "first\nauthor",
        XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmWspacing, (short)0); argcnt++;
    XtSetArg(args[argcnt], XmWx, 3); argcnt++;
    XtSetArg(args[argcnt], XmWy, 20); argcnt++;
    toggleButton4 = XtCreateWidget("toggleButton4",
        xmToggleButtonWidgetClass,
        radioBox1,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtAddCallback(toggleButton4, XmNarmCallback, sortfirstau,
        (caddr_t)0);

    XtManageChild(toggleButton4);

    argcnt = 0;
    XtSetArg(args[argcnt], XmWlabelString, (xmstr[0]=XmStringCreateLtoR(
        "report\nnumber",
        XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmWspacing, (short)0); argcnt++;
    XtSetArg(args[argcnt], XmWx, 3); argcnt++;
    XtSetArg(args[argcnt], XmWy, 37); argcnt++;
    toggleButton5 = XtCreateWidget("toggleButton5",
        xmToggleButtonWidgetClass,
        radioBox1,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtAddCallback(toggleButton5, XmNarmCallback, sortrepnum,
        (caddr_t)0);
    XtManageChild(toggleButton5);

    argcnt = 0;
    XtSetArg(args[argcnt], XmWlabelString, (xmstr[0]=XmStringCreateLtoR(
        "date",
        XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmWspacing, (short)0); argcnt++;
    XtSetArg(args[argcnt], XmWx, 3); argcnt++;
```

hepcrea.c

```
  XtSetArg(args[argcnt], XmMy, 54); argcnt++;
  toggleButton6 = XtCreateWidget("toggleButton6",
      xmToggleButtonWidgetClass,
      radioBox3,
      args,
      argcnt);
  XmStringFree( xmstr[0] );
  XtAddCallback(toggleButton6, XmArmCallback, sortdate,
      (caddr_t) 0);
  XtManageChild(toggleButton6);

  argcnt = 0;
  XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
              "default",
              XmSTRING_DEFAULT_CHARSET))); argcnt++;
  XtSetArg(args[argcnt], XmNset, True); argcnt++;
  XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNx, 3); argcnt++;
  XtSetArg(args[argcnt], XmNy, 3); argcnt++;
  toggleButton7 = XtCreateWidget("toggleButton7",
      xmToggleButtonWidgetClass,
      radioBox4,
      args,
      argcnt);
  XmStringFree( xmstr[0] );
  XtAddCallback(toggleButton7, XmArmCallback, set_def,
      (caddr_t) 0);
  XtManageChild(toggleButton7);

  argcnt = 0;
  XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
"full",
              XmSTRING_DEFAULT_CHARSET))); argcnt++;
  XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNx, 3); argcnt++;
  XtSetArg(args[argcnt], XmNy, 20); argcnt++;
  toggleButton8 = XtCreateWidget("toggleButton8",
      xmToggleButtonWidgetClass,
      radioBox4,
      args,
      argcnt);
  XmStringFree( xmstr[0] );
  XtAddCallback(toggleButton8, XmArmCallback, set_full,
      (caddr_t) 0);

  XtManageChild(toggleButton8);

  argcnt = 0;
  XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
      "brief",
      XmSTRING_DEFAULT_CHARSET))); argcnt++;
  XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNx, 3); argcnt++;
  XtSetArg(args[argcnt], XmNy, 37); argcnt++;
  toggleButton9 = XtCreateWidget("toggleButton9",
      xmToggleButtonWidgetClass,
      radioBox4,
      args,
      argcnt);
  XmStringFree( xmstr[0] );
  XtAddCallback(toggleButton9, XmArmCallback, set_brief,
      (caddr_t) 0);
  XtManageChild(toggleButton9);
```

```
  argcnt = 0;
  XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
      "tex",
      XmSTRING_DEFAULT_CHARSET))); argcnt++;
  XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNx, 3); argcnt++;
  XtSetArg(args[argcnt], XmNy, 54); argcnt++;
  toggleButton10 = XtCreateWidget("toggleButton10",
      xmToggleButtonWidgetClass,
      radioBox4,
      args,
      argcnt);
  XmStringFree( xmstr[0] );
  XtAddCallback(toggleButton10, XmArmCallback, set_tex,
      (caddr_t) 0);
  XtManageChild(toggleButton10);

  argcnt = 0;
  XtSetArg(args[argcnt], XmNmarginWidth, (short)2); argcnt++;
  XtSetArg(args[argcnt], XmNmarginHeight, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
      "exact match",
      XmSTRING_DEFAULT_CHARSET))); argcnt++;
  XtSetArg(args[argcnt], XmNset, True); argcnt++;
  XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNx, 3); argcnt++;
  XtSetArg(args[argcnt], XmNy, 3); argcnt++;
  toggleButton11 = XtCreateWidget("toggleButton11",
      xmToggleButtonWidgetClass,
      radioBox,
      args,
      argcnt);
  XmStringFree( xmstr[0] );
  XtAddCallback(toggleButton11, XmArmCallback, set_match,
      (caddr_t) 0);
  XtManageChild(toggleButton11);

  argcnt = 0;
  XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
      "begins with",
      XmSTRING_DEFAULT_CHARSET))); argcnt++;
  XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNx, 3); argcnt++;
  XtSetArg(args[argcnt], XmNy, 23); argcnt++;
  toggleButton12 = XtCreateWidget("toggleButton12",
      xmToggleButtonWidgetClass,
      radioBox,
      args,
      argcnt);
  XmStringFree( xmstr[0] );
  XtAddCallback(toggleButton12, XmArmCallback, unset_match,
      (caddr_t) 0);
  XtManageChild(toggleButton12);

  argcnt = 0;
  XtSetArg(args[argcnt], XmNmarginWidth, (short)2); argcnt++;
  XtSetArg(args[argcnt], XmNmarginHeight, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
      "none",
      XmSTRING_DEFAULT_CHARSET))); argcnt++;
  XtSetArg(args[argcnt], XmNset, True); argcnt++;
  XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
  XtSetArg(args[argcnt], XmNx, 5); argcnt++;
  XtSetArg(args[argcnt], XmNy, 1); argcnt++;
```

```
toggleButton13 = XtCreateWidget("toggleButton13",
    xmToggleButtonWidgetClass,
    radioBox2,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtAddCallback(toggleButton13, XmArmCallback, date_none,
    (caddr_t) 0);
XtManageChild(toggleButton13);

argcnt = 0;
XtSetArg(args[argcnt], XmNmarginWidth, (short)2); argcnt++;
XtSetArg(args[argcnt], XmNmarginHeight, (short)0); argcnt++;
XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
    "date - ",
    XmSTRING_DEFAULT_CHARSET)); argcnt++;
XtSetArg(args[argcnt], XmNset, False); argcnt++;
XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
XtSetArg(args[argcnt], XmNx, 3); argcnt++;
XtSetArg(args[argcnt], XmNy, 3); argcnt++;
toggleButton14 = XtCreateWidget("toggleButton14",
    xmToggleButtonWidgetClass,
    radioBox2,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtAddCallback(toggleButton14, XmArmCallback, date_mq,
    (caddr_t) 0);
XtManageChild(toggleButton14);

argcnt = 0;
XtSetArg(args[argcnt], XmNmarginWidth, (short)2); argcnt++;
XtSetArg(args[argcnt], XmNmarginHeight, (short)0); argcnt++;
XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
    "date <- ",
    XmSTRING_DEFAULT_CHARSET)); argcnt++;
XtSetArg(args[argcnt], XmNset, False); argcnt++;
XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
XtSetArg(args[argcnt], XmNx, 3); argcnt++;
XtSetArg(args[argcnt], XmNy, 3); argcnt++;
toggleButton15 = XtCreateWidget("toggleButton15",
    xmToggleButtonWidgetClass,
    radioBox2,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtAddCallback(toggleButton15, XmArmCallback, date_le,
    (caddr_t) 0);
XtManageChild(toggleButton15);

argcnt = 0;
XtSetArg(args[argcnt], XmNmarginWidth, (short)2); argcnt++;
XtSetArg(args[argcnt], XmNmarginHeight, (short)0); argcnt++;
XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
    "date >= ",
    XmSTRING_DEFAULT_CHARSET)); argcnt++;
XtSetArg(args[argcnt], XmNset, False); argcnt++;
XtSetArg(args[argcnt], XmNspacing, (short)0); argcnt++;
XtSetArg(args[argcnt], XmNx, 3); argcnt++;
XtSetArg(args[argcnt], XmNy, 3); argcnt++;
toggleButton16 = XtCreateWidget("toggleButton16",
    xmToggleButtonWidgetClass,
    radioBox2,
    args,
    argcnt);
XmStringFree( xmstr[0] );
XtAddCallback(toggleButton16, XmArmCallback, date_ge,
    (caddr_t) 0);
XtManageChild(toggleButton16);

/*
    argcnt = 0;
    XtSetArg(args[argcnt], XmNeditMode, XmMULTI_LINE_EDIT); argcnt++;
    XtSetArg(args[argcnt], XmNeditable, False); argcnt++;
    XtSetArg(args[argcnt], XmNcolumns, (short)40); argcnt++;
    XtSetArg(args[argcnt], XmNrows, (short)3); argcnt++;
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
     diagnostics = XtCreateWidget("text",
        xmTextWidgetClass,
        scrolledWindow,
        args,
        argcnt);
    XtManageChild( diagnostics);
*/
    return( retval );
}

Widget
CreatebulletinBoard1(parent)
Widget parent;
{
    Arg         args[512];
    int         argcnt;
    Widget      retval;
    XmString    xmstr[37];
    XmString    *xmstrTable[8];
    Widget      bulletinBoard1;
    Widget label12, label13;
    Widget      separator5;
    Widget pushButton11, pushButton12, pushButton13;
    Widget pushButton14, pushButton15;
    XmRegisterConverters();

    argcnt = 0;
    XtSetArg(args[argcnt], XmNresizePolicy, XmRESIZE_NONE); argcnt++;
/*
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
*/
    XtSetArg(args[argcnt], XmNwidth, 500); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 417); argcnt++;

    bulletinBoard1 = XtCreateWidget("bulletinBoard1",
                                    xmBulletinBoardWidgetClass,
                                    parent,
                                    args,
                                    argcnt);
    retval = bulletinBoard1;

    argcnt = 0;
    XtSetArg(args[argcnt], XmNscrollingPolicy, XmAPPLICATION_DEFINED); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 150); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 480); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 200); argcnt++;
    scrolledWindow1 = XtCreateWidget("scrolledWindow1",
        xmScrolledWindowWidgetClass,
```

```
        bulletinBoard1,
        args,
        argcnt);
    XtManageChild(scrolledWindow1);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
        "show session\n\
history",
        XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 210); argcnt++;
    XtSetArg(args[argcnt], XmNy, 20); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 140); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton11 = XtCreateWidget("pushButton11",
        xmPushButtonWidgetClass,
        bulletinBoard1,
        args,
        argcnt);
/*  XtAddCallback(pushButton11, XmNarmCallback, show_session_history,
                (caddr_t)0);
*/
    XtAddCallback(pushButton11, XmNarmCallback, not_implemented,
                (caddr_t)nlbox);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton11);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
  "show\n\
search",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 210); argcnt++;
    XtSetArg(args[argcnt], XmNy, 70); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 140); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton12 = XtCreateWidget("pushButton12",
                                xmPushButtonWidgetClass,
                                bulletinBoard1,
                                args,
                                argcnt);
    XmStringFree( xmstr[0] );
    XtAddCallback(pushButton12, XmNarmCallback, show_search,
                (caddr_t)syntax);
/*
    XtAddCallback(pushButton12, XmNarmCallback, not_implemented,
                (caddr_t)nlbox);
*/
    XtManageChild(pushButton12);

/*
    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "show\n\
search",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 530); argcnt++;
    XtSetArg(args[argcnt], XmNy, 20); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 100); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 50); argcnt++;
    pushButton13 = XtCreateWidget("pushButton13",
                                xmPushButtonWidgetClass,
                                bulletinBoard1,
                                args,
                                argcnt);
    XmStringFree( xmstr[0] );
    XtAddCallback(pushButton15, XmNarmCallback, show_search,
                (caddr_t)0);
    XtManageChild(pushButton15);
*/
    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
  "download result",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 360); argcnt++;
    XtSetArg(args[argcnt], XmNy, 20); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 100); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton13 = XtCreateWidget("pushButton13",
        xmPushButtonWidgetClass,
        bulletinBoard1,
        args,
        argcnt);
    XtAddCallback(pushButton13, XmNarmCallback, not_implemented,
                (caddr_t)nlbox);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton13);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "clear result",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 360); argcnt++;
    XtSetArg(args[argcnt], XmNy, 70); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 100); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pushButton14 = XtCreateWidget("pushButton14",
        xmPushButtonWidgetClass,
        bulletinBoard1,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(pushButton14);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNvalue,
"Number of records in current search result : "); argcnt++;
    XtSetArg(args[argcnt], XmNeditMode, XmMULTI_LINE_EDIT); argcnt++;
    XtSetArg(args[argcnt], XmNeditable, False); argcnt++;
    XtSetArg(args[argcnt], XmNwordWrap, True); argcnt++;
    XtSetArg(args[argcnt], XmNblinkRate, (int)0); argcnt++;
    XtSetArg(args[argcnt], XmNcolumns, (short)24); argcnt++;
    XtSetArg(args[argcnt], XmNrows, (short)3); argcnt++;
    XtSetArg(args[argcnt], XmNcursorPositionVisible, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 20); argcnt++;
    XtSetArg(args[argcnt], XmNy, 40); argcnt++;
    numrec = XtCreateWidget("text4",
        xmTextWidgetClass,
        bulletinBoard1,
        args,
        argcnt);
    XtManageChild(numrec);
```

hepcrea.c

```
    argcnt = 0;
    XtSetArg(args[argcnt], XmX, 10); argcnt++;
    XtSetArg(args[argcnt], XmY, 110); argcnt++;
    XtSetArg(args[argcnt], XmWidth, 480); argcnt++;
    XtSetArg(args[argcnt], XmHeight, 30); argcnt++;
    separator5 = XtCreateWidget("separator5",
        xmSeperatorWidgetClass,
        bulletinBoard1,
        args,
        argcnt);
    XtManageChild(separator5);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNfontList, FONT_LIST(parent, "-adobe-helvetica-bold-o-norma
l--12-120-75-75-p-69-iso8859-1")); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
 "Search Result Content (prototype displays first 50 records)",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNalignment,XmALIGNMENT_BEGINNING); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 120); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 430); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    label13 = XtCreateWidget("label13",
        xmLabelWidgetClass,
        bulletinBoard1,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(label13);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNfontList, FONT_LIST(parent, "-adobe-helvetica-bold-o-norma
l--12-120-75-75-p-69-iso8859-1")); argcnt++;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
                "Search Result Management",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 10); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 230); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 30); argcnt++;
    XtSetArg(args[argcnt], XmNalignment, XmALIGNMENT_BEGINNING);
     argcnt++;
    label12 = XtCreateWidget("label12",
        xmLabelWidgetClass,
        bulletinBoard1,
        args,
        argcnt);
    XmStringFree( xmstr[0] );
    XtManageChild(label12);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNeditMode, XmMULTI_LINE_EDIT); argcnt++;
    XtSetArg(args[argcnt], XmNwordWrap, True); argcnt++;
    XtSetArg(args[argcnt], XmNeditable, False); argcnt++;
    XtSetArg(args[argcnt], XmNcolumns, (short)75); argcnt++;
    XtSetArg(args[argcnt], XmNrows, (short)500); argcnt++;
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    ResultText = XtCreateWidget("text5",
        xmTextWidgetClass,
        scrolledWindow1,
        args,
        argcnt);
    XtManageChild(ResultText);
    XtAddCallback(pushButton14, XmNarmCallback, clear_result,
                (caddr_t) ResultText);
    return( retval );

}

Widget
CreateShowElem(parent)
Widget parent;
{
    int argcnt;
    Widget bb;
    Widget sw, el_text;
    Widget pb;
    Arg args[10];
    XmString    xmstr[32];
    XmString    *xmstrTable[8];

    spires_command("in rdo out show elements;;");
    hlca_area[HLI_AREA_SIZE] = '\0';
    format_elt_result();

    argcnt = 0;
    XtSetArg(args[argcnt], XmNwidth, 280); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 360); argcnt++;
    bb = XtCreateWidget("bulletinBoard1",
                        xmBulletinBoardWidgetClass,
                        Shell002,
                        args,
                        argcnt);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
 "qa",
                XmSTRING_DEFAULT_CHARSET))); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 60); argcnt++;
    XtSetArg(args[argcnt], XmNy, 340); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 100); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pb = XtCreateWidget("pb",
        xmPushButtonWidgetClass,
        bb,
        args,
        argcnt);
    XtAddCallback(pb, XmNarmCallback, close_sebox,
                (caddr_t) 0);

    XmStringFree( xmstr[0] );
    XtManageChild(pb);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNscrollingPolicy, XmAPPLICATION_DEFINED); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 10); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 260); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 320); argcnt++;
    sw = XtCreateWidget("sw",
        xmScrolledWindowWidgetClass,
        bb,
        args,
```

```
        argcnt);
    XtManageChild(sw);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNeditMode, XmMULTI_LINE_EDIT); argcnt++;
    XtSetArg(args[argcnt], XmNwordWrap, True); argcnt++;
    XtSetArg(args[argcnt], XmNeditable, False); argcnt++;
    XtSetArg(args[argcnt], XmNvalue, hlcs_data); argcnt++;
    XtSetArg(args[argcnt], XmNcolumns, (short)40); argcnt++;
    XtSetArg(args[argcnt], XmNrows, (short)600); argcnt++;
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    el_text = XtCreateWidget("el_text",
        xmTextWidgetClass,
        sw,
        args,
        argcnt);
    XtManageChild(el_text);
    return(bb);
}


Widget
CreateHelp(parent)
Widget parent;
{
    int argcnt;
    Widget bb;
    Widget sw, help_text;
    Widget pb;
    Arg args[10];
    XmString    xmstr[32];
    XmString    *xmstrTable[8];

    argcnt = 0;
    XtSetArg(args[argcnt], XmNwidth, 480); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 280); argcnt++;
    bb = XtCreateWidget("bulletinBoard",
                        xmBulletinBoardWidgetClass,
                        Shell00),
                        args,
                        argcnt);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNlabelString, (xmstr[0]=XmStringCreateLtoR(
"ok",
                XmSTRING_DEFAULT_CHARSET)); argcnt++;
    XtSetArg(args[argcnt], XmNrecomputeSize, False); argcnt++;
    XtSetArg(args[argcnt], XmNx, 160); argcnt++;
    XtSetArg(args[argcnt], XmNy, 260); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 100); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 40); argcnt++;
    pb = XtCreateWidget("pb",
        xmPushButtonWidgetClass,
        bb,
        args,
        argcnt);
    XtAddCallback(pb, XmNarmCallback, close_helpbox,
                  (caddr_t) 0);

    XmStringFree( xmstr[0] );
    XtManageChild(pb);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNscrollingPolicy, XmAPPLICATION_DEFINED); argcnt++;
    XtSetArg(args[argcnt], XmNx, 10); argcnt++;
    XtSetArg(args[argcnt], XmNy, 10); argcnt++;
    XtSetArg(args[argcnt], XmNwidth, 460); argcnt++;
    XtSetArg(args[argcnt], XmNheight, 240); argcnt++;
    sw = XtCreateWidget("sw",
        xmScrolledWindowWidgetClass,
        bb,
        args,
        argcnt);
    XtManageChild(sw);

    argcnt = 0;
    XtSetArg(args[argcnt], XmNeditMode, XmMULTI_LINE_EDIT); argcnt++;
    XtSetArg(args[argcnt], XmNwordWrap, True); argcnt++;
    XtSetArg(args[argcnt], XmNeditable, False); argcnt++;
    XtSetArg(args[argcnt], XmNvalue,
"To use the Remote HLP Client:\n\
\n\
1. Decide on whether you wish to search on title, author, or topic.\n\
2. Enter your search value in the field on the Specify Query area. \n\
3. Press the appropriate button (e.g., <show sample title words> \n\
<find author>, etc. \n\
4. A search result window will popup and you can scroll through the \n\
result. \n\
5. To execute another search, or adjust the filter or output format \n\
-- click on the main window \n\
-- change to desired filter or format \n\
If you want to change the search value, select the whole value \n\
currently in that field and enter your new value. \n\
Repeat step 3. \n\n\n\n\
For help with questions about search strategies, etc.\n\
phone L. Addis or R. Gex, x2411 or send email to \n\
email to ADDIS@SLACVM.slac.stanford.edu or \n\
LIRYC@SLACVM.slac.stanford.edu. \n\
\n\
 Send bug reports to Cynthia Hertzer, chertzer@usl.lbl.gov\n\
");
        argcnt++;
    XtSetArg(args[argcnt], XmNcolumns, (short)40); argcnt++;
    XtSetArg(args[argcnt], XmNrows, (short)600); argcnt++;
    XtSetArg(args[argcnt], XmNx, 0); argcnt++;
    XtSetArg(args[argcnt], XmNy, 0); argcnt++;
    help_text = XtCreateWidget("el_text",
        xmTextWidgetClass,
        sw,
        args,
        argcnt);
    XtManageChild(help_text);
    return(bb);
}
```

```
#include <X11/X10.h>
#include <X11/Xutil.h>
#include <X11/Xos.h>
#include <stdio.h>
#include "eventnames.h"

#define CMS

#define MONO(rd,gn,bl) (((rd)*11 + (gn)*16 + (bl)*5) >> 5)  /*.31R+ .5G+ .17B*/

#ifdef CMS
#include "allglue.h"
#endif

#define maxwd 10

#define MYMAPSIZ 257
#define IMGSIZ 33371

int domono=0;

int x_setupd_();
void eventloop_(),make_cursors(),x_make_image();

char *display_name = NULL;
Display *display;
int screen,depth;
Window win_root,win;
XSetWindowAttributes attr;
long event_mask;
XColor cells[256];
char imagefile[30],rawfile[30];

GC gc;
Colormap dcmap,cmap,cmap_use,cmapnew;
Visual *visual;
int class;
XEvent event;
Cursor curl,cur2;

int cmap_static;

int status;
int win_xorg,win_yorg;
unsigned int win_width,win_height,win_border,win_depth;

XImage *image[maxwd];
short old_magx[maxwd],old_magy[maxwd];
Window wd_id[maxwd];
Colormap clr_map[maxwd];

int nwin,windex;
unsigned char cmapdata[256][100];

void eventloop_()
{
    while (1)
    {
        XNextEvent(display,&event);
        /printf(stderr," event: %s in window %d \n",
                event_names[event.type],event.xany.window);
        for(windex=0 ;windex<nwin;windex++)
                if(wd_id[windex] == event.xany.window) break;
        printf("window in loop %d %d %d\n",windex,wd_id[windex],event.xany.window);
        if (windex ==0)
            XPutImage(display,wd_id[0],gc,image[0],
                0,0,0,0,100,256);
        else
            XPutImage(display,wd_id[1],gc,image[1],
                0,0,0,0,151,221);
/*
        XGetGeometry(display,win,&win_root,&win_xorg,&win_yorg,
                &win_width,&win_height,&win_border,&win_depth);
        printf("geometry %d  %d %d %d\n",win_xorg,win_yorg,win_width,win_height);
*/
    }
}
```

```
int m_setupd_(n_colors,cmap_ro, scr_width_pix,scr_width_mm,
                                        scr_height_pix,scr_height_mm)
    int *n_colors,*cmap_ro,*scr_width_pix,*scr_width_mm;
    int          *scr_height_pix,*scr_height_mm;
{

        printf("ncolors %d\n",*n_colors);
        display =XOpenDisplay(display_name);
        printf("display %d\n",display);
        fflush(stdout);
        if( display == NULL) {
            printf("Could not open display!/n");
            return 0;
        }
        screen = DefaultScreen(display);
        win_root = RootWindow(display,screen);
        depth = DisplayPlanes(display,screen);
        gc = DefaultGC(display,screen);
        *n_colors =  DisplayCells(display,screen);
        printf(" depth & colors %d %d\n",depth,n_colors);
        *scr_width_pix  = DisplayWidth(display,screen);
        *scr_width_mm   = 0.1 * DisplayWidthMM(display,screen);
        *scr_height_pix = DisplayHeight(display,screen);
        *scr_height_mm  = 0.1 * DisplayHeightMM(display,screen);

        visual = DefaultVisual(display,screen);
/*
        visual->class = DirectColor;
*/
        class = visual->class;
        printf("visual class %d\n",class);
        dcmap = DefaultColormap(display,screen);
        cmap = dcmap;
        cmap_use = cmap;
        cmap_static =    (visual->class == StaticGray) ||
                         (visual->class == StaticColor) ||
                         (visual->class == TrueColor);
        *cmap_ro = cmap_static;
        printf("%d %d %d %d %d %d \n",GrayScale,PseudoColor,DirectColor,
        StaticGray,StaticColor,TrueColor);

        make_cursors();
        return 1;
}
```

```
void make_cursors()
{
        Pixmap shape,mask;
/*  Bitmap data of cursors' shape and mask.
*/

        static char curl_shape[] ={     0x40, 0x00, 0x40, 0x00, 0x40, 0x00,
                                        0x40, 0x00, 0x40, 0x00, 0x00, 0x00, 0x1F, 0x1F,
                                        0x00, 0x00, 0x40, 0x00, 0x40, 0x00, 0x40, 0x00,
                                        0x40, 0x00, 0x40, 0x00  };
        static char curl_mask[] ={      0xE0, 0x00, 0xE0, 0x00, 0xE0, 0x00,
                                        0xE0, 0x00, 0xE0, 0x00, 0x1F, 0x1F, 0x1F, 0x1F,
                                        0x1F, 0x1F, 0xE0, 0x00, 0xE0, 0x00, 0xE0, 0x00,
                                        0xE0, 0x00, 0xE0, 0x00  };
        static char cur2_shape[] ={ 0x01, 0x10, 0x02, 0x08, 0x04, 0x04,
                                    0x08, 0x02, 0x10, 0x01, 0x00, 0x00, 0x00, 0x00,
                                    0x00, 0x00, 0x10, 0x01, 0x08, 0x02, 0x04, 0x04,
                                    0x02, 0x08, 0x01, 0x10 };
        static char cur2_mask[] = { 0x03, 0x18, 0x07, 0x1C, 0x0E, 0x0E,
                                    0x1C, 0x07, 0x18, 0x03, 0x10, 0x01, 0x00, 0x00,
                                    0x10, 0x01, 0x18, 0x03, 0x1C, 0x07, 0x0E, 0x0E,
                                    0x07, 0x1C, 0x03, 0x18 };
        XColor white,black,hwhite,hblack;
        XLookupColor(display,cmap,"WHITE",&white,&hwhite);
        printf("lookup color %d %d %d %ld \n",white.red,white.green,white.blue,white.pixe
l);
        XLookupColor(display,cmap,"BLACK",&black,&hblack);
        printf("lookup color %d %d %d %ld \n",black.red,black.green,black.blue,black.pixe
l);
        printf("color looked up\n");
        shape = XCreatePixmapFromBitmapData(display,win_root,
                        curl_shape,13,13,1,0,1);
        mask =  XCreatePixmapFromBitmapData(display,win_root,
                        curl_mask,13,13,1,0,1);
        printf("pixmap created\n");
        curl=XCreatePixmapCursor(display,shape,mask,&white,&black,6,6);
        shape = XCreatePixmapFromBitmapData(display,win_root,
                        cur2_shape,13,13,1,0,1);
        mask =  XCreatePixmapFromBitmapData(display,win_root,
                        cur2_mask,13,13,1,0,1);
        printf("pixmap created\n");
        cur2=XCreatePixmapCursor(display,shape,mask,
                        &white,&black,6,6);
}
```

ximage.c

```
void x_make_window_(windex,xp,yp,havepos,width,height,icomp,banner,normal)
    int *windex;
    int *xp,*yp;
    unsigned int *width,*height;
    char *banner;
    int *havepos,*icomp,*normal;
{
    unsigned long flags;
    static unsigned int border_width = 0;
        flags = CWEventMask | CWBackingStore | CWBackingPlanes;
        if (! cmap_static) flags = flags | CWColormap;
        if(*normal) flags = flags | CWCursor;
        event_mask =    ExposureMask | StructureNotifyMask |
                        ButtonMotionMask | KeyPressMask    ;
        attr.colormap = cmapnew;
        attr.cursor = curl;
        attr.event_mask = event_mask;
        attr.backing_store = Always;
        attr.backing_planes = 255;
        printf("xcreatewindow %d %d\n",*xp,*yp);
        win = XCreateWindow(display,win_root,*xp,*yp,*width,*height,
            border_width,depth,InputOutput,visual,flags,&attr);
        printf("index %d\n",*windex);
        wd_id[*windex] = win;
        cir_map[*windex] = cmap_use;
        printf ("banner %s\n",banner);
        XStoreName(display,win,banner);
        XMapRaised(display,win);
        XSelectInput(display,win,event_mask);
        status = XGetGeometry(display,win,&win_root,&win_xorg,&win_yorg,
                &win_width,&win_height,&win_border,&win_depth);
        printf("geometry %d  %d %d %d\n",win_xorg,win_yorg,win_width,win_height);
}
```

```
void my_image(imagefile,pixels)
char *imagefile;
int pixels[];

{
    FILE *ifile;
    static unsigned char idat[IMGSIZ];
    int kount;
    int i;

    ifile = fopen (imagefile,"r");
    if(ifile == NULL) {
        printf("could not open imagefile named %s\n",imagefile);
        exit(0);
    }
    kount = fread(idat,1,IMGSIZ,ifile);
    printf("file %s read with %d bytes\n",imagefile,kount);
    kount = 0;
    for(i=0;i<IMGSIZ;i++) {
        idat[i] =  pixels[ idat[i]];
        if(idat[i] <0 || idat[i]>7246) kount++;
    }
    printf("kount %d\n",kount);
    x_make_image(1,idat,151,221);
    printf("image made\n");
}


void x_make_image(index,data,width,height)
int index;
unsigned char *data;
unsigned int width;
unsigned int height;
{
    image[index] = XCreateImage(display,visual,depth,ZPixmap,0,data,
                                width,height,8,0);
}
```

```
void rih_cmap(cmap,mymap,myname,pixels)
Colormap cmap;
Colormap *mymap;
char *myname;
int pixels[];
{
    int base;
    int k;
    int ncolors,nstore;
    char red,green,blue;
    int ired,igreen,iblue;
    unsigned int width,height;
    char tchar;
    int i,j;
    Status status;
    FILE *rfile,*gfile,*bfile,*ifile,*rgbfile;
    XStandardColormap bestmap;
    int planes[10];
    int nplanes,npixels;
    char pixflag;
    int kount;
    FILE *mapfile;
    int mon;

    for(i=0;i<256;i++) cells[i].pixel=i;
    XQueryColors(display,cmap,cells,256);
/*
    for(i=0;i<256;i++) printf("readonly %d %d %d %d %d %d\n",i,cells[i].pixel,cells[i].r
ed,cells[i].green,cells[i].blue,cells[i].pad);
*/
    pixflag = DoRed | DoGreen | DoBlue;
    nplanes = 0;
    npixels =MYMAPSIZ;
    ncolors = npixels;
    printf ("alloc %d %d %d %d %d\n",npixels,display,cmap,False);
    status = XAllocColorCells(display,cmap,False,planes,nplanes,pixels,npixels);
    printf("status %d\n",status);
    fflush(stdout);
    if(status)
        status=0;
        while (!status && npixels ) {
                status =XAllocColorCells (display, cmap, False,
                        planes, nplanes, pixels, npixels);
                if(!status)  npixels--;
        }
        printf("\ncan allocate %d spaces in colormap\n",npixels);
    }
    printf("need %d spaces in colormap\n\n",ncolors);
    fflush(stdout);
    if(npixels < ncolors) status =0;
    if(!status)
        printf("Unable to allocate needed space in existing colormap\n");

    mapfile = fopen("fixmap map","r");
    if(mapfile == NULL) {printf("cannot open file\n");exit(0);}
    else printf("open ok\n");
    fflush(stdout);
    if(status) base=0;
    else {
        npixels = ncolors;
        base=256-ncolors;
        XQueryColors(display,cmap,cells,256);
        for(i=base;i<base+npixels;i++) pixels[i]=i;
    }
    printf("base npixels %d %d\n",base,npixels);
    fflush(stdout);
    for(i=base;i<base+npixels;i++) {
        kount=fscanf(mapfile,"%d %d %d\n",&ired,&igreen,&iblue);
        cells[i].pixel = pixels[i];
        if(domono) {
            mon=MONO(ired,igreen,iblue)<<8;
            cells[i].red   = mon;
            cells[i].green = mon;
            cells[i].blue  = mon;
        }
        else
        {
            cells[i].red=ired <<8;
            cells[i].green=igreen <<8;
            cells[i].blue=iblue <<8;
        }
        cells[i].flags = DoRed | DoGreen | DoBlue;
/*
        printf("allocated color cells %d %d %d %d %d %d %d %d\n",i,cells[i].pixel,cells[i
].red,cells[i].green,cells[i].blue,ired,igreen,iblue);
*/
    }
    printf("filled cells\n");
    fflush(stdout);

    if(status) {
        cmapnew=cmap;
        nstore = npixels;
    }
    else {
        cmapnew = XCreateColormap(display,win_root,wvisual,AllocAll);
        nstore = 256;
/*      npixels = ncolors; */
    }
    printf("storing %d colors at base %d with new %d\n",nstore,base,npixels);
    fflush(stdout);
    XStoreColors(display,cmapnew,&cells[0],nstore);
    *mymap = cmapnew;
    XInstallColormap(display,cmapnew);
    for(i=0;i<npixels;i++) pixels[i]=pixels[i]+base;
}
```

```
void kiuge_imagn_()
{
    char red,green,blue;
    unsigned int width,height;
    char tchar;
    int i,j;

    XColor color,hcolor;
    tchar = (char)color.pixel;
    XLookupColor(display,cmap,"RED",&color,&hcolor);
    XAllocColor(display,cmap,&color);
    red = (char) color.pixel;
    printf("lookup red %d %d %d %ld %d\n",color.red,color.green,color.blue,color.pixel,r
ed);
    for (i=0;i<87;i++) for (j=0;j<100;j++) cmapdata[i][j]  = red;
    XLookupColor(display,cmap,"GREEN",&color,&hcolor);
    XAllocColor(display,cmap,&color);
    green = (char) color.pixel;
    printf("lookup green %d %d %d %ld %d\n",color.red,color.green,color.blue,color.pixel
,green);
    for(i=87;i<164;i++) for(j=0;j<100;j++) cmapdata[i][j] = green;
    XLookupColor(display,cmap,"BLUE",&color,&hcolor);
    XAllocColor(display,cmap,&color);
    blue = (char) color.pixel;
    printf("lookup blue %d %d %d %ld %d\n",color.red,color.green,color.blue,color.pixel,
blue);
    for (i=164;i<256;i++) for(j=0;j<100;j++) cmapdata[i][j] = blue;
    x_make_image(0,cmapdata,100,256);
}

void cmap_image_()
{
    int i,j;

    for (i=0;i<256;i++) for(j=0;j<100;j++) cmapdata[i][j] = i/16;
    x_make_image(0,cmapdata,100,256);
}
```

```
void main(argc,argv)
int argc;
char **argv;
{
    int ok;
    int n_colors,cmap_ro,scr_width_pix,scr_width_mm;
    int          scr_height_pix,scr_height_mm;
    int xp,yp;
    unsigned int width,height;
    unsigned int cmapwidth,cmapheight;
    char banner;
    int havepos,icomp,normal;
    int pixels[256];

    n_colors = 99;

    printf("hello--enter image name!\n");
    scanf("%s",imagefile);

    strcat(imagefile, " image");
    printf("filename %s\n",imagefile);
    ok =  x_setup_(&n_colors,&cmap_ro, &scr_width_pix,&scr_width_mm,
                                       &scr_height_pix,&scr_height_mm);
/*    XSynchronize(display,1); */
/*  printf("asynchronize\n\n"); */
        xp = 100;
        yp = 100;
        havepos = 0;
        width = 151;
        height = 221;
        cmapwidth = 100;
        cmapheight = 256;
        icomp = 0;
        normal = 1;
        rlh_cmap(cmap,&cmap_use,"filemap",pixels);
#ifdef DEBUG
        printf("maps %d %d %d\n",cmap,cmapnow,cmap_use);
        fflush(stdout);
#endif
        nwin = 0;
/*   x_make_window_(&nwin,&xp,&yp,&havepos,&cmapwidth,&cmapheight,&icomp,"cmap",&norma'
*/
        fflush(stdout);
/* cmap_image_();*/
        nwin++;
        xp=xp+300;
        my_image(imagefile,pixels);
        x_make_window_(&nwin,&xp,&yp,&havepos,&width,&height,&icomp,imagefile,&normal);
  eventloop_();
}
```

```
/* xmspire.h
 * header file for Motif version of Xspire
 */

#define MLI_AREA_SIZE 200*80

/* The following are the external routines for calling SPIRES */
extern int spires_command();
extern int spires_wait();
extern int spires_startup();

extern char hica_area[];
extern char hica_data[];
extern int user_i;
/*extern char dispnum[];*/
```

```
/* $XConsortium: Varargs1.h,v 1.8 89/12/14 14:55:30 swick Exp $ */
/*

Copyright 1985, 1986, 1987, 1988, 1989 by the
Massachusetts Institute of Technology

Permission to use, copy, modify, and distribute this
software and its documentation for any purpose and without
fee is hereby granted, provided that the above copyright
notice appear in all copies and that both that copyright
notice and this permission notice appear in supporting
documentation, and that the name of M.I.T. not be used in
advertising or publicity pertaining to distribution of the
software without specific, written prior permission.
M.I.T. makes no representations about the suitability of
this software for any purpose.  It is provided "as is"
without express or implied warranty.

*/

#ifndef _Varargs1_h_
#define _Varargs1_h_

#include <resource.h>


/* temporary hack for varargs stuff till function prototypes are fully implemented */
#if NeedVarargsPrototypes
#undef NeedFunctionPrototypes
/*#define NeedFunctionPrototypes 1*/
#endif

#ifdef __STDC__
# ifndef HISSING_STDARG_H
#  include <stdarg.h>
# endif
# define Va_start(a,b) va_start(a,b)
#else
#ifdef IBMCPP
# ifndef HISSING_STDARG_H
#  include <stdarg.h>
# endif
# define Va_start(a,b) va_start(a,b)
#else /* !IBMCPP */
# include <varargs.h>
# define Va_start(a,b) va_start(a)
#endif
#endif

typedef struct _XtTypedArg {
    String      name;
    String      type;
    XtArgVal    value;
    int         size;
} XtTypedArg;

/* private routines */

extern void _XtCountVaList(
#if NeedFunctionPrototypes
    va_list /*var*/, int* /*total_count*/, int* /*typed_count*/
#endif
);

extern void _XtVaToArgList(
#if NeedFunctionPrototypes
    Widget /*widget*/, va_list /*var*/, int /*max_count*/, ArgList* /*args_return*/, Card
inal* /*num_args_return*/
#endif
);

extern void _XtVaToTypedArgList(
#if NeedFunctionPrototypes
    va_list /*var*/, int /*count*/, XtTypedArgList* /*args_return*/, Cardinal* /*num_args
_return*/
#endif
);

extern XtTypedArgList _XtVaCreateTypedArgList(
#if NeedFunctionPrototypes
    va_list /*var*/, int /*count*/
#endif
);

#endif /* _Varargs1_h_ */
```

```
/* kludgebucket for crippled C/170 */

#define BulletinBoard1 BB1
#define CreateBulletinBoard1 CreBB1
#define separator1 sep1
#define separator2 sep2
#define separator3 sep3
#define separator4 sep4
#define separator5 sep5
#define separator6 sep6
#define separator7 sep7
#define separator8 sep8
#define pushButton1 pushb1
#define pushButton2 pushb2
#define pushButton3 pushb3
#define pushButton4 pushb4
#define pushButton5 pushb5
#define pushButton6 pushb6
#define pushButton7 pushb7
#define pushButton8 pushb8
#define pushButton9 pushb9
#define pushButton10 pushb10
#define pushButton11 pushb11
#define pushButton12 pushb12
#define pushButton13 pushb13


#define toggleButton1 toggb1
#define toggleButton2 toggb2
#define toggleButton3 toggb3
#define toggleButton4 toggb4
#define toggleButton5 toggb5
#define toggleButton6 toggb6
#define toggleButton7 toggb7
#define toggleButton8 toggb8
#define toggleButton9 toggb9
#define toggleButton10 toggb10

#define radioBox1 radb1
#define radioBox2 radb2
#define radioBox3 radb3
#define radioBox4 radb4
#define radioBox5 radb5

#define scrolledWindow1 scroll1
#define scrolledWindow2 scroll2
#define scrolledWindow3 scroll3
```

myivp.c

```
#pragma runopts(isa(48),isainc(48))
/*
#pragma runopts(report)
*/
/*
 * sivp.c
 *
 * process that listen()s on an IVP, awaiting connections from the
 * Connection Manager. When a connection is received, the IVP launches
 * the database client interface.
 *
 */

#define CLOSE 1

#include <stdio.h>
#include <sllglue.h>

#include <ctype.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <saiucv.h>
#include <netinet/in.h>
#include <netdb.h>


/* creates a listening IUCV stream socket */
void
make_listener(sock, name)
int *sock;
char *name;
{
    struct sockaddr_iucv listensock;

/* establish an endpoint for connection */
    if((*sock = socket(AF_IUCV, SOCK_STREAM, 0)) < 0){
        perror("socket");
        exit(1);
    }
/* assign a name to the socket */
    memset(&listensock, 0, sizeof(listensock));
    strncpy(listensock.siucv_nodeid, "        ", 8);
    strncpy(listensock.siucv_userid, "        ", 8);
    strncpy(listensock.siucv_name, "        ", 8);
    listensock.siucv_family = AF_IUCV;
    strncpy(listensock.siucv_name, name, 8);
    strncpy(listensock.siucv_userid, "RDOSCS  ", 8);
/*
    printf("siucv_port = %d\n", listensock.siucv_port);
    printf("siucv_addr = %d\n", listensock.siucv_addr);
    printf("siucv_nodeid = %s\n", listensock.siucv_nodeid);
    printf("siucv_name = %s\n", listensock.siucv_name);
    printf("siucv_userid = %s\n", listensock.siucv_userid);
*/
    if(bind(*sock, (struct sockaddr *) &listensock, sizeof(listensock)) < 0){
        perror("binding stream socket");
        exit(1);
    }

    if(listen (*sock, 0) < 0){
        perror("listen");
        exit(1);
    }
    printf("listening \n");
}

/* read dbid, display address, etc. from Connection Manager */
void
receive_msg(activesock)
int activesock;
{
    unsigned int displen, hostlen, db_id;
    int msg, rc;
    char *hostname, *display;
    char cmd[100];

    if(read(activesock, &hostlen, sizeof(int)) != 4){
        perror("wrong hostlen");
        exit(1);
    }
    hostlen = ntohl(hostlen);
    printf("hostlen = %d\n", hostlen);

    if(read(activesock, &displen, sizeof(int)) != 4){
        perror("wrong displen");
        exit(1);
    }
    displen = ntohl(displen);
    printf("displen %d\n", displen);

    if(read(activesock, &db_id, sizeof(int)) != 4){
        perror("wrong db_id");
        exit(1);
    }
    db_id = ntohl(db_id);

    display = (char *) malloc(displen +1);
    if(read(activesock, display, displen) != displen){
        perror("wrong display addr");
        exit(1);
    }
    display[displen] = '\0';
    hostname = (char *) malloc(hostlen +1);
    if(read(activesock, hostname, hostlen) != hostlen){
        perror("wrong hostname");
        exit(1);
    }
    hostname[hostlen] = '\0';

    printf("received from Connection Manager:\n");
    printf("\tdb_id: %d\n", db_id);
    printf("\tdisplay: %s\n", display);
    printf("\thostname: %s\n", hostname);
    sprintf(cmd, "hepmain -display %s\n", display);
    printf("cmd = %s\n", cmd);

    system("mapmem");
    system("progmap");

    rc = system(cmd);
    fprintf(stderr, " finished rc = %d\n", rc);
    /* tell Connection Manager that we're finished */
    msg = CLOSE;
    if(write(activesock, &msg, sizeof(int)) != 4){
        perror("writing sock");
/*      exit(1); */
    }
    fprintf(stderr, "closing sock\n");
```

eventnam.h

```
static char *event_names[] = {
    "",
    "",
    "KeyPress",
    "KeyRelease",
    "ButtonPress",
    "ButtonRelease",
    "MotionNotify",
    "EnterNotify",
    "LeaveNotify",
    "FocusIn",

    "FocusOut",
    "KeymapNotify",
    "Expose",
    "GraphicsExpose",
    "NoExpose",
    "VisibilityNotify",
    "CreateNotify",
    "DestroyNotify",
    "UnmapNotify",
    "MapNotify",

    "MapRequest",
    "ReparentNotify",
    "ConfigureNotify",
    "ConfigureRequest",
    "GravityNotify",
    "ResizeRequest",
    "CirculateNotify",
    "CirculateRequest",
    "PropertyNotify",
    "SelectionClear",

    "SelectionRequest",
    "SelectionNotify",
    "ColormapNotify",
    "ClientMessage",
    "MappingNotify"
};
```

```
    close(activesock);
}

/* accept()s... and print a message when a connection arrives */
int
do_accept(sock)
int sock;
{
    struct sockaddr_iucv  active;
    int activesock, namelen;
    struct hostent *hp;
    char buf[1024];

    namelen = sizeof(active);
    if((activesock = accept(sock, &active, &namelen)) < 0){
        perror("accept");
        exit(1);
    }

    else {
/*      if((hp = gethostbyaddr((char *) &active.sin_addr,
                               sizeof(struct in_addr),
                               AF_INET)) == NULL){
            perror("gethostbyaddr");
            exit(1);
 }*/
    }
    printf("[connection accepted]\n");

    fflush(stdout);
    return(activesock);  \
}

main(argc, argv)
int argc;
char **argv;
{
    int sock, asock;

    make_listener(&sock, argv[1]);
    while(1){
        asock = do_accept(sock);
        receive_msg(asock);
    }
}
```

myivp.c

```
#pragma runopts(isa(48),isainc(48))
/*
#pragma runopts(report)
*/
/*
 * aivp.c
 *
 * process that listen()s on an IVP, awaiting connections from the
 * Connection Manager. When a connection is received, the IVP launches
 * the database client interface.
 *
 */

#define CLOSE 1

#include <stdio.h>
#include <aliglue.h>

#include <ctype.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <saiucv.h>
#include <netinet/in.h>
#include <netdb.h>


/* creates a listening IUCV stream socket */
void
make_listener(sock, name)
int *sock;
char *name;
{
    struct sockaddr_iucv listensock;

/* establish an endpoint for connection */
    if((*sock = socket(AF_IUCV, SOCK_STREAM, 0)) < 0){
        perror("socket");
        exit(1);
    }
/* assign a name to the socket */
    memset(&listensock, 0, sizeof(listensock));
    strncpy(listensock.siucv_nodeid, "        ", 8);
    strncpy(listensock.siucv_userid, "        ", 8);
    strncpy(listensock.siucv_name, "        ", 8);
    listensock.siucv_family = AF_IUCV;
    strncpy(listensock.siucv_name, name, 8);
    strncpy(listensock.siucv_userid, "RDOSCS  ", 8);
/*
    printf("siucv_port = %d\n", listensock.siucv_port);
    printf("siucv_addr = %d\n", listensock.siucv_addr);
    printf("siucv_nodeid = %s\n", listensock.siucv_nodeid);
    printf("siucv_name = %s\n", listensock.siucv_name);
    printf("siucv_userid = %s\n", listensock.siucv_userid);
*/
    if(bind(*sock, (struct sockaddr *) &listensock, sizeof(listensock)) < 0){
        perror("binding stream socket");
        exit(1);
    }

    if(listen (*sock, 0) < 0){
        perror("listen");
        exit(1);
    }
    printf("listening \n");
}

/* read dbid, display address, etc. from Connection Manager */
void
receive_msg(activesock)
int activesock;
{
    unsigned int displen, hostlen, db_id;
    int msg, rc;
    char *hostname, *display;
    char cmd[100];

    if(read(activesock, &hostlen, sizeof(int)) != 4){
        perror("wrong hostlen");
        exit(1);
    }
    hostlen = ntohl(hostlen);
    printf("hostlen = %d\n", hostlen);

    if(read(activesock, &displen, sizeof(int)) != 4){
        perror("wrong displen");
        exit(1);
    }
    displen = ntohl(displen);
    printf("displen %d\n", displen);

    if(read(activesock, &db_id, sizeof(int)) != 4){
        perror("wrong db_id");
        exit(1);
    }
    db_id = ntohl(db_id);

    display = (char *) malloc(displen +1);
    if(read(activesock, display, displen) != displen){
        perror("wrong display addr");
        exit(1);
    }
    display[displen] = '\0';
    hostname = (char *) malloc(hostlen +1);
    if(read(activesock, hostname, hostlen) != hostlen){
        perror("wrong hostname");
        exit(1);
    }
    hostname[hostlen] = '\0';

    printf("received from Connection Manager:\n");
    printf("\tdb_id: %d\n", db_id);
    printf("\tdisplay: %s\n", display);
    printf("\thostname: %s\n", hostname);
    sprintf(cmd, "hepmain -display %s\n", display);
    printf("cmd = %s\n", cmd);

    system("mapmem");
    system("progmap");

    rc = system(cmd);
    fprintf(stdvrr, " finished rc = %d\n", rc);
    /* tell Connection Manager that we're finished */
    msg = CLOSE;
    if(write(activesock, &msg, sizeof(int)) != 4){
        perror("writing sock");
/*      exit(1); */
    }
    fprintf(stdvrr, "closing sock\n");
```

```
    close(activesock);
}

/* accept(s... and prints a message when a connection arrives */
int
do_accept(sock)
int sock;
{
    struct sockaddr_iucv  active;
    int activesock, namelen;
    struct hostent *hp;
    char buf[1024];

    namelen = sizeof(active);
    if((activesock = accept(sock, &active, &namelen)) < 0){
        perror("accept");
        exit(1);
    }

    else {
/*      if((hp = gethostbyaddr((char *) &active.sin_addr,
                               sizeof(struct in_addr),
                               AF_INET)) == NULL){
            perror("gethostbyaddr");
            exit(1);
 }*/
    }
    printf("[connection accepted]\n");

    fflush(stdout);
    return(activesock);
}

main(argc, argv)
int argc;
char **argv;
{
    int sock, asock;

    make_listener(&sock, argv[1]);
    while(1){
        asock = do_accept(sock);
        receive_msg(asock);
    }
}
```

CM1.C

```
/*
 * cm.c - Connection Manager main loop
 */
/*#define DEBUG */
#include <manifest.h>
#include <bsdtocms.h>
#include <time.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>

#include "cm.h"

/* maximum # of characters in display address, hostname */
#define MAXDISP 24
#define MAXHOST 80

extern void log_connection();
extern void update_cm_log();
extern IVP_PTR freelist;
extern void send_message();
extern char *ivpuserids[];
extern char *ivpnames[];

/* array with data for each IVP */
IVP_DATA ivps[NUMIVPS];

/* global log-file fd */
FILE *fp;
FILE *diagfile;

/*
 * initialize logging structure
 * ivps are listening at consecutive ports >= 4124
 */
void
init_ivps()
{
    int i, nextivp;

    nextivp = 0;
    freelist = &ivps[nextivp];

    for(i = 0; i < NUMIVPS; i++){
        ivps[i].num = i;
        ivps[i].startersock = INVALID;
        ivps[i].name = ivpnames[i];
        ivps[i].userid = ivpuserids[i];
        ivps[i].state = LOGGEDOFF;
        ivps[i].db_id = 0;
        ivps[i].displayaddr = NULL;
        ivps[i].hostname = NULL;
        ivps[i].free = &ivps[i+1];
    }
    ivps[NUMIVPS-1].free = NULL;
#ifdef DEBUG
    printf("In init_ivps()\n");
    dump_ivps();
    dump_freelist();
#endif
}

main(argc, argv)
int argc;
char **argv;
{
    int maxsock, newsock, listensock, ivpsock;
    int i, j, db_id, ivpnum, data;
    char display[MAXDISP], hostname[MAXHOST];
    fd_set *readfds, *writefds;
    time_t ltime;

    readfds = (fd_set *) malloc(sizeof(fd_set));
/* open cm log file and write first entry */
    open_cm_log();
    open_diag_file();
    update_cm_log();
/* initialize ivp data structures */
    init_ivps();

/* setup listening socket */
    make_listener(&listensock);
    maxsock = listensock;
    listen(listensock, 5);

/* main loop */
    while(1){
        FD_ZERO(readfds);

        printf("fd (%08x) -", readfds);
        for(j = 0; j < 7; j ++){
            printf(" %08x", (*readfds).fds_bits[j]);
        }
        printf("\n");

        FD_SET(listensock, readfds);
        for (i = 0; i < NUMIVPS; i++) {
/*
            if (ivps[i].startersock != INVALID) {
                if(ivps[i].startersock > maxsock){
                    maxsock = ivps[i].startersock;
                }
                FD_SET(ivps[i].startersock, readfds);
            }
*/
            if (ivps[i].ivpsock != INVALID) {
                if(ivps[i].ivpsock > maxsock){
                    maxsock = ivps[i].ivpsock;
                }
                FD_SET(ivps[i].ivpsock, readfds);
            }
        }

        printf("fd (%08x) -", readfds);
        for(j = 0; j < 7; j ++){
            printf(" %08x", (*readfds).fds_bits[j]);
        }
        printf("\n");

        j=select(maxsock + 1, readfds, NULL, NULL, NULL);

        printf("select() rc=%d\n", j);
        printf("fd (%08x) -", readfds);
        for(j = 0; j < 7; j ++){
            printf(" %08x", (*readfds).fds_bits[j]);
```

```
        }
        printf("\n");

        /* accept incoming connection from starter */
        if (FD_ISSET(listensock, readfds)) {
            newsock = accept(listensock, 0, 0);
            if (newsock < 0)
                perror("accept");
            if (newsock > maxsock)
                maxsock = newsock;
            if(!authorize(newsock)){
                close(newsock);
                maxsock--;
            }
            else {
                receive_msg_from_starter(newsock, db_id,
                                         display, hostname);
                /* connect to IVP */
                                printf("freelist = %x\n", freelist);
                                if(freelist != NULL){
                                    ivpsock = open_ivp(&ivpnum);
                                    ivps[ivpnum].startersock = newsock;
                                    if(ivpsock > 0){
                                                send_message(ivpsock, db_id, display, ho
stname);

                                                /* log connection -- map IVP to starter
*/
 \
                                                log_connection(newsock, ivpsock, ivpnum,

                                                               db_id, display, hostname)
}
                \
                                                fprintf(fp,
                                                        "connection accepted from %s\n",
 hostname);
                                                fprintf(diagfile,
                                                        "connection accepted from %s\n",
 hostname);


                                                time(&ltime);
                                                fprintf(fp, "on %s", ctime(&ltime));
                                                fprintf(diagfile, "on %s", ctime(&ltime)
);
                                                update_cm_log();
                                    }
                                }
                                else {
                                    fprintf(diagfile,
                                            "No ivp avail for %s\n", hostname);
                                    /* do the right thing here */
                                    /*                  close( ) ; */
                                }
            }
        } else {
            for (i = 0; i < NUMIVPS; i++) {
                if(ivps[i].ivpsock != INVALID){
                    newsock = ivps[i].ivpsock;
                    if (FD_ISSET(newsock, readfds)) {
                        printf("reading from sock %d\n", newsock);
                        if(read(newsock, &data, sizeof(int)) < 0){
                            perror("Read");
                            exit(1);
                        }
                        else{
                            /* close connection to ivp, unlog */
                                printf("closing sock to ivp %d\n", i);
                                fprintf(diagfile,
                                    "closing sock to ivp %d\n", i);
                                unlog_connection(i);
                                add_to_freelist(&ivps[i]);
                                fprintf(fp, "disconnected from %s\n",\
                                        hostname);
                                fprintf(diagfile, "disconnected from %s\n",\
                                        hostname);
                                time(&ltime);
                                fprintf(fp, "at %s", ctime(&ltime));
                                fprintf(diagfile, "at %s", ctime(&ltime));
                                update_cm_log();
                                close(newsock);
                        }
                    }
                } /* if */
            }/* for */
        }
    }
}
```

```
/*
 * cm.h - data structs for
 * Connection Manager
 *
 * VM prototype
 *
 * Lawrence Berkeley Laboratory
 * 1991
 */

/* port the Connection Manager listen()s at */
#define PORT            6123

#define NUMIVPS          3

#define INVALID          0

/* IVP status */
#define LOGGEDOFF        0
#define DISCONNECTED     1
#define TRYING           2
#define CONNECTED        3

typedef struct ivp_data {
    int num;             /* IVP number - same as index in array 1 */
    int starterSock;     /* socket connected to a starter */
    int ivpsock;         /* socket connected to ivp */
    char *name;          /* name process is bound to */
    char *userid;        /* userid process is bound to */
    int state;           /* LOGGEDOFF, DISCONN... etc */
    int db_id;           /* database identifier */
    char *displayaddr;   /* X server identifier */
    char *hostname;      /* X server hostname */
    struct ivp_data *free;      /* next IVP in the free list */
} IVP_DATA, *IVP_PTR;

typedef struct {
    int db_id;
    char display[19];
    char hostname[33];
} STARTER_DATA;

extern void make_listener();
extern int open_ivp();
extern int authorize();
```

cmauth.c

```c
/* cmauth.c
 * verify that connecting site is authorized
 */
#include <sllglue.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <stdio.h>

/* authorized sites */
#define LBL0 "131.243."
#define LBL1 "128.3."
#define SLAC0 "134.79."
#define SLAC1 "36.95.0."
#define SSC0 "134.3."
#define SSC1 "143.202."

char *peeraddr;

int
authorize(sock)
int sock;
{
    struct sockaddr_in peer;
    int length;

    length = sizeof(peer);
    peeraddr = (char *) malloc(16);
    getpeername(sock, &peer, &length);
    peeraddr = inet_ntoa(peer.sin_addr);

    printf("checking authorization for site %s\n", peeraddr);
/*  if(peeraddr[0] != '1'){
        log_reject(peeraddr);
        return(0);
    }
*/
    if(strncmp(peeraddr, LBL0, 8) == 0)
        return(1);
    else if(strncmp(peeraddr, LBL1, 6) == 0)
        return(1);
    else if(strncmp(peeraddr, SLAC0, 7) == 0)
        return(1);
    else if(strncmp(peeraddr, SLAC1, 8) == 0)
        return(1);
    else if(strncmp(peeraddr, SSC0, 6) == 0)
        return(1);
    else if(strncmp(peeraddr, SSC1, 8) == 0)
        return(1);
    else {
        log_reject(peeraddr);
        return(0);
    }
}
```

cmlog.c

```c
/*
 * cmlog.c - logging module for Connection Manager
 *
 */
/*
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
*/
#include <time.h>
#include <stdio.h>

#include "cm.h"

/* global variables */
extern IVP_DATA ivps[];
extern FILE *fp;
extern FILE *diagfile;

/* open log file */
open_cm_log()
{
    time_t ltime;
    time(&ltime);
    if((fp = fopen("cm log a", "w")) == NULL)
        perror("can't open log file");
        exit(1);
    }
    fprintf(fp, "Log opened on %s", ctime(&ltime));
}

/* open diagnostics file */
open_diag_file()
{
    if((diagfile = fopen("cm out a", "w")) == NULL)
        perror("can't open diagnostics file ");
        exit(1);
    }
}

/* write new entry in log file */
void
update_cm_log()
{
    int i;
    fprintf(fp, "\n*****************************************************\
*****************\n");
    for(i = 0; i < NUMIVPS; i++) {
        fprintf(fp, "-----------------------------------------------------\
------------------\n\n");
        fprintf(fp, "IVPNUM: %d\n", i);
        fprintf(fp, "\tSTARTERHOST %s\n", (ivps[i].hostname==NULL?"NULL"\
: ivps[i].hostname));
        fprintf(fp, "\tSTARTERSOCK %d\n", ivps[i].startersock);
        fprintf(fp, "\tDISPLAYADDR %s\n", ivps[i].displayaddr);
        fprintf(fp, "\tDBID %d\n", ivps[i].db_id);
        fprintf(fp, "\tIVPNAME %s \n", ivps[i].name);
        fprintf(fp, "\tUSERID %s\n", ivps[i].userid);
    }
    fprintf(fp, "\n*****************************************************\
*****************\n\n");
}
```

```c
/* keep track of rejected connections */
void
log_reject(addr)
char *addr;
{
    fprintf(fp, "Connection from %s rejected.\n", addr);
}



/*
 * logs starter-CM connection in data structure
 */
void
log_connection(startersock, ivpsock, ivpnum, db_id, dispaddr, hostname)
int startersock, ivpsock, ivpnum, db_id;
char *dispaddr, *hostname;
{
    ivps[ivpnum].startersock = startersock;
    ivps[ivpnum].state = CONNECTED;
    ivps[ivpnum].db_id = db_id;

    ivps[ivpnum].displayaddr = (char *) malloc(strlen(dispaddr) + 1);
    strcpy(ivps[ivpnum].displayaddr, dispaddr);
    ivps[ivpnum].hostname = (char *) malloc(strlen(hostname) +1);
    strcpy(ivps[ivpnum].hostname, hostname);
    printf("disp = %s\n", ivps[ivpnum].displayaddr);
    printf("host = %s\n", ivps[ivpnum].hostname);
}

/*
 * un-logs starter-CM connection for ivps[i]
 */
unlog_connection(i)
int i;
{
    ivps[i].displayaddr = (char *) NULL;
    ivps[i].hostname = (char *) NULL;
    ivps[i].startersock = INVALID;
    ivps[i].db_id = 0;

}
```

```c
/*
 * cmtcp.c
 *      reads and writes messages on the tcp connections to
 *      setup or end connections
 */
/* #define DEBUG*/
#include <allglue.h>
#include <bedtocms.h>
#include <sys/types.h>
#include <string.h>
#include <sys/socket.h>
#include <sslucv.h>
#include <netinet/in.h>
#include <netdb.h>
#include <signal.h>
#include <stdio.h>
#include <errno.h>
#include "cm.h"

extern FILE *diagfile;
extern IVP_DATA ivps[];

/* list of free IVPs */
extern IVP_PTR freelist;
extern char *peeraddr;

char *ivpuserids[] = {"RDOIVP01", "RDOIVP02", "RDOIVP03"};
char *ivpnames[] = {"Rdoivp01", "Rdoivp02", "Rdoivp03"};

/*
 * creates an AF_INET socket listening for connections from starters
 * at port 6123
 */
void
make_listener(listensock)
int *listensock;
{
    struct sockaddr_in server;
    struct hostent *hp;

    *listensock = socket(AF_INET, SOCK_STREAM, 0);
    if (*listensock < 0) {
        perror("opening stream socket");
        close(*listensock);
        exit(0);
    }

    setsockopt(*listensock, SOL_SOCKET, SO_REUSEADDR, 0, 0);
    server.sin_family = AF_INET;
    server.sin_addr.s_addr = INADDR_ANY;
    server.sin_port = htons(PORT);


    if (bind(*listensock, &server, sizeof(server)) < 0) {
        perror("binding stream socket");
        exit(1);
    }
    printf("connection manager listening at port [%d]\n", PORT);
    fprintf(diagfile,
        "connection manager listening at port [%d]\n", PORT);
}

/*
 * initiates a connection to a waiting IVP
 * returns local fd
 */
int
open_ivp(ivpnum)
int *ivpnum;
{
    int sock, length, done = 0;
    struct sockaddr_iucv server;
    char buf[1024];
    IVP_DATA ivp;

    ivp = *freelist;
#ifdef DEBUG
    printf("freelist->num = %d\n", freelist->num);
    printf("freelist->userid = %s\n", freelist->userid);
#endif

    memset(&server, 0, sizeof(server));
    server.slucv_family = AF_IUCV;
    strncpy(server.slucv_nodeid, "        ", 8);
    strncpy(server.slucv_userid, "        ", 8);
    strncpy(server.slucv_name, "        ", 8);
    /* create socket */
    /* try to connect to one of the IVPs */
    while(!done){
       if((sock = socket(AF_IUCV, SOCK_STREAM, 0)) < 0){
            perror("opening stream socket");
            exit(1);
        }

        *ivpnum = ivp.num;
        printf("ivpnum = %d\n", *ivpnum);
        strncpy(server.slucv_name, ivp.name, 8);
        strncpy(server.slucv_userid, ivp.userid, 8);
/*
        printf("slucv_port = %d\n", server.slucv_port);
        printf("slucv_addr = %d\n", server.slucv_addr);
        printf("slucv_nodeid = %s\n", server.slucv_nodeid);
*/
        printf("slucv_name = %s\n", server.slucv_name);
        printf("slucv_userid = %s\n", server.slucv_userid);

        fprintf(diagfile, "slucv_port = %d\n", server.slucv_port);
        fprintf(diagfile, "slucv_addr = %d\n", server.slucv_addr);
        fprintf(diagfile, "slucv_nodeid = %s\n", server.slucv_nodeid);
        fprintf(diagfile, "slucv_name = %s\n", server.slucv_name);
        fprintf(diagfile, "slucv_userid = %s\n", server.slucv_userid);
        ivps[*ivpnum].ivpsock = sock;
        if(connect(sock, (struct sockaddr *)&server, sizeof(server))< 0){
            perror("connecting stream socket");
            fprintf(diagfile, "error connecting stream socket to\
ivp %d\n", *ivpnum);
            if(ivp.free != NULL){ /* try next IVP in freelist */
                ivp = *ivp.free;
            }
            else { /* no IVPs available */
                printf("no more IVPs\n");
                /* tell starter no more IVPs left */
                fprintf(diagfile, "no more IVPs\n");
                return(0);
            }
        }
        else { /* connection accepted */
            done = 1;
```

```
        }
    }
    delete_from_freelist(slvp);
    dump_freelist();
    fdump_freelist(diagfile);
    printf("--> connection accepted\n");

    return(sock);
}

void
send_message(sock, db_id, display, hostname)
int sock, db_id;
char *display, *hostname;
{
    unsigned int displen, hostlen;
    hostlen = strlen(hostname);
    displen = strlen(display);
    hostlen = htonl(hostlen);
    if(write(sock, &hostlen, sizeof(int)) != 4){
        perror("wrong hostlen");
        fprintf(diagfile, "write: wrong hostlen");
        exit(1);
    }
    displen = htonl(displen);
    if(write(sock, &displen, sizeof(int)) != 4){
        perror("wrong displen");
        fprintf(diagfile, "write: wrong displen");
        exit(1);
    }
    db_id = htonl(db_id);
    if(write(sock, &db_id, sizeof(int)) != 4){
        perror("wrong db_id");
        fprintf(diagfile, "write: wrong db_id");
        exit(1);
    }
    if(write(sock, display, displen) != displen){
        perror("wrong dispaddr");
        fprintf(diagfile, "write: wrong dispaddr");
        exit(1);
    }
    if(write(sock, hostname, hostlen) != hostlen){
        perror("wrong hostname");
        fprintf(diagfile, "wrong hostname");
        exit(1);
    }
}

/*
 * read message header from connecting starter
 *
 * message format:
 *
 * int displen
 * int hostnamelen
 * int db_id
 * char displayaddr[displen]
 * char hostname[hostnamelen]
 */
void
receive_msg_from_starter(sock, db_id, display, hostname)
int sock;
unsigned int *db_id;
char *display, *hostname;
{
    unsigned int displen, hostnamelen, len;
    char *disbuf, *hostbuf, *addr;
    struct sockaddr_in client;
    addr = (char *) malloc(14);

    if(read(sock, &displen, sizeof(int)) < 0 ){
        perror("displen incorrect");
        close(sock);
        exit(1);
    }
    displen = ntohl(displen);

    if(read(sock, &hostnamelen, sizeof(int)) < 0){
        perror("hostnamelen incorrect");
        close(sock);
        exit(1);
    }
    hostnamelen = ntohl(hostnamelen);

    /* allocate space for strings */
    disbuf = (char *) malloc(displen +1);
    hostbuf = (char *) malloc(hostnamelen +1);

    /* read message from starter */
    if(read(sock, db_id, sizeof(int)) < 0){
        perror("db_id incorrect");
        close(sock);
        exit(1);
    }
    *db_id = ntohl(*db_id);
    if(read(sock, disbuf, displen) != displen){
        perror("getting display address");
        exit(1);
    }
    ASCclp(disbuf, displen+1);
    if(read(sock, hostbuf, hostnamelen) != hostnamelen){
        perror("getting hostname");
        exit(1);
    }
    ASCclp(hostbuf, hostnamelen+1);
    printf("received from starter:\n");
    printf("db_id: %d\n", *db_id);
    printf("display: %s\n", disbuf);
    printf("hostname: %s\n", hostbuf);
    sprintf(display, "%s%s", addr, disbuf);
    printf("in cmtcp peeraddr = %s\n", peeraddr);
    sprintf(display, "%s%s", peeraddr, disbuf);
    printf("*** display = %s\n", display);
    strcpy(hostname, hostbuf);
    close(sock);
}
```

```c
/*
 * manage.c - manages IVPs for Connection Manager
 *
 * free ivps are kept in a FIFO queue (implemented as a linked list)
 */

   #define DEBUG
#include <stdio.h>
#include "cm.h"

/* global variables */
extern IVP_DATA ivps[];

IVP_PTR freelist;  /* list of free ivps */

/* debugging tool - print ivp array */
void
dump_ivps()
{
    int i;

    printf("****dumping ivps****\n");
    for (i = 0; i < NUMIVPS; i++){
        printf("ivp num: %d\n", ivps[i].num);
        printf("\tstartersock: %d\n", ivps[i].startersock);
        printf("\tivpsock: %d\n", ivps[i].ivpsock);
        printf("\tstate: %d\n", ivps[i].state);
        printf("\tdb_id: %d\n", ivps[i].db_id);
        printf("\tdisplay addr: %s\n", ivps[i].displayaddr);
        printf("\thostname: , %s\n", ivps[i].hostname);
        printf("\tfree: %x\n", ivps[i].free);
    }
    printf("****done****\n");
}

/* print list of free ivps */
void
dump_freelist()
{
    IVP_PTR ptr;
    printf("####dumping free ivp list####\n");
    printf("freelist = %x\n", freelist);
    for(ptr = freelist; ptr != NULL; ptr = ptr->free){
        printf("ivp num: %d\n", ptr->num);
        printf("ivp userid: %s\n", ptr->userid);
        printf("ivp: %x\n", *ptr);
        printf("&ivp %x\n", ptr);
    }
    printf("####done####\n");
}

/* print list of free ivps */
void
fdump_freelist(dfp)
FILE *dfp;
{
    IVP_PTR ptr;
    fprintf(dfp, "####dumping free ivp list####\n");
    fprintf(dfp, "freelist = %x\n", freelist);
    for(ptr = freelist; ptr != NULL; ptr = ptr->free){
        fprintf(dfp, "ivp num: %d\n", ptr->num);
        fprintf(dfp, "ivp userid: %s\n", ptr->userid);
        fprintf(dfp, "ivp: %x\n", *ptr);
        fprintf(dfp, "&ivp %x\n", ptr);
    }
    fprintf(dfp, "####done####\n");
}

/* add free ivp to freelist
 * freelist is sorted by ivp index num
 */
void
add_to_freelist(ivp)
IVP_PTR ivp;
{
    IVP_PTR  rptr, ptr;

    ivp->ivpsock = INVALID;
    if(freelist == NULL){
        freelist = ivp;
        return;
    }
    for(rptr = NULL, ptr = freelist; ptr != NULL; rptr = ptr, ptr = ptr->free){
        if(ivp->num < ptr->num){
            /* insert */
            if(rptr == NULL){
                /* insert at front of list */
                ivp->free = ptr;
                freelist = ivp;
#ifdef DEBUG
    printf("just added ivp head of to freelist \n");
    printf("num = %d\t userid = %d\n", ivp->num, ivp->userid);
    dump_freelist();
#endif
                return;
            }
            else {
                rptr->free = ivp;
                ivp->free = ptr;
#ifdef DEBUG
    printf("just added ivp to freelist \n");
    printf("num = %d\t userid = %d\n", ivp->num, ivp->userid);
    dump_freelist();
#endif
                return;
            }
        }
    }
}

/* removes ivp from head of free list */
void
delete_from_freelist(ivp)
IVP_PTR ivp;
{
    IVP_PTR rptr, ptr;
    if(freelist == NULL){
        fprintf(stderr, "Error: cannot delete ivp from NULL freelist\n");
    }
    else if(freelist->free == NULL){
        freelist = NULL;
    }
    else { /* freelist has at least 2 elements */
        for(rptr = NULL, ptr = freelist; ptr != NULL;
            rptr = ptr, ptr = ptr->free){
            if(ivp->num == ptr->num){
                /* delete */
                if(rptr == NULL) { /* ivp to be deleted is head  */
```

```
}
    freelist = ptr->free;
    lvp->free = NULL;
    return;
}
else { /* rptr != NULL */
    rptr->free = NULL;
    freelist = ptr;
    return;
}
```

starter.c

```
/*
 * starter.c - UNIX, X11/Xaw version
 *
 * This client program creates a socket and initiates a connection to a
 * hardwired endpoint. It sends the display address of an X11
 * server to the ConnectionManager (CM) process
 * on the other side of the wire.
 *
 *
 * 09/26/90
 * 10/21/91 revised
 */

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>

/* db host info */
#define HOST "slacvm.slac.stanford.edu"
#define DBID 666
#define PORT 6123
#define DISPLAY ":0.0"

typedef struct {
    int displaylen;
    int hostlen;
    int db_id;
    char *displayaddr;
    char *hostname;
} STARTER_MSG;

extern void draw_error_dialog();

void init_connection()
{
    int sock, maglen, len= 80;
    struct sockaddr_in server;
    struct hostent *hp, *gethostbyname();
    STARTER_MSG msg;
    char buf[1024];

    msg.displaylen = htonl(strlen(DISPLAY) + 1);
    msg.hostlen = htonl(len);
    msg.db_id = htonl(DBID);
    msg.displayaddr = (char *) malloc(strlen(DISPLAY) +1);
    msg.hostname = (char *) malloc(len);
    strcpy(msg.displayaddr, DISPLAY);
    gethostname(msg.hostname, len);

/* create starter-side socket */
    if((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0){
        perror("opening stream socket");
        exit(1);
    }

    /* connect socket to endpoint specified in command line */
    server.sin_family = AF_INET;
    if((hp = gethostbyname(HOST)) == 0){
        fprintf(stderr, "%s: unknown host\n", HOST);
        exit(2);
    }
    bcopy(hp->h_addr, &server.sin_addr, hp->h_length);
    server.sin_port = htons(PORT);

    printf("Client trying to connect to server at [ %s ]\n",
           inet_ntoa(server.sin_addr));

/* connect to server-side socket */
    if(connect(sock, &server, sizeof(server))< 0){
        perror("connecting stream socket");
        draw_error_dialog("Starter failed to connect to DB host\n");
    }
    else {
        printf("--> connection accepted on fd %d from [ %s ]\n",
                sock, inet_ntoa(server.sin_addr));

        printf("now sending ... ");
        maglen = msg.displaylen + msg.hostlen + 12;

        if(write(sock, &msg.displaylen, 4) != 4){
            perror("writing stream message 1");
            exit(1);
        }

        if(write(sock, &msg.hostlen, 4) != 4){
            perror("writing stream message 2");
            exit(1);
        }
        if(write(sock, &msg.db_id, 4) != 4){
            perror("writing stream message 3");
            exit(1);
        }
        if(write(sock, msg.displayaddr, msg.displaylen) != msg.displaylen){
            perror("writing stream message 4");
            exit(1);
        }
        if(write(sock, msg.hostname, msg.hostlen) != msg.hostlen){
            perror("writing stream message 5");
            exit(1);
        }

        printf("done.\n");
        close(sock);
    }
}
```

Xstarter.c

```
/*
 * Xstarter.c -  UNIX, X11 version of starter.c
 *
 * This module contains the main program.
 *
 * usage: Xstart
 *
 * starter.c is the tcp/ip module.
 *
 * 09/26/90
 *
 */

#include <stdio.h>

#include <X11/Xatom.h>
#include <X11/Intrinsic.h>
#include <X11/StringDefs.h>

#include <X11/Xaw/Command.h>
#include <X11/Xaw/Form.h>

Widget toplevel;
Widget form;
Widget button;
Display *display;
Widget popup, dialog;

extern void init_connection();
extern Pixmap make_icon();
extern void create_dialog();

Pixmap bm;

void
make_window(w)
Widget w;
{
    form = XtVaCreateManagedWidget("", formWidgetClass, toplevel,
                                   NULL);

    bm = make_icon();
    button = XtVaCreateManagedWidget("pressme",
                                     commandWidgetClass, form,
                                     XtNbitmap, bm,
/*                                   XtNheight, 100,
                                     XtNwidth, 100,

*/
/*                                   XtNforeground,
                                         (XtArgVal) &values.foreground,
                                     XtNbackground,
                                         (XtArgVal) &values.background,
*/
                                     XtNfromVert, NULL,
                                     XtNfromHoriz, NULL,
                                     NULL);
    XtAddCallback(button, XtNcallback, init_connection, (XtPointer) NULL);
}

main(argc, argv)
int argc;
char **argv;
{
    XtAppContext app_con;

    toplevel = XtAppInitialize(&app_con, "Xapim", NULL, 0, &argc,
                               argv, NULL, NULL, 0);
    display = XtDisplay(toplevel);

    make_window(toplevel);
    create_dialog();
    XtRealizeWidget(toplevel);
    XtAppMainLoop(app_con);
}
```

Xstarter.c

```
/*
 * Xstarter.c -  UNIX, X11 version of starter.c
 *
 * This module contains the main program.
 *
 * usage: Xstart
 *
 * starter.c is the tcp/ip module.
 *
 * 09/26/90
 *
 */

#include <stdio.h>

#include <X11/Xatom.h>
#include <X11/IntrinsicP.h>
#include <X11/StringDefs.h>

#include <X11/Xaw/Command.h>
#include <X11/Xaw/Form.h>

Widget toplevel;
Widget form;
Widget button;
Display *display;
Widget popup, dialog;

extern void init_connection();
extern Pixmap make_icon();
extern void create_dialog();

Pixmap bm;

void
make_window(w)
Widget w;
{
    form = XtVaCreateManagedWidget("", formWidgetClass, toplevel,
                                   NULL);
    bm = make_icon();
    button = XtVaCreateManagedWidget("pressme",
                                     commandWidgetClass, form,
                                     XtNbitmap, bm,
/*                                   XtNheight, 100,
                                     XtNwidth, 100,
*/
/*                                   XtNforeground,
                                         (XtArgVal) &values.foreground,
                                     XtNbackground,
                                         (XtArgVal) &values.background,
*/
                                     XtNfromVert, NULL,
                                     XtNfromHoriz, NULL,
                                     NULL);
    XtAddCallback(button, XtNcallback, init_connection, (XtPointer) NULL);
}

main(argc, argv)
int argc;
char **argv;
{
    XtAppContext app_con;

    toplevel = XtAppInitialize(&app_con, "Xapire", NULL, 0, &argc,
                               argv, NULL, NULL, 0);
    display = XtDisplay(toplevel);

    make_window(toplevel);
    create_dialog();
    XtRealizeWidget(toplevel);
    XtAppMainLoop(app_con);
}
```

```
/* dialog.c */

#include <stdio.h>

#include <X11/Xatom.h>
#include <X11/Intrinsic.h>
#include <X11/StringDefs.h>
#include <X11/Shell.h>

#include <X11/Xaw/Box.h>
#include <X11/Xaw/Cardinals.h>
#include <X11/Xaw/Command.h>
#include <X11/Xaw/Dialog.h>

extern Display *display;
extern Widget popup, dialog;
extern Widget button;

void
okay()
{
    XtUnrealizeWidget(popup);
}


void
create_dialog()
{

    Arg         arg[5];
    Position    x, y;
    Dimension   width, height;
    Cardinal    n;

    n = 0;
    XtSetArg(arg[0], XtNwidth, &width); n++;
    XtSetArg(arg[1], XtNheight, &height); n++;
    XtGetValues(button, arg, n);
    XtTranslateCoords(button, (Position) (width / 2), (Position) (height / 2),
                      &x, &y);

    n = 0;
    XtSetArg(arg[n], XtNx, x);                          n++;
    XtSetArg(arg[n], XtNy, y);                          n++;


    popup = XtCreatePopupShell("prompt",  transientShellWidgetClass, button,
                               arg, n);
    dialog = XtCreateManagedWidget("dialog", dialogWidgetClass, popup,NULL, 0);
    XawDialogAddButton(dialog, "ok", okay, (XtPointer) dialog);
    XtVaSetValues(dialog, XtNlabel,
                  "Connection Refused; Try again later", NULL);

}

void
draw_error_dialog(str)
char *str;
{
    XtPopup(popup, XtGrabNone);
    XtRealizeWidget(popup);
}
```

```
/* drawbit.c */
#include <X11/Intrinsic.h>
#include <X11/Xlib.h>
#include <X11/StringDefs.h>
#include <X11/Xaw/Label.h>
#include <X11/Xaw/Form.h>
#include "hepstart.h"

extern Widget toplevel;

Pixmap
make_icon()
{
    XGCValues values;
    Pixmap bm;
    int width, height;
    char *bits;

    width = hepstart_width; height = hepstart_height; bits = hepstart_bits;

    bm = XCreateBitmapFromData(XtDisplay(toplevel),
                               RootWindowOfScreen(XtScreen(toplevel)), bits,
                               width, height);
    return(bm);
}
```

hepstart.h

```
#define hepstart_width 32
#define hepstart_height 32
static char hepstart_bits[] = {
   0x07, 0x00, 0x00, 0xe0, 0x0f, 0x00, 0x00, 0xf0, 0x1e, 0x00, 0x00, 0x18,
   0xff, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x91, 0xf1, 0x00,
   0x00, 0x92, 0x90, 0x00, 0x00, 0x9e, 0xf1, 0x00, 0x00, 0x91, 0x10, 0x00,
   0x00, 0x92, 0x13, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfc, 0xff, 0xff, 0x1f,
   0x00, 0x0e, 0x70, 0x00, 0x00, 0x1c, 0x38, 0x00, 0x04, 0x38, 0x1c, 0x00,
   0x3e, 0xe0, 0x07, 0x00, 0xef, 0xff, 0xff, 0x3f, 0x70, 0xc0, 0x03, 0x20,
   0x70, 0xe0, 0x07, 0x20, 0x26, 0x70, 0x0e, 0x20, 0x3e, 0x38, 0x1c, 0x20,
   0x2f, 0x1c, 0x38, 0x20, 0x20, 0x0e, 0x70, 0x20, 0x20, 0xb7, 0xed, 0x20,
   0xe6, 0xb7, 0xed, 0x21, 0xfe, 0xb7, 0xed, 0x27, 0xcf, 0xb6, 0x6d, 0x27,
   0x60, 0xb6, 0xed, 0x3f, 0x70, 0xb6, 0x6d, 0x00, 0x38, 0xff, 0xff, 0x1c,
   0x38, 0xff, 0xff, 0x1c, 0x00, 0x00, 0x00, 0x00};
```

Makefile

```
#
#       Makefile for starter.c
#
EXEC=   Xstart
#
#       DEFINES
#
CFLAGS= -g
#
#
#
LIBS=   -lXaw -lXmu -lXt -lXext -lX11
#
#
#
OBJECTS=        Xstarter.o starter.o drawbit.o dialog.o
#
#
$(EXEC):        $(OBJECTS)
                cc  $(OBJECTS) -o $(EXEC) $(LIBS)
#
#
```

```
/* Mac interface includes */
#include <types.h>
#include <errors.h>
#include <files.h>
#include <fonts.h>
#include <memory.h>
#include <packages.h>
#include <quickdraw.h>
#include <resources.h>
#include <sysequ.h>
#include <toolutils.h>
#include <windows.h>
#include <dialogs.h>
#include <desk.h>
#include <CursorCtl.h>

#include <stdio.h>

/* TCP/IP includes */
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/errno.h>
#include <netinet/in.h>
#include <netdb.h>
#include <res_async.h>

#include "resources.h"

#define DISPLAY ":0.0"
#define PORT 6123
#define HOST "SLACVM.SLAC.stanford.edu"
#define DBID 32
#define MAXDISPLEN 25

extern acurPtr my_cursors;
extern int errno;
```

```
/* draw dialog */
static void AlertUser( char *msg) {
        paramtext( nil, msg, nil, nil);
        Alert( rUserAlert, nil);
}

void
send_request(arghandle)
Handle arghandle;
{

        int sock, rc;
        unsigned int displaylen, hostlen, db_id;
        struct sockaddr_in server;
        struct hostent *hp, *gethostbyaddr();
        char *hostname, *displayaddr, *localhostname;
        CursHandle *handle;


        displaylen = strlen(DISPLAY) +1;
        hostlen = 80;
        db_id = htonl(DBID);
        displayaddr = (char *) malloc(strlen(DISPLAY) +1);
        hostname = (char *) malloc(hostlen);
        localhostname = (char *) malloc(hostlen);
        strcpy(hostname, HOST);
        strcpy(displayaddr, DISPLAY);

        /* open local socket */
        if((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0){
                AlertUser("error creating socket... quit program?");
                SystemTask();
                /*
                s_perror("opening stream socket");
                fprintf(stderr, "sock rc = %d\n", sock);
                fprintf(stderr, "errno = %d\n", errno);
                */
```

```
        exit(1);
        }

/* lookup db host */
server.sin_family = AF_INET;
if((hp = gethostbyname(HOST)) == 0){
        /* network or nameserver may be down */
        AlertUser("error getting db hostname... quit program?");
        SystemTask();
        exit(2);
        }

bcopy(hp->h_addr, &server.sin_addr, hp->h_length);
server.sin_port = htons(6123);

/* connect to db host */

AlertUser("connecting to host... one moment please...");
SystemTask();
SetCursor(*GetCursor(20011));
SpinCursor(4);

if((rc = (connect(sock, (struct sockaddr *) &server, sizeof(server)))) < 0) {
        AlertUser("error connecting to host... quit program?");
        SystemTask();
        exit(3);
}

/* lookup local host */
gethostname(localhostname, hostlen);

/* get our own display address */
/*      getsockname(sock, (struct sockaddr *) &client, &len) ;*/

/*      disp = (char *)inet_ntoa(client.sin_addr);
        sprintf(displayaddr, "%s:0.2", disp);
        displaylen = htonl(strlen(displayaddr) +1);
*/
```

```
/* send */

if(send(sock, (char *)&displaylen, 4, 0) <0){
        AlertUser("error writing stream message 1...");
        SystemTask();
        exit(1);
}
if(send(sock, (char *)&hostlen, 4, 0) <0){
        AlertUser("error writing stream message 2...");
        SystemTask();
        exit(1);
        }
if(send(sock, (char *)&db_id, 4, 0) <0){
        AlertUser("error writing stream message 3...");
        SystemTask();
        exit(1);
        }
if(send(sock, displayaddr, htonl(displaylen), 0) <0){
        AlertUser("error writing stream message 4...");
        SystemTask();
        exit(1);
        }
if(send(sock, localhostname, htonl(hostlen), 0) <0){
        AlertUser("error writing stream message 5...");
        SystemTask();
        exit(1);
        }
close(sock);
exit(0);
}
```

```
        This file contains resources for the Macintosh interface...

** ************************************************ */

#include "Types.r"
#include "resources.h"
data 'hep1' (128) {
};

data 'hep1' (0, "Owner resource") {
        $"00"                                   /* . */
};

data 'BNDL' (128) {
        $"6865 7031 0000 0001 4652 4546 0000 0000"          /* hep1....FREF.... */
        $"0080 4943 4E23 0000 0080 0080"                    /* .ÃICN#...Ã.Ã */
};

data 'FREF' (128) {
        $"6865 7031 0080 00"                        /* hep1.Ã. */
};

data 'ics#' (128, "APPL ICON") {
        $"C003 FFFF 0DF0 0D70 0DC0 FFFF 0660 E3C0"          /* ¿...¬.¬p¬¿...`.¿ */
        $"FFFF E3C1 E661 2FF1 FFF9 FFFF 7FFE 6FF6"          /* ...¡.a/.....o. */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"          /* ..... */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"          /* ..... */
};

data 'ALRT' (130, purgeable) {
        $"0028 0023 00A5 01E0 0082 4444"                    /* .(.#.•...ÇDD */
};

data 'STR#' (128, purgeable) {
        $"000C 1B4E 6F74 2065 6E6F 7567 6820 6D65"          /* ...Not enough me */
        $"6D6F 7279 2074 6F20 6C61 756E 6368 1844"          /* mory to launch.D */
        $"6972 6563 746F 7279 2064 6F65 7320 6E6F"          /* irectory does no */
        $"7420 6578 6973 741C 566F 6C75 6D65 2028"          /* t exist.Volume ( */
```

```
        $"6469 736B 2920 646F 6573 206E 6F74 2065"      /* disk) does not e */
        $"7869 7374 1D50 6174 6873 206D 7573 7420"      /* xist.Paths must  */
        $"6265 2066 756C 6C79 2071 7561 6C69 6669"      /* be fully qualifi */
        $"6564 1E44 6F63 756D 656E 7420 2866 696C"      /* ed.Document (fil */
        $"6529 2064 6F65 7320 6E6F 7420 6578 6973"      /* e) does not exis */
        $"7421 4170 706C 6963 6174 696F 6E20 2866"      /* t!Application ((f */
        $"696C 6529 2064 6F65 7320 6E6F 7420 6578"      /* ile) does not ex */
        $"6973 741B 436F 756C 6420 6E6F 7420 7265"      /* ist.Could not re */
        $"7365 7420 6861 6E64 6C65 2073 697A 651C"      /* set handle size. */
        $"436F 756C 6420 6E6F 7420 6361 7420 646F"      /* Could not cat do */
        $"6320 746F 2061 7267 6861 6E64 1853 6574"      /* c to arghand.Set */
        $"566F 6C20 6572 726F 724F 7065 6E57 4420"      /* Vol errorOpenWD  */
        $"6572 726F 720E 4765 7446 496E 666F 2065"      /* error.GetFInfo e */
        $"7272 6F72 0C4C 6175 6E63 6820 6572 726F"      /* rror.Launch erro */
        $"720F 4765 7457 4449 6E66 6F20 6572 726F"      /* r.GetWDInfo erro */
        $"72"                                           /* r */
};

data 'CURS' (20010, preload) {
        $"07C0 1F30 3F08 7F04 7F04 FF02 FF02 FFFE"      /* .¿.0?........... */
        $"81FE 81FE 41FC 41FC 21F8 19F0 07C0 0000"      /* Å.Å.A.A.!....¿.. */
        $"07C0 1FF0 3FF8 7FFC 7FFC FFFE FFFE FFFE"      /* .¿..?........... */
        $"FFFE FFFE 7FFC 7FFC 3FF8 1FF0 07C0 0000"      /* ........?....¿.. */
        $"0007 0007"                                    /* .... */
};

data 'CURS' (20011, preload) {
        $"07C0 1FF0 3FF8 5FF4 4FE4 87C2 8382 8102"      /* .¿..?._.O.$¬ÉÇÂ. */
        $"8382 87C2 4FE4 5FF4 3FF8 1FF0 07C0 0000"      /* ÉÇ$¬O._.?....¿.. */
        $"07C0 1FF0 3FF8 7FFC 7FFC FFFE FFFE FFFE"      /* .¿..?........... */
        $"FFFE FFFE 7FFC 7FFC 3FF8 1FF0 07C0 0000"      /* ........?....¿.. */
        $"0007 0007"                                    /* .... */
};

data 'CURS' (20012, preload) {
        $"07C0 19F0 21F8 41FC 41FC 81FE 81FE FFFE"      /* .¿..!.A.A.Å.Å... */
        $"FF02 FF02 7F04 7F04 3F08 1F30 07C0 0000"      /* ........?..0.¿.. */
        $"07C0 1FF0 3FF8 7FFC 7FFC FFFE FFFE FFFE"      /* .¿..?........... */
```

```
        $"FFFE FFFE 7FFC 7FFC 3FF8 1FF0 07C0 0000"          /* .......?...¿.. */
        $"0007 0007"                                        /* .... */
};

data 'CURS' (20013, preload) {
        $"07C0 1830 2008 701C 783C FC7E FEFE FFFE"          /* .¿.0 .p.x<.~.... */
        $"FEFE FC7E 783C 701C 2008 1830 07C0 0000"          /* ...~x<p. ..0.¿.. */
        $"07C0 1FF0 3FF8 7FFC 7FFC FFFE FFFE FFFE"          /* .¿..?........... */
        $"FFFE FFFE 7FFC 7FFC 3FF8 1FF0 07C0 0000"          /* .......?....¿.. */
        $"0007 0007"                                        /* .... */
};

data 'acur' (128) {
        $"0004 0000 4E2A 0000 4E2B 0000 4E2C 0000"          /* ....N*..N+..N,.. */
        $"4E2D 0000"                                        /* N-.. */
};

data 'ICON' (128) {
        $"F000 0003 F800 0007 FC00 000E FFFF FFFF"          /* ................ */
        $"0000 0000 0127 3C00 0124 2400 01E6 3C00"          /* .....'<..$$...<. */
        $"0124 2000 0127 2000 0000 0000 FFFF FFFF"          /* .$ ..' ........ */
        $"00FE 0E00 007F 1C00 603F B806 7C1F 703E"          /* .......`?∏.|.p> */
        $"F40E E02F 041D F020 0439 F820 6470 FC20"          /* ...../... .9. dp.  */
        $"7FFF FFFE F1C0 3FEF 0380 1F20 0700 0FA0"          /* .....¿?..Ã. ...† */
        $"0E00 07E0 1C6D B3F0 386D B1F8 706D B0FC"          /* .....m≥.8m±.pm═. */
        $"E06D B07E C06D BFFF 80FF F81F 00FF F80F"          /* .m═~¿m═.Ã..... */
};

data 'ICN#' (128, "APPL ICON") {
        $"F008 0003 F800 0007 FC00 000E FFFF FFFF"          /* ................ */
        $"0000 0000 0093 9E00 0092 1200 00F3 1E00"          /* .....10..f..... */
        $"0092 1000 0093 9000 0000 0000 FFFF FFFF"          /* .f..lè......... */
        $"00FE 0E00 007F 1C00 603F B806 7C1F 703E"          /* .......`?∏.|.p> */
        $"F40E E02F 041D F020 0439 F820 6470 FC26"          /* ...../... .9. dp.& */
        $"7FFF FFFE F1C0 3F2F 0380 1FA0 0700 0FE0"          /* .....¿?/.Ã.†.... */
        $"0E6D B7E0 1C6D B3F0 386D B1F8 706D B0FC"          /* .m∑..m≥.8m±.pm═. */
        $"E06D B07E C06D BFFF 80FF F81F 00FF F80F"          /* .m═~¿m═.Ã..... */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"          /* ................ */
```

```
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"        /* ................ */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"        /* ................ */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"        /* ................ */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"        /* ................ */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"        /* ................ */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"        /* ................ */
        $"FFFF FFFF FFFF FFFF FFFF FFFF FFFF FFFF"        /* ................ */
};

data 'DITL' (130, purgeable) {
        $"0003 0000 0000 0000 0000 00A0 01D1 0000"        /* ..........†.—... */
        $"0000 0000 000A 0037 005F 01AE 8806 5E30"        /* .......7._..Æà.^0 */
        $"3A20 5E31 0000 0000 000D 000B 002D 002B"        /* : ^1.....¬...-.+ */
        $"A002 0080 0000 0000 0050 014A 0073 0183"        /* †..Ä.....P.J.s.É */
        $"0402 4F4B"                                      /* ..OK */
};
```

```
# File:     hep.make
# Target:   hep
# Sources:  hep.c
# Created:  Wednesday, September 18, 1991 2:26:29 PM


OBJECTS = hep.c.o main.c.o


COptions = -i "{SockIncludes}" -i "{TCPort includes}"

TCPortlibs =    "{TCPort Libraries}"tcp_interface.o ∂
                        "{TCPort Libraries}"udp_interface.o ∂
                        "{TCPort Libraries}"ip_interface.o ∂
                        "{TCPort Libraries}"reslib.o ∂
                        "{TCPort Libraries}"citimlib.o ∂
                        "{TCPort Libraries}"socket.o

hep ƒƒ hep.make hep.r
        Rez hep.r -append -o hep

hep ƒƒ hep.make {OBJECTS}
        Link -w -t APPL -c hep1 -sym on -mf ∂
                {OBJECTS} ∂
                {TCPortlibs} ∂
                "{CLibraries}"CSANELib.o ∂
                "{CLibraries}"Math.o ∂
                #"{CLibraries}"Complex.o ∂
                "{CLibraries}"StdClib.o ∂
                "{CLibraries}"CInterface.o ∂
                "{CLibraries}"CRuntime.o ∂
                "{Libraries}"Interface.o ∂
                "{Libraries}"Runtime.o ∂
                "{Libraries}"ToolLibs.o ∂
                #"{Libraries}"Stubs.o ∂
                -o hep
main.c.o ƒ hep.make main.c
        C -sym on main.c {COptions}
```

```
hep.c.o ƒ hep.make hep.c
        C -sym on hep.c {COptions}
```

```
/* Mac Interface Includes */
#include <types.h>
#include <errors.h>
#include <files.h>
#include <fonts.h>
#include <memory.h>
#include <packages.h>
#include <quickdraw.h>
#include <resources.h>
#include <sysequ.h>
#include <toolutils.h>
#include <windows.h>
#include <dialogs.h>
#include <desk.h>
#include <CursorCtl.h>
#include "resources.h"

void send_request();

/* display a dialog if there is a problem */
static DoAlert(char *diag, char *data) {
        char msg[ 256];

        sprintf( msg, "%s %s", diag, data);
        ParamText( nil, msg, nil, nil);
        Alert( rUserAlert, nil);
        ExitToShell();
}

acurPtr my_cursors;

main() {
        short   refn;
        Handle  arghandle;
        char    appname[ 256];
```

```
/* Initialize the environment (the following are quickdraw routines) */
InitGraf(( Ptr) &qd.thePort);
InitFonts();
InitWindows();
InitDialogs( nil);
InitCursor();
InitCursorCtl(&my_cursors);                    /* so I can use SpinCursor, etc */

/* store name of this application */
GetAppParms( appname, &refn, &arghandle);
ParamText( appname, nil, nil, nil);

/* send request */
send_request(/* arghandle*/);

/* die */
ExitToShell();
}
```