STAR Offline Computing and Software Training School

Lawrence Berkeley National Laboratory

January 9 - 12, 1998

‘‘Introduction and Overview’’

Lanny Ray, The University of Texas at Austin

1. The Star Computing Challenge
2. Organization and People
3. Computing Environment
4. Computing Tools
5. STAF
6. Offline Software Design
7. User Programming Guidelines
8. STAR Web Documents
9. Steps in Typical SAS Projects

1. The STAR Computing Challenge

- The STAR experiment will generate over 200 TB of raw data each year, plus simulation data, reconstructed event data summaries (DST), reduced event summaries (micro-DST) and physics analysis data. Over the lifetime of STAR this amounts to many PetaBytes or many millions of GBytes of data.

- The STAR simulation, event reconstruction and physics analysis will demand enormous cpu capacity, of order 200 GFlops or enough to keep about 2000 Pentium PRO 200 MHz processors going full blast 24 hrs a day, 365 days a year!

- The STAR infrastructure and analysis software is being developed by ~50 part-time people at many STAR institutions and must eventually be made available to the entire 400+ member collaboration. Turnover rate is high.

- STAR's computing and software capabilities must continue throughout the ~15 year lifetime of the experiment in the midst of a world wide explosion in computing and software technology.

- The relativistic heavy-ion collision events are very complex, each contains of order 10-20 MB of raw data corresponding to several thousand charged particle tracks.

- Physics analysis of STAR data will likely involve making many, complex data base queries over huge amounts of DST volume by many users.

- Each of these go beyond the present experience in HE/NP by at least an order of magnitude.

The Solution:

- At the RHIC wide level the RHIC Computing Facility (RCF) at BNL attempts to address the data volume, cpu, bandwidth and data access issues.

- The STAR Software Infrastructure (SOFI) group has developed a plan and is well along the way implementing a computing environment to meet these challenges. Much remains to do and workers are urgently needed!

- A number of SOFI developed tools and commercial software products are in use and will be discussed at this school.

- Industry products and standards are used as much as possible.

- The STAR Simulations and Analysis Software (SAS) group has developed a general design plan for the offline software and a lot of simulation and analysis code has been written, however much remains to do with STAR data less than 2 years away.

Simulations and Analysis Software:

Description:

- Simulate particle production from pp, pA and AA collisions; particle propagation in the STAR detector; physical and electronic response of the various STAR detectors.

- Analysis of real and simulated data using magnetic field information and calibration constants to produce lists of particle tracks, decay vertices, energies, event characteristics and an overall event summary for data archives.

- Analysis of calibration, slow controls, monitor and other data to generate correction factors and calibration constants.

Essential Functionality:

- Simulation - generate lists of produced particles, ADC values, hits in the detectors, space points, tracks and event summaries.

- Reconstruction - Tracking, primary and secondary vertices, PID, neutral and charged energy, background track flagging, event characteristics and event data summaries.

- Calibration - Determine corrections for magnetic field, pedestals, gains, timing offsets, energy deposition, space point position distortions, beam-beam luminosity profiles, local and global alignment.

  The calibration software must handle data for laser tracks, cathode emission, charge injection, sources, light into PMTs, straight tracks, cosmic rays, wire pulses, gas monitor chambers, NMR and Hall probes, slow controls and monitors.

Elements:

- Infrastructure tools
- Physicist written data structures (tables) and analysis modules (~functions)
- Run time procedural control scripts (KUIP macros or kumac files)

2. Organization and People:

See http://www.rhic.bnl.gov/STAR/html/comp_l/org/CompOrg.gif

See http://www.rhic.bnl.gov/STAR/html/comp_l/sofi/swat

STAR Simulation and Analysis Software Group

Co-Leader: Lanny Ray

Co-Leader: Peter Jacobs

TPC: Iwona Sakrejda

D. Weerasundara, X. Zhang, M. Lisa, P. Jones, C. Ogilvie, E. Hjort, R. Bossingham, N. Stone, T. Trainor, K. Barish, F. Wang, W. Love, W. Betts, H. Long, M. Gilkes, H. Ward

SVT: Ken Wilson

S. Pandey, I. Kotov, H. Caines, V. Rykov, S. Paganis, O. Barannikova, J. Sheen, J. Takahashi, D. Read, L. Muresan

OFL Trigger: Pablo Yepes

Z. Milosevich, T. Trainor, E. Judd

Visualization: Fred Moore

H. Ward, M. Gilkes

CTB/TOF: Pablo Yepes

G. Eppley

EMC: Tom LeCompte

B. Christie, W. Llope, V. Rykov, K. Medved, K. Shestermanov, S. Vasiliev, C. Allgower, M. Tonjes, O. Tsai, A. Ogawa

Event Gen: Ron Longacre

J. Seger, E. Scannapieco, S. Klein

FTPC: Janet Seyboth

M. Konrad, J. Romero, J. Draper, A. Schuettauf, H. Hummler

Global: Spiros Margetis

M. Lisa, H. Caines, W. Llope, P. Jones, L. Ray, D. Weerasundara

GSTAR/g2t: Pavel Nevski

W. Llope

January 1998

3. General Computing Environment

UNIX is the general computing environment for STAR; Windows NT is under consideration.

Computing Platforms:

SGI irix
IBM aix
Hewlett-Packard hpux
DEC OSF1
SUN sun4
SUN Solaris sun4os5
Intel Pentium/Solaris sun4os5pc

Computing Hardware:

4 Silicon Graphics processors plus an 8 processor farm (rsgi00)

4 IBM processors plus SP-1 8 processor farm

10 Intel Pentium Pro-200 MHz processor farm (prototype for final system)

140 GBytes Disk capacity

2 8mm Tape robots

A 3 TByte DLT Storage Tek Tape Robot (access via /star/mds/...)

Computing Languages:

For programming:

Fortran 77 - with VAX extensions
- with C precompiler
- support legacy code

C

C++

Scripting Languages:

KUIP - interface language to STAF and PAW
- will eventually go away

Tcl/Tk - possible replacement for KUIP

IDL - interface user written data structures and modules to C++ (CORBA) objects in STAF.

PERL - for SOFI tools.

STAR File System (not an exhaustive list)

/star/u2 /u2a /u2b /u2c /u2d	STAR user disks
/star/data /scr2a /scr2b ...	STAR data, scratch space tmp file space, etc.
/star/tpc /svt /emc /tof /trg ...	STAR group disks
/star/mds	STAR Mass Data Storage (3 TB tape robot)
/afs/rhic/star/...	STAR Library, Analysis code, documentation

AFS = Andrew File System - managed files for distributed computing at non-BNL sites, restricted access to directories, group memberships, group ownership, privileges, access control lists,...

Much of the AFS area is version managed by CVS = Concurrent Versions System

STAR AFS Directory Hierarchy (not an exhaustive list)

/afs/rhic/star/

/CVSREP/dev	-- CVS Repository for STAR software, source codes, idl and inc files, but not the compiled binaries and executables.
/arch	-- Architecture dependent software releases and development software, *.inc files and compiled libraries, libPKG.a files.
/bin	-- SOFI tools (binary executables)
/doc	-- Web documentation files
/login	-- Setup and login scripts
/starlib	-- Reference copies of software organized by branch and software packages.

The CVS repository, reference area, software documentation area and binaries are organized with respect to the STAR Library Branches:

BRANCH	Description
ana	event rec. and calibration
daq	data acquisition
ofl	offline production
onl	online
phy	post DST physics analysis
sim	simulations
sys	system
trg	trigger

STAR Software PACKAGES:

- In STAR we use the term ‘‘Package’’ (abbreviated as PKG) for a specific sub-directory structure which contains source code, data struct’s, html files, *.inc files, macros, etc... associated with a well defined software task.

- Each STAR Branch is divided into a number of Packages. For example under /ana there are 65 packages (Dec. 1997) which handle event rec. and calibration. Under /sim there are 27 packages (Dec. 1997) which include event generators, Geant and detector response simulators.

- Each PKG contains the following directory structure according to the current STAR Library 97a (SL97a) release:

/PKG	/bin	/aix /irix ...	-- STAF executables specific to PKG or user for each architecture
	/doc		-- html documentation files
	/inc		-- *.idl files for data structure definitions; may contain 0,1,2... tables.
	/lib	/aix /irix ...	-- *.inc files, data structure header files, templates for f77 and C modules specific to PKG, compiled binaries for each arch., libPKG.a, interface code to STAF for each module.
	/solib		-- Similar to /lib for shared libraries
	/src		-- Source code for analysis modules (PAMs) and *.idl files for modules; may contain 0,1,2... modules.
	/wrk		-- Work area - KUIP macros or kumac files, saved PAW ntuples, soft-links to exec., Tcl scripts in future.

Directory Paths to Areas of Interest:

CVS Repository

/afs/rhic/star/CVSREP/dev/"BRANCH"/PKG/...

Arch. dependent software releases, libraries, *.inc files

/afs/rhic/star/arch/"ARCH"/...

CVS Reference area

/afs/rhic/star/starlib/ref/"BRANCH"/PKG/...

Web Documentation for PKG software

/afs/rhic/star/doc/www/code/"BRANCH"/PKG/doc

Web Documentation for STAR Groups

/afs/rhic/star/doc/www/...

Useful Directory Definitions:

$STAR_ROOT	=	/afs/rhic/star
$CVSROOT	=	$STAR_ROOT/CVSREP
$STAR_REF	=	$STAR_ROOT/starlib/ref
$STAR_ARCH	=	e.g. irix, aix, sun4os5pc, ...
$STAR_LIB	=	$STAR_ROOT/arch/$STAR_ARCH
$STAR_SYS_LEVEL	=	e.g. SL97a

4. General Computing Tools:

STAR Library Tools:

STAF	--	Scientific Table-based Analysis Framework; a.k.a. STar Analysis Framework
CVS	--	Concurrent Versions System used for software version control and cataloging of most STAR software, code development management tool.
DSL	--	Data Set Library; STAR I/O software for reading and writing datasets (~directory like structures) and tables (~files) using XDR from STAF to disk.
XDR	--	eXternal Data Representation standard (1987) for self describing, compressed data format.
IDL	--	Interface Definition Language, used by STAR to define table types, analysis module interfaces and analysis service package interfaces to CORBA objects, similar to C++ .
sl_*	--	STAR Library scripts which invoke CVS commands to create, check-out and check-in PKGs, also to add or remove files in PKGs from CVS control.
SLM	--	STAR Library Makefiles for setup, compiling, building STAF executables and installing binaries.
Stic	--	STAR IDL Compiler - Processes table and analysis module *.idl files and produces C header files, Fortran include files, interface code which STAF uses to provide C++ wrappers for the analysis modules, and C and Fortran templates specific to each analysis module in the package.
docfor, idldoc	--	Automatic html doc file generation from modules and idls.
/sys/cds	--	Automatic coding style checker.

Data Base Software -- Currently this is the RDBMS product ORACLE, but very little use is being made of it in STAR. Plans are underway to implement an OODBMS product called OBJECTIVITY. Watch sofi email for announcements!

CERN Software Products

CERNLIB -- CERN mathematics, statistics, analysis, graphics, and plotting software.

PAW -- Physics Analysis for Workstations - data analysis, histogramming, fitting and plotting tools for the HE/NP community. STAF is directly linked to PAW.

ROOT(?) -- Object oriented replacement for PAW under study, may be used by Phobos.

Physics Simulation Tools

Event Generators -- Very simple multiplicity generators up to very complex parton cascade models for pp, pA and AA relativistic collisions. Included are Fritiof, Hijing, Hijet, Vini, RQMD, HBT, and an assortment of ‘Longacre’ models.

GSTAR/g2t -- Implementation of Geant for STAR and Geant Zebra to STAF Datasets in XDR.

5. STAF

- STAF is a highly modular data analysis and software development framework written mainly in C++ in which the numerous software components are linked and communicate via a software bus using CORBA compliant interface standards.

- STAF provides CORBA compliant (C++ like) encapsulation of user written analysis modules which may be written in either Fortran 77, C or C++.

- STAF permits seamless integration of user written data structures and modules with system like software tools.

- The first production release of STAF was in June 1996; the current version is SL97a (Dec. 1997); STAF has been adopted RHIC-wide; STAF is the brain child of Craig Tull and Doug Olson at LBNL.

- The basic elements of STAF include:

    * KUIP user interface
    * Software Bus
    * Analysis Service Packages (ASP)
    * Physics Analysis Modules (PAM)
    * Datasets, data structure types and data structure instances (tables)
    * DSL and XDR

- A STAF session is initiated by typing a specific STAF executable filename; action occurs by way of KUIP commands which invoke STAF and PAW functions.

STAF Architecture

- Example STAF configuration with 5 ASPs and 9 PAMs loaded --

- All ASPs communicate with each other via the software bus (SOC)

- Each PAM interfaces to a C++ object and, in turn, to the rest of the system.

Analysis Service Packages

Software packages which serve system-like functions.

AMI -- Analysis Module Interface - PAM interface to STAF; calls PAMs at run time.

ASU -- Analysis Service Utility - messages and memory allocation statistics.

DIO -- Data Input/Output - Read/Write external files and event streams.

DUI -- Dataset Unix like Interface - Unix-like commands for dataset creation and navigation at run time. i.e.:

cd, ls, mkdir, pwd

Data tables (~files) are stored in datasets (~directories) in a hierarchical Unix-like tree structure at run time in STAF. DIO acts on the entire dataset rather than the individual tables.

LEV -- Log Environment and Version

SOC -- Service and Object Catalog - STAF object manager

SPX -- Service Package eXample

TBR -- Table BRowser - initiates GUI for browsing variables and numerical values in tables.

TDM -- Table and Dataset manager - create datasets and tables; get and set maximum number of rows, number of rows used, get and put numerical values.

Note: The table idl file determines the table type. TDM is used to create instances of a table type at run time (see following example tutorials).

TNT -- Tables to NTuples - fill column-wise PAW ntuples from STAF tables.

TOP -- Table OPerators - operate on tables; cut, project and join.

ASPs are invoked in STAF via calls which have the general syntax:

/ASPname/Adjectives.../action [name of object acted on] [parameters]

Examples of these commands in STAF will be given in the following tutorials.

Physics Analysis Modules

- User written f77, C and C++ functions which do the simulation, event reconstruction, calibration and post-DST physics analysis.
- Interface to STAF and other PAMs are thru table arguments; these consist of a table header part and the variables part.
- PAMs are interfaced to STAF via idl files.
- User written functions are linked to STAF at the package level only, i.e. not individual tables or modules.
- PAMs may use tables belonging to different PKGs; tables belonging to a PKG may be used by modules in other PKGs.
- PAMs, however, are called individually at run time using the ami/call STAF command.

Example Physics Analysis Module:

Example PAM idl file:

STAF Data Structure Definitions - tables

- Data structures in STAF are defined as variable length arrays of C structs, in table idl files, which follow the naming convention

  PKG_tablename.idl

  These are located in the PKG/inc sub-directory.

- All data communication between STAF and PAMs and among the PAMs are via tables.

- The maximum row length is assigned at run time and may be dynamically increased or decreased at run time.

- The actual number of rows used can be assigned either at the KUIP level using a STAF command or within an analysis module.

- One or more instances of a table type as defined in an idl file can be assigned at run time, it will be given a local run time name and placed in a dataset (~directory) in STAF.

- FORTRAN include files and C, C++ struct files are created automatically using Stic.

Variable types allowed in STAF:

IDL	Ansi-C	C++	FORTRAN 77
short	short	short	INTEGER*2
unsigned short	unsigned short	unsigned short	INTEGER*2
long	long	long	INTEGER*4
unsigned long	unsigned long	unsigned long	INTEGER*4
char	char	char	CHARACTER*1
octet	unsigned char	unsigned char	LOGICAL*1
float	float	float	REAL*4
double	double	double	REAL*8

Example IDL table type definitions:

tcl_tphit.idl

tpt_track.idl

Reference to table variables:

- In FORTRAN modules:

  localtablename(i).variable

  where i = 1,2... is the row number

- In C and C++ modules:

  localtablename[i].variable

  where i = 0,1,2... is the row number

- At the KUIP level as 'table_instance_name[i].variable'

  where i = 0,1,2... is the row number

- Tables with only 1 row are referenced as:

  localtablename(1).variable

  localtablename[0].variable

- Table attributes are referenced in PAMs as:

  Maximum number of allocated rows → *localtablename_h.maxlen*

  Number of used rows → *localtablename_h.nok*

Primary and Foreign Keys in tables:

- **Primary Key -** A variable in a STAF table which uniquely identifies a row or set of related rows. This might or might not be set equal to the row number.

  example: tableX(i).id

- **Foreign Key -** A variable in a STAF table that refers to a row or set of related rows in the same or in another table. The reference is not a pointer to a memory address, but is indicated by the numerical value assigned to the foreign key.

- For example, refer to the tcl_tphit and tpt_track tables which contain TPC space points and tracks. In order to relate all the TPC space points which were assigned to a certain TPC track we set the foreign keys in tcl_tphit according to:

  tcl_tphit(i).track = tpt_track(j).id

  (Foreign key) (Primary key)

  for i = row number of all assigned space points to the track on row j in table tpt_track.

The Foreign key values are equatted to the appropriate Primary key value,

NEVER TO THE ROW NUMBER

~~*tcl_tphit(i).track = j*~~

- SOFI has developed several tools to facilitate Primary/Foreign key usage in STAF modules. These tools and other important guidelines are presented in:

  ‘‘Proposed Rules and Tools for PAM Initialization and Foreign Key Usage’’

  http://www.rhic.bnl.gov/STAR/html/sofi_l/info/margetis/key-usage.html

7. User Programming Guidelines

- FORTRAN 77 and C analysis module templates
  generated by Stic.

- STAR Coding Style Manual
  for FORTRAN 77, C and C++

  -- see http://www.rhic.bnl.gov/STAR/html/ssd_l/style/style.html

- Code Style Checker -- Automatically checks your f77 and C
  analysis modules for compliance with the STAR Coding Style
  Manual; it issues error messages and warnings.
  Compliance is voluntary so far.

  -- see system software package /sys/cds

- PAM initialization and Primary key - Foreign key usage
  described in the preceding.

8. STAR WWW Documentation and Organization

Useful and up-to-date description of STAR documentation procedures are available on the web at:

STAR Home page > How to > Document submission

Useful STAR Computing and Software Web Pages:

STAR Home >	How to	> Getting Started > Tutorials > Document Submission > Software Code pkg authorization	(Comp. Environ.)
STAR Home >	Computing	> Offline Software > Software Infrastructure > etc...	
STAR Home >	news	> SOFI	(changes, updates, patches)
STAR Home >	documents	> STAR Notes	(STAR technical reports)
STAR Home >	group pages	> TPC > Software > Global, SAS, SOFI, ...	(example kumacs and cosmic ray data analysis)
STAR Home >	mail archives	> STAR SAS > STAR SOFI > STAR SOFT > STAR SVT-Soft > STAR Global	(email discussion, announcements, mail archives)

9. Steps in a Typical Simulation Project:

(Examples of these steps will be included in the STAF tutorials.)

- Determine what detectors, reconstruction and physics analysis are required. Survey the relevant documentation to get an idea of what software is available and which PKGs, PAMs and tables are involved.

- Create a local, dummy PKG directory tree structure to work in, edit the Makefile.control file to link the required ASPs, PAMs (by PKG), user LIBS and CERN Libs.; build the STAF exec. on the desired platform.

- Run a STAR event generator to produce simulated events or use existing events in /star/mds/data.

- Run GSTAR/g2t using the event generator output, the appropriate STAR geometry files, and the necessary physics process switches and threshold parameters.

- Develop the KUIP macro (kumac file) to control the execution of STAF for your specific application. This involves loading the necessary initialization and setup kumacs for each PAM, determining the dataset organization, GSTAR/g2t event stream input, calling the modules, saving the desired output tables in PAW ntuples. The best approach is to edit a similar, existing kumac file.

- Run the STAF executable and analyze the results with PAW.

Steps in a Typical Software Development Project:

- Determine what existing PKGs, PAMs and tables are necessary, and if existing PKGs need to be modified or if new PKGs need to be created.

- If a new PKG is needed, submit a Package Proposal Form to the appropriate Software Branch Manager. If the new PKG is approved the manager will create the new PKG using CVS and the corresponding AFS group. The STAR web servant (Al Saulys) will set the necessary links to install the PKG documentation files.

- ‘‘Check-out’’ the PKG into your local directory area. This is where all development, debugging, testing and running is done.

- Edit or write the PKG_table.idl files in the PKG/inc directory.

- Edit or write the analysis module(s) and the PKG_pam.idl files in the PKG/src directory.

- Edit the Makefile.control file and compile and build STAF. When the PKG builds satisfactorily, edit Makefile.control and include any other required ASPs and PAMs to build a complete STAF executable for testing and futher simulation studies.

- When the new or modified PKG is working and tested to your satisfaction, write or update the PKG/doc/*.html files.

- Add/remove files in the PKG which you want CVS to manage. These should be text files only, no binaries! Do not include temporary files, working junk files, etc.

- ‘‘Check-in’’ the PKG to the CVS repository. Install the binary object libraries and *.inc files.

- Verify installation including doc files and libraries.

- Announce new or modified PKG to starsoft-l.

STAR Computing and Software Glossary

AFS -	Andrew File System; managed, distributed data
ARCH -	Computing platform and operating system, e.g. irix.
ASP -	Analysis Service Package, system-like functions in STAF.
CORBA -	Common Object Request Broker Arch., interface standard.
CVS -	Concurrent Versions System; code version management
Dataset -	Container for STAF tables; analogous to UNIX directory.
DLT -	Digital Linear Tape
DSL -	DataSet Library
DST -	Data Summary Tape; event reconstruction summary
GUI -	Graphical User Interface
html -	Hypertext mark-up language, for web documents.
IDL -	Interface Definition Language; interface for tables and PAMS
KUIP -	Kit for User Interface Package; Fortran-like scripting language
kumac -	KUIP macro; files for convenient running of STAF and PAW.
MDS -	Managed Data Store, a.k.a. Mass Data Store.
ntuple -	PAW table-like data structure in compressed format (Zebra).
OODBMS -	Object Oriented Data Base Managment Software
OS -	Operating System (e.g. irix 6.0)
PAM -	Physics Analysis Module; user written analysis code in STAF
PAW -	Physics Analysis (for) Workstation(s); CERN data analysis product.
PKG -	Package; related set of software items as understood in STAR.
RCF -	RHIC Computing Facility
RDBMS -	Relational Data Base Managment Software
ROOT -	Rene’s Object Oriented Toy (just kidding!) OO replacement for PAW.
SAS -	Simulations and Analysis Software (group in STAR).
SLM -	STAR Library Makefiles
SOFI -	SOFtware Infrastructure (group in STAR)
STAF -	Scientific Table-based Analysis Framework; a.k.a. STAR Analysis Fr.
Stic -	STAR idl compiler
table -	Basic data structure element in STAF
Tcl/Tk	Tool Command Language/Tool kit; scripting language with GUI.
WWW -	World Wide Web
XDR -	eXternal Data Representation; data compression standard.