A small workshop on database tools was held at LIGO Caltech on 22,23 October 1998. LIGO invited four individuals with different backgrounds in the use of databases and database management systems. The purpose was to glean information that would be useful in assisting LIGO to select a database tool for manipulating LIGO data. Following are materials presented by the presenters.

In attendance at the workshop were:

- Stuart Anderson, LIGO
- J. Kent Blackburn, LIGO
- Chaitan Baru, SDSC DB2/IBM HPSS expert
- Julian Bunn, CACR and CERN CMS Experiment
- Damir Buskulic, VIRGO Annecy Data Group
- Thomas Handley, JPL/Caltech IPAC
- Barbara Kratochiwill, LIGO
- Albert Lazzarini, LIGO
- Walid Majid, LIGO
- Edward Maros, LIGO
- Thomas Prince, LIGO
- Fons Rademakers, CERN/ROOT
- Gary Sanders, LIGO
- Larry Wallace, LIGO
- Rai Weiss, LIGO
- John Zweizig, LIGO

Workshop on Databases for LIGO

22,23 October 1998

LIGO Laboratory

Caltech

Pasadena, CA

Agenda

Thursday 22 October

Introduction - Albert Lazzarini

LIGO Astrophysics - Rai Weiss

LIGO Data Types, Data Products - A. Lazzarini

LIGO Data Analysis System Architecture
- Kent Blackburn

LIGO Data Archive - Roy Williams

LIGO Data Uses/Needs - R. Weiss, Daniel Sigg

Presentation by C. Baru, SDSC

Presentation by T. Handley, JPL/IPAC

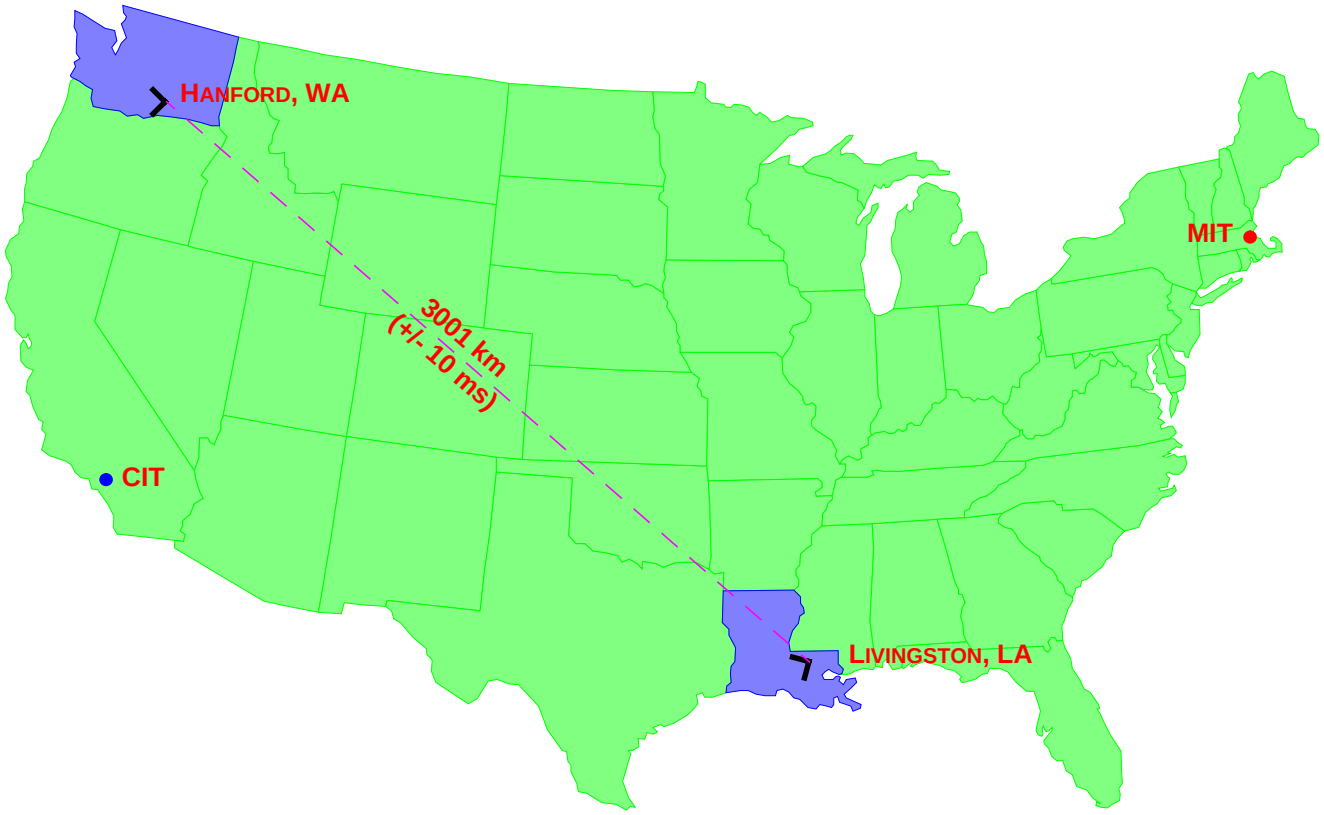
Presentation by J. Bunn, CERN

Presentation by F. Rademakers, CERN

Introduction
- Albert Lazzarini

LIGO LABORATORY SITES

Observatories & Universities



HANFORD, WASHINGTON

- LOCATED ON U.S. DOE RESERVATION
- TREELESS, SEMI-ARID HIGH DESERT
- APPROX. 25 KM FROM RICHLAND, WA (POPULATION :140,000)

LIVINGSTON, LOUISIANA

- LOCATED IN FORESTED RURAL AREA
- MIXED FOREST; LOW-LYING; POOR DRAINAGE
- APPROX. 50 KM FROM BATON ROUGE, LA (POPULATION :450,000)

LIGO

- NSF funded facility [$290M] dedicated to interferometric detection gravitational waves in the 40Hz - 3kHz band arising from astrophysical processes
- Interferometers are a practical realization of isolated test-masses in “free fall” -- inertial frames
  - High power IR laser (10W @ 1064 nm) to sense displacement with adequately small shot noise
  - Massive, high quality mirriors as test “charges” for gravity (10 kg,~ λ/1000 figure error)
  - Long arms (4km) and many reflections of light (~50x) to magnify effect of gravitationally induced strain on space-time
  - Ultra high vacuum ($p < 10^{-7}$ torr) to suppress light scattering (backgrounds)
  - Large volume (V ~ 10000 m^3 per facility)
  - Isolation @ 100 Hz ~ 175 dB from seismic environment

LIGO

- Two Sites: Hanford, Washington & Livingston, Louisiana
  - 3000 km baseline isolates sites and helps localize sources by time (phase) delay of signals
  - Arms oriented “parallel” to one another to maximize senstivity to same polarization state of the GW
  - Two 4km interferometers & one 2 km interferometer (Hanford): amplitude discrimination
  - Coincident observations among all three interferometers reduces terrestrial backgrounds
- Initial Sensitivity: $h_{rms} \leq 10^{-21}$ within 100 Hz band centered at maximum sensitivity [~100Hz]

LIGO Astrophysics
- Rai Weiss

Elementary Properties of Gravitational Waves and Sources

Required by all relativistic theories of gravitation

In vector and tensor theories, sources are accelerated masses

Wave propagation velocity is assumed as c

General Relativity

Lowest order source is a quadrupole:

One sign of mass/energy

Momentum conservation

In weak field approximation:

Electromagnetic analog for radiated energy holds: $Q^2 \rightarrow GM^2$

Wave equation: $(\nabla^2 - \delta^2/c^2\delta t^2)h_{\mu\nu} = 0$

THE RADIATION FIELD

Transverse Plane Wave Solutions with "Electric" and "Magnetic" Terms

Geometric Interpretation

$$d\,s^2 \;=\; g_{ij}\,dx^i\,dx^j$$

$$g_{ij} \;=\; \eta_{ij} \;+\; h_{ij} \qquad \text{weak field}$$

$$\eta_{ij} \;=\; \begin{pmatrix} 1 & & & 0 \\ & -1 & & \\ 0 & & -1 & \\ & & & -1 \end{pmatrix} \qquad \text{Minkowski Metric of Special Relativity}$$

Gravity Wave Propagating in the x_1 Direction

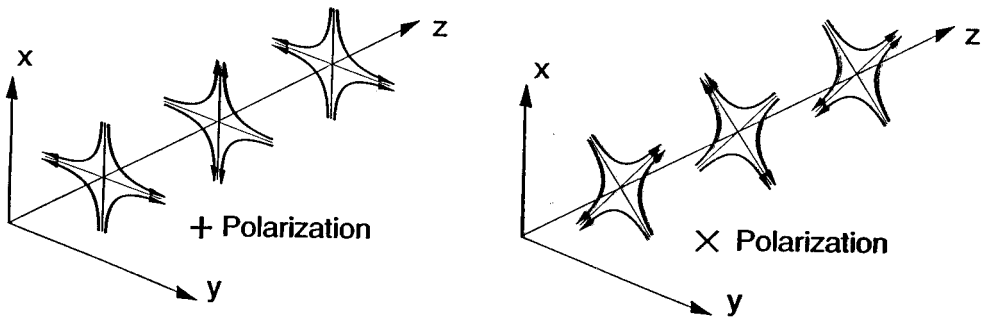
$$h_{ij} \;=\; \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & h_{22} & h_{23} \\ 0 & 0 & h_{32} & h_{33} \end{pmatrix} \qquad \text{all } h_{ij} \ll 1$$

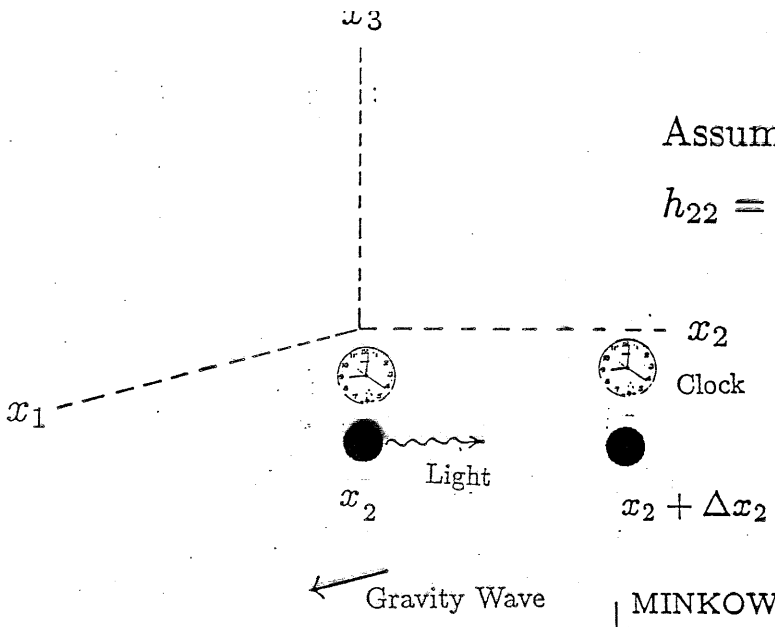
Plane Wave

$\mathbf{h_{22} \;=\; -h_{33}} \qquad \mathbf{h_{23} \;=\; h_{32}}$

$\mathbf{+\,polarization} \qquad \mathbf{\times polarization}$

And All Only Function of $x_1 \;-\; ct$


x
z
y
+ Polarization
x
z
y
× Polarization



Assume + Polarization

$$h_{22} = h\sin(kx_1 - \omega t)$$

$$\Delta s^2 = 0 = c^2\Delta t^2 - \Big(1 + h\sin(kx_1 - \omega t)\Big)\,\Delta x_2^2$$

LIGHT RAY

Let $\quad \Delta t \ll \dfrac{1}{\omega} \qquad h \ll 1$

$$c\,\Delta t \cong \Big(1 + \frac{h}{2}\sin(kx_1 - \omega t)\Big)\Delta x_2$$

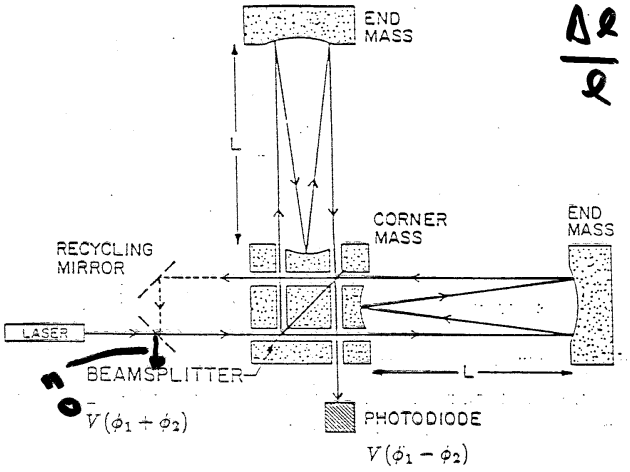
INFERRED DISTANCE BETWEEN POINTS

$$\frac{\delta(c\,\Delta t)}{\Delta\, x_2} = \frac{h}{2}\sin(kx_1 - \omega t)$$

Time Dependent **Strain**

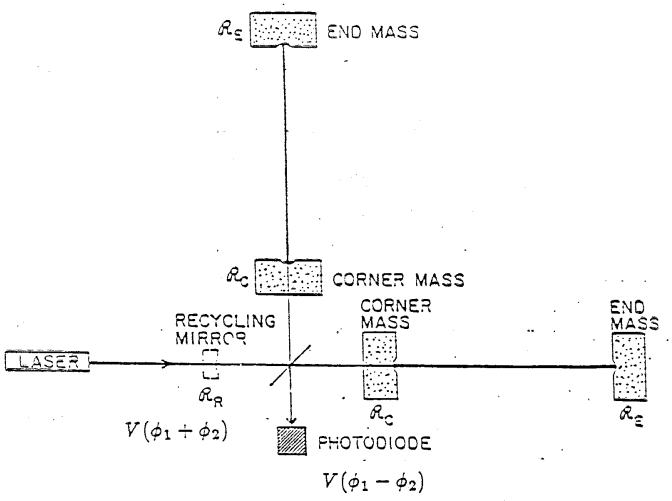
$$\frac{\Delta\, l}{l} = \frac{h}{2}$$

The Measurable Quantity

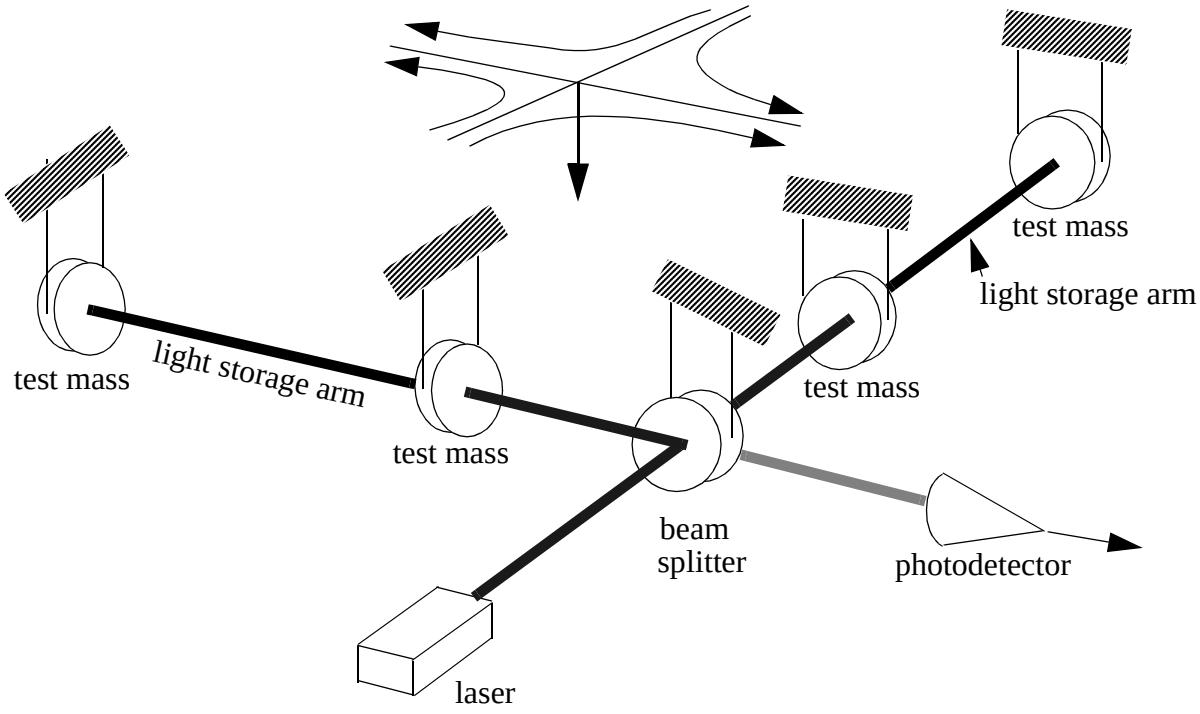


$\frac{\Delta \ell}{\ell} < 10^{-21}$

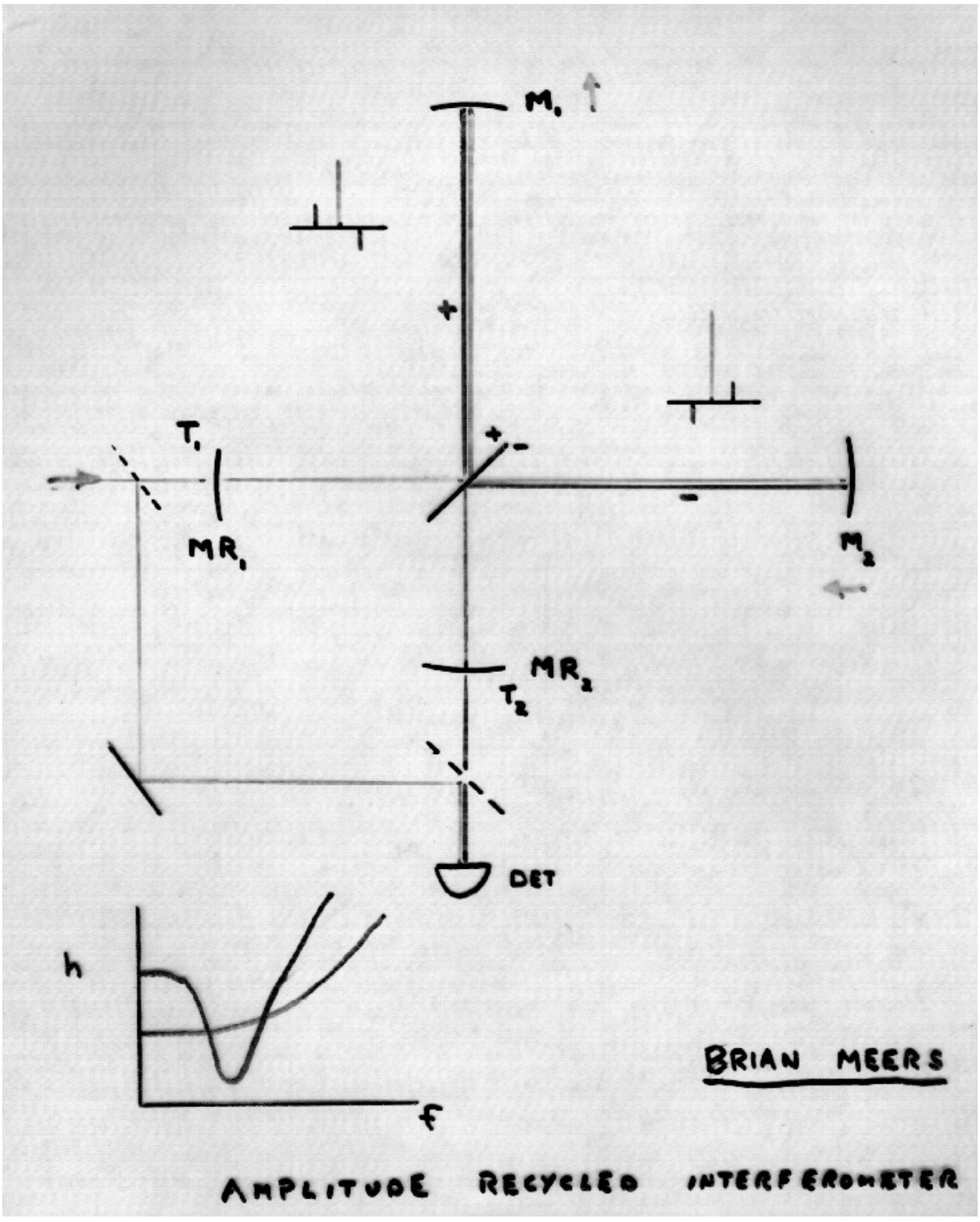
Schematic Michelson Delay Line Interferometer



Schematic Fabry - Perot Interferometer



b) laser interferometric detector



BRIAN MEERS

AMPLITUDE RECYCLED INTERFEROMETER

NOISE SOURCES

Noise Terms Influencing the Strain Measurement

* Shot (Poisson) Noise

  Light Amplitude Noise

  Laser Frequency Fluctuations

  Scattering of Light by

  1) Moving Sources
  2) Stationary Sources

  Laser Beam Position and Angle Jitter

  Residual Gas Column Density Fluctuations

Fluctuation Forces Moving the End Points

* Seismic Noise

* Thermal Noise in the Suspension Elements

  Thermal Noise Driving the Mirror Normal Modes

  Optical / Mechanical Imbalance Radiation Pressure Force

  "Radiometer" Force Driven by Light Amplitude Noise

  Fluctuating External Gravitational Gradients

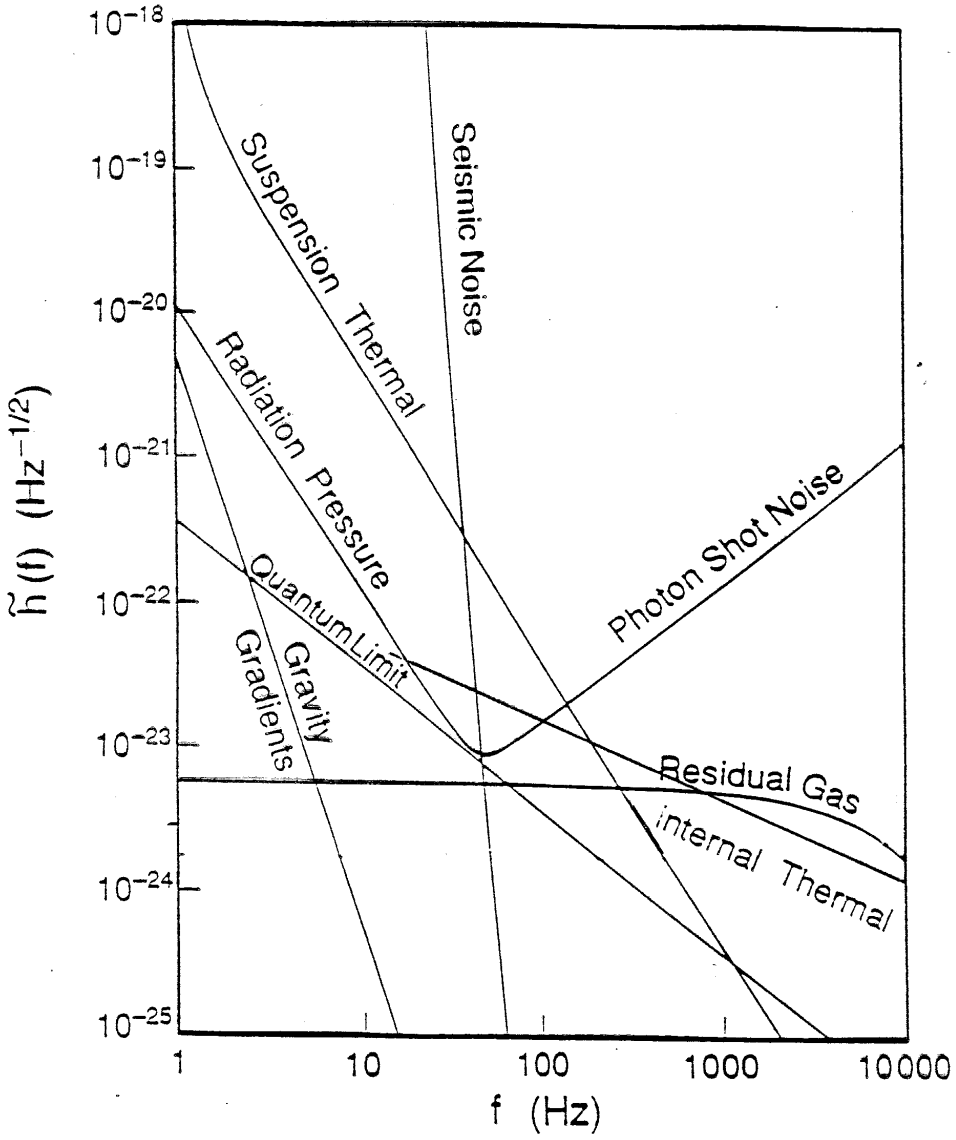
  Fluctuating "Patch" Electric Fields

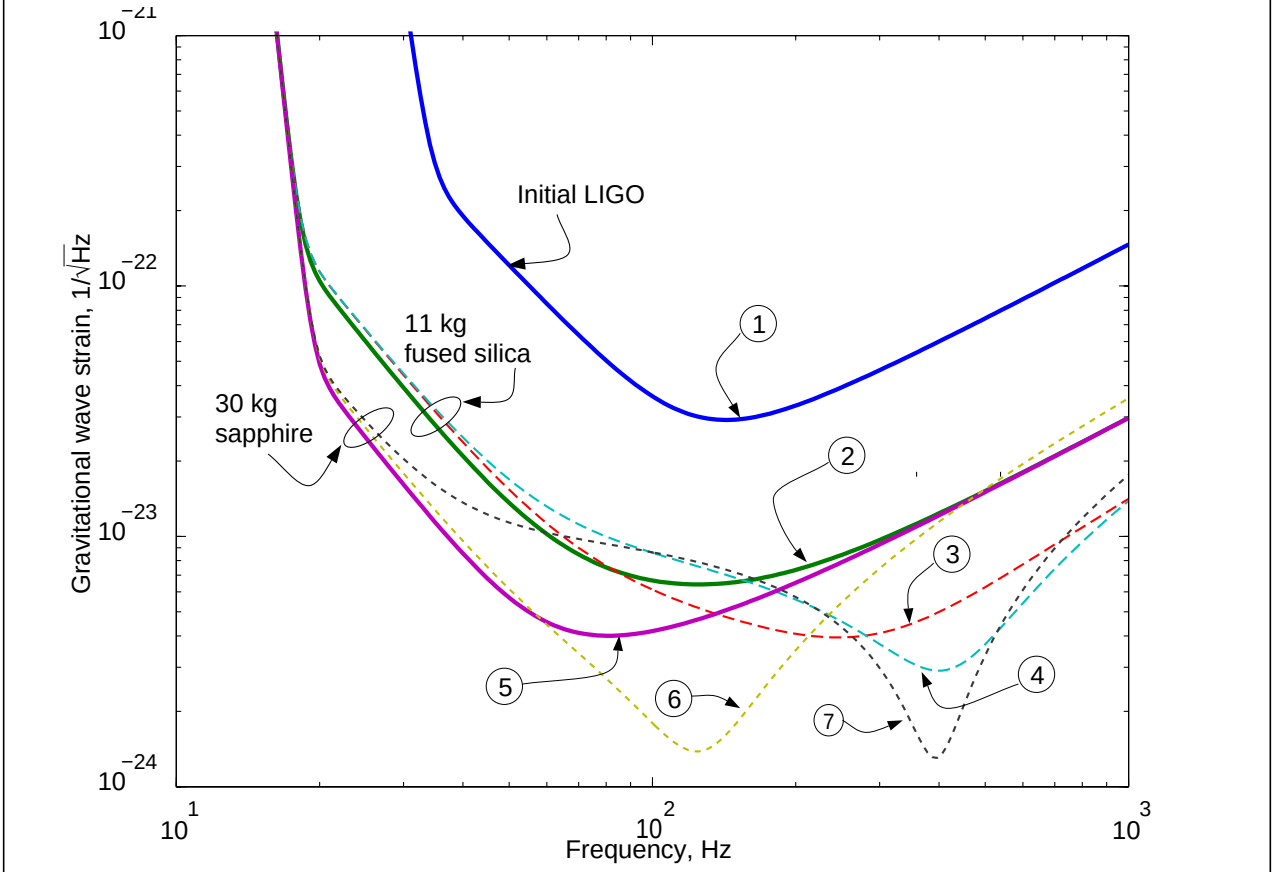
  Fluctuating Magnetic Fields Acting on Iron Impurities

  Cosmic Ray Muons

  The "Naive" Quantum Limit

* Important Terms Influencing *Initial* Sensitivity Goals


10^{-18}
10^{-19}
10^{-20}
10^{-21}
10^{-22}
10^{-23}
10^{-24}
10^{-25}
$\tilde{h}(f)$ $(Hz^{-1/2})$
Suspension Thermal
Seismic Noise
Radiation Pressure
Photon Shot Noise
Quantum Limit
Gravity Gradients
Residual Gas
Internal Thermal
1
10
100
1000
10000
f (Hz)



Parameter	Curve 1	Curve 2	Curve 3, 4	Curve 5, 6, 7
Parameter	Initial LIGO I value	Double suspension, 100 W laser, thermal de-lensing	Signal tuned configuration	Alternative test mass material
Input power to recycling mirror	6w	62w	140w	
Mirror loss (transmission+scatter)	50 ppm	20 ppm		
Effective power recycling	30	93		
Substrate absorption	5ppm/cm	0.4 ppm/cm		17 ppm/ cm
Thermal lensing correction	(none)	factor 10		
Suspension fiber	steel wire, $Q = 1.6\times10^5$	fused silica $Q = 3 \times 10^7$		
Test mass	fused silica, 10.8 kg, $Q = 1\times10^6$	fused silica, 10.8 kg, $Q = 3\times10^7$		sapphire, 30 kg, $Q = 2\times10^8$
Signal recycling mirror transmission	(none)		T=0.6 (curve 3) T=0.15 (curve 4)	Curve 5: none T=0.3 (curve 6) T=0.09 (curve 7)
Tuning phase			0.7 rad (curve 3) 0.45 rad (curve 4)	1.3 rad (curve 6) 0.45 rad (curve 7)

Figure and Table 1 : Performance and parameters of interferometers described in program.

.

RELATION OF THE INTENSITY AND THE METRIC COMPONENTS

$$I_g = \frac{c^3}{16\pi G} < \dot{h}_+^2 + \dot{h}_\times^2 > \qquad \text{Einstein 1918}$$

$$c^3/G \sim 4 \times 10^{38}\, ergs/cm^2\, Hz$$

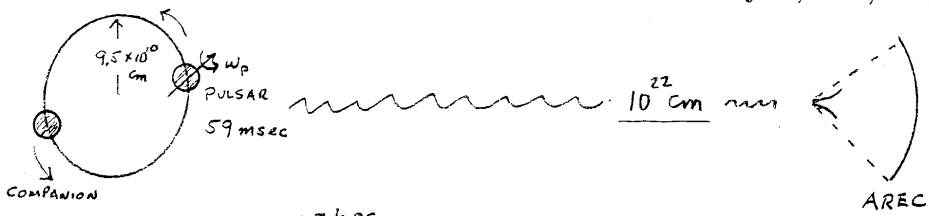
COMBINING QUADRUPOLE RADIATION FORMULA WITH INTENSITY

$$h \sim \frac{G\,M}{R\,c^2}\,\frac{v^2}{c^2}$$

$$M_\odot,\ R = 10\,kpc \quad v/c \sim 1 \quad h \sim 10^{-17}$$

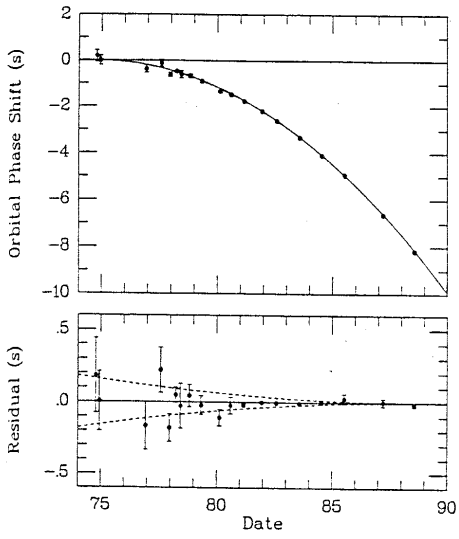
The Binary Pulsar PSR 1913 + 16

Taylor, J.H., Weisberg, J.M.



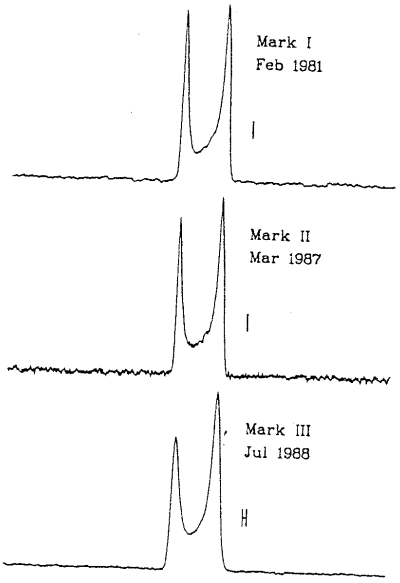
ORBITAL PERIOD ~ 7.75 hrs

$e \sim 0.6$ $\quad \frac{v_{ORB}}{c} \sim 10^{-3}$ $\quad \frac{GM}{2ac^2} \sim 10^{-6}$

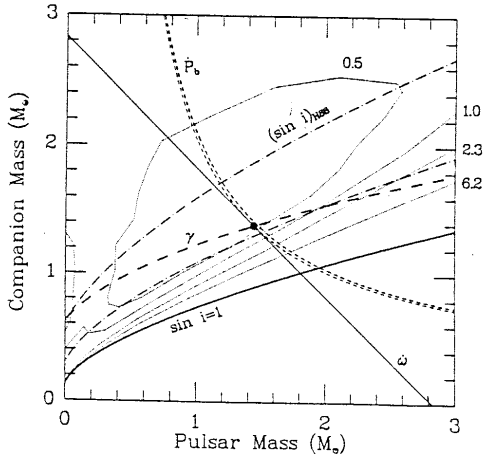


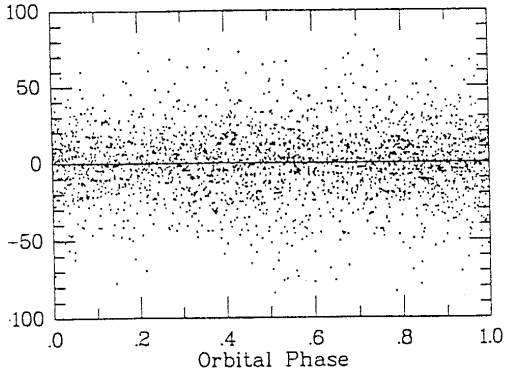
Energy loss to gravitational radiation

$$P = G(\dddot{Q})^2/45c^5$$

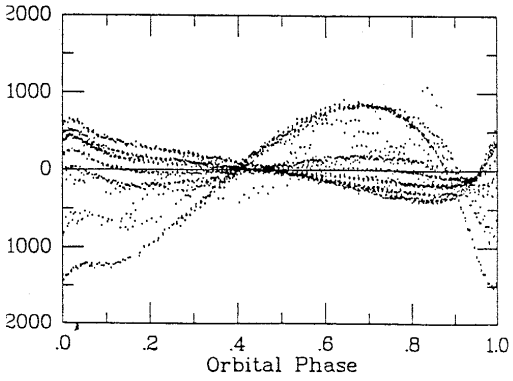


Detected Pulsar pulses





Post fit residuals



Post fit residuals if $\gamma \rightarrow 0$

Taylor, J.H., Weisberg, J.M.

Properties of the waves and sources

The waves

- The detector measures h, the field amplitude, directly. Corresponds to an envelope detector in E&M.
- The depth of the observations varies as 1/h, the volume of space opened to observation varies as $1/h^3$.
- The waves interact weakly with matter - the scattering is negligible. Gravitational waves are the most penetrating interactions in nature.

The sources

- The waves are radiated by the coherent accelerations of large aggregates of mass in the source - not strongly by the independent motions of many small objects.
- The waves will originate from the inner motions usually shrouded in E&M.

  Back to the beginning, before recombination in the universe
  Inside of a stellar collapse
  At the horizon of a black hole

- Scaling

$$\tau \approx \frac{1}{\sqrt{G\rho}} = \sqrt{\frac{R^3}{GM}} \xrightarrow[\text{black hole}]{} \frac{\mathbf{GM}}{\mathbf{c^3}}$$

10^{-4} sec surface NS 10^{-4} sec solar mass BH

COMPACT BINARY DECAY

$$f_{\substack{orb \\ max}} \;=\; \frac{1}{2\pi}\left(\frac{G\pi\rho}{3}\right)^{1/2} \qquad \rho = 10^{14} gm/cm^3 \qquad f_{\substack{gw \\ max}} \;\sim\; 1kHz$$

If Damped Only by Gravitational Radiation

Number of Cycles Near f

$$n \;=\; \frac{f^2}{df\,/dt} \;=\; \frac{5}{96\pi}\;\left[\frac{c^3}{\pi G\,M\,f}\right]^{5/3}$$

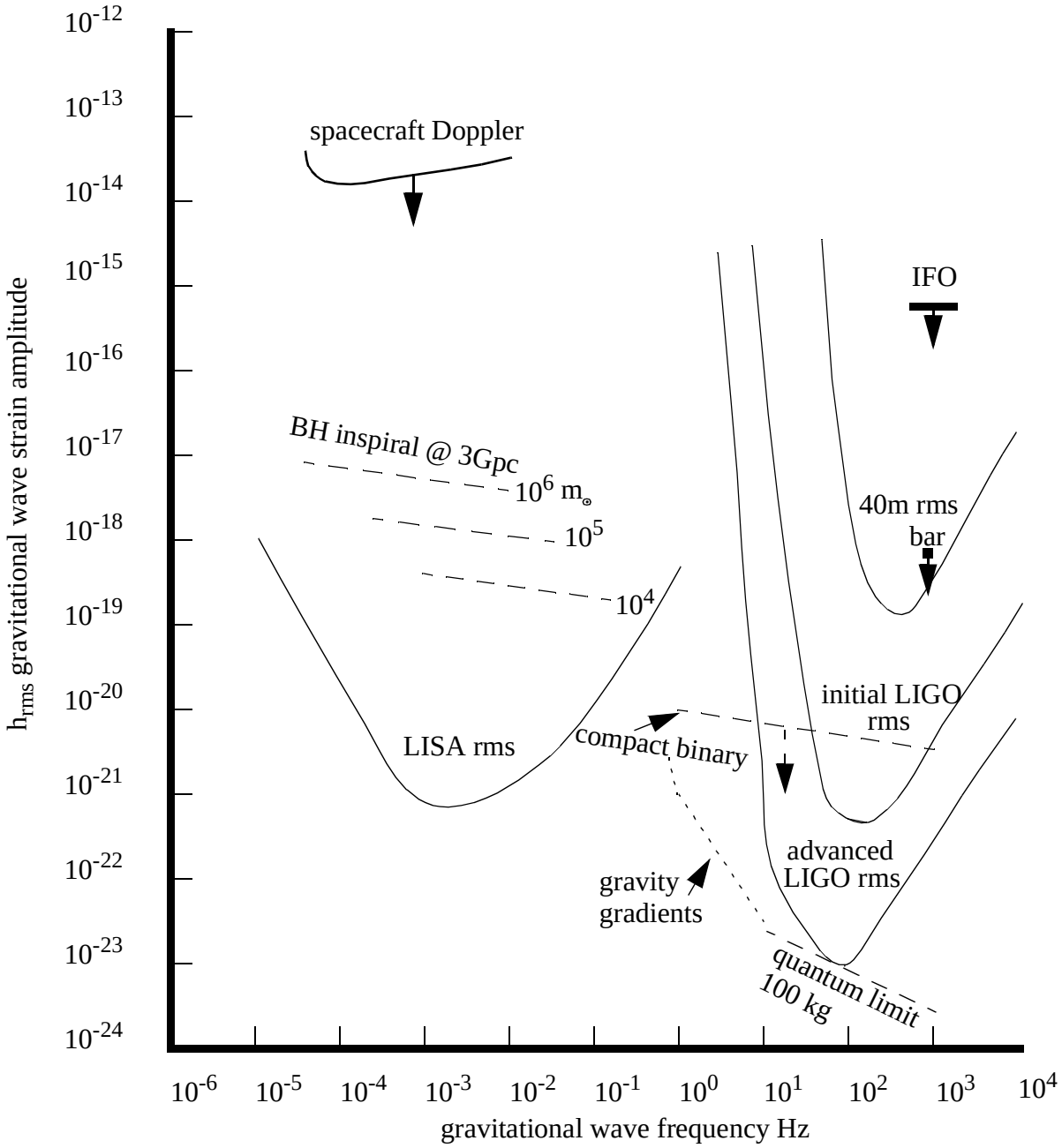
$$< h > n^{1/2} \;=\; \left(\frac{2}{15\pi}\right)^{1/2}\;\frac{G\,m}{r\,c^2}\;\left[\frac{c^3}{\pi G\,M\,f}\right]^{1/8}$$

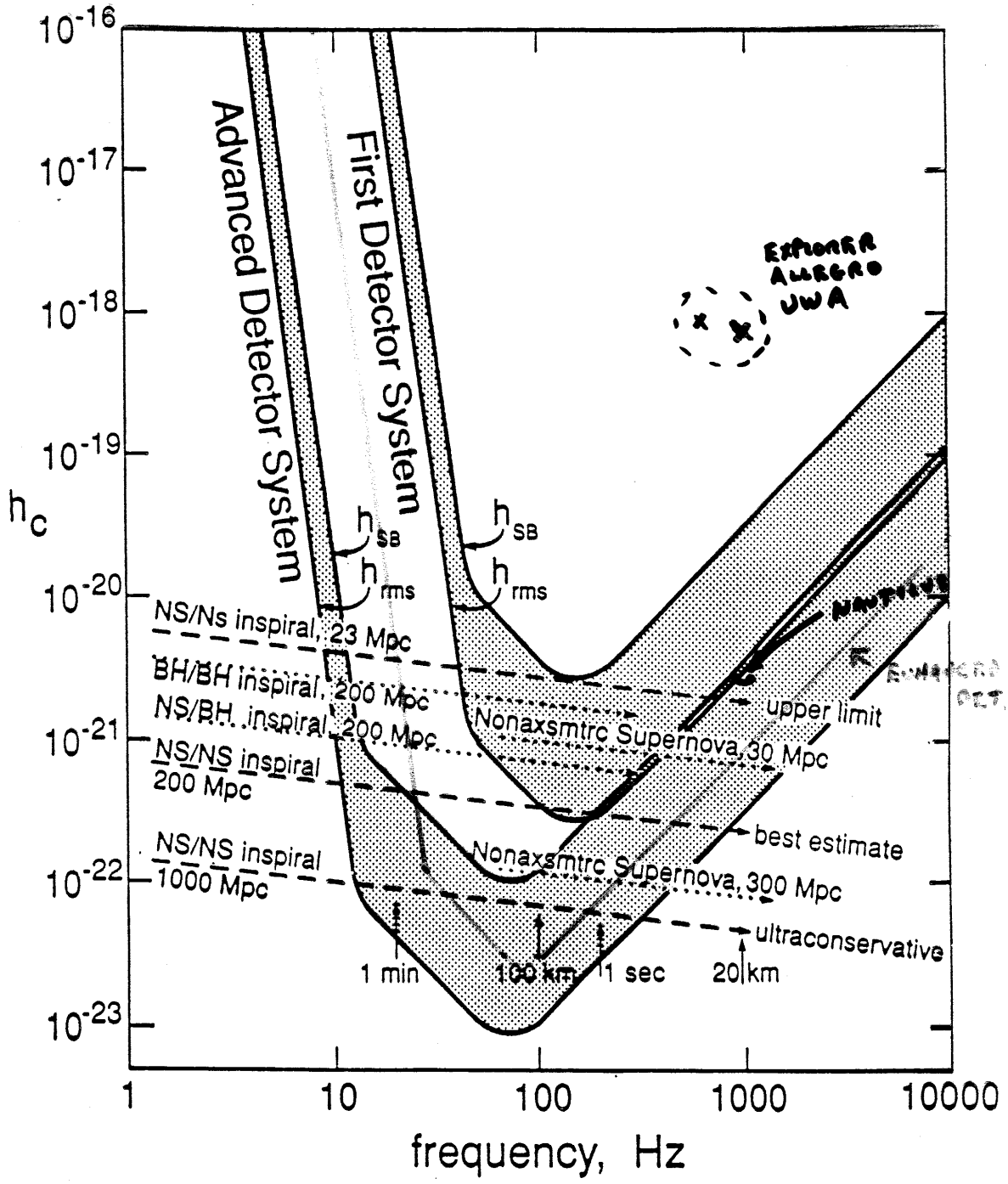
SUPERNOVA

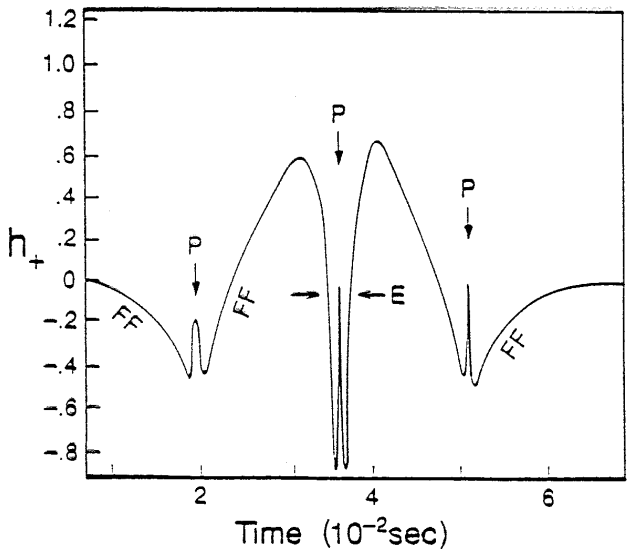
$$h \;=\; 5\times 10^{-22}\;\left[\frac{\epsilon}{10^{-3}}\right]\;\left[\frac{15\,Mpc}{r}\right]\;\left[\frac{1\,kHz}{f}\right]^{1/2}$$

PULSARS

$$h \;=\; 10^{-19}\,\epsilon\,\frac{(f/1\,kHz)^2}{(r/10\,kpc)}$$

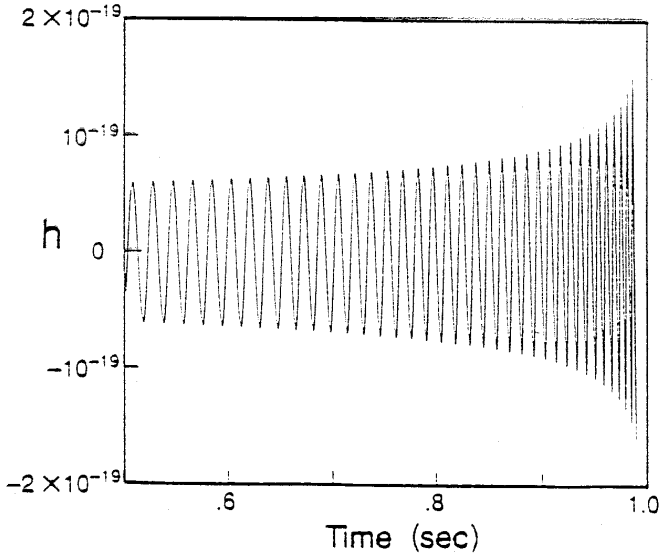

10^{-12}
10^{-13}
10^{-14}
10^{-15}
10^{-16}
10^{-17}
10^{-18}
10^{-19}
10^{-20}
10^{-21}
10^{-22}
10^{-23}
10^{-24}
h_{rms} gravitational wave strain amplitude
spacecraft Doppler
IFO
BH inspiral @ 3Gpc
10^6 $m_\odot$
10^5
10^4
40m rms
bar
initial LIGO
rms
LISA rms
compact binary
advanced
LIGO rms
gravity
gradients
quantum limit
100 kg
10^{-6} 10^{-5} 10^{-4} 10^{-3} 10^{-2} 10^{-1} 10^{0} 10^{1} 10^{2} 10^{3} 10^{4}
gravitational wave frequency Hz

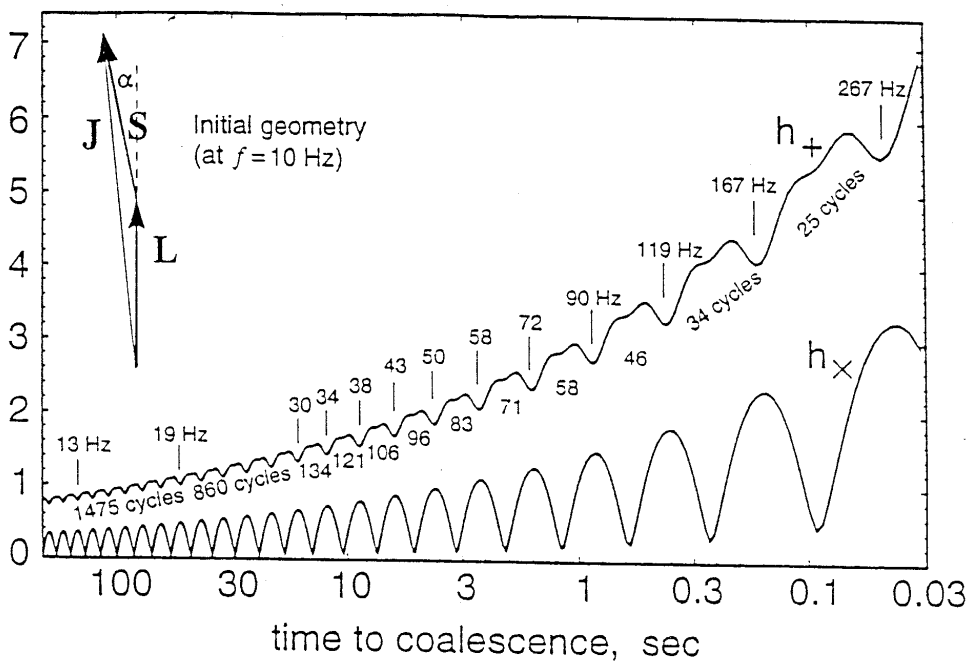

10^{-16}
10^{-17}
10^{-18}
10^{-19}
10^{-20}
10^{-21}
10^{-22}
10^{-23}
h_c
Advanced Detector System
First Detector System
h_{SB}
h_{rms}
h_{SB}
h_{rms}
EXPLORER
ALLEGRO
UWA
NS/Ns inspiral, 23 Mpc
BH/BH inspiral, 200 Mpc
NS/BH inspiral, 200 Mpc
NS/NS inspiral
200 Mpc
NS/NS inspiral
1000 Mpc
Nonaxsmtrc Supernova, 30 Mpc
upper limit
best estimate
Nonaxsmtrc Supernova, 300 Mpc
ultraconservative
1 min
100 km
1 sec
20 km
1
10
100
1000
10000
frequency, Hz



Signal from a stellar collapse to a neutron core

Saenz, R. A., Shapiro, S. I





Modulation envelopes for the two waveforms from a NS spiraling into a spinning BH. The vertical scale is arbitrary but linear, and is the same for h_+ and $h_\times$.

C. Cutler et al

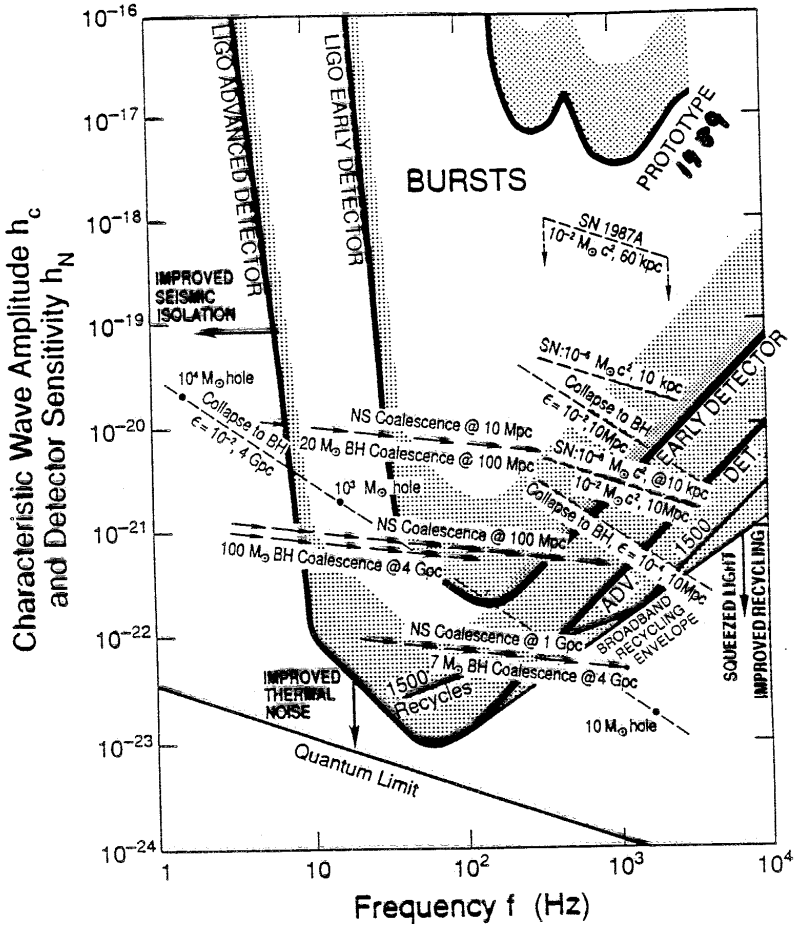


Figure A–4a The estimated characteristic amplitude $h_c \simeq h\sqrt{n}$ and frequency f for gravitational-wave bursts from various sources (dashed lines and arrows); and benchmark sensitivities $h_N \simeq \tilde{h}(f)\sqrt{f}$ (solid curves and stippled strips atop them), for interferometric detectors today and in the proposed LIGO. n is the number of cycles in the burst for which the amplitude is near h and the frequency is near f (see text) and $\tilde{h}(f)$ is the square root of the spectral density of the interferometer's noise; (see Equations (A.18), (A.19) and Appendix F). For each detector the solid curve corresponds to the characteristic amplitude of a gravitational wave burst which would be detected with unity signal–to–noise ratio, for optimal polarization and arrival direction. The top of the stippled strip corresponds to the characteristic amplitude of gravitational waves which would give significant detection of bursts occurring as infrequently as three times per year, with random polarizations and arrival directions, in detectors which are free from spurious non-Gaussian noise bursts. Stated more precisely, it corresponds to a confidence level of 90% that the signals are not false alarms caused by statistical fluctuations in Gaussian noise, in a cross-correlation experiment between two detectors at separate sites. The lowest solid line (Quantum Limit) is the limit to the sensitivity curve for the advanced detector set by the quantum limit for the test masses. "Envelope" defines the range of broad-band recycling improvements to the sensitivity, for detectors optimized for each frequency in the range shown. Abbreviations: SN—supernova; NS—neutron star; BH—black hole; $M_\odot c^2$—one solar rest mass (units for energy carried by waves); ϵ—the fraction of the mass of a black hole emitted in the waves. The assumed distances to the sources include 10 kpc—10 kiloparsecs (center of our galaxy); 50 kpc (Large Magellanic Cloud); 10 Mpc—10 megaparsecs (Virgo cluster of galaxies); 4 Gpc—4 gigaparsecs (the edge of the observable universe).

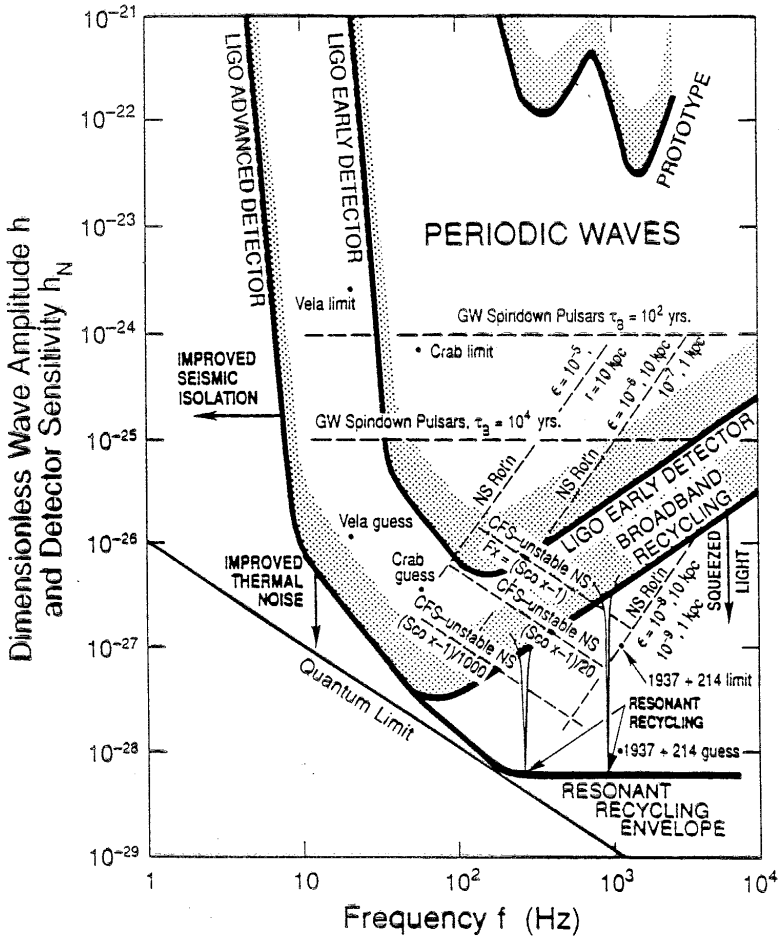


Figure A–4b The estimated dimensionless amplitude h for periodic gravitational waves from various sources (dashed lines and dots); and benchmark sensitivities (solid curves and stippled strips atop them) for interferometric detectors today and in the proposed LIGO. For each detector the solid curve corresponds to the dimensionless amplitude of a periodic gravitational wave, of previously known frequency, which would be detected at unity signal-to-noise ratio after integration for $\hat{\tau} = 10^7$ seconds, for optimal polarization and arrival direction. This amplitude, h_N, is related to the square root of the interferometer's spectral density of strain noise, $\tilde{h}(f)$, by $h_N = \tilde{h}(f)/\sqrt{\hat{\tau}}$ (see Equations (A.18), (A.19) and Appendix E). The top of the stippled strip corresponds to the dimensionless amplitude of a periodic gravitational wave of previously known frequency, with random polarization and arrival direction, which would give significant detection in 10^7 seconds. Stated more precisely, it corresponds to a confidence level of 90% that the signal is not a false alarm caused by a statistical fluctuation in Gaussian noise.

The lower right-hand part of the advanced detector curve indicates the envelope of peak responses of a resonant recycling detector when it is tuned for optimal operation at each frequency in the range shown. The slanting part of the advanced detector curve above this indicates the response of a broad-band recycling detector. The solid line near the bottom of the figure indicates limits to the advanced detector sensitivity curves set by the quantum li..it for the test masses.

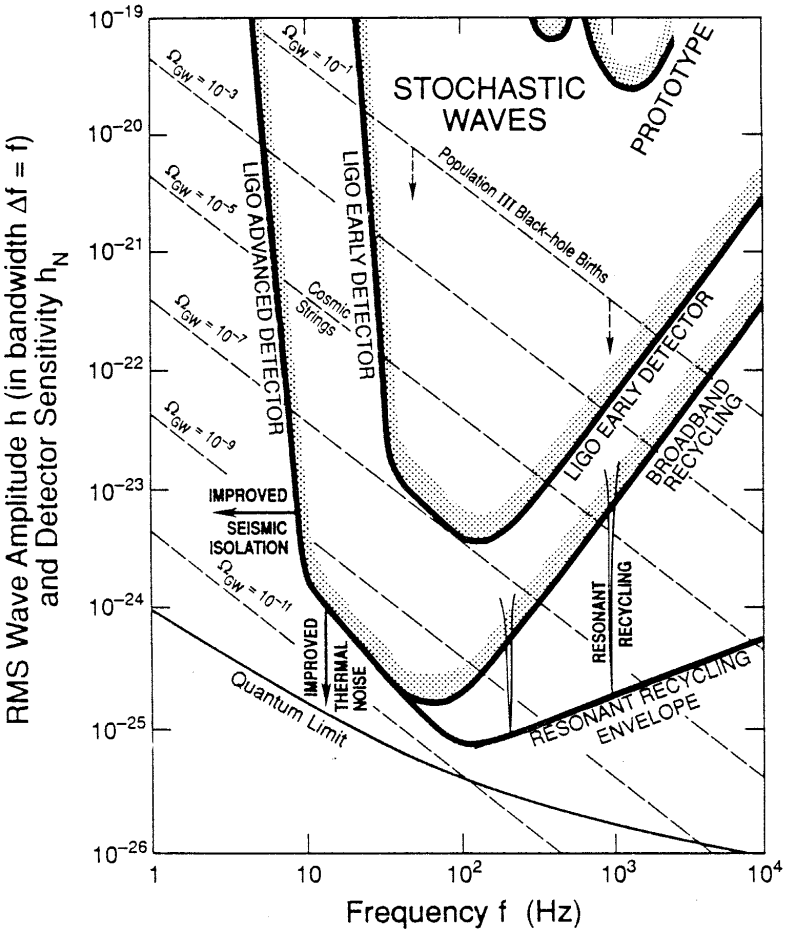


Figure A–4c The estimated rms amplitude h in a bandwidth $\Delta f = f$ for stochastic backgrounds of gravitational waves from various sources (dashed lines); and benchmark sensitivities $h_N \simeq \sqrt{5}\tilde{h}(f)[2f/\tilde{\tau}]^{1/4}[1 + fD/c]^{1/2}$ (see Equation (A.20)) (solid curves and stippled strips atop them) for interferometric detectors today and in the proposed LIGO. For each detector the solid curve corresponds to the rms amplitude in a bandwidth equal to the frequency for a stochastic background which would be detected at unity signal-to-noise ratio in a cross-correlation experiment between two interferometers at different LIGO sites, after integration for $\tilde{\tau} = 10^7$ seconds. The top of the stippled strip corresponds to the rms amplitude in a bandwidth equal to the frequency of a stochastic background which would give significant detection in 10^7 seconds. Stated more precisely, it corresponds to a confidence level of 90% that the signal is not a false alarm due to a statistical fluctuation in Gaussian noise. The lower right-hand part of the advanced detector curve indicates the envelope of peak responses of a resonant recycling detector system when tuned for optimal operation at each frequency in the range shown. Each of the resonance curves above this indicates response of a particular resonant recycling system. The solid line near the bottom of the figure indicates the limit to the advanced detector sensitivity curves set by the quantum limit for the test masses.

$$\left(\frac{\Delta \ell}{\ell}(f)\right)^2_{\text{geom}} \geq \frac{\left(\frac{\Delta \ell}{\ell}(f)\right)^2_{\text{Noise}}}{(\Delta f\, t_{int})^{1/2}}$$

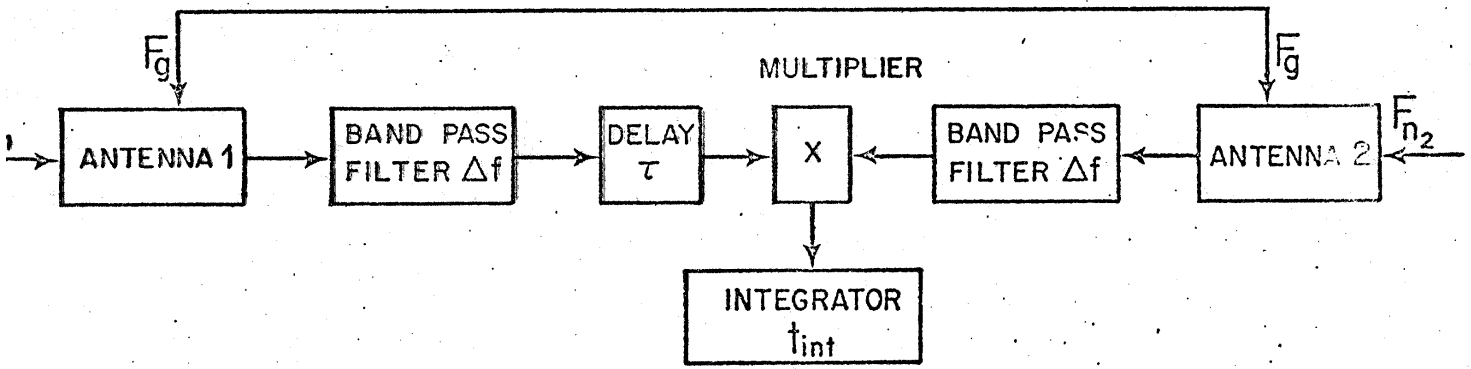
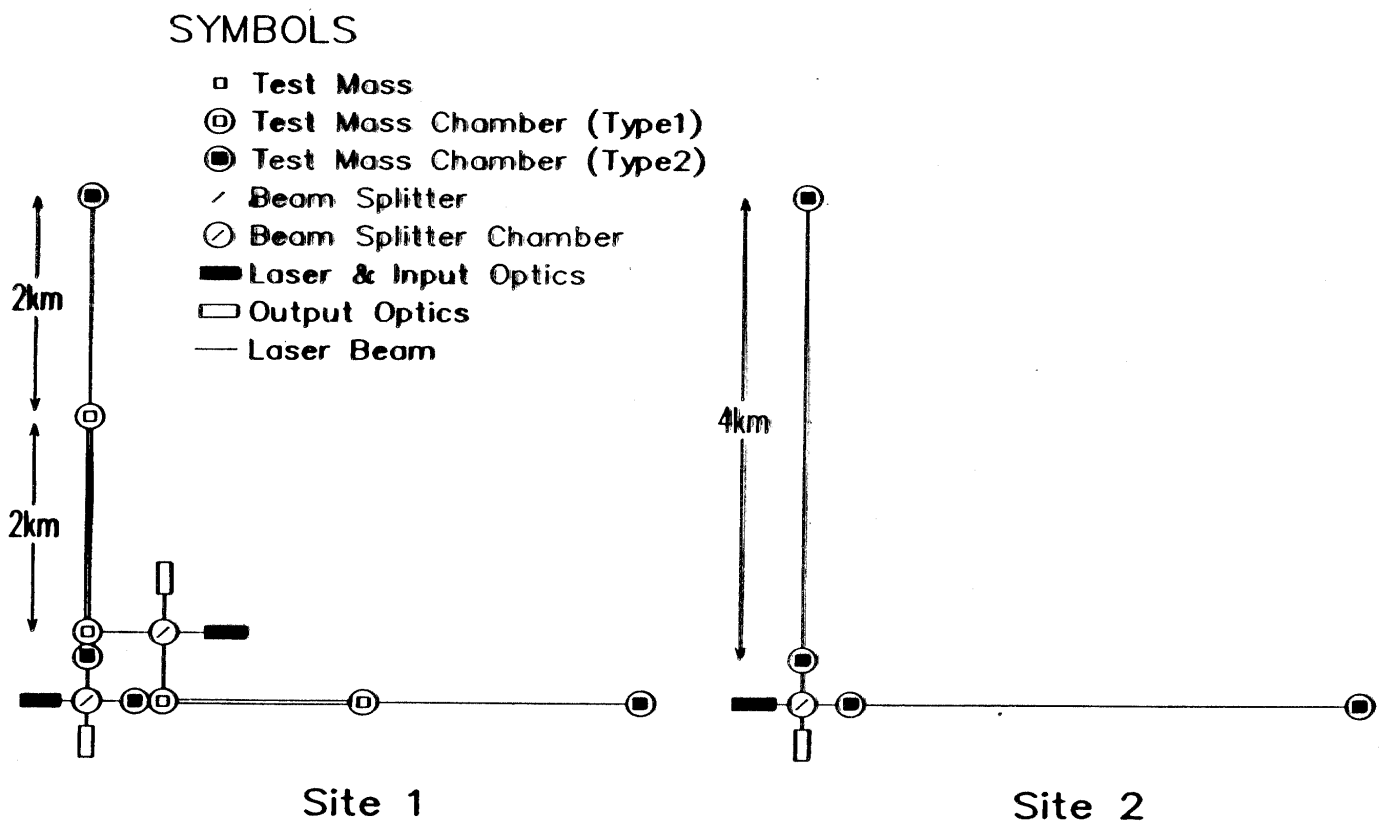


Figure 2. Schematic of a cross correlation experiment.


SYMBOLS
Test Mass
Test Mass Chamber (Type1)
Test Mass Chamber (Type2)
Beam Splitter
Beam Splitter Chamber
Laser & Input Optics
Output Optics
Laser Beam
2km
2km
4km
Site 1
Site 2

GRAVITATIONAL BURST DETECTION STRATEGY

Operation of interferometers at widely separated locations

Coincidence measurements: $R_{12} = \tau_w R_1 R_2$

$$\tau_w = \tau_p + 2D/c$$

$D >$ environmental noise correlation length

Operation of an environmental and instrument monitoring system

Reduce R_1 and R_2.

seismic noise monitor	acoustic noise monitor
magnetic field monitor	radio frequency interference monitor
cosmic ray shower monitor	electrical power transient monitor
residual gas column density	instrument housekeeping monitor

Operation of a half length interferometer at one site

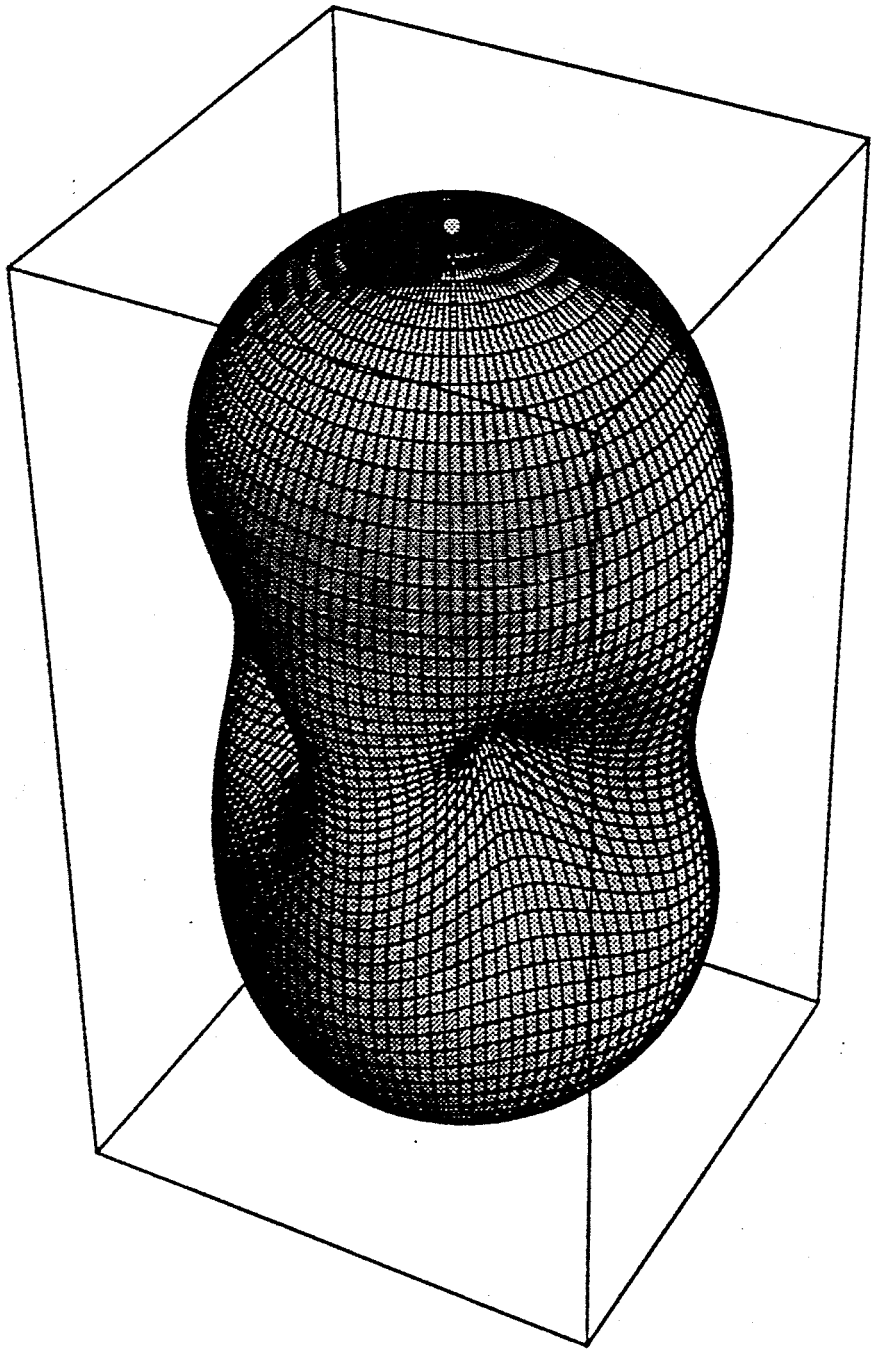
Gravitational wave signal proportionality to length as a discriminant.

Triple coincidence detection with some correlation due to common vacuum system and location.

$$R_{123} = (\tau_p + 2D/c)\tau_p R_1 R_2 R_3$$

correlations increase accidental triple coincidence by

$$\Delta R_{123} = R_{c1} R_2 (\tau_p + 2D/c)$$



IF GRAVITATIONAL WAVES ARE DETECTED!

Tests of general relativity

- Direct evidence for time dependent metric – waves
- Tests of strong field gravity – black hole signatures
- Spin of the graviton – polarization of the waves
- Rest mass of the graviton – propagation velocity

Different View of the Universe

- Most likely inner dynamics of processes hidden from eletromagnetic astronomies
- Cores of supernovae
- Dynamics of neutron stars and large scale nuclear matter
- The earliest moments of the evolution of the cosmic explosion – the Planck Epoch

Serendipity: A new instrument, a new field. The precedent is the unexpected.

Risks: Are the sources strong enough?
Will the technology work at the required level?

PAYOFF IS ENORMOUS

LIGO Data Types, Data Products
- Albert Lazzarini

LIGO Datastream Characteristics

- LIGO datastream consists of continuous broadband signals
  - Audio frequency (16384 samples/s, 16 bit) digitization & acquisition of key channels *(lower sample rates for ancillary channels)*
  - LIGO detection band: 40 Hz < f < 3 kHz

- No directionality
  - Require signal processing to deduce source locations
    - modulation of CW sources due to Earth motion
    - Time delay between coincident responses along 3000km baseline

LIGO Datastream Characteristics (cont.)

- LIGO signals expected to be at limits of detectability
  - Instantaneous SNR ~ 10^{-4}(strong chirp;pulsars) (weakest EM pulsars: SNR ~ 5 x 10^{-3}, require T_{int}~ hours to detect)
  - Need to integrate over entire (most) of the waveform to generate detectable SNRs (~ 10).
  - Requires coherent detection & signal processing
  - False alarms: validation of an "event" requires ability to preclude all other (terrestrial) interpretations -- vetoes

- LIGO acquisition data rates are high
  - Many parallel channels of instrumentation to monitor instrument behavior, environment, anthropogenic disturbances, ...
  - GW channel: 100kB/s for three interferometers (IFOs) -- 16384 samples/s @ 2bytes = 32kB/s per IFO
  - Data acquisition is ~ 10 MB/s for 3 IFOs
    - 573 channels *per* IFO (only 1 is GW channel)
    - 610 channels on physical environment monitors (PEM) (at each site)
  - Science (astrophysics) channels constitute as little as ~ [100 kB/s]/[10MB/s] ~ 1%

Initial LIGO Sources

Table 1: Initial LIGO Sources and Estimated Analysis Capability Requirements

	Sources	Initial LIGO Performance Estimate	Data Analysis Requirements		
			CPU	Storage	Comments
Burst Signals $\Delta T < 1$ s	Supernovae & Accretion-induced collapse of white dwarfs	$\Re_0 \sim 2-3$/ yr @ 15 Mpc If sufficiently asymmetric; however, ΔE_{GW} expected to be significantly less than 10^{-7} M_{solar}	Minimal	Minimal Need PEM/houskeeping data for veto	• On-line analysis desirable for correlation with other astrophysics: **EW** • **visible/radio/γ** • **ν** Gravity • VIRGO/GEO • Resonant bars • Waveforms unknown • 2x/3x IFO correlation of events
	BH/BH Collisions	$\Re_0 \sim 1$/ yr(?) @ 500 Mpc;			
Chirped Waveform $10s < \Delta T < 1000s$	NS/NS Inspirals	$\Re_0 \sim 3$/ yr (?) @ 23 Mpc; for $M_{NS} \sim M_{solar}$ $\Delta T \sim 36 \times T_{inspiral} = 360s$	~ 7.2 GFLOPS (WA)	Templates/Data ~5 GB /~24 MB	• On-line analysis appears feasible down to ~ 1 M_{solar} • 1x/2x/3x correlations feasible depending on SNR. • Coalescence event may generate correlated (EW) signals as above. • PEM/housekeeping needed for vetoing • Template matching (Wiener filtering) or wavelet analysis in f-t domain.
	BH/BH & NS/BH Inspirals	$\Re_0 \sim 1$/ yr @ 150 Mpc; for $M_{BH} \sim 3M_{solar}$ $\Delta T \sim 36 \times T_{inspiral} = 60$ s	~ 330 MFLOPS (WA)	~ 41 MB /~4 MB	

Initial LIGO Sources

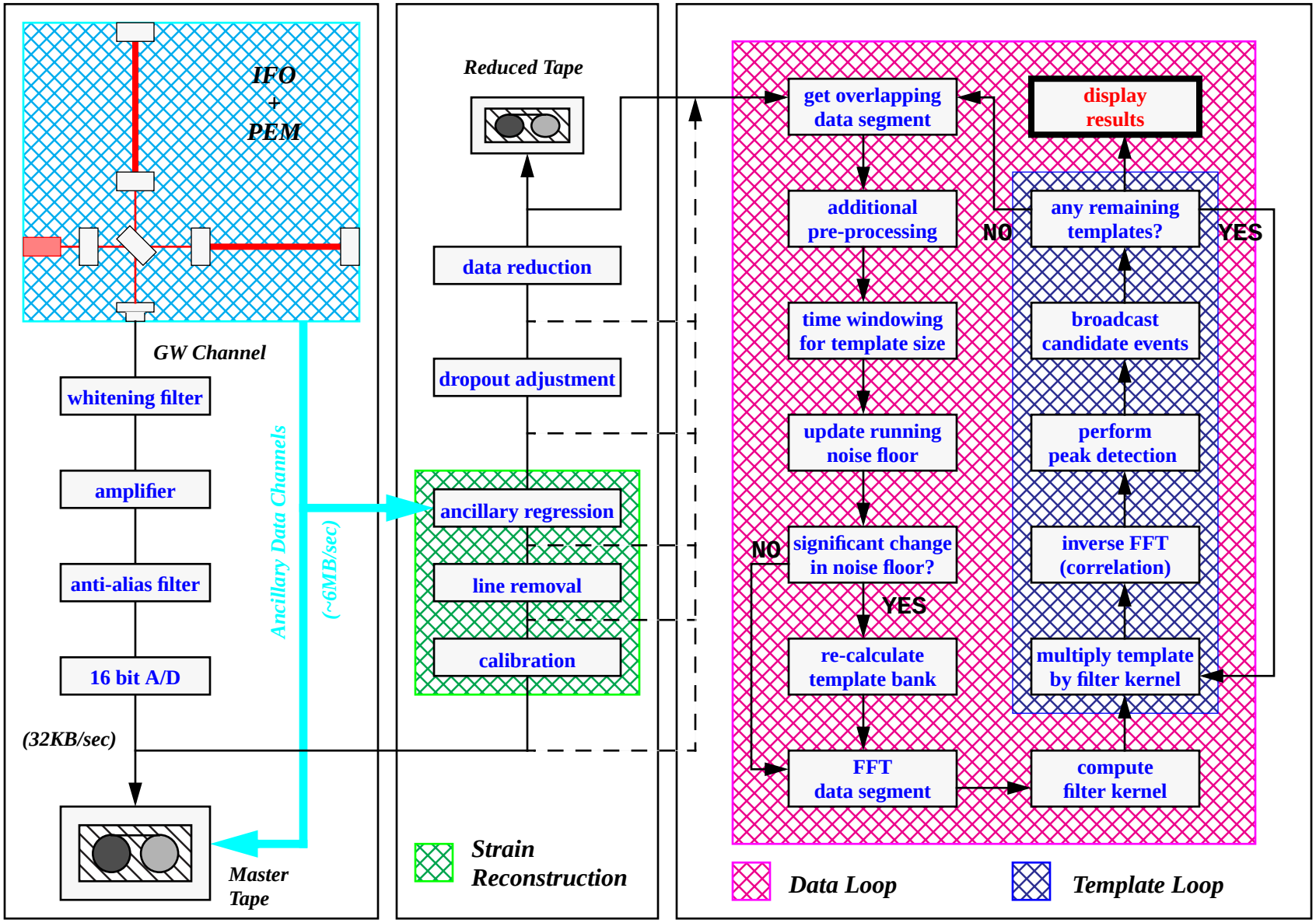
Table 2: Initial LIGO Sources and Estimated Analysis Capability Requirements

	Sources	Initial LIGO Performance Estimate	Data Analysis Requirements		
			CPU	Storage	Comments
Periodic Signal $\Delta T \sim 10^6 - 10^7$ s	Pulsars with mass asymmetry $\frac{S}{N} \approx 1.5\left(\frac{\varepsilon}{10^{-6}}\right)\left(\frac{10\text{kpc}}{r}\right)\left(\frac{1\text{ms}}{P}\right)^{\frac{5}{2}}\sqrt{\frac{T_{int}}{1\text{month}}}$ $\tau_{spindown} \sim 830\text{yr}\left(\frac{f_{rot}}{1\text{kHz}}\right)^{-4}\left(\frac{\varepsilon}{10^{-6}}\right)^{-2}$	$\frac{S}{N} \approx 8$ $\varepsilon = 10^{-5}$ r=10 kpc P=1 ms $T_{int} = 10^7$ s	Only directed searches feasible for nearby sources	10 GB for 10^6s (GW waveform)	• Off-line analysis • Detection less sensitive to non-Gaussian noise; more sensitive to calibration drifts. • Detection techniques as for pulsars -- narrow line sources with modulated frequency. • Correlations among interferometers may be performed (if needed) after detection. • A 4π sr. search requires decomposition of the sky into a very large number of pixels. Exact number is sensitive details of stacking.
Broadband Signals $\Delta T \sim 10^6 - 10^7$ s	Stochastic Background $\zeta \equiv \frac{\Omega_{gw}}{\Omega_0}$	$\zeta \geq 3\times10^{-6}$ $40\text{Hz} < f < 300\text{Hz}$ $T_{int} = 10^7$ sec	Minimal requirements -- analysis may done on single workstations; study of systematic correlated noise effects may require significantly more processing.		• Off-line analysis • Requires multiple interferometers to be correlated

LIGO Data Analysis
Challenges

- Techniques are those for detecting the possible presence of weak signals embedded in noise (radar, sonar, pulsar searches, ...):
  - Continuous processing of interferometer output
  - Parallelization
  - Frequency-domain spectral analysis (spectral cross-correlation)
    - Optimal matched filtering; [1-3] x 10^4 physics-based templates; 90+% of CPU time spent of Fourier transformations.
    - Frequency-time analyses (spectrograms, Wigner-Ville distributions, etc.); pattern/ridge detection (2-D)
  - Wavelet analysis; novelty detection; phenomenology
  - Kalman filtering to remove instrumental signatures -- data conditioning
- Data archival & Distribution
  - volume reduction by 10X => reduction/veto algorithms
  - access to archived data => database engine/network tools
  - 100% processing => computational power
    - LIGO inspiral search requires ~300 kFLOP/Byte of data
    - This is much higher than typical processing requirements
      - radar/sonar ~ 100 FLOP/Byte (many fewer templates, shorter durations!)
      - directed EM pulsar searches ~ 1 kFLOP/Byte
    - Compare: blind EM pulsar searches ~ 100-1000 kFLOP/Byte

LIGO Data Flow (Model)



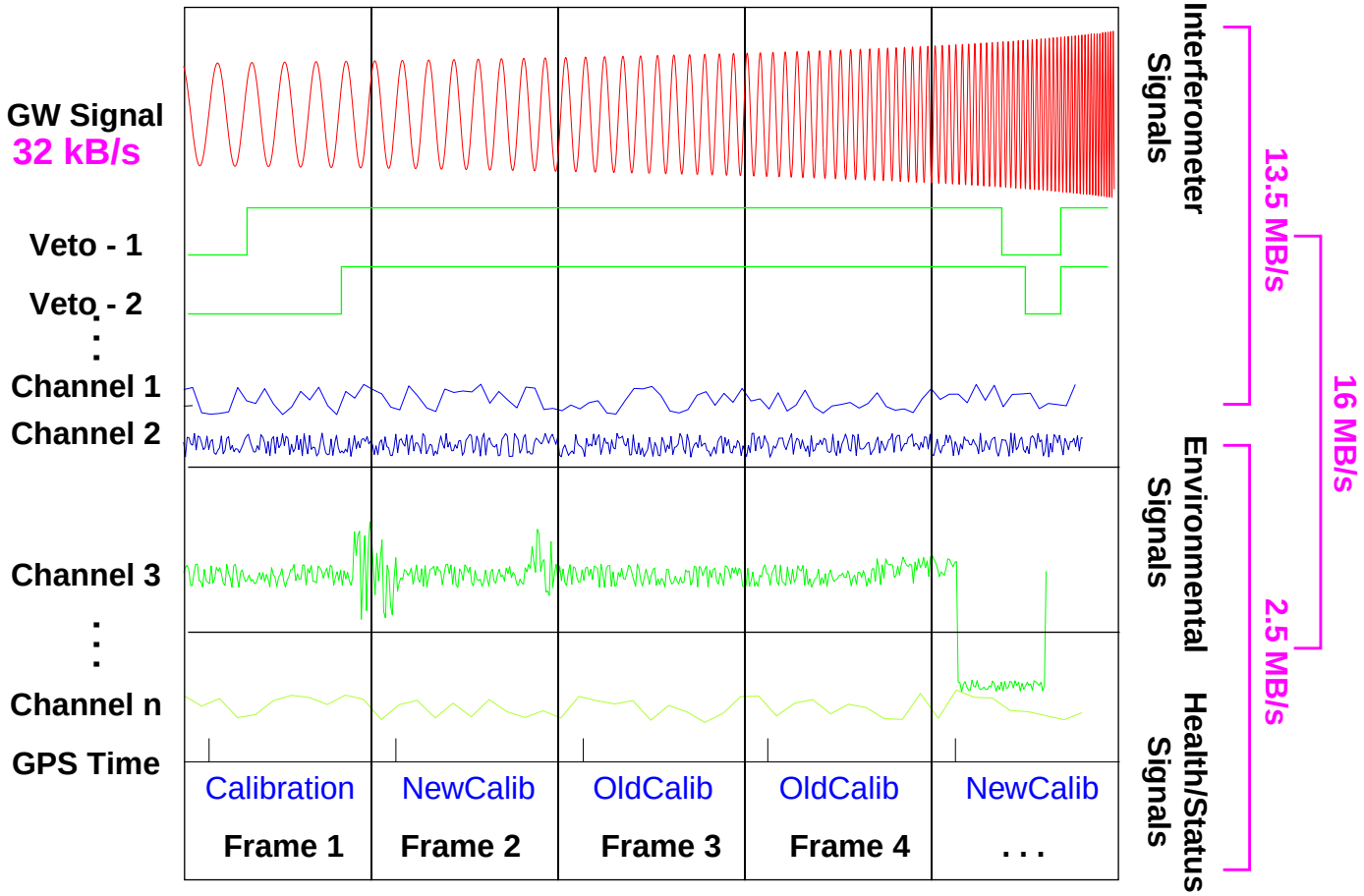
LIGO Data Products/Types

- The full [raw] detector datastream will be acquired and recorded as data frames.
  - Format for data frames has been unified with VIRGO in anticipation of being able to share software (now) and data (at some future date)
  - Other major interferometer projects have adopted standard
    - GEO
    - TAMA

LIGO Datastream

Frame Design

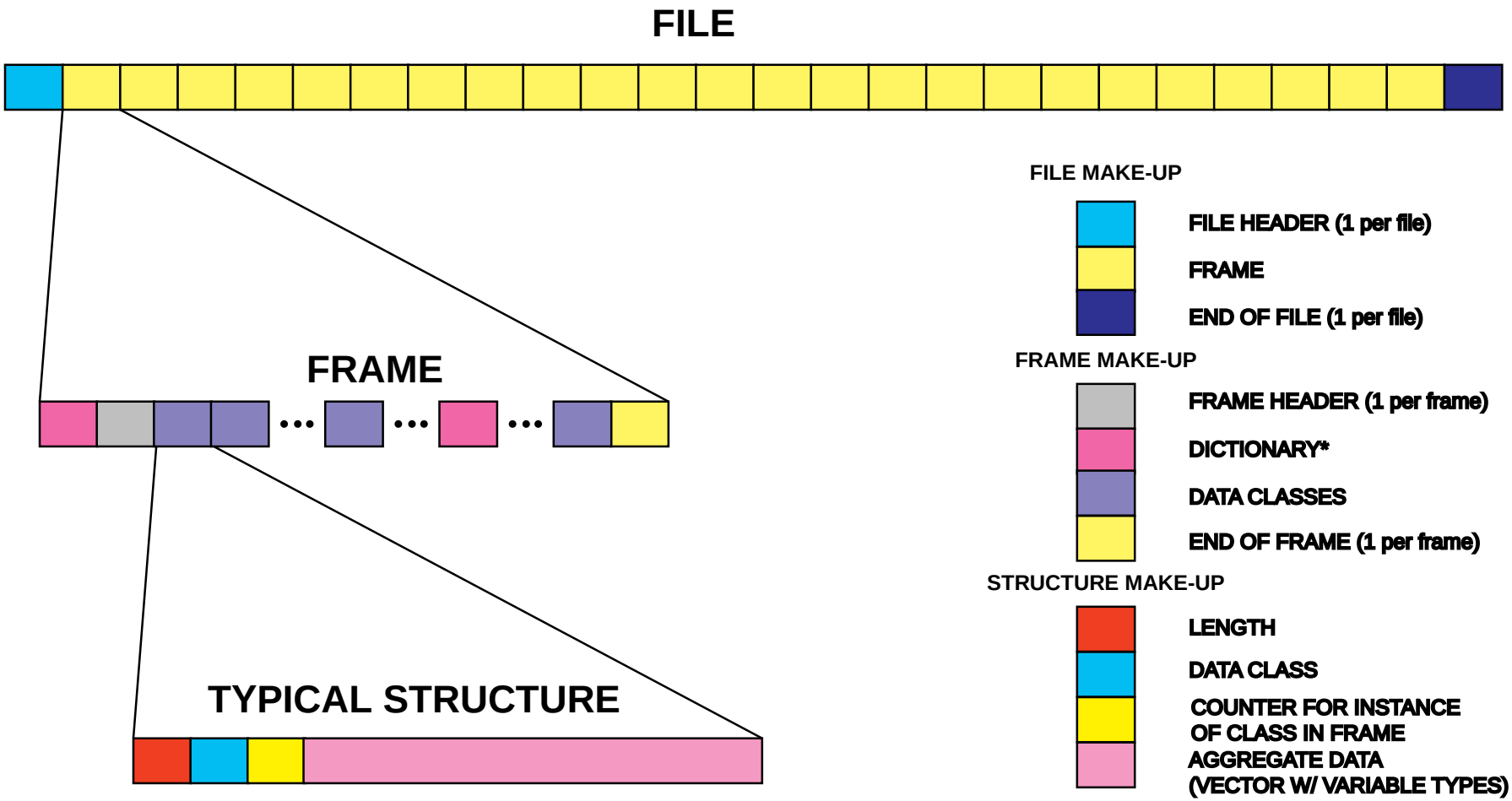
[Ref. LIGO Frame Format chart, B1 at end of talk]



- Frame is (structured) self-contained snapshot of data for a period of time
  - GW channel & ancillary IFO channels
  - Environmental monitoring (veto) channels
  - Facilities/Vacuum health & status
  - Hierarchical organization of data reflects IFO subsystems for more efficient veto utilization
- Full datastream could be ~ 300TB/yr
  - Plan to reduce (and compress) to ~ 50 TB/yr

Frame Format Implementation

Frame Composition



*** Dictionary structure behavior is unique in that:**
1. It preceeds header for first frame of file;
2. Dictionary is built up incrementally as addititional structures are incorporated into frame
3. It is valid for entire file (persistent)

Lightweight Data Format
XML

- Reduced, processed, or otherwise non-frame data will be recorded in a LIGO-standard lightweight data format (LigoLW)
  - Metadata (data about data: frame catalog indices, operator logs, textual data, etc.)
  - Event data [event specification still TBD]
  - Spectra, time series snippets, intermediate analyses performed with commercial/public-domain tools (MATLAB, Mathematica, ROOT, Triana, ...)
  - LigoLW is based on XML to anticipate web-distribution, network distributed processing
    - Metadata: tags, keywords, elements, attributes
    - Data: encoded binary; ASCII; raw binary(?); other objects; ...
- Need a lightweight format to complement frames:
  - interprocess data communications (@ socket level)
  - easily readable/parsable format for end users
    - quick-look products, single channels
    - spectra
    - plots
    - events
    - metadata
  - estimated data volume: ~600 GB/yr reduced data; ~135 MB/yr metadata

LDAS Reduced Data and Metadata - C1

SOURCE	Data	Data Types	Basis of size estimate #Parameters #Bins #Pixels #Samples	#Bytes/Unit	#/Hr	LW Data Volume/Year [GB]	MetaData Volume/Year [MB]
LIGO - Interferometer	Machine state vector	String[XML]	2048	1	10	0.0	90
		Binary	128	1	10	0.01	0.0
	Operator Logs	Strings	20480	1	20	0.0	180
		Graphics[JPEG]	32768	1	10	2.9	89.8
	Diagnostics	Video	4096	1	60	2.2	538.6
		Spectra/Fast Scopes	2048	2	20	0.7	179.5
		Calibrations - Spectra	2048	4	10	0.7	89.8
		Calibrations - Coeffi-cients	4096	1	10	0.4	89.8
		Calibrations - Matrices	2048	4	10	0.7	89.8
		Triggers/Discrete Logic	128	2	60	0.1	538.6
	Frame Data Catalog	String[XML]	1024	1	3600	0.0	64630.0
LIGO - Environment [PEM]	Facilities state vector	String[XML]	512	1	10	0.0	134.6
	Seismometers	Spectra	1024	2	60	1.1	538.6
	Magnetometers	Spectra	1024	2	60	1.1	538.6
	Tiltmeters	Time Series@0.1 Hz Stored 1/Hr	16	1	360	0.1	9.0
	Acoustic Sensors	Spectra	8192	2	60	8.6	538.6
	Diagnostics - Calibrations	Matrices/coefficients	2048	1	0.41666667	0.01	3.7
	Diagnostics - Triggers	String[XML]: Model parameters	1024	1	0.41666667	0.004	7.5
		Discrete logic	128	2	60	0.1	538.6

LDAS Reduced Data and Metadata - C2

SOURCE	Data	Data Types	Basis of size estimate #Parameters #Bins #Pixels #Samples	#Bytes/Unit	#/Hr	LW Data Volume/Year [GB]	MetaData Volume/Year [MB]
Non-LIGO	Seismic	String[XML]	512	1	10	0.0	89.8
	Electromagnetic storms	String[XML]	256	1	100	0.2	897.6
	Astrophysics - GRBs	String[XML]	256	1	0.04	0.0	0.4
	Astrophysics - neutrinos	String[XML]	256	1	0.00	0.0	0.0
	Astrophysics - visible	String[XML]	256	1	0.00011408	0.0	0.0
	Astrophysics - gravitational	String[XML]	2048	1	10	0.2	89.8
LDAS Events	Event Lists	String[XML]	2048	1	3600	64.6	32315.0
		Images/Graphics[GIF]	8192	2	3600	517.0	32315.0
				Total Database [GB]	== >	**600.8**	**134.5**

LigoLW
Example -- Metadata

```
<?xml version="1.0"?>
<!DOCTYPE LIGO_LW SYSTEM "Ligolw.dtd">
<LIGO_LW>
<!-- First the Metadata ------------------------------ -->
 <Metadata>
  <Creator>Tom Prince</Creator>
  <Creator>Roy Williams</Creator>
  <Date>28 Sept 98</Date>
  <Comment>LIGO power spectrum of 32 magnetometers at 64 frequencies</Comment>
  <Key>
   <Name>LIGOType</Name>
   <Comment>The Ligo data type is defined here...</Comment>
   <Value>Power Spectrum</Value>
  </Key>
  <Key>
   <Name>StartDate</Name>
   <Comment>Can't remember exactly but this date is close!</Comment>
   <Value>03/21/97</Value>
  </Key>
  <Key>
   <Name>FreqSamp</Name>
   <Unit>Hz</Unit>
   <Comment>This is the sampling frequency</Comment>
   <Value>1024</Value>
  </Key>
 </Metadata>
```

LigoLW
Example -- Data

```
<!-- Now for the Data objects ------------------------ -->
  <Object>
    <Name>Magnetometer</Name>
    <Array>
      <Dimension>64</Dimension>
      <Dimension>32</Dimension>
      <Type>double</Type>
    </Array>
<!-- This Array is at Cacr, Hanford, and on a tape -->
    <Link>
      <Encoding>bigendian</Encoding>
      <Timeout>600</Timeout>
      <Ref>file://hpss.cacr.caltech.edu/magval_09_25_97.bin</Ref>
    </Link>
    <Link>
      <Encoding>base64</Encoding>
      <Ref>file://hanford.ligo.caltech.edu/magval_09_25_97.bin</Ref>
    </Link>
    <Link>
      <Ref>tape://347846-6/756473</Ref>
    </Link>
  </Object>
   <Object>
    <Name>Magscale</Name>
    <Array><Dimension>32</Dimension></Array>
<!-- Embedded data -->
    <Data>
      1.28374 1.23453 1.94847 2.148474 2.39484 2.84746 3.10928 4.92827
      5.28374 5.23453 5.94847 6.148474 6.39484 6.84746 7.10928 8.92827
      9.28374 9.23453 9.94847 10.18474 10.3984 10.8446 11.1928 12.9827
      13.2874 13.2453 13.9847 14.18474 14.3984 14.8446 15.1928 16.9827
    </Data>
  </Object>
  <Object>
    <Name>Magoffset</Name>
    <Comment>This is the magnetic offset</Comment>
    <Array><Dimension>32</Dimension></Array>
<!-- Data follows from the end of the previous Object in the same stream -->
    <Follows/>
  </Object>

</LIGO_LW>
```

Database Management Systems
DBMS

- LIGO has four data types that need to be managed:
  - raw, framed data -- HPSS or equivalent network file system - ~100TB [2 years of data @ 10X reduction]
  - lightweight data -- HPSS or database management system (DBMS) - ~1 TB [2 years of data]
  - events (as they are generated, cataloged) -- DBMS -? GB
  - metadata -- DBMS - ~300 MB [2 years of data]
    - catalogs & indices
    - operator logs
    - trends and high-level descriptions of detector performance
    -
- Process of deciding DBMS for LIGO
  - Options to be considered:
    - relational [deemed sufficient for LIGO needs]
      - ORACLE (CIT license for campus MIS)
      - PostgresSQL (INFORMIX precursor; public domain - 'free')
      - miniSQL (similar to above)
    - object-oriented DBMS
      - Objectivity
  - Issues: Buy-in costs; operational costs; upgrades if we start too low; metadata only vs (metadata+data); ...
  - Want a decision by January 1999

Database Management Systems
DBMS

LIGO data flow

- Constraints
  - Data are generated at two observatories
  - Data are shipped [raw frames]/ transmitted[LigoLW,metadata] to Caltech
  - Data are processed at Caltech for reduction and ingestion into the archive
  - Shipping & transmission processes are a “steady-state” flow 365 days/year.
  - Data ingestion & reduction must keep up with data arrival -- around the clock prospect @ Caltech
- Availability
  - Very recent data [T < 16 hours] available only from observatories on-line disk cache
    - Framed, raw data
    - Metadata, lightweight reduced data from on-line processing
  - Recent data [16 hours < T< 2 weeks] may NOT be available (in transit, not yet ingested)
    - Framed data only as backup media at observatories -- retrieval could prove difficult
    - Metadata, lightweight data transmitted
  - Older data [3 weeks < T < 2 years] available only from Caltech archive

Database Management Systems
DBMS

LIGO data flow

- Consequences
  - Database management systems must be “aware” of different segments of the database and their locations
  - Seamless update of location metadata as data inventory moves around
- Users
  - Access to DBs must be provided to LIGO staff at all four sites: Hanford, WA; Livingston, LA; Caltech; MIT
  - In addition as LIGO comes on-line as an astrophysics facility, collaboration scientists will need to access databases
    - Syracuse
    - Penn State
    - Univ. Wisconsin
    - LSU
    - Univ. Fla
    - Univ. of Oregon
    - ... and growing

Database Management Systems
DBMS

- Data use model
  - Many data *PERUSERS*:
    - Trend data summaries.
    - Logs
    - Metadata
    - Download data to local workstation
  - Many (but fewer than *PERUSERS*) data *USERS*:
    - Diagnostics
    - Regressions
    - Modeling
    - Debugging
    - Algorithm debugging/prototyping with data
    - Download data to local workstation ...
  - Few data *ABUSERS*:
    - Gigapoint FFTs;
    - End-to-end analysis of 250 GB of data (3 month integration of the GW channel)
    - Regression of 10 x 10 channels for the last 6 months...
    - Process data at Caltech LIGO Archive using CACR resources

Database Management Systems
DBMS

- Unresolved issues for LIGO:
  - What fraction of data is stored in DBs and what fraction is on unix file systems?
    - Raw data [100TB] on HPSS or equivalent robot with ~1TB disk cache connected to servers
    - Metadata are in a DBMS, supporting queries/sorts/etc.
    - Lightweight data could be in either place ...?
  - Does LIGO need web-based DBMS licenses (expensive), or can we require users to set up local Caltech/LIGO accounts and rlogin/telnet/ssh in to use resources?
  - How many users will we have?
    - Plan for tens of perusers;
    - One dozen users;
    - <5 “abusers”;
  - Do we need “industrial-grade” security? At what price? At what inconvenience?
  - If we commit to tables-only relational DBMS solution, when (if ever) will we outgrow it?
    - Will we experience a graceful degradation or abrupt cessation of quality?
    - How complex will an upgrade be? Time consuming?

LIGO Data Analysis System Architecture
- Kent Blackburn

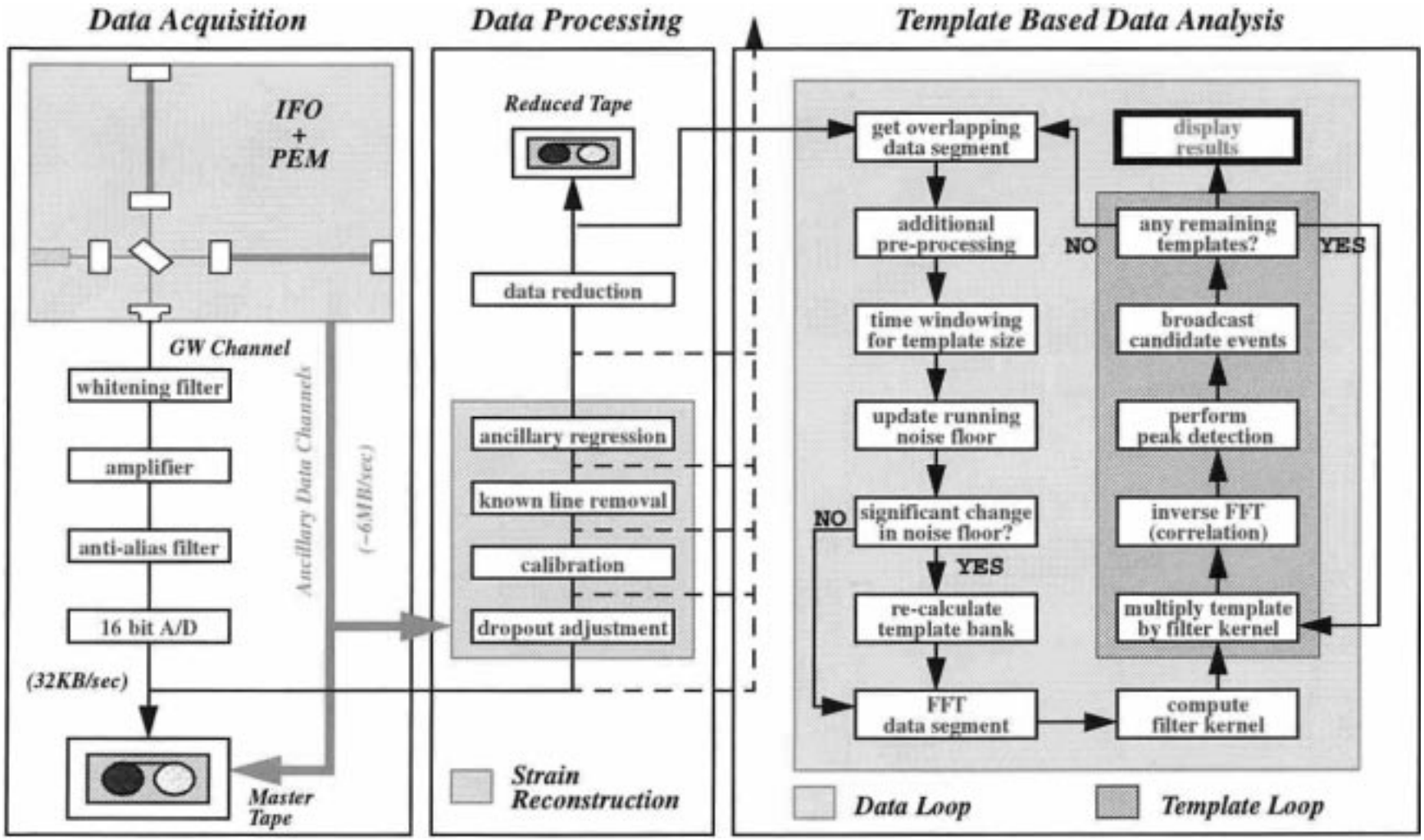
LIGO Data Analysis System "LDAS"

LDAS Database Workshop

October 22-23, 1998

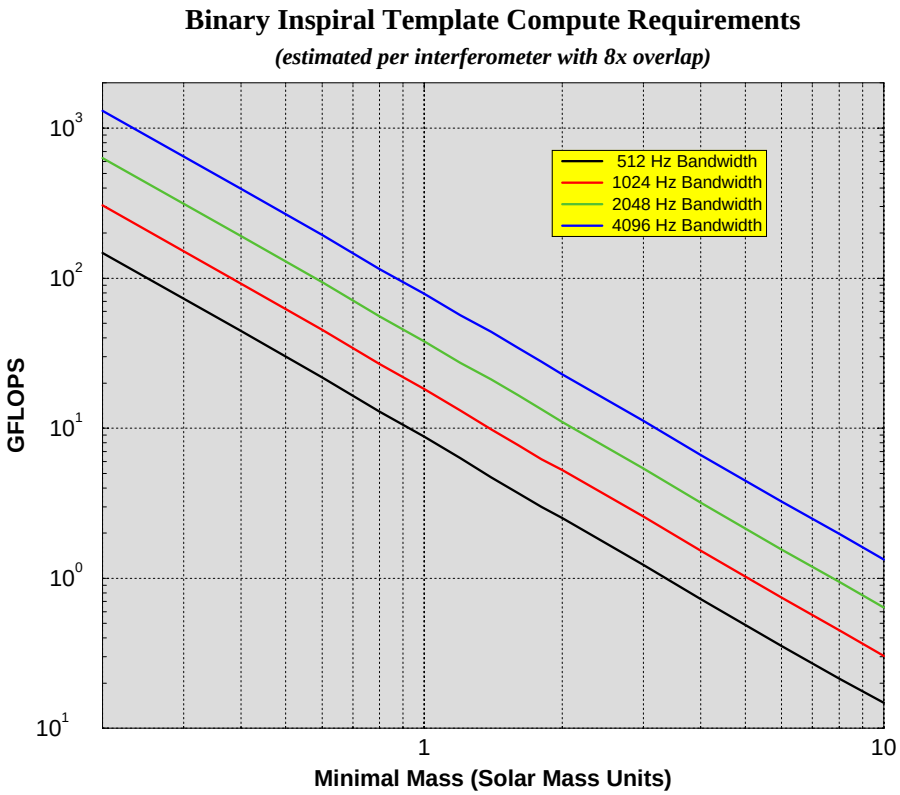
James "Kent" Blackburn

LDAS Data Flow Model

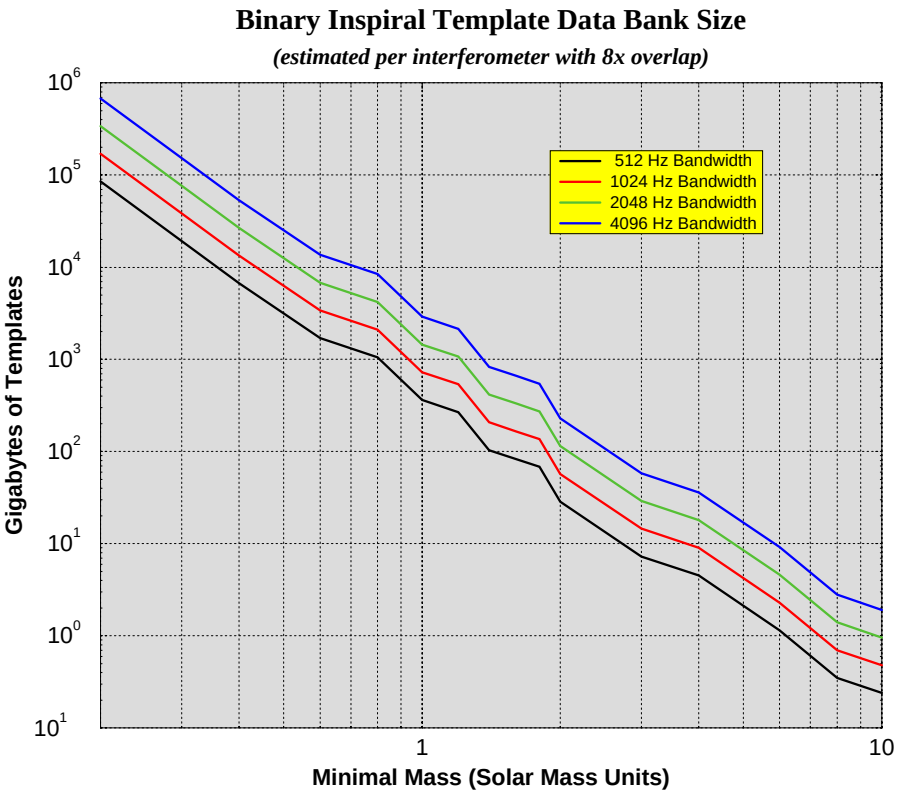


LDAS Binary Inspiral Search "Optimal Filtering Demands"

Computation:



Templates:



LDAS Hardware Design

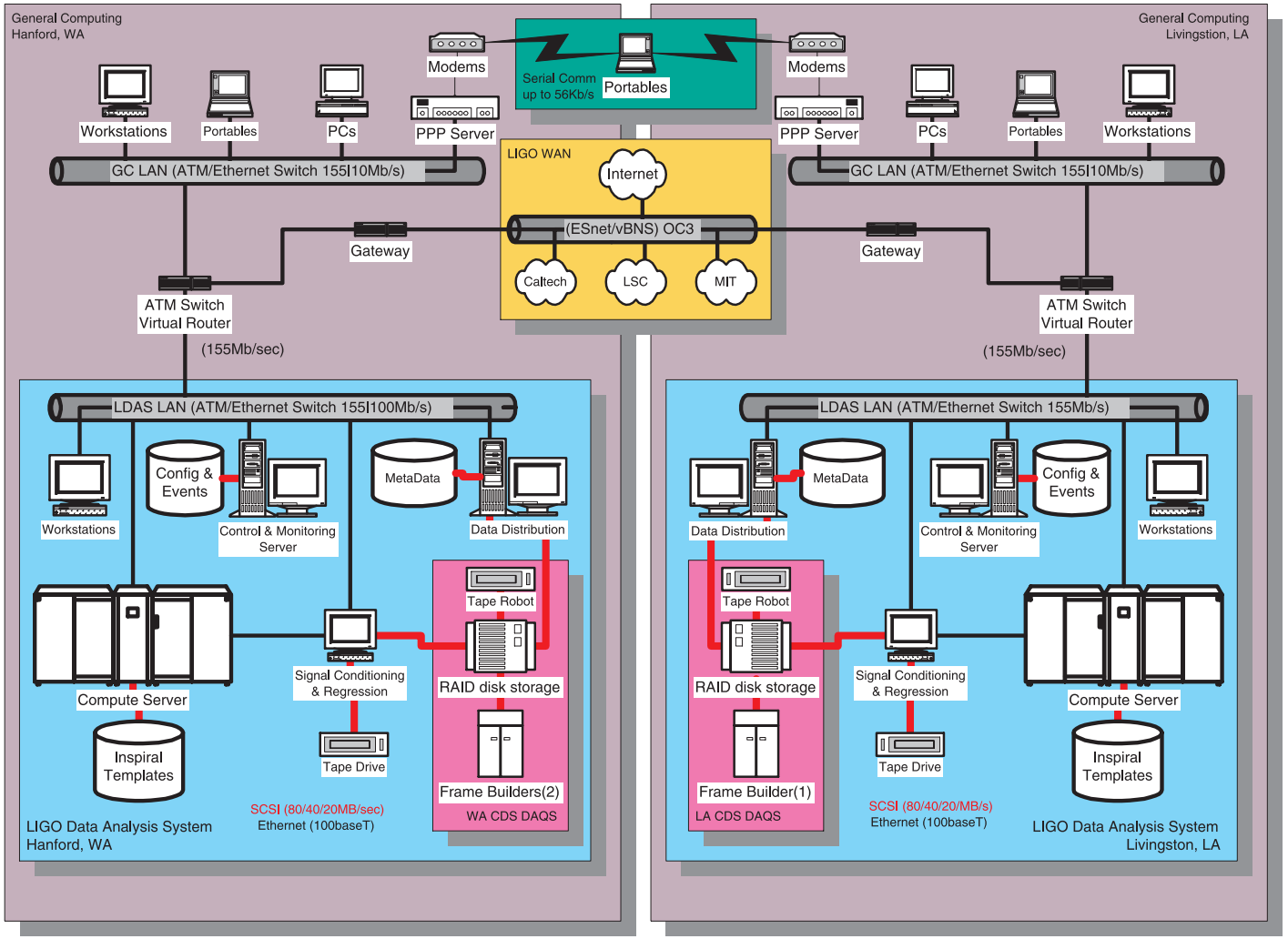
Requirements:

1. *10-100 GFLOPS of Compute Performance & 0.5-5 TB of uncompressed Templates*
2. *1-10 MB per second of point-to-point bandwidth*
3. *500 GB of On-Line Disk Cache*
4. *50-500 TB Archived Data per Year*
5. *50-500 GB of Metadata per Year*
6. *LAN & WAN Networks*

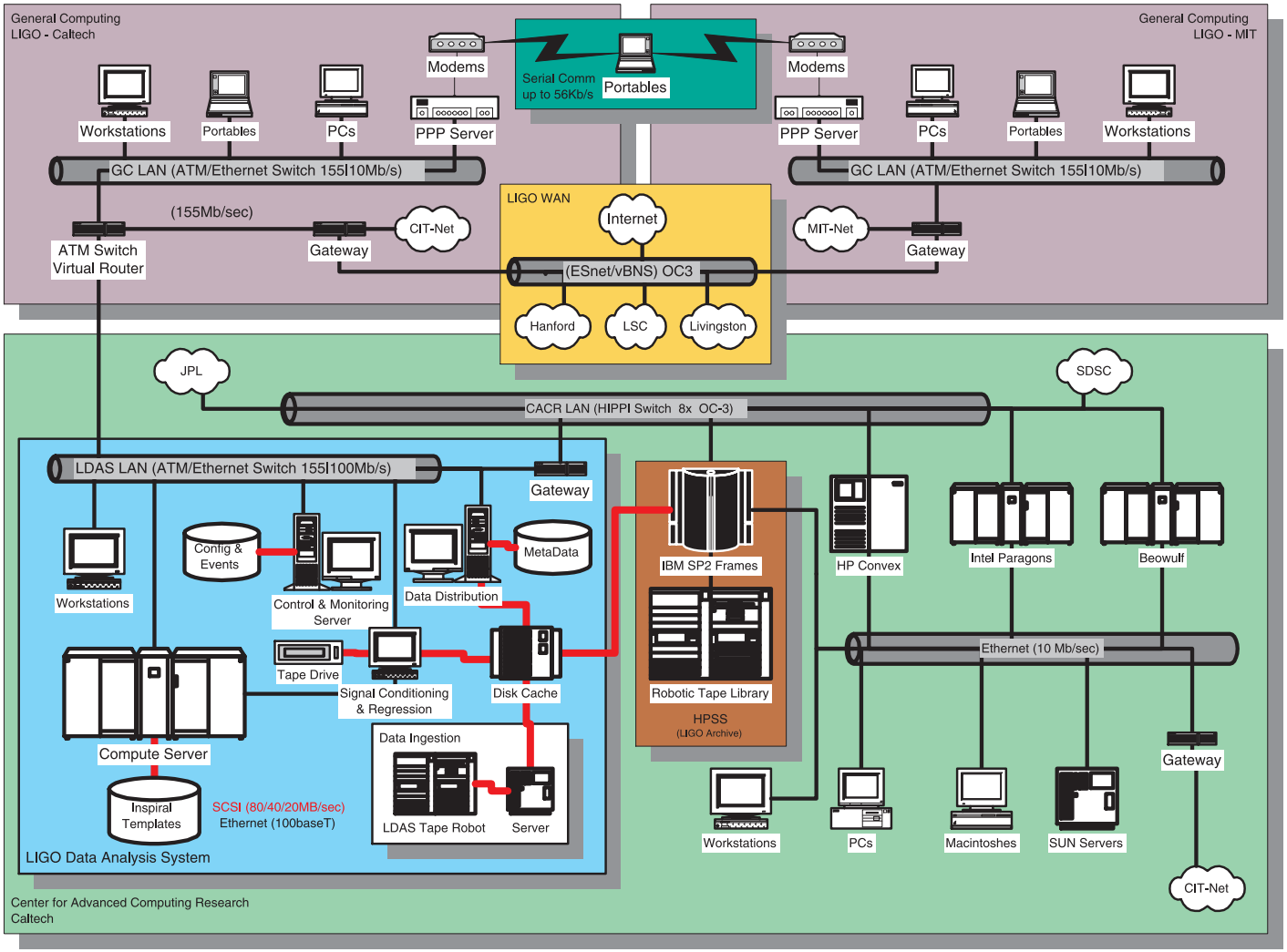
Components:

1. *Beowulf (PC clusters)*
2. *Switched ATM & Fast-Ethernet backbones*
3. *SCSI Hard Disk Storage*
4. *Optical Tapes ?!?*
5. *Database Capable of handling this Volume*
6. *LIGO LANs using ATM & Fast-Ethernet with ESNet and vBNS connectivity to the Observatories*

LDAS On-Line Hardware Architecture



LDAS Off-Line Hardware Architecture



LDAS Software Design

Requirements:

1. *Portability -*
   - POSIX
   - ANSI C/C++
   - TCL/TK
2. *Extensibility -*
   - OOP Design
   - Modular/Reusable Code
3. *Maintainability -*
   - Expressly Coded w/ OO Languages when possible
   - CVS Source Code Management
4. *Flexibility -*
   - Class Design
   - Modular Libraries
   - Distributed Processing

Components:

1. *Data Formats -*
   - Frames
   - Lightweight (metadata, Events, Templates, ...)
2. *Libraries -*
   - Data Format I/O
   - Numerical (FFT's, filters, etc.)
   - POSIX & Socket interfaces to OS
3. *API's -*
   - High Level Supervisors to Data I/O
   - Control, Monitor, and Management
   - MPI Level Communications
   - Filtering and Analysis
4. *UI's -*
   - TCL/TK (Wish Shell) GUI
   - TCL Interpreter Command Line
   - TCLet Plug-ins for Web Browsers

LDAS Data Formats

Frame Format:

1. Structured (C-like) Format
   - Samples of these structures:
     - ✔ ADC, Static, Detector, Trigger,
     - ✔ Simulated, History, Logs, ...
2. Jointly Developed with VIRGO
   - LIGO-T970130-B-E
   - VIRGO-SPE-LAP-5400-102
3. Original I/O Library in C
4. C++ Class I/O Library (FCL)
5. Primary Uses:
   - Data Acquisition
   - Data Archival
   - *Subsystem of Diagnostics*

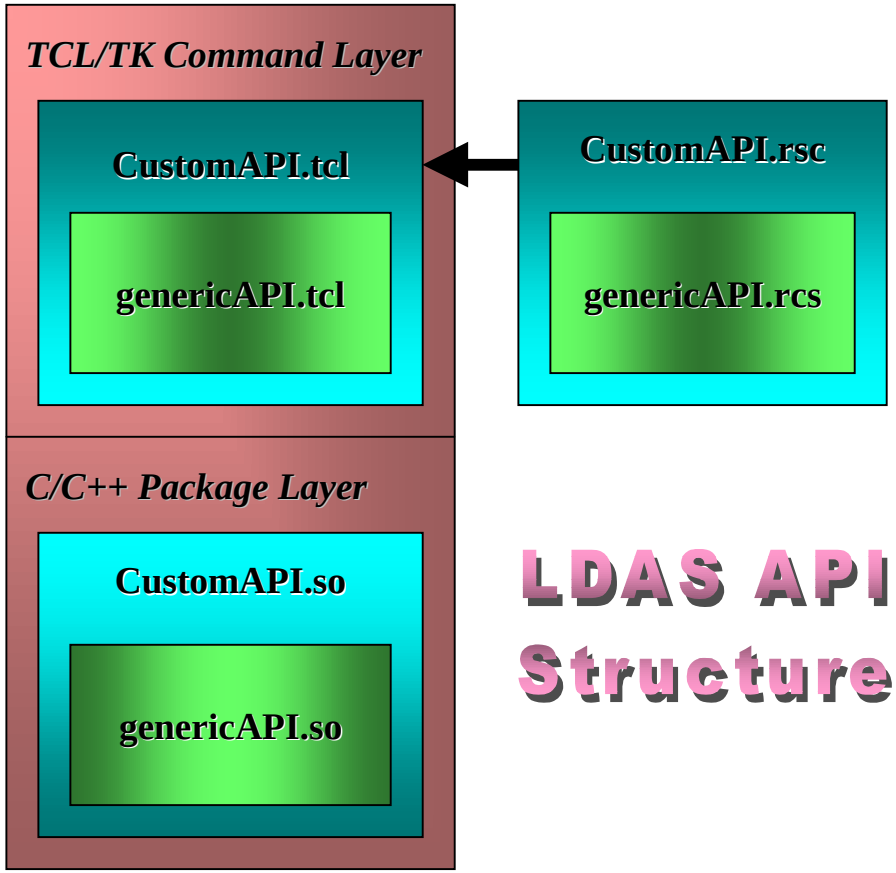
Lightweight Format:

1. Tagged (XML-based) Format
2. Easily Parsed *(and written)*
   - <int_s format="ascii">57 7 15</int_s>
3. LIGO Defined Objects included
   - tables (n-tuplets),
   - arrays (matrix, vector),
   - vectors (time-series, power-spectra)
   - Some Revisions expected
4. Uses Complement the Frame
   - LIGO Event data
   - LIGO Metadata
   - Spectra & Time-series data
   - Inter-process Communications

LDAS Layered API Design

LDAS API's:

1. Two Layers:
   - TCL/TK
   - C/C++ (extends TCL Language)
   - SWIG Unifies Layers
2. GenericAPI (core) Module:
   - Communications
     - TCL <-> C++
     - API <-> API
   - Common TCL proc's:
     - Help
     - Logging
     - Command Socket Management
     - Resource Management
   - Common C/C++ methods:
     - Data Socket Management
     - Internal Light-Weight Data Management
     - Class Save & Restore

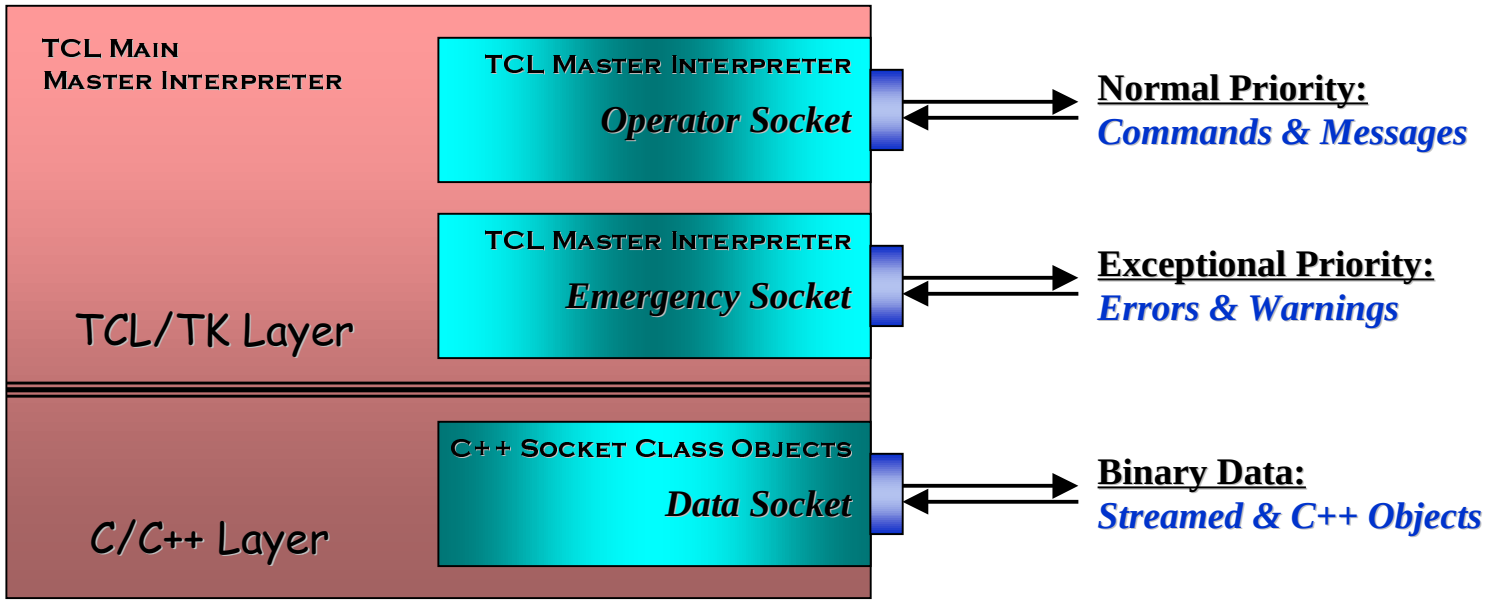


LDAS API Structure

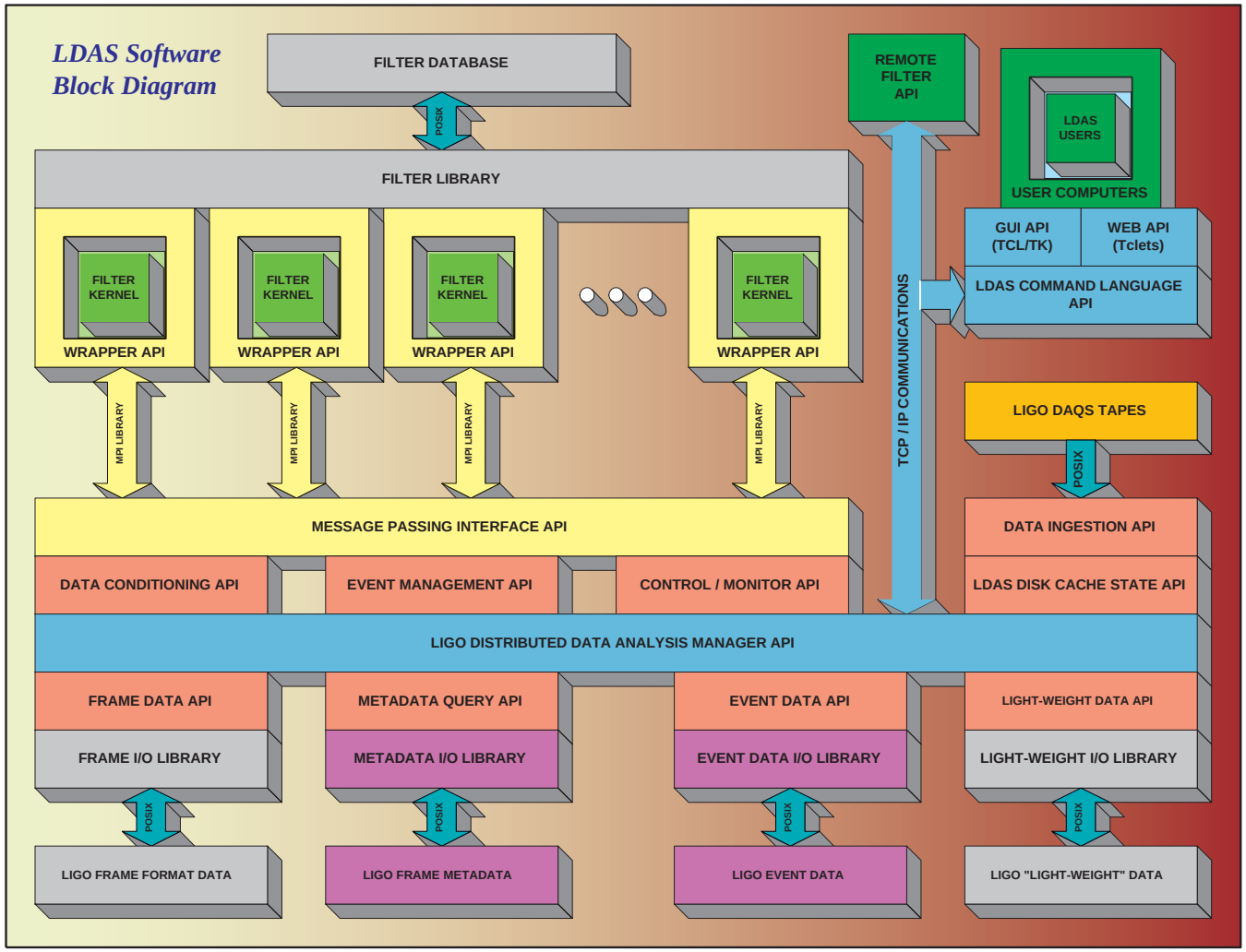
LDAS API Communications

3 Distinct Types of Socket Communications in API's:

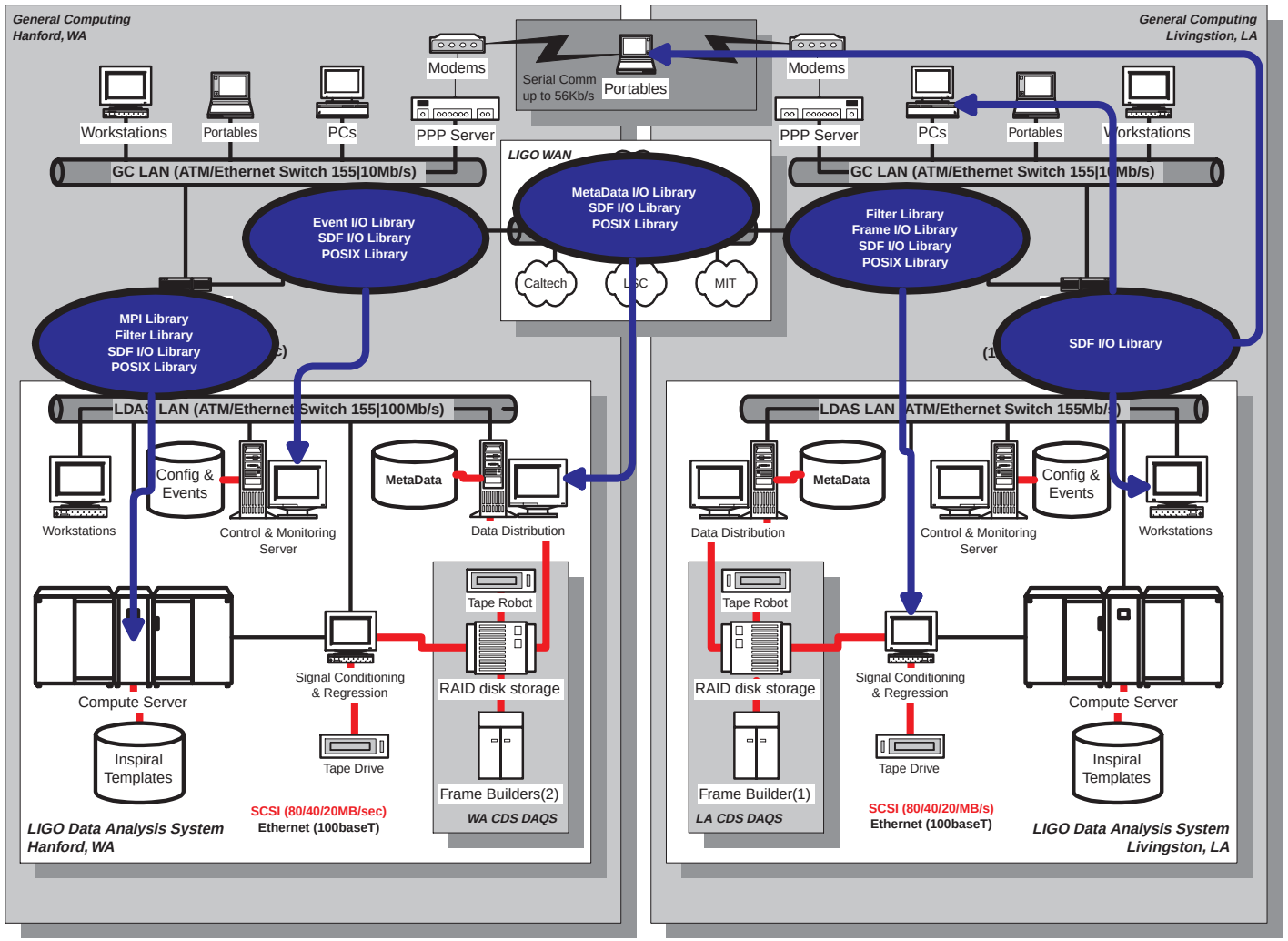
1. Operator Sockets - Normal Inter-process Commands & Messages
2. Emergency Sockets - Error & Warning Commands & Messages
3. Data Sockets - Binary Data in either Raw Streams or C++ Objects



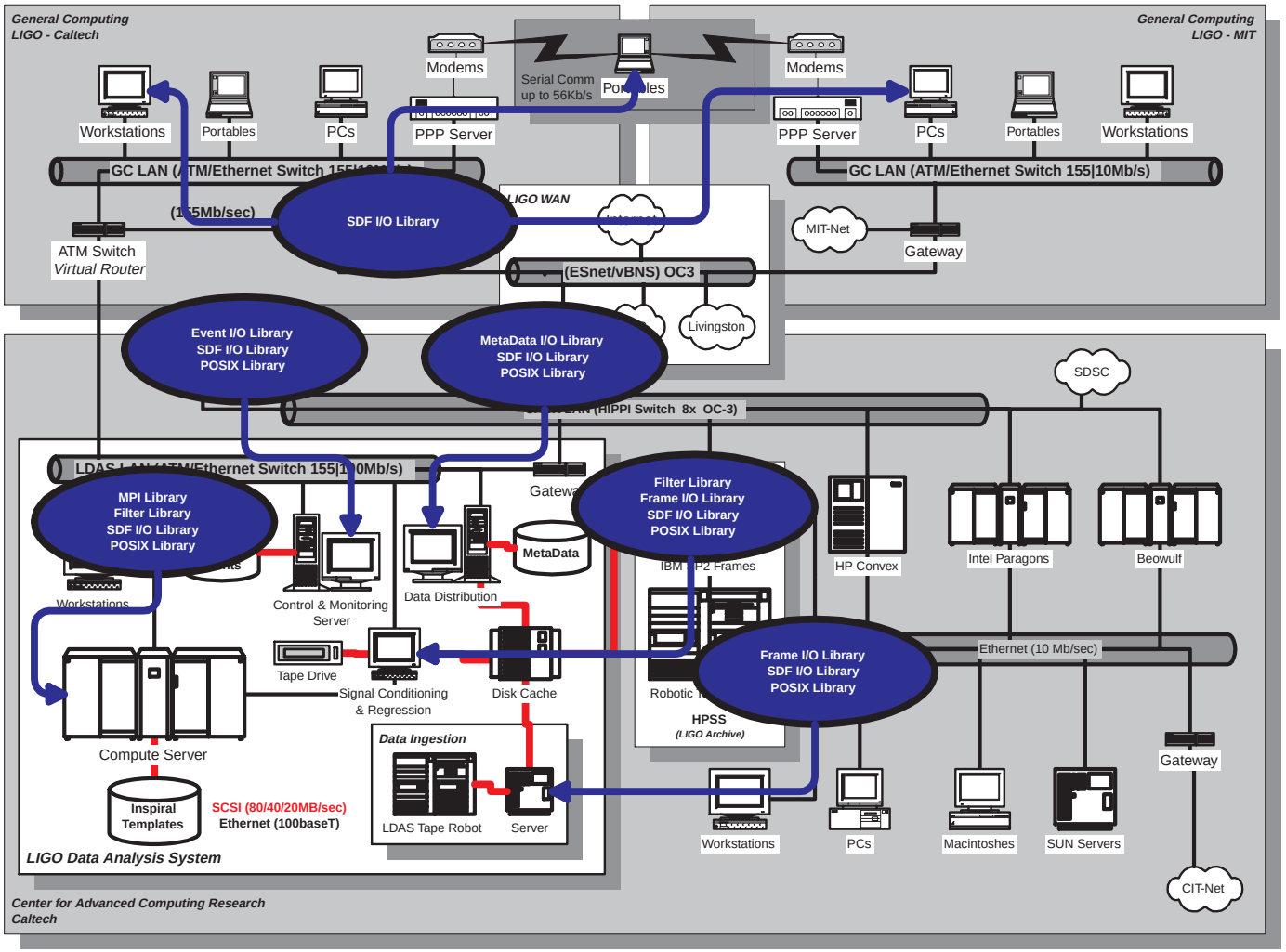
LDAS Software Block Architecture



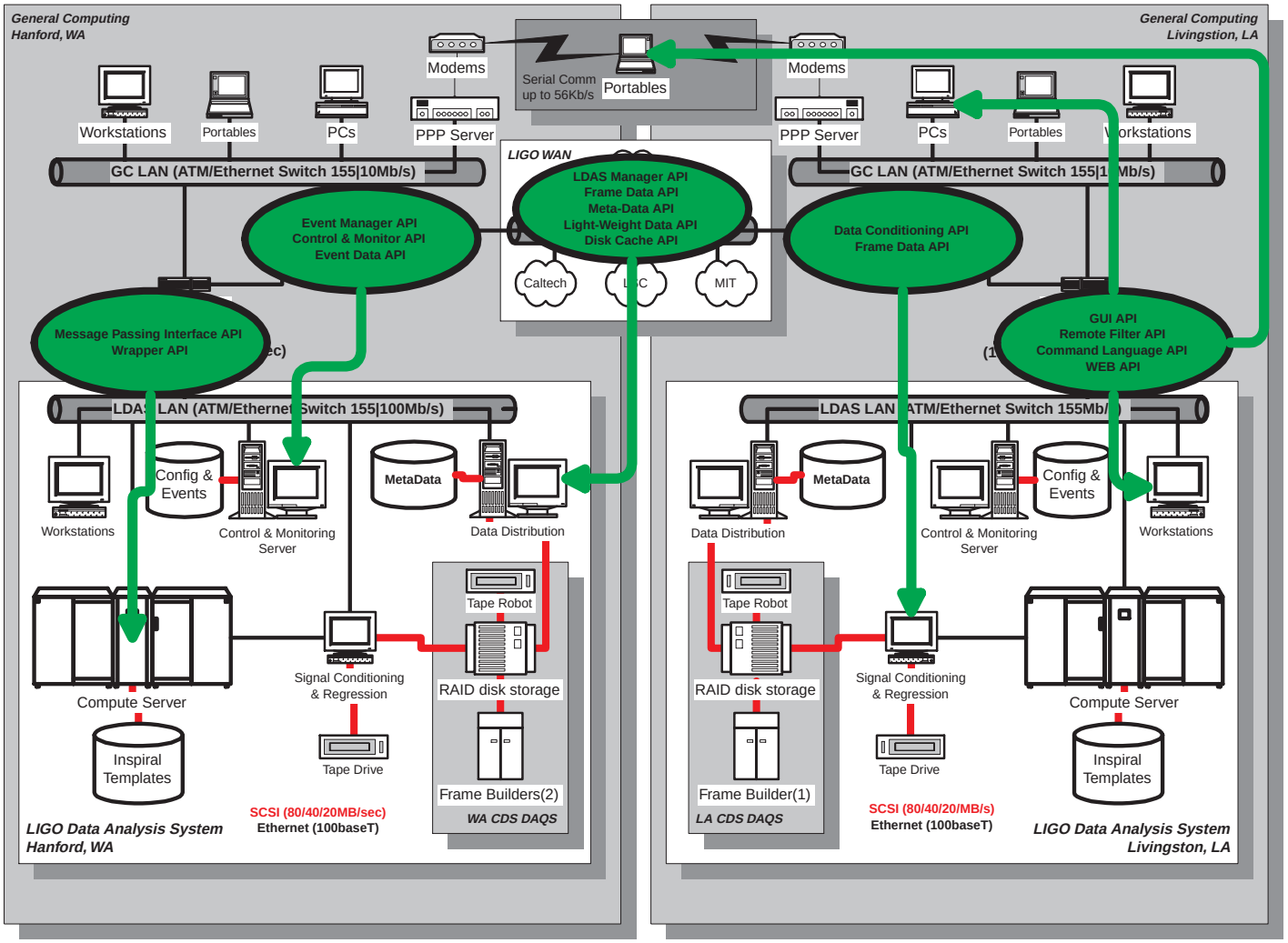
LDAS Software/Hardware On-Line Library Map



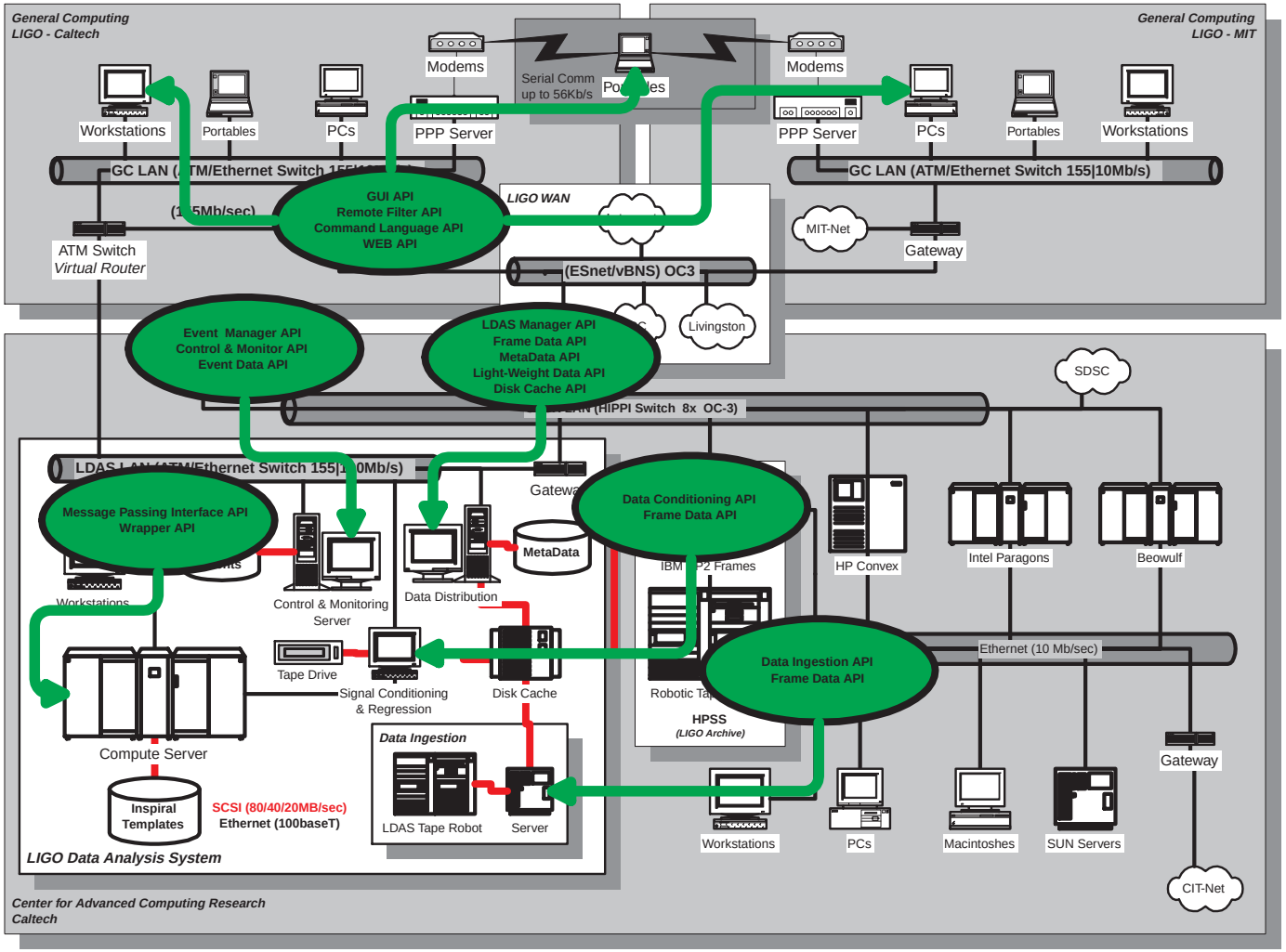
LDAS Software/Hardware Off-Line Library Map



LDAS Software/Hardware On-Line API Map



LDAS Software/Hardware Off-Line API Map



LDAS Software/Hardware Status

Hardware Design Complete:

1. Wide Area Network established to Sites
2. Expecting delivery of "canned" Beowulf System
   - Alta Technology's Altacluster 16 Pentium CPU w/ Extreme Linux
3. Data Distribution Server Hardware Identified

Software Design Complete:

1. GenericAPI component developed
2. FrameAPI, ManagerAPI being Specified
3. Expect to add DataConditioningAPI in time for Site Support
4. Myriad of Proto-typing activities underway
   - Web-based Data Distribution
   - Data-flow for Periodic Searches
   - Frame-to-XML Translation & Viewing
5. MetadataAPI and EventdataAPI components need **database** definition!

LIGO Data Archive
- Roy Williams

LIGO Data Archive

who, what, how, why

Roy Williams

Center for Advanced Computing Research
California Institute of Technology

Who are the Users of LDAS?

Peruser (60%)

Trends
Audio/Video/Heartbeat
Highly reduced data
Long-term summaries
Candidate events
Emphasis is Reliability and Ease of use
interface: Browser and GUI

Menu of Data

User (30%)

Data downloads
Mining
Exploration/Visualization
Applying filters from a menu
Custom summaries
Emphasis is Reliability and Functionality
interface: GUI and Command-line and Browser

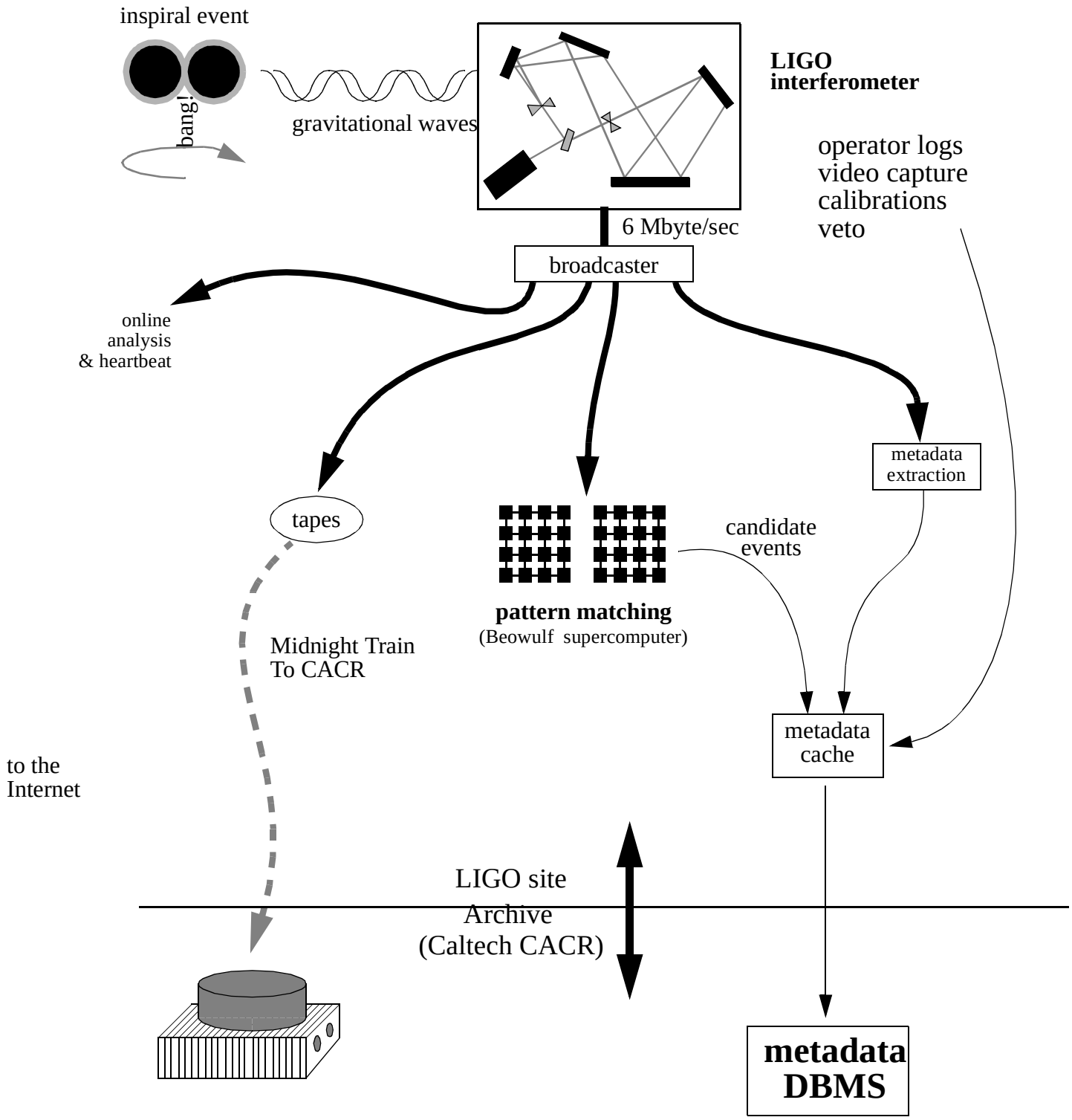
Menu of Data Customization

Implementer (10%)

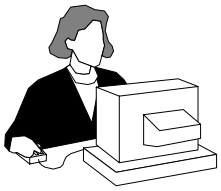
Algorithm development
Compiling and Linking
Stressing archive and computing
Emphasis is Reliability and Efficiency
interface: Command-line

I'll do it My Way

LIGO Data Ingestion



LIGO is a Library!



Visualization Software

Hardcore Code

Browser

X, xterm

Other Servers
seismic data
astrophysics data
weather data

http
telnet
"object"

Presentation Services

Scheduling

Summarize

Graphics

Format Conversion

Heterodyning

Catalog Services

Trending

Principle Components

Pattern Matching

parallel transfer

Foundation Services

Object Service

http Service

Object Service

http Service

NFS Service

HPSS Service

metadata DBMS

creation of reduced data products

5 year archive

Data Archive
(HPSS tape robot)

tape drive

Requests are Queries and Scripts (not Filenames)
Responses are Objects and Streams (not Files)

Query input

Keyword-value, for example in input to the frame catalogue
Tmin=1998_09_19_09_34
Tmax=1998_10_03_09_34
Tgap=3600

SQL-like
SELECT Channels.IFO, Channels.Lock AS Expr FROM Channels
WHERE ((Channels.Lock.Value>55) AND (Time > T0) AND (Time < T1));

XML
<Input><Link>jdbc://db.ligo.caltech.edu/.....</Link></Input>
<Query><SQL>SELECT Apples,Bananas from Fruit</SQL></Query>
<Filter><Heterodyne freq="500Hz"/></>
<Output><Format>Frame</Format>

Query Output is Objects and Streams

-- read by human, read by computer

Frames
Virgo-Ligo standard

Web pages
with graphs, tables, etc

LigoLW (an XML dialect)
Catalogue
Array, Table, Frame, MimeObjects

DAQD real-time protocol
other real-time eg. heartbeat monitor

Some Foundation Services

Frame files

Give me a file (equivalent to ftp)

FILE=14nov94.1/94-11-14-10-21-54

Give me the data and format that I want

CHANNEL=seismometer
FORMAT=SDF

Intelligent requests

Give me certain channels between Tstart and Tend
But only when conditions are satisfied
When there is a long contiguous chunk
In the format that I want

Metadata about the frames (the "Frame Catalogue")

When is the instrument working?

Tstart=76083737,Tend=76086574

produces a list of time intervals

76083737 76083739
76083743 76083748
76083760 76083769

Given a frame file, what are the names of the channels in it?

Multimedia

Images of beam shape

External events

Such as earthquakes, gamma-ray bursts
Each event has a time, a significance, and "other data", with XML table output

```
76083737 3.78 blah blah
76086458 2.33 blah blah
```

Candidate events from Ligo data,

Given a time interval T0,T1 what is in the collection
From which template set?
Authority information such as authors, likelihood, processing documentation

Instrument Calibrations and Settings

Hierarchical
Historical.
Collections of comments logged by operators

Bakeout database

"A Micro-Scale model of LDAS"

Online at http://www.cacr.caltech.edu/ligo/bakeout

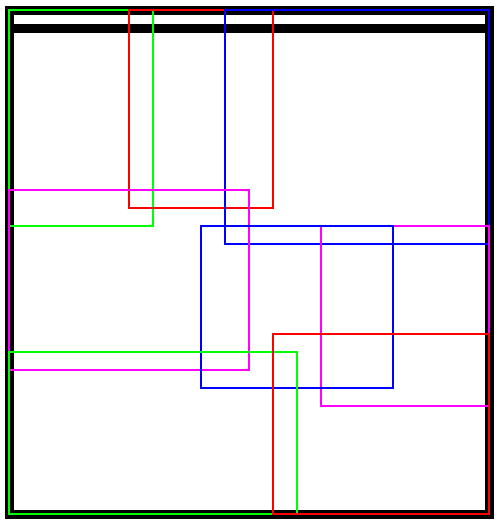
Data arrives each week as ~80 Mbyte MS Access database
representing a table with 700 channels, measured each minute (10000 rows)

Access DB is ingested, creating metadata and binary objects
Ingestion is administered via web interface
Takes 3 hours on WindowsNT machine with WebBase software

Foundation server runs on CACR web server
Stitches together different patches of the quilt (table)
Can run from a basic form interface to the **Foundation Server** or from the
Presentation Service (Tcl GUI) showing map of the beamtube

Foundation server is independent of Presentation server (Tclet)
Communication through agreed query language
Request: Tmin, Tmax, Tgap, list of channels
Response: Table of results in ASCII

Next: Next bakeout, Gas analyser, Weather, Operator logs



LigoLW

An XML language for Scientific Data Objects and Metadata

"A Human-Readable Object Serialization"

Observation: **Scientific data is** generally two parts:
Metadata (parameter files, file headers, lists of filenames)
+ **Data** (large, binary)

Metadata can be written as:
Name = Value (+ units + comment) **the "Key" element of LigoLW**

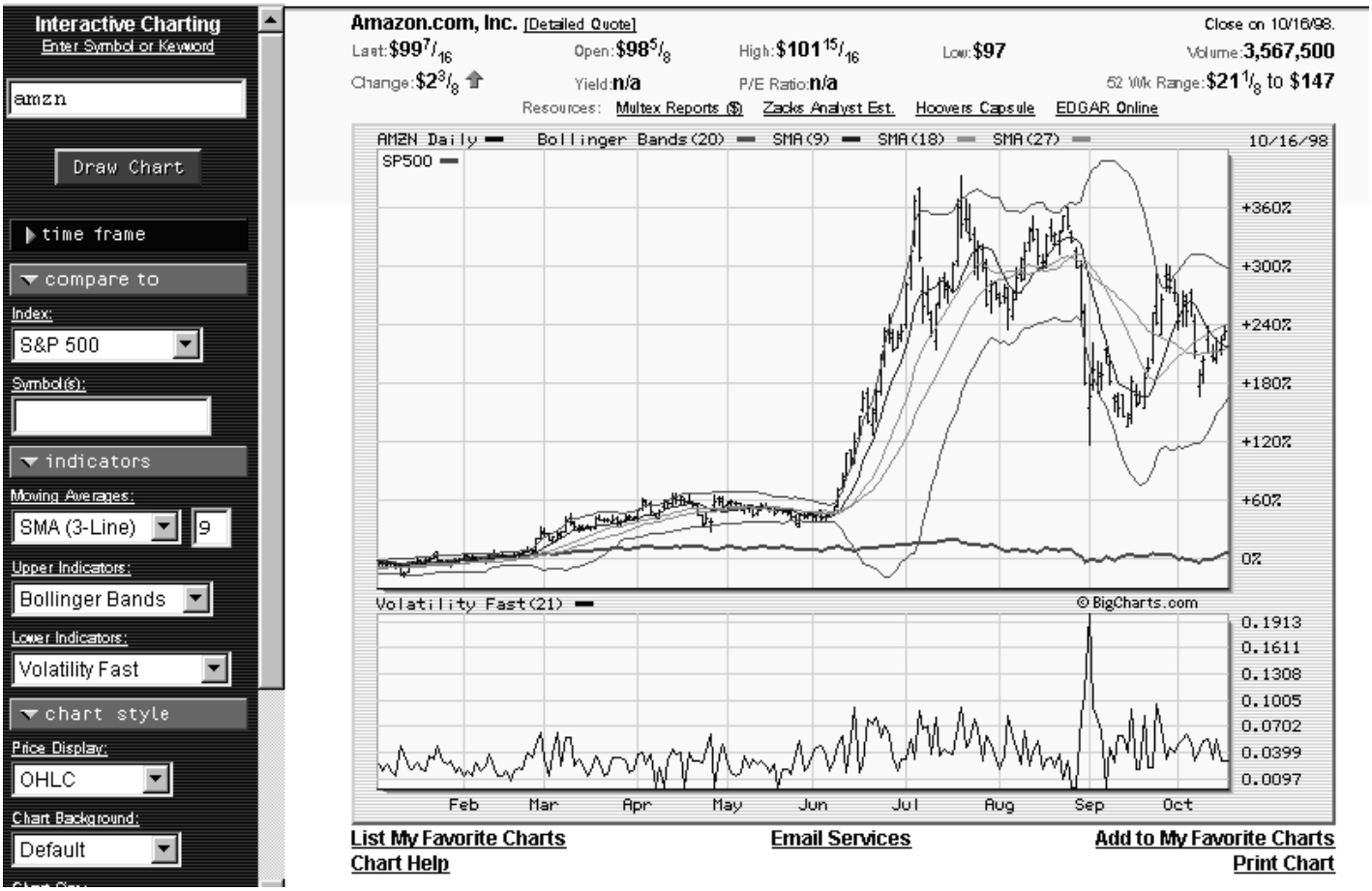
Data can be:
"pure" **binary**, eg 10^7 doubles
representing Array, Table, Frame etc.
a specific **file format** (eg. Fits, Frame, Jpeg)
inherited from MIME-typed files
-- or a collection of the above **the "Objects" of LigoLW**

LigoLW is good because:
-- It can be read by computer or human (also sorted, edited, browsed)
-- Based on industry-standard XML
-- Can be Metadata only or Metadata+Data
-- Flexible specification of data location and encoding
-- Can be used as "catalog card" for files in other formats
-- Many parsers in many languages available
-- Hierarchical, Extensible and Flexible
-- No special equipment necessary but sophisticated tools available

Status
Specified at theoretical level
LIGO T980091-00
http://www.cacr.caltech.edu/ligo/ligolw
First implementation (R.Williams & S.Anderson):
Pulsar detection stacks from 40-meter data converted
to LigoLW, plus parser and browser.
In progress:
Extend LigoLW with Frame metadata (W.Majid)
Next:
Bakeout data delivered as LigoLW tables
Frame catalogs from 1997 runs delivered from DBMS
as LigoLW
Firmer specification 99Q1 in light of experience.

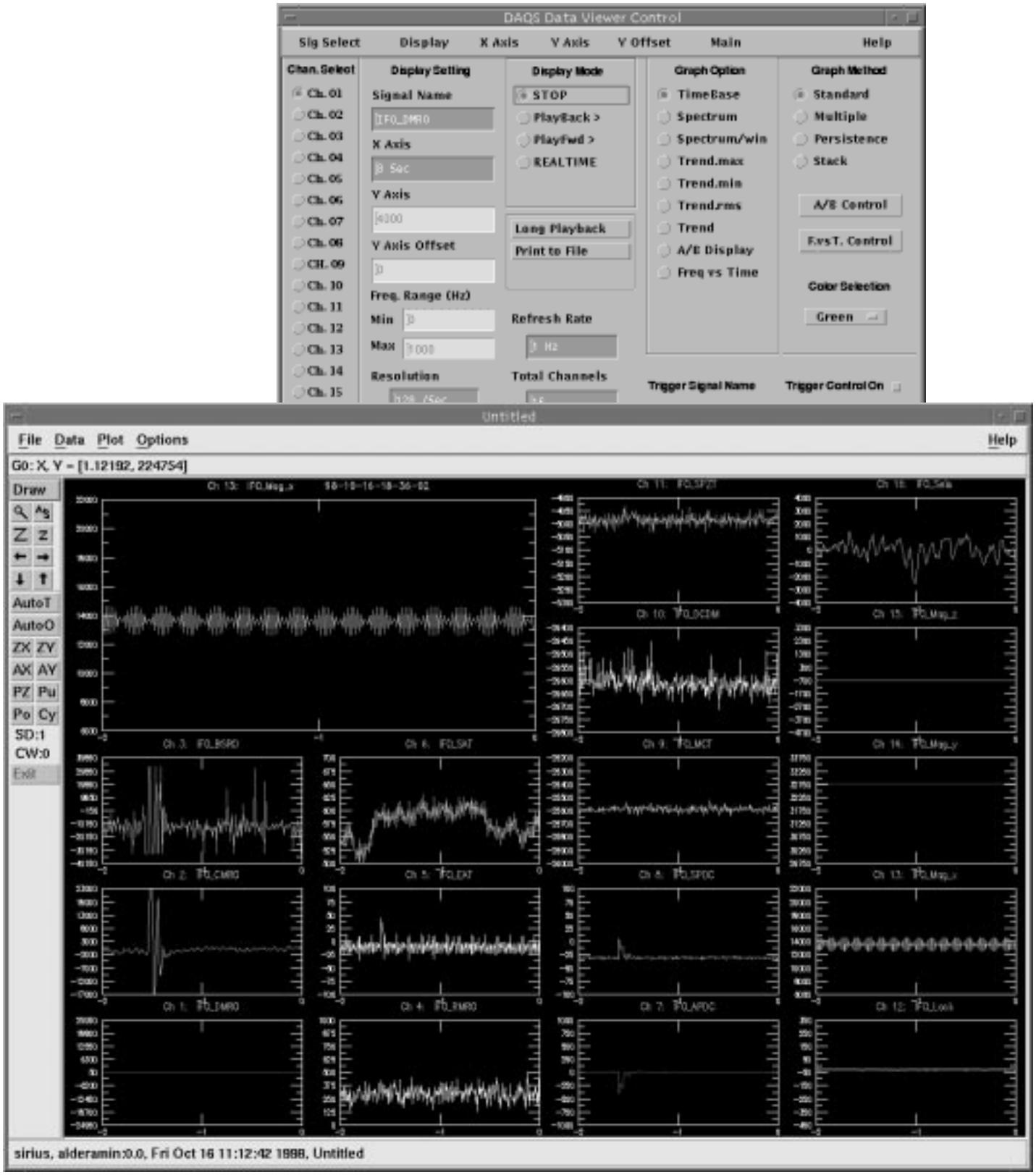
LIGO Visualizations

Like this?



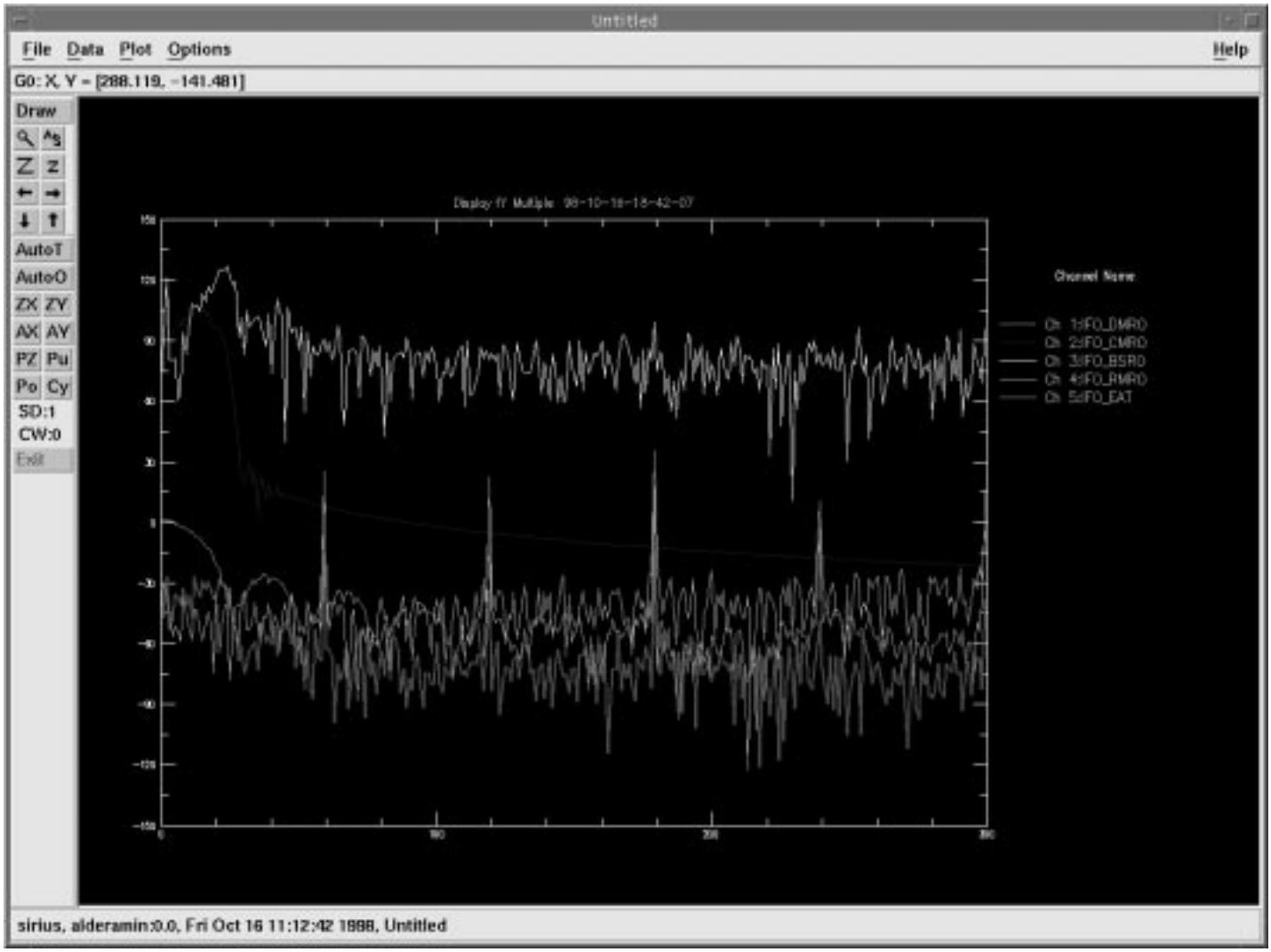
Data Viewer

(H. Ding, LIGO Data Acquisition)



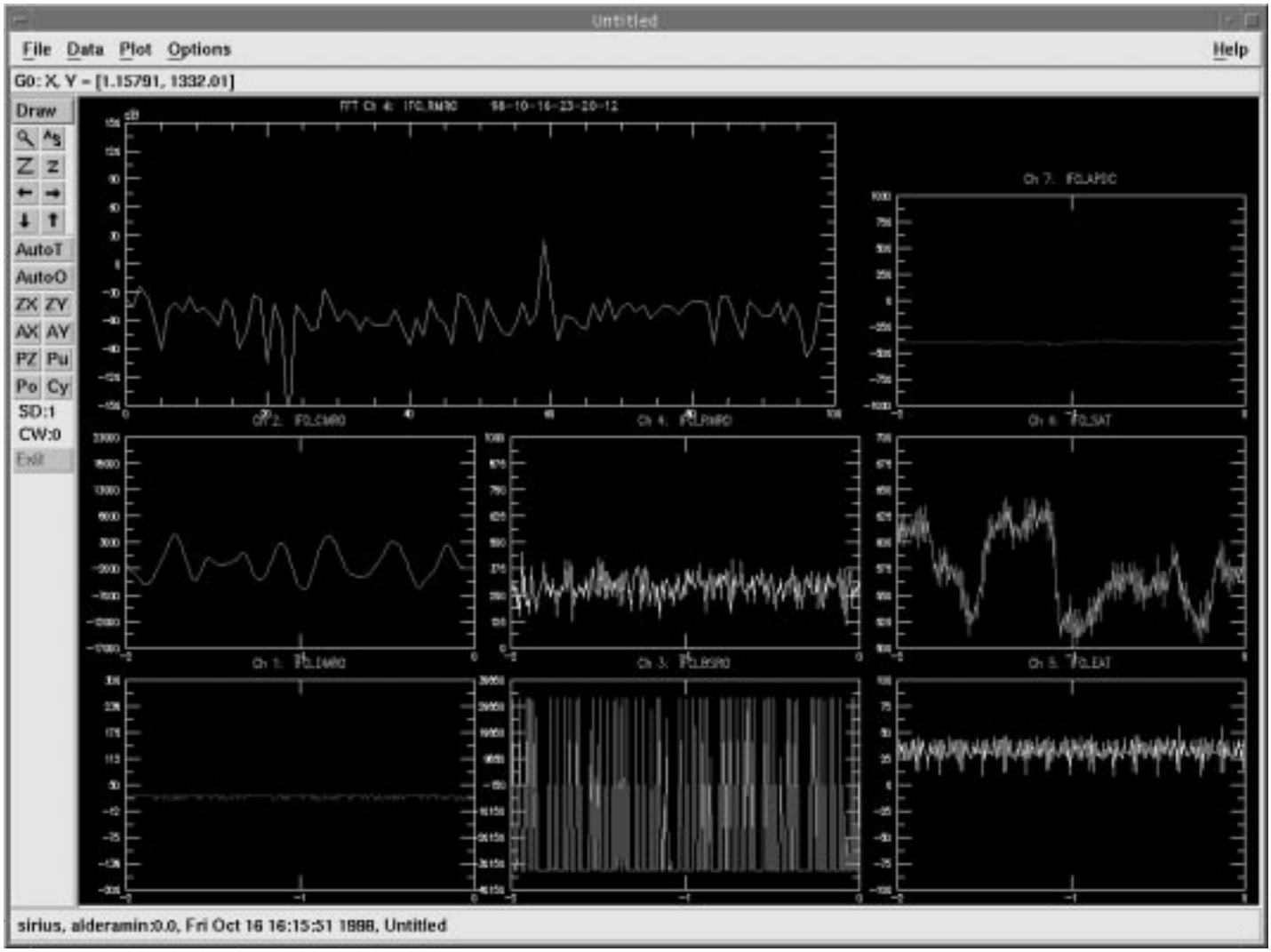
Data Viewer

(H. Ding, LIGO Data Acquisition)



Data Viewer

(H. Ding, LIGO Data Acquisition)



What can we do with the Data

Find all the multichannel events which have the same shape as this one
-- this is where the relational model is not so good
-- Does Ligo have “event signatures” to decide how to classify it?

Take some raw data, do some analysis, store the results in the database
-- easy to access with API, sorting, reports

Record everything in the database
-- when the instrument has a “random” error, can look for previous examples/precursors

Object and Relational Databases

Object databases are GOOD for
- Programming (API)
- Navigating and indexing the data
- Prefetching
- Object placement and clustering

Object databases are BAD
- Management
- Security
- Language is “not quite C++”
- Scaloing the administration tools

Relational databases are easier to manage, more scalable, easier to port platforms

Object Example: Hanford Bakeout

The table has 100,000 rows and 690 columns -- temperatures every minute for weeks

And the **column names change**

And there is **missing data**

And some things are recorded every 15 minutes, not every minute

This is a big table for any RDBMS!

(And this is just the beginning of the bakeout!)

Current solution is **objects**

Each object is a fully-populated mini-table with some columns and some rows

The software makes the quilt from the patches

Surely these columns are hierarchial ...

"PUMP_69 assembly consists of

PUMP_69_1 and PUMP_69_2 and PUMP_69_3"

This is a **virtual column**. It can be created easily with an object database.

Now lets extend it. How can we get a timeseries from an object database?

```
public class RealTimeSeries extends PersistentObject {
    Real getValue(Time T) {...}
    Real GetMovingAverage(Time T, TimeInterval Ti) {...}
    PowerSpectrum GetPowerSpectrum(Time Tstart, Time Tend) {...}
}
```

This is a time series, not just a bunch of numbers

LIGO Data Uses/Needs

- Rai Weiss
- Daniel Sigg

Reduced Data Sets

- How to get one's arms around the data?
  - Small enough body of data to enable on line analysis
  - Methods to "visualize" the state of the system
  - Methods to characterize the stationarity of the noise
  - Fast methods to discern changes
    - Diagnostics
    - Unexpected sources and surprises
  - Easy recognition of out of bound conditions

EXAMPLE :

- Dark port data using current best estimators in:
  - regression to environmental parameters
  - regression to ancillary interferometer signals
  - calibration in h(t) or h(f)
  - removal of instrument signatures

Reduced Data Sets

- Statistical information
  - ›› simple variances: rms, h^2 (f) vs t (“sonograms”)
  - ›› averaged cross power spectra between dark port and other signals
  - ›› the results of a covariance analysis - principal value decomposition of the spectral components
    - eigenvalues and eigenvectors
- Atlas of detected waveforms
  - ›› Wavelet characterization
    - amplitude
    - rate of occurence

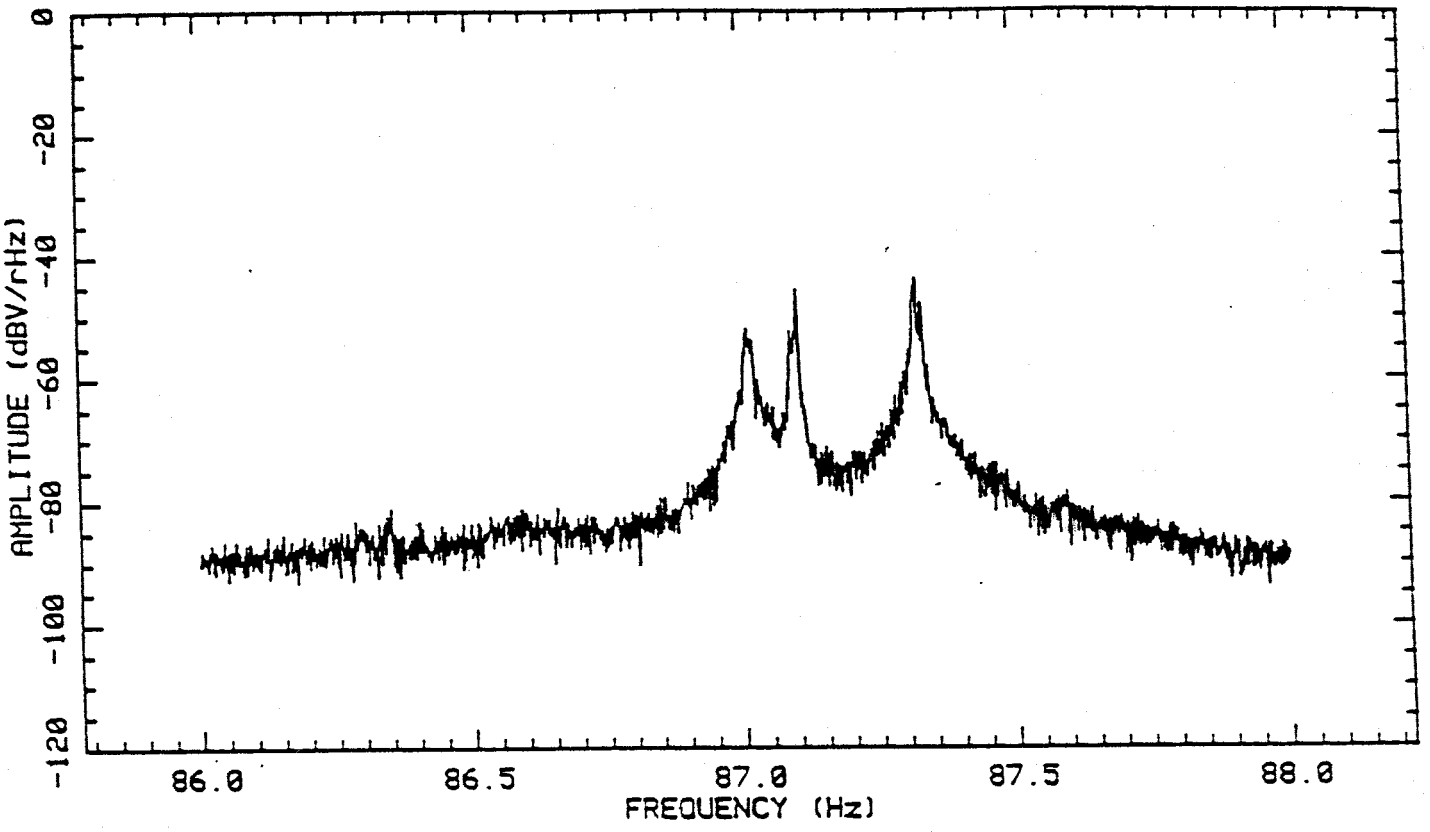


Figure 5.19
Suspension wire "string" mode resonances.

J. LIVAS

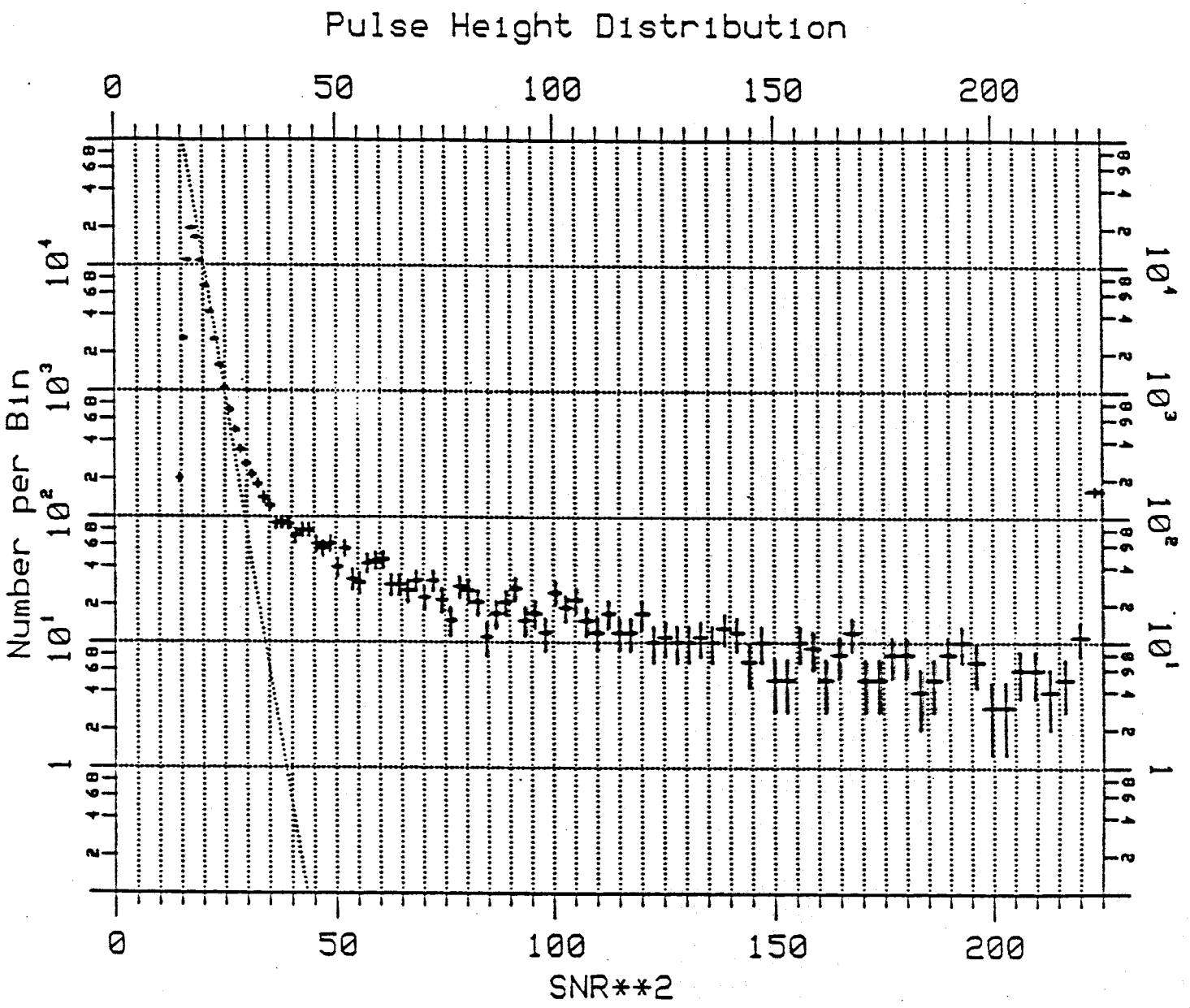


Figure 7.17 PHD from All Templates—No Multiple Detection Removal

D. DEWEY

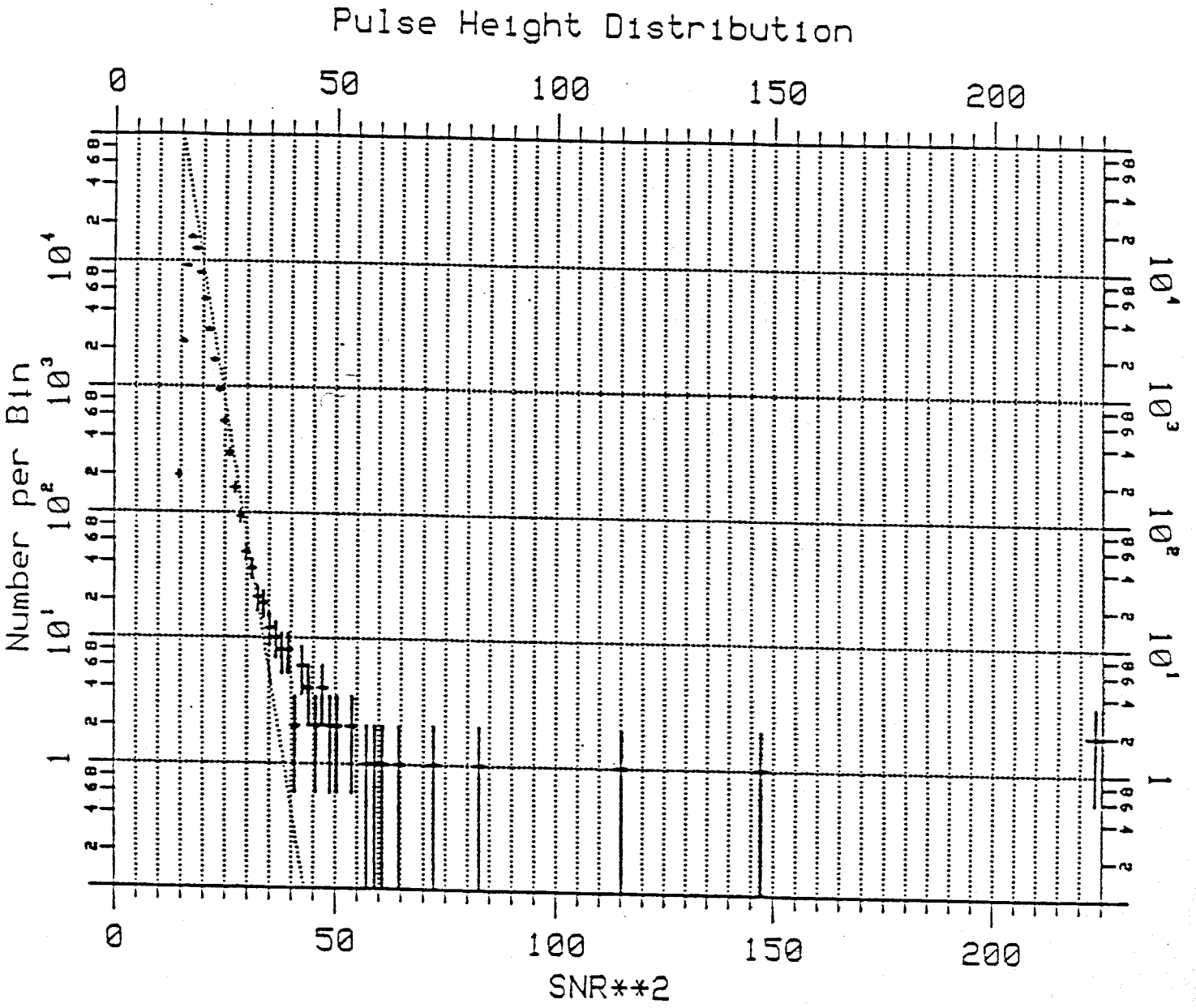


Figure 7.20 PHD from All Templates Excluding the "102" Events

D.ORWEY

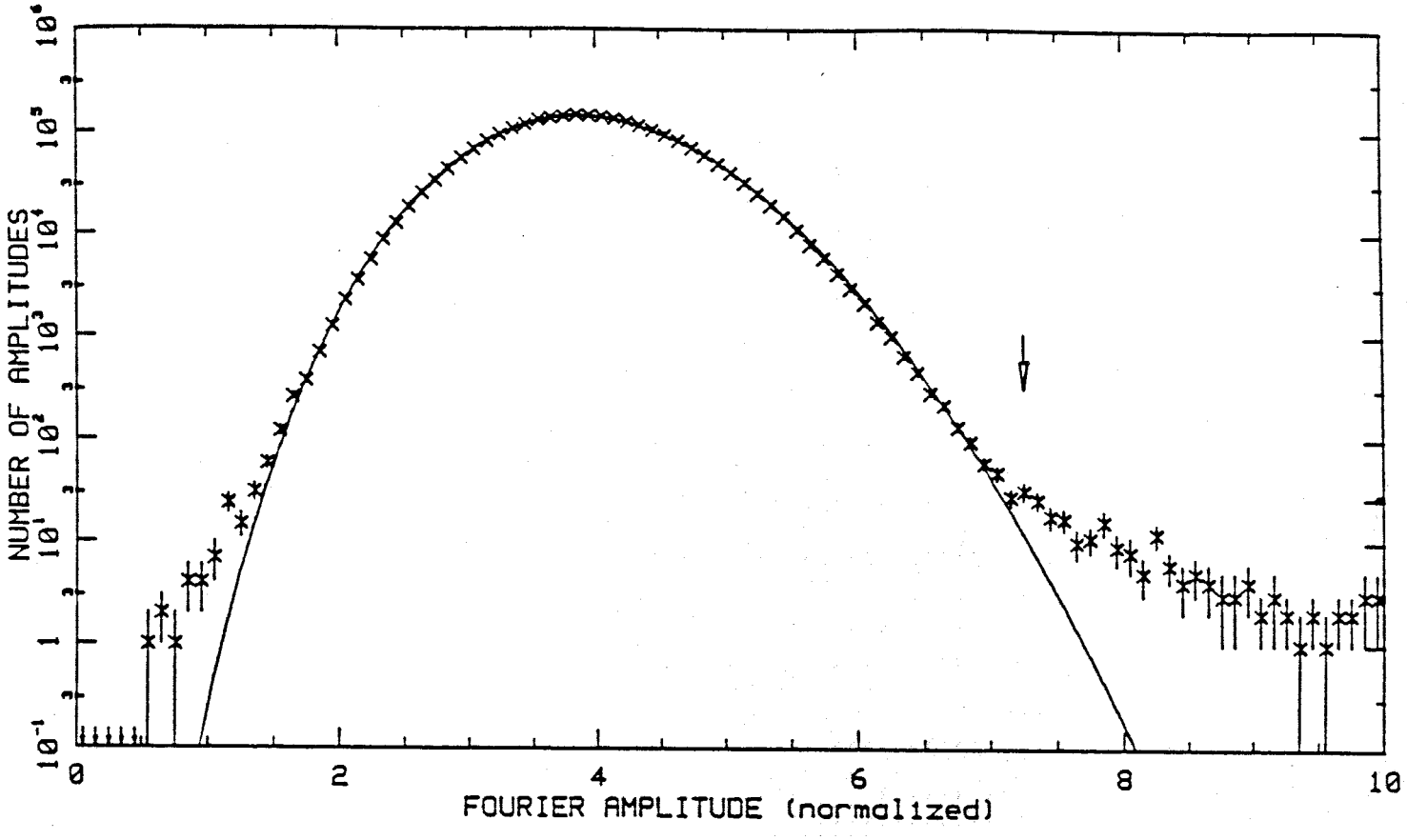


Figure 5.14

8-tape rms averaged power spectrum amplitude distribution.

Presentation by
Chaitan Baru, SDSC

Data-Intensive Computing at SDSC

Chaitanya Baru

Senior Principal Scientist

Enabling Technologies

San Diego Supercomputer Center

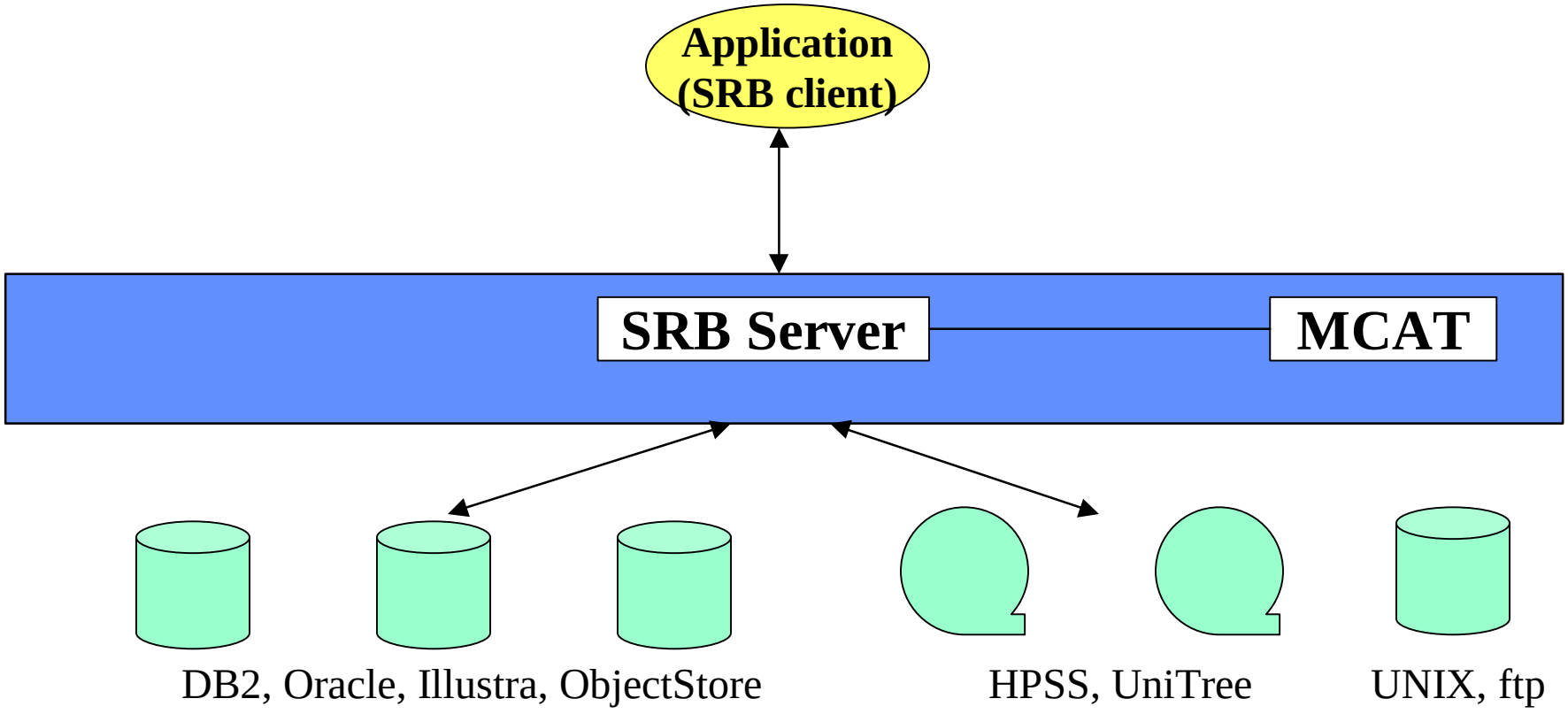
Challenges and Needs

- **Efficient storage management of very large collections**
  - 100's of 1TB data sets
  - 10^6 to 10^9 of 1K data sets
- **Ad hoc querying to identify data sets**
- **Access to distributed collections**
- **Ability to manage collections for multiple disciplines**
- **Representation and querying of semistructured metadata / data**

DICE Technology Solutions

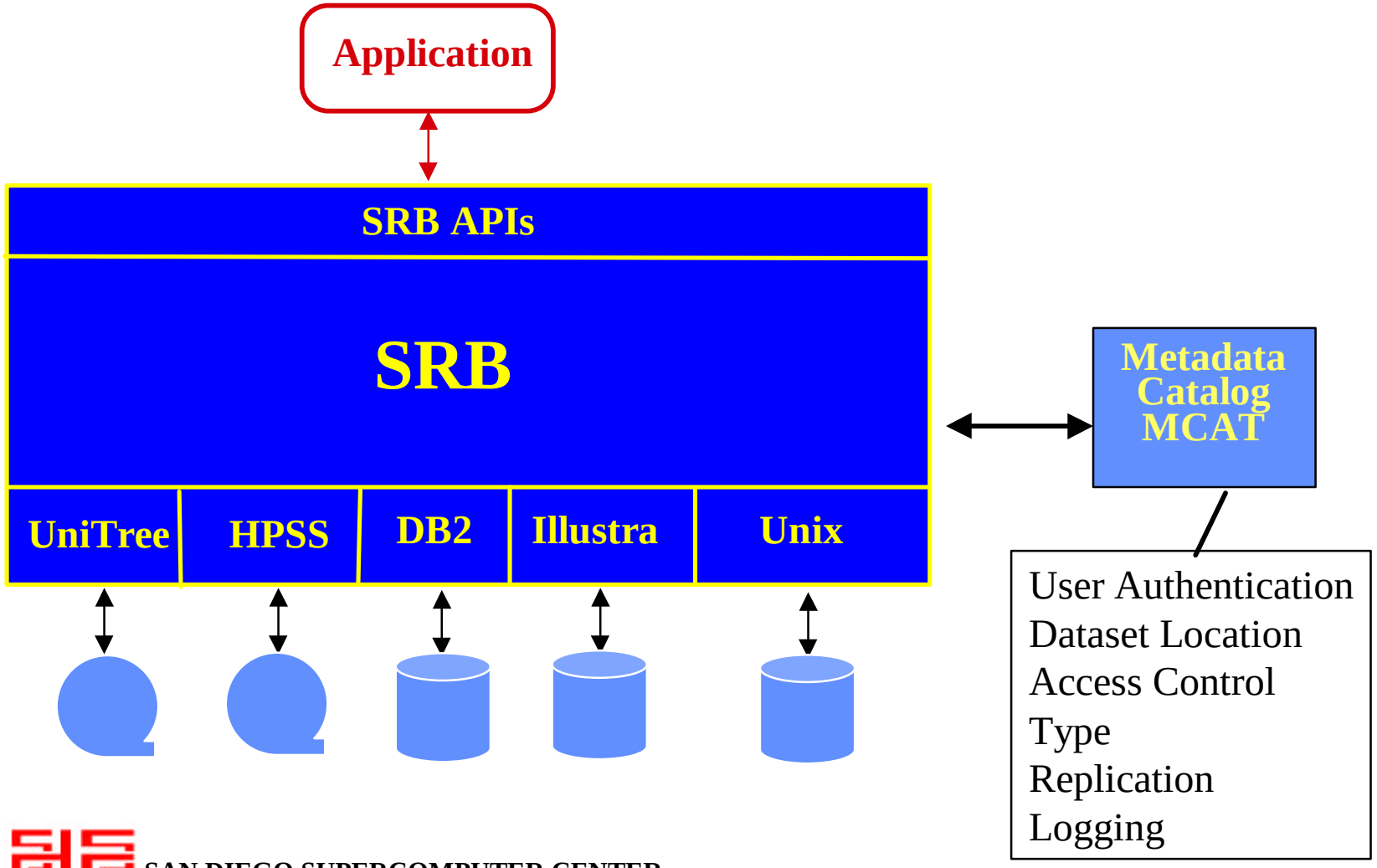
- **HPSS**
- **Storage Resource Broker (SRB) with Metadata Catalog (MCAT)**
- **Integrated DB2/HPSS system**
- **SDSC Encryption and Authentication system (SEA)**
- **IBM Digital Library, GPFS, Datalinks, Oracle/DB2/Informix DBMS**
- **XML-based mediator system**
- **Object-based information model and use of XML for representation**

The SDSC Storage Resource Broker (SRB)



Distributed Storage Resources

Architecture of the SRB Agent



SRB Features

- **Support for *Collection* hierarchy**
  - grouping of heterogeneous data sets
  - hierarchical access control, with ticket mechanisms
- **Replication**
  - optional replication at the time of creation
  - can choose replica on *read*
- **Proxy operations**
  - supports *proxy* ***move*** and ***copy*** operations
- **Monitoring capability**

MCAT Features

- **Stores *system* and *user/application-specific* metadata for data sets (and storage resources) based on relational model**
- **API for ad hoc querying of metadata attributes**
- **Ability to define new schemas**
- **Data sets can be associated with multiple schemas**
- **Ability to modify schemas**
- **Can relate metadata attributes across schemas**
- **Implemented in Oracle and DB2**

DB2/HPSS Integration

- **Collaboration with IBM TJ Watson Research Center**
- **Features:**
  - Prototype, works with DB2 UDB (Version 5)
  - DB2 handles DCE authentication and read/write to HPSS
  - HPSS file is defined as a DB2 container
  - *Regular* as well as *long (LOB)* DB2 columns can be in HPSS
  - DB2 can also manage an optional disk cache between DB2 and HPSS

DB2/HPSS Integration

Create Long Tablespace
HPSS-SPACE
Managed By Database Using

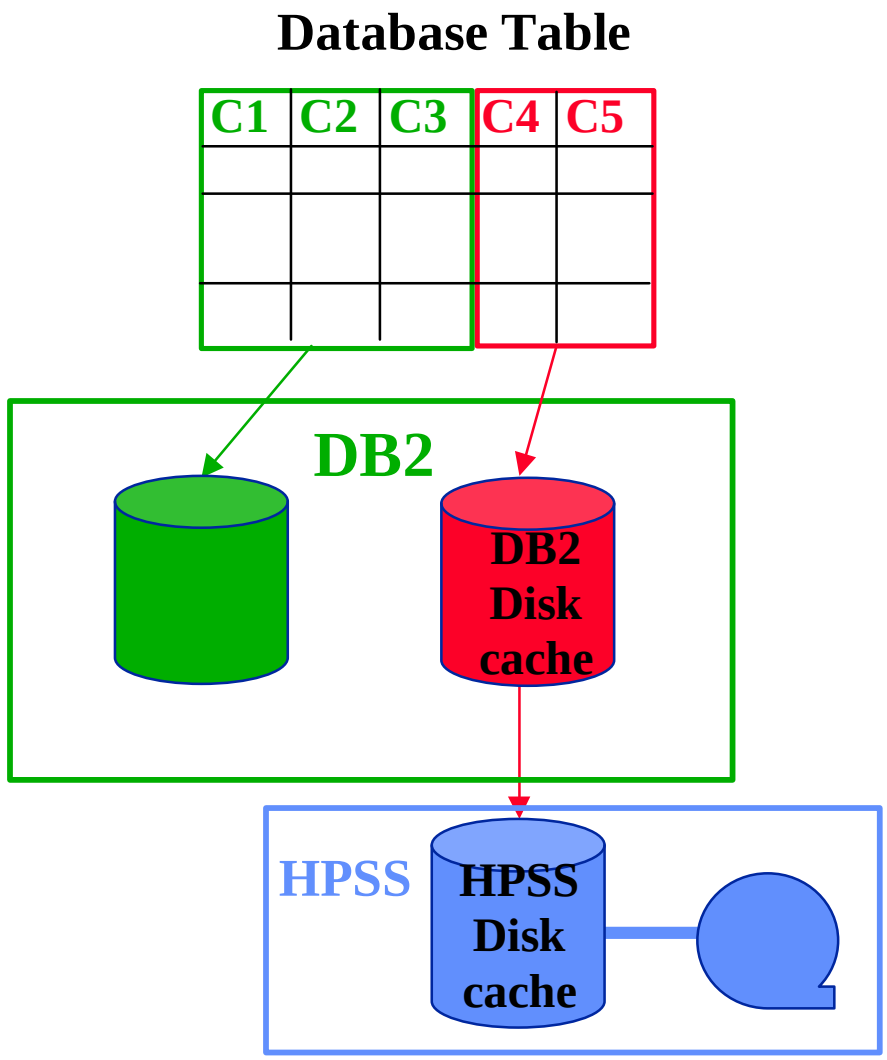
FILE (**HPSS** <hpss-filename> <size>
DISKBUF <path> <size>);

Create Tablespace
REGULAR-SPACE
Managed By Database Using

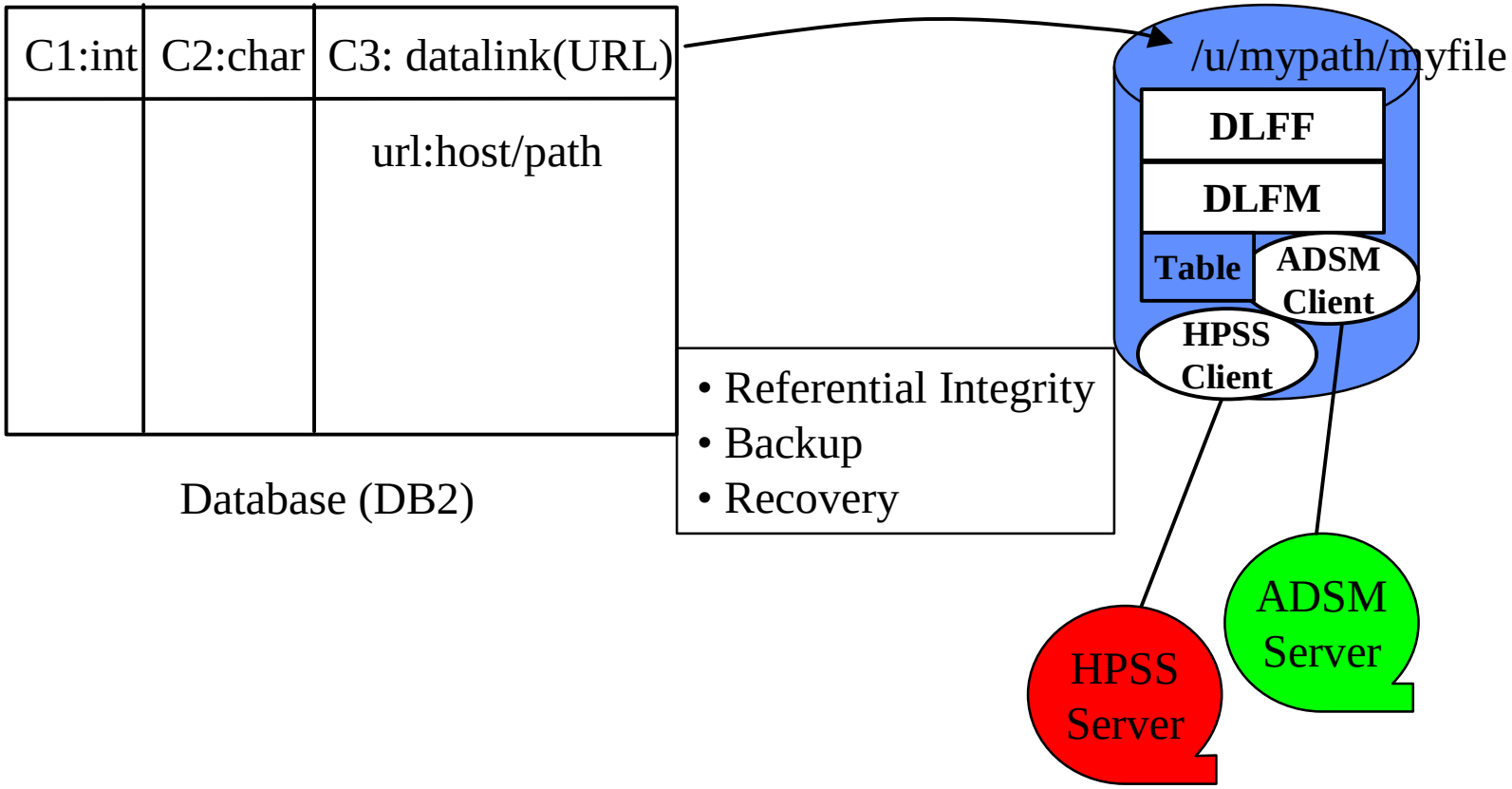
FILE (<unix-filename> <size>);

Create Table
SAMPLE-TABLE
(**C1** int, **C2** float, **C3** char,
C4 CLOB, **C5** BLOB)

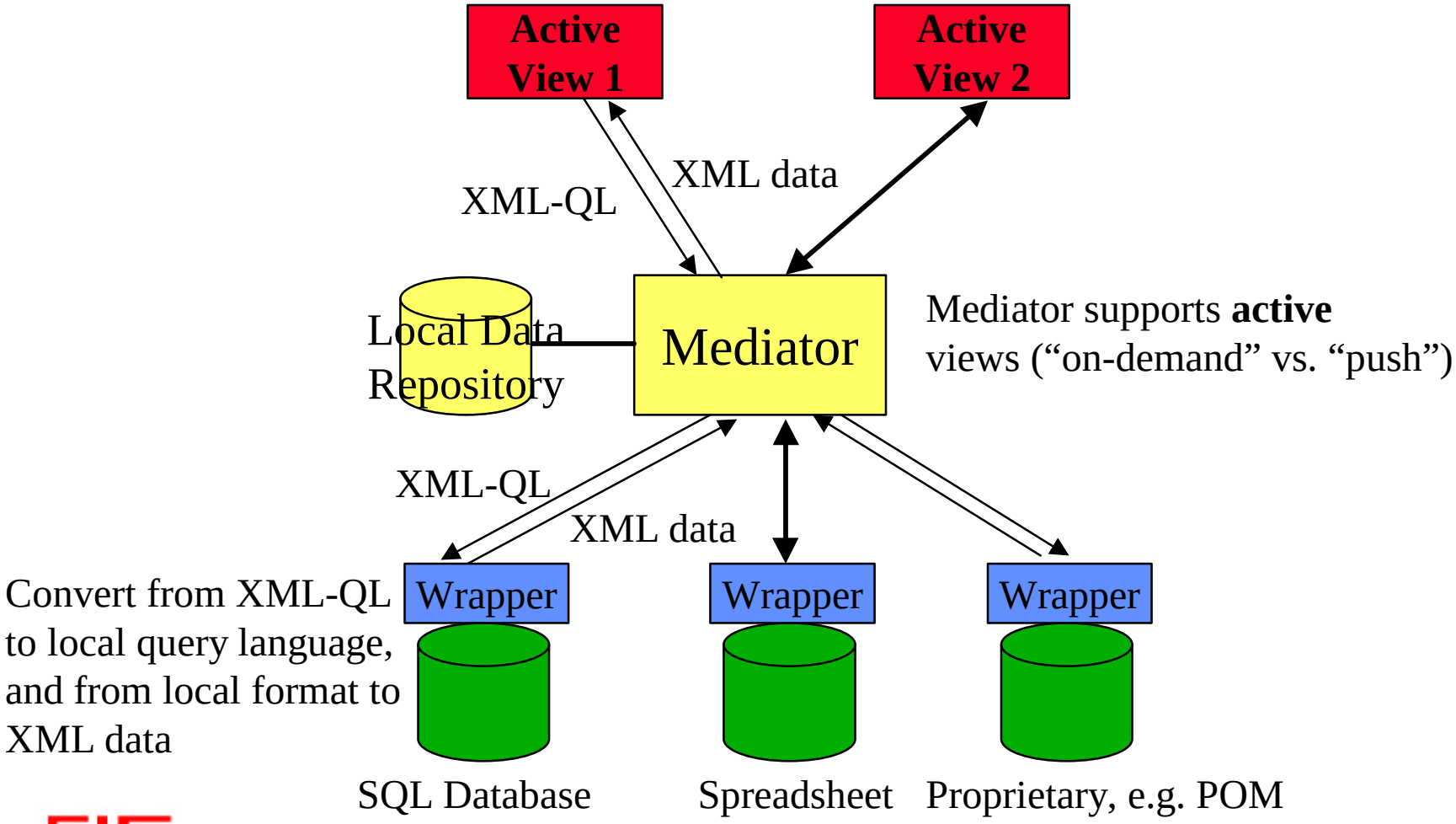
In **REGULAR-SPACE**
Long In **HPSS-SPACE**



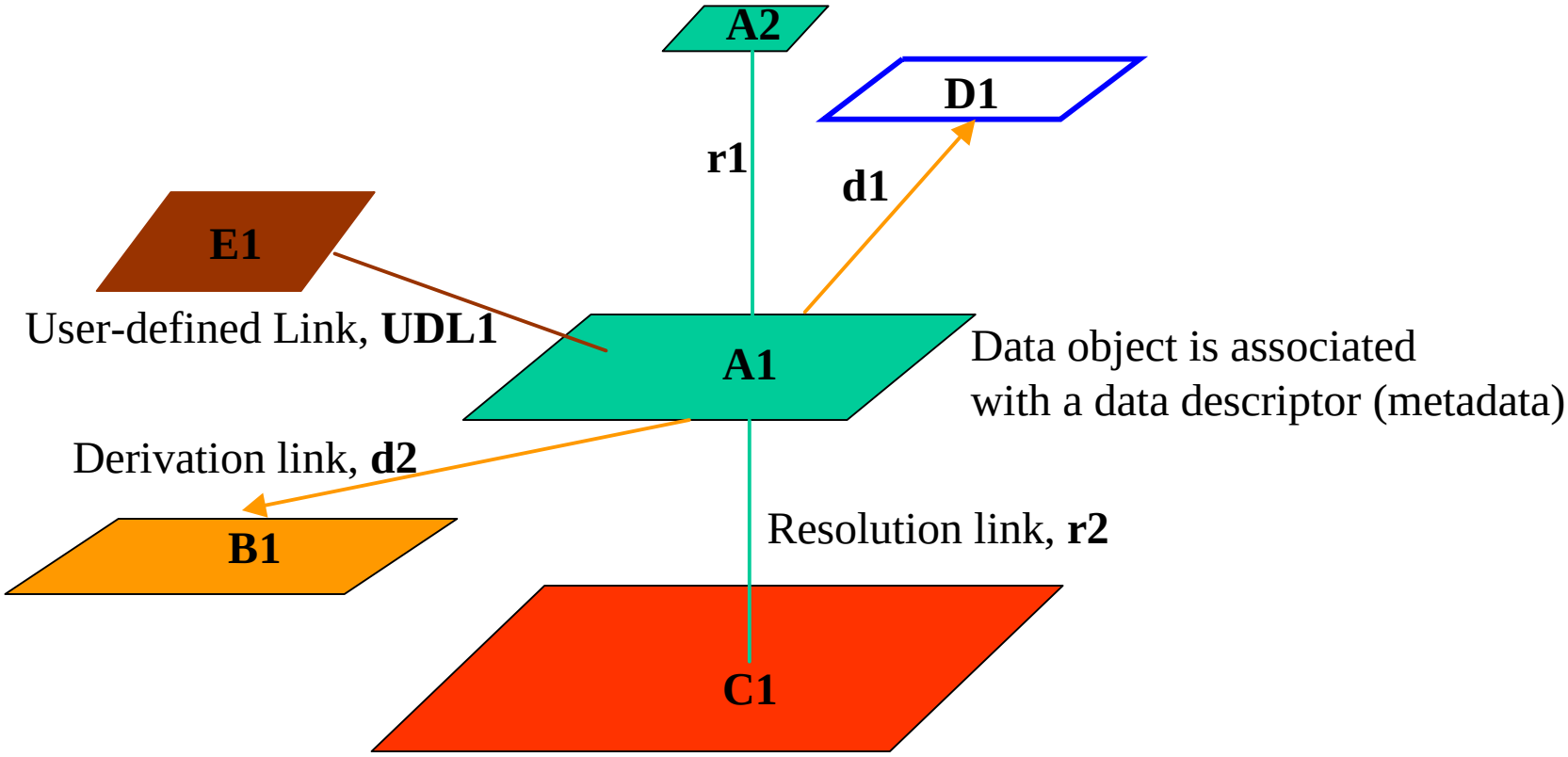
Datalinks


C1:int
C2:char
C3: datalink(URL)
url:host/path
Database (DB2)
/u/mypath/myfile
DLFF
DLFM
Table
ADSM Client
HPSS Client
• Referential Integrity
• Backup
• Recovery
HPSS Server
ADSM Server

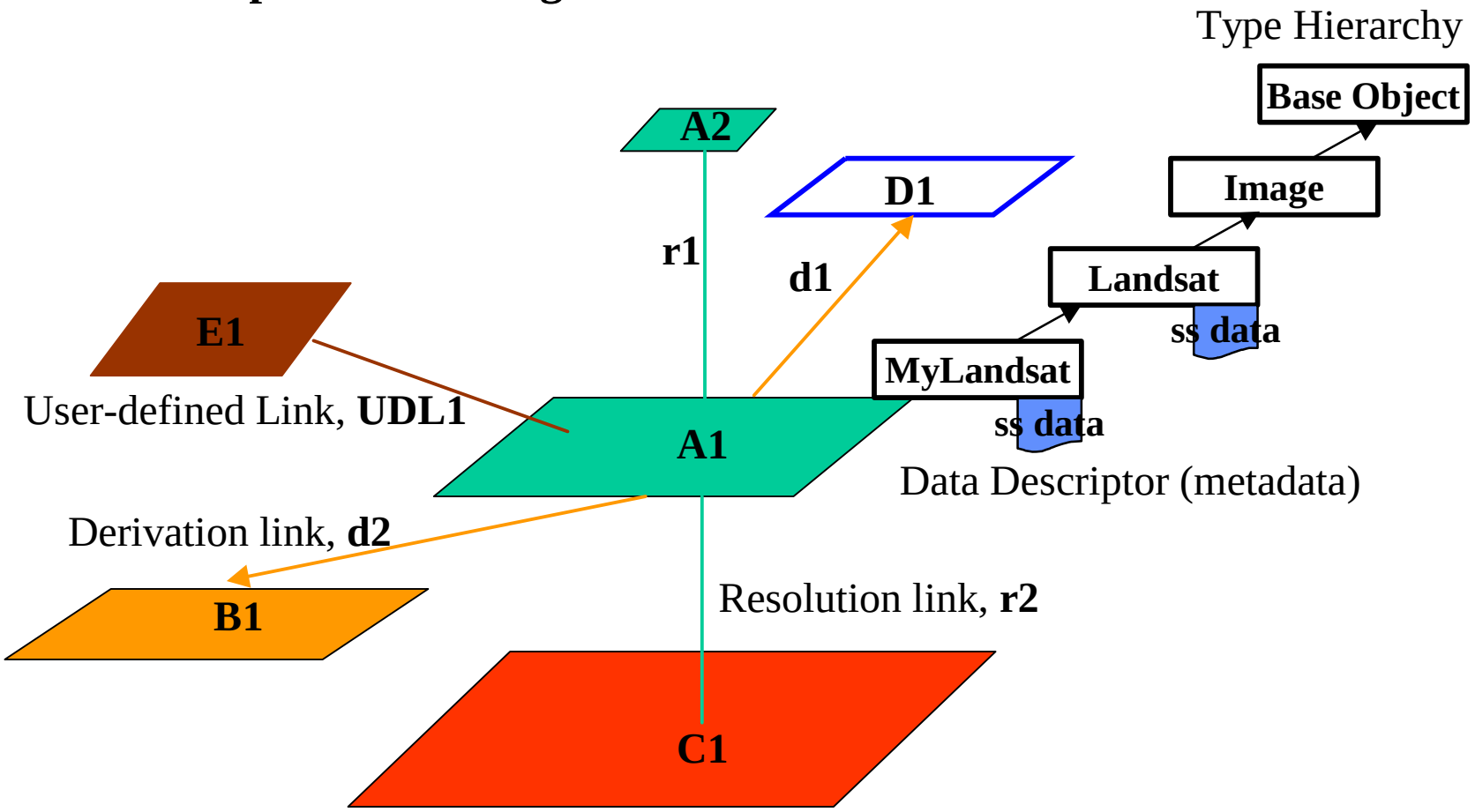
XML-Based Mediator System



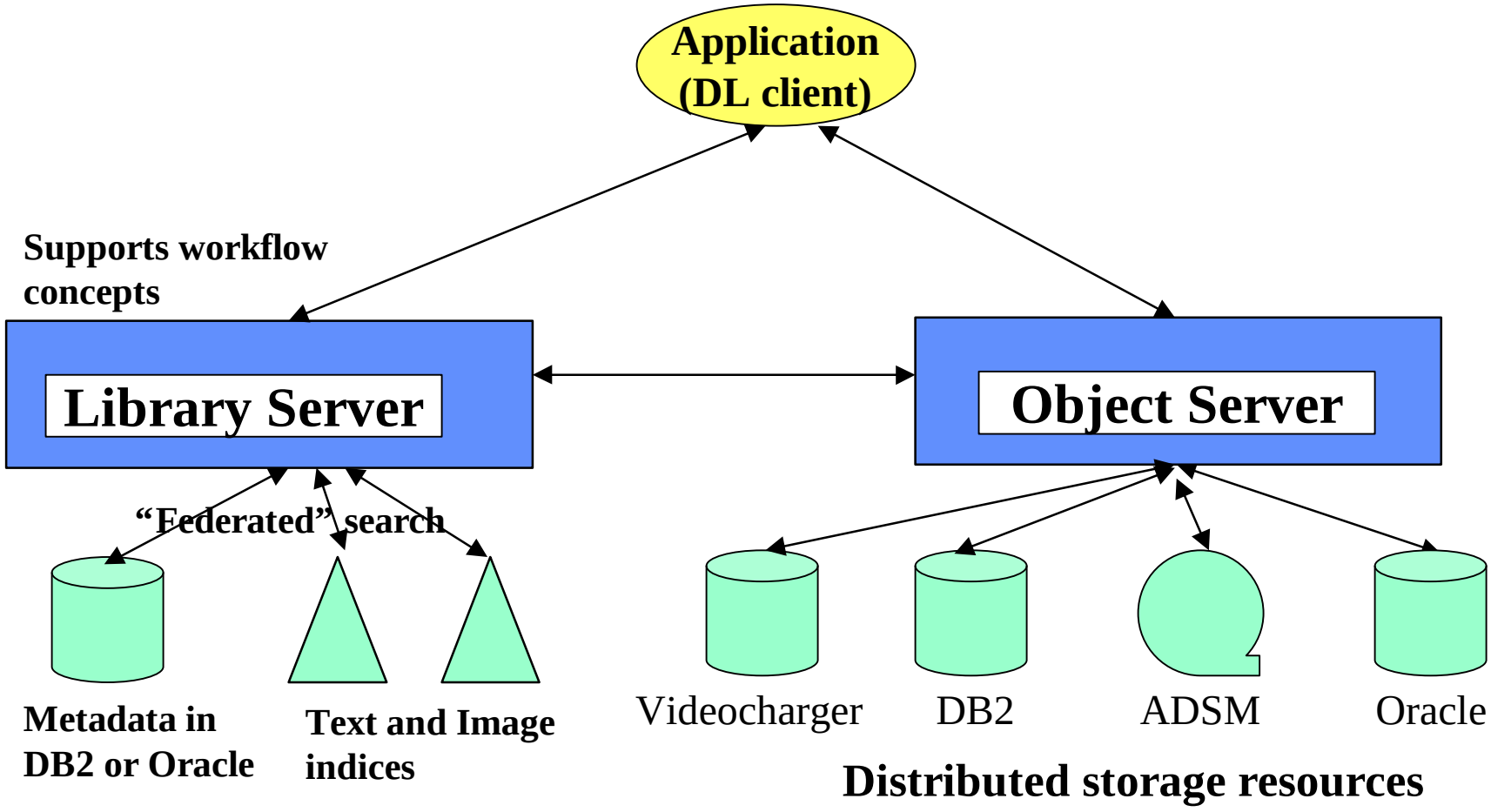
Object-Based Information Model



- **Data descriptors may be semistructured (XML representation)**
- **Objects can be represented using XML**



The IBM Digital Library



Unresolved Issues

- **Ubiquitous security mechanism**
- **Support for versioning (database and schema)**
- **Support for "time travel"**
  - access to archives
  - incorporation/integration of predictive models in/with database
- **Details of wrapping and mediation**
- **Need for information flow models (like workflow)**

Presentation by
Thomas Handley, JPL/IPAC

Presentation by
Julian Bunn, CERN

Experience with the use of ODBMS for HEP Data Storage, Reconstruction and Analysis

The GIOD Project
(Globally Interconnected Object Databases)

A Joint Project between Caltech(HEP and CACR), CERN and Hewlett Packard

http://pcbunn.cithep.caltech.edu/

LIGO Database Workshop

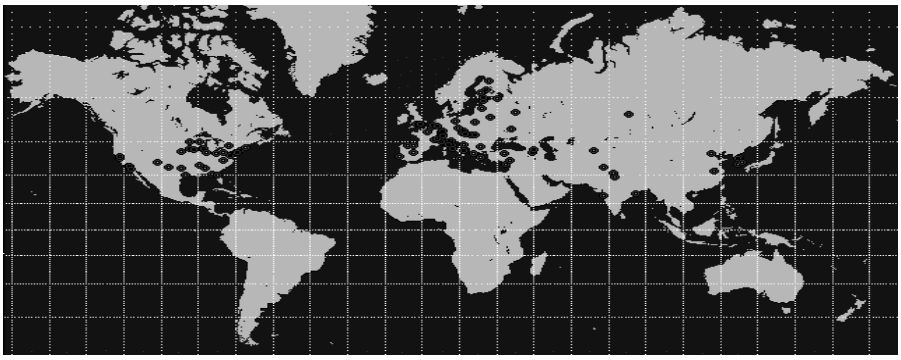
Julian Bunn/CERN
October 1998

CERN's Large Hadron Collider- 2005 to >2025

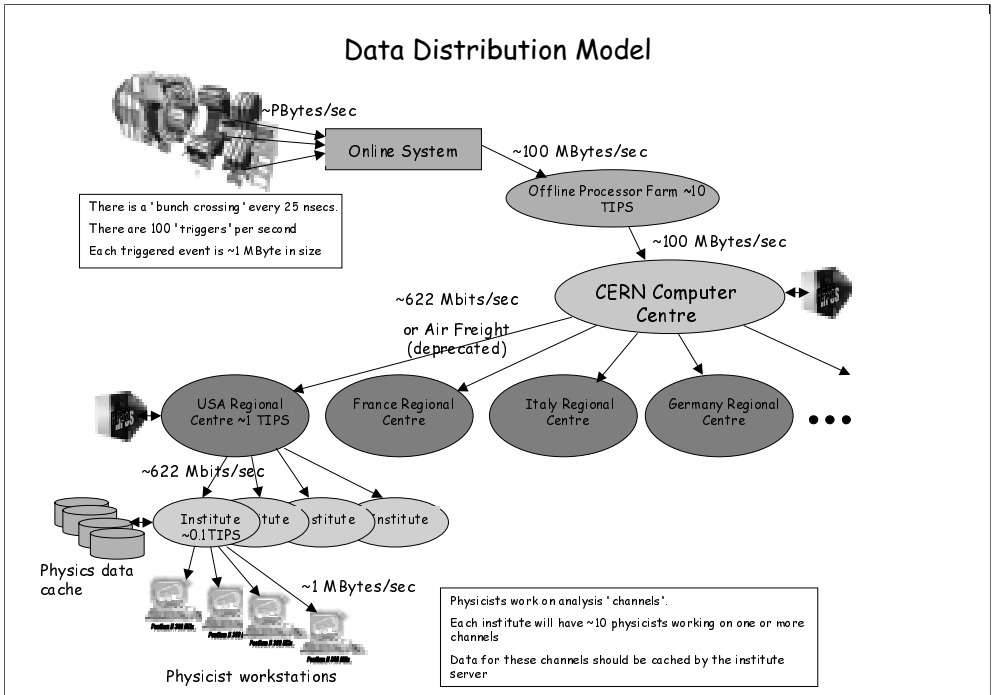
The Large Hadron Collider (LHC)

- Biggest machine yet built: a proton-proton collider
- Four experiments: ALICE, ATLAS, CMS, LHCb

WorldWide Collaboration



- CMS
  - >1700 physicists
  - 140 institutes
  - 30 countries
- 100 Mbytes/sec from online systems
- ~1 Pbyte/year raw data
- ~1 Pbyte/year reconstructed data
- Data accessible across the globe

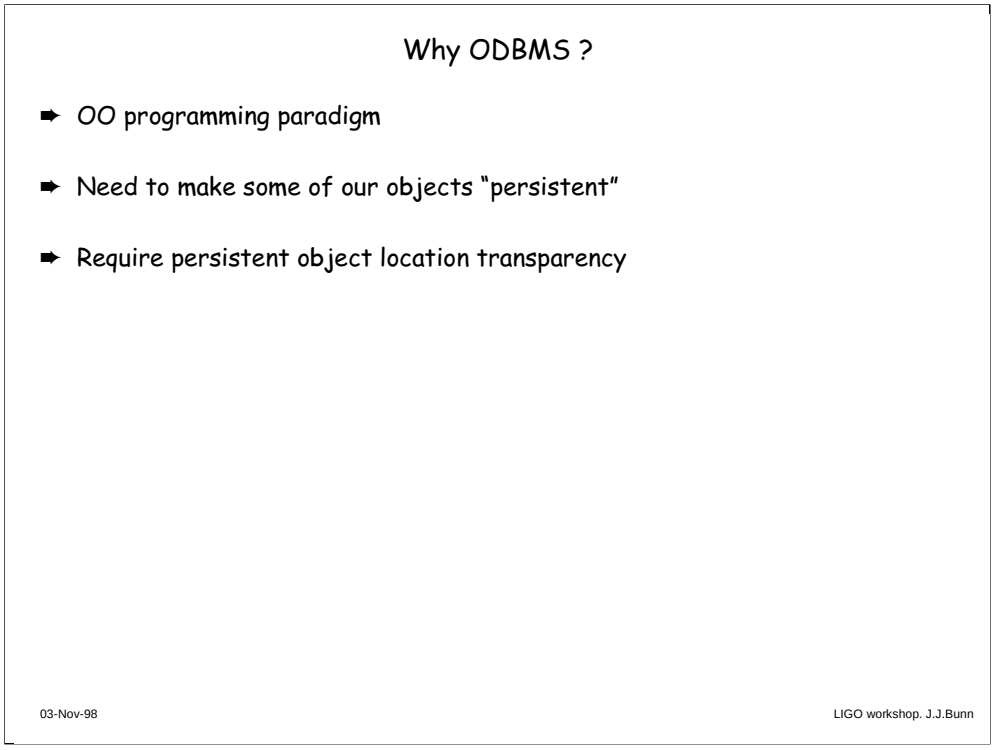

Data Distribution Model
~PBytes/sec
Online System
~100 MBytes/sec
Offline Processor Farm ~10 TIPS
There is a 'bunch crossing' every 25 nsecs.
There are 100 'triggers' per second
Each triggered event is ~1 MByte in size
~100 MBytes/sec
~622 Mbits/sec
or Air Freight (deprecated)
CERN Computer Centre
USA Regional Centre ~1 TIPS
France Regional Centre
Italy Regional Centre
Germany Regional Centre
~622 Mbits/sec
Institute ~0.1TIPS
Physics data cache
~1 MBytes/sec
Physicist workstations
Physicists work on analysis 'channels'.
Each institute will have ~10 physicists working on one or more channels
Data for these channels should be cached by the institute server
03-Nov-98
LIGO workshop. J.J.Bunn

Computing Model

- Computing Hardware, Network and Software systems to support timely and competitive analysis by a worldwide collaboration
- Hierarchical networked ensemble of heterogeneous, data-serving and processing computing systems
- Key technology:
  - Object-Oriented Software
  - Object Database Management Systems (ODBMS)
  - Hierarchical Storage Management Systems
  - Networked Collaborative Environments

The GIOD Project Goals

- Build a small-scale prototype Regional Centre using:
  - Object Oriented software, tools and DB management system
  - Large scale data storage software and equipment
  - High bandwidth WAN and LANs
- Measure, evaluate and tune the components of the Centre for LHC Physics
- Use measurements of the prototype as input to simulations of realistic LHC Computing Models for the future



OO programming paradigm

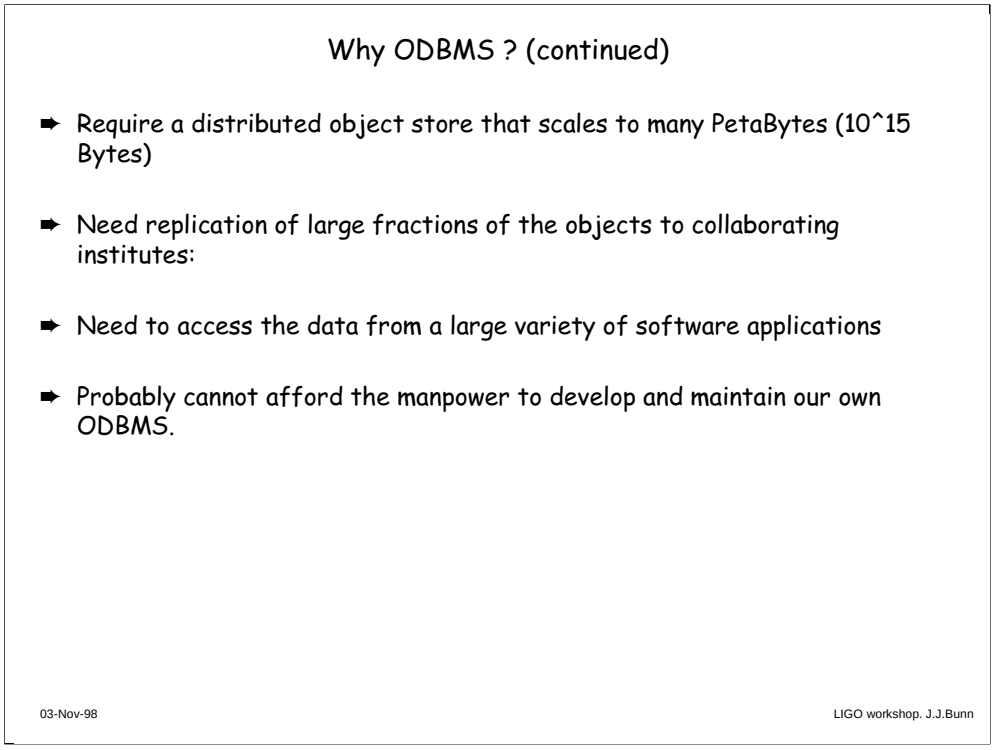
- is the modern, industry direction
- supported by C++, Java high level languages
- excellent choice of both free and commercial class libraries
- suits our problem space well: rich hierarchy of complex data types (raw data, tracks, energy clusters, particles, missing energy, time-dependent calibration constants)
- Allows us to take full advantage of industry developments in software technology

Need to make some of our objects "persistent"

- raw data
- newly computed useful objects
- an object store that supports our evolving data model

Require persistent object location transparency

- No huge "logbooks" containing correspondences between event numbers, run numbers, event types, tag information, file names, tape numbers, site names
- No worries about software versions used to create persistent objects: should be done "automatically"



Require a distributed object store that scales to many PetaBytes (10^15 Bytes)

RDBMS wont cut it: think of a virtual table with 10^9 rows and many many columns - just for events accumulated in ~1 year of LHC operation

Need replication of large fractions of the objects to collaborating institutes:

Achievable transparently across the network or when using freight of tapes/DVDs

Object store must keep track of location and access mathod for all objects in the networked, distributed system

The user shouldn't be concerned with the location of objects in the system

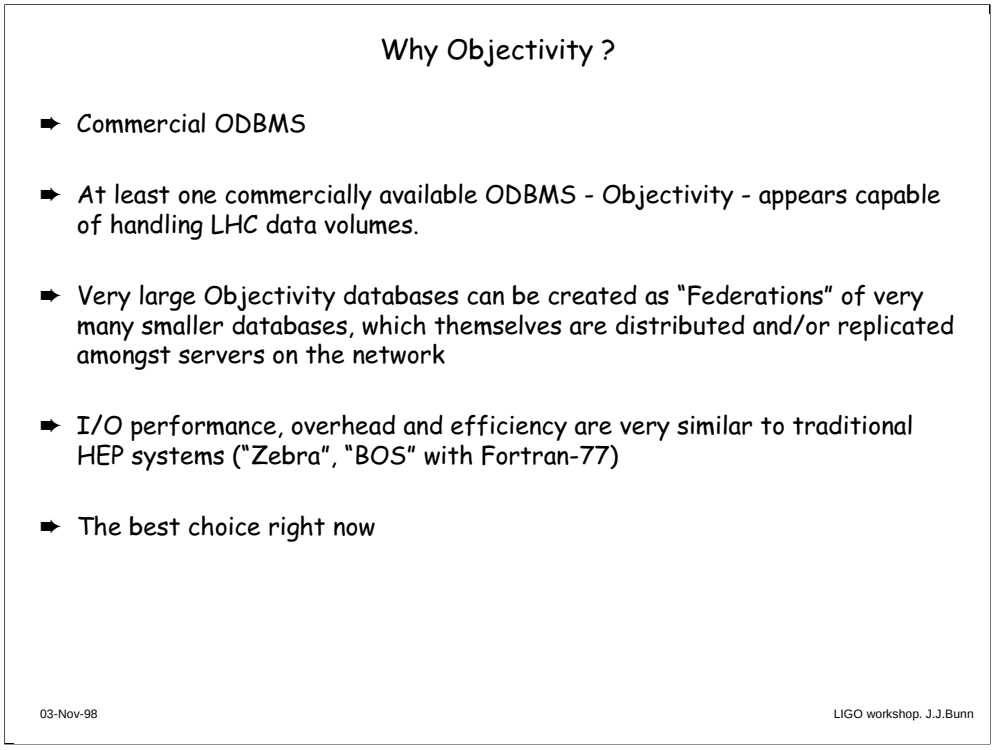
Need to access the data from a large variety of software applications

Need several language bindings

Need support for major operating systems (NT and Unix odours)

Need standards conformance(safely predict C++ -> Java -> ?? -> ?? before 2025)

Probably cannot afford the manpower to develop and maintain our own ODBMS.



Commercial ODBMS

embody hundreds of person-years of effort to develop

tend to conform to standards

offer rich set of management tools & language bindings

At least one commercially available ODBMS - Objectivity - appears capable of handling LHC data volumes.

Very large Objectivity databases can be created as “Federations” of very many smaller databases, which themselves are distributed and/or replicated amongst servers on the network

I/O performance, overhead and efficiency are very similar to traditional HEP systems (“Zebra”, “BOS” with Fortran-77)

The best choice right now in terms of

Architecture (federations, data replication, fault tolerance)

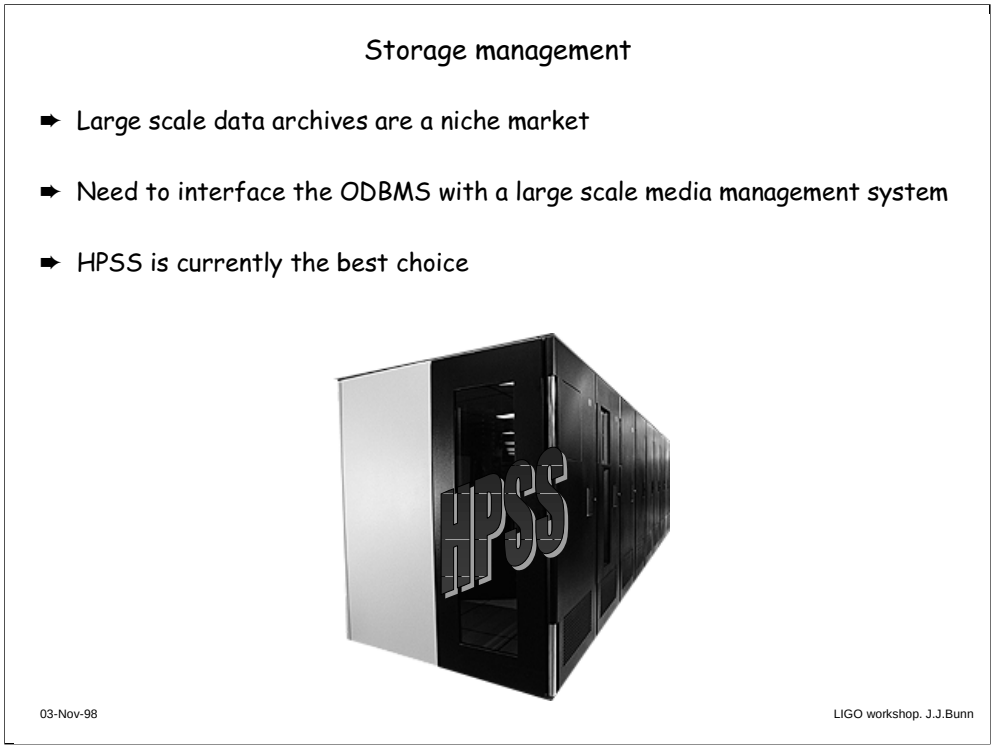
OS support (NT, Solaris, Linux (imminent), Irix, AIX, HP-UX, etc..)

Language bindings (C++, Java, [C, SmallTalk, SQL++ etc.])

Commitment to HEP as target business sector

Close relationships built up with the company, at all levels from the CEO “down” to the engineers

Attractive licensing schemes for HEP (e.g. CERN and SLAC, BaBar)



Large scale data archives are a niche market

Continued reliance on Tapes is foreseen

Slow evolution of costs and technology over time.

Reliability not enough to avoid “backup copies”

Need to interface the ODBMS with a large scale media management system

Have chosen HPSS - appears to have scalability into the Petabyte range

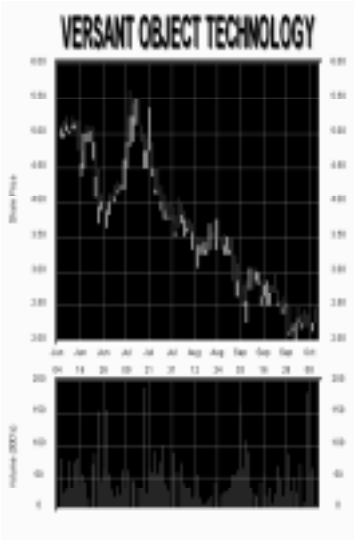
HPSS far from being multi-platform, and not yet robust enough for production use

Heavy investment of CERN/Caltech/SLAC… effort to make *HPSS* evolve in directions suited for HEP

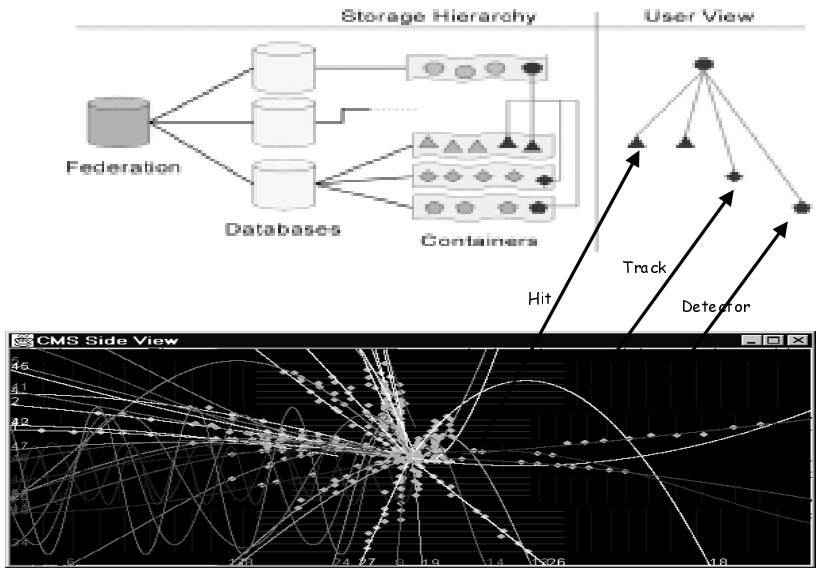
Have developed a layer between the ODBMS and the filesystem that restores newly requested databases from tertiary HPSS storage when required

ODBMS worries

- Bouyancy of the commercial marketspace?
  - Introduction of Computer Associates "Jasmine" pure ODBMS (but targetted at multimedia data)
  - Oracle et al. paying lip-service to OO with Object features "bolted on" to their fundamentally RDBMS technology
  - Breathtaking fall of Versant stock
  - Still no IPO for Objectivity
- Conversion of "legacy" ODBMS data from one system to another
  - 100 PetaBytes via an ODMG-compliant text file?!
  - Good argument for keeping raw data outside the ODBMS, in simple binary files (maybe doubles storage needs)



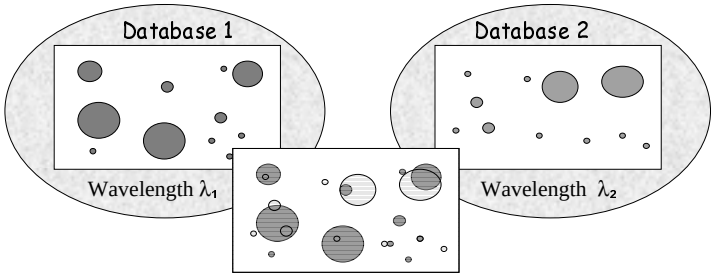
Overview of the Storage and User Views of the Data



Objectivity tests

- Developed simple scaling application: matching 1000s of sky objects at different wavelengths
- Runs entirely in cache (can neglect disk I/O performance), applies matching algorithm between pairs of objects in different databases.
- Looked at usability, efficiency and scalability for
  - number of objects
  - location of objects
  - object selection mechanism
  - database host platform

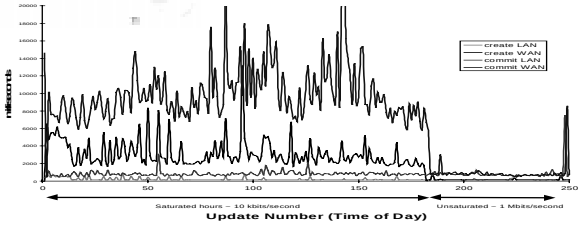
- Results
  - Platform independence of the application
  - Platform independence of the database
  - Fastest access when objects are "indexed"
  - Slowest when using predicates
  - No performance hit when database remote from application

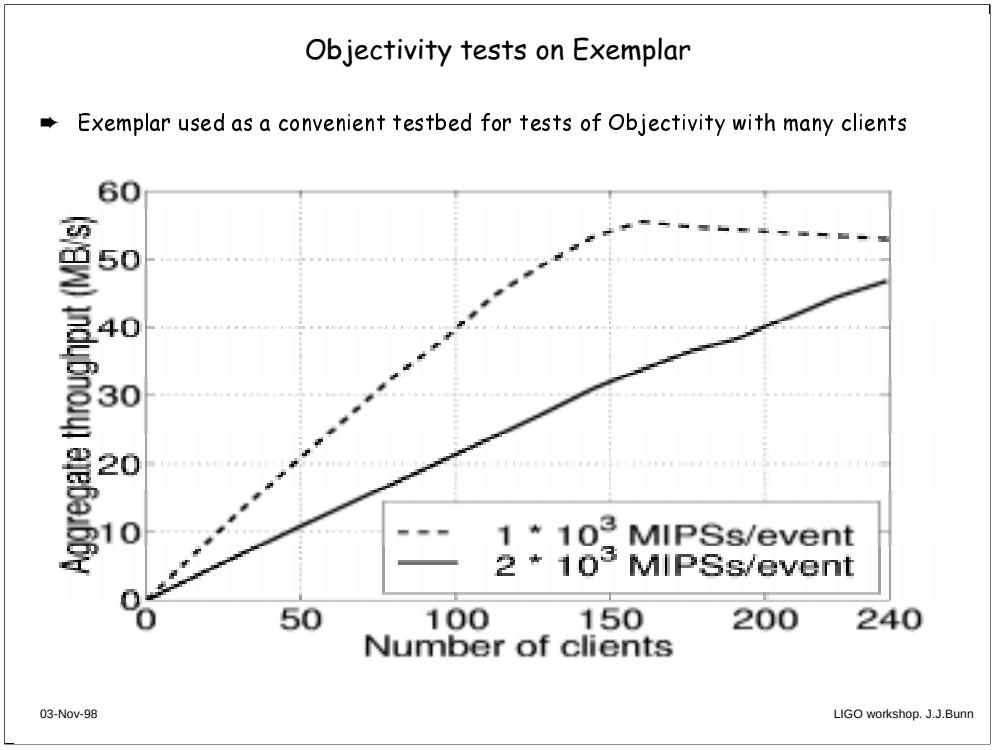


Match using indexes, predicates or cuts

Objectivity tests

- Other Tests:
  - Looked at Java binding performance (~3 times slower)
  - Created federated database in HPSS managed storage, using NFS export
  - Tested database replication from CERN to Caltech





Exemplar used as a convenient testbed for tests of Objectivity with many clients

Tested clients running simulated Track reconstruction: CPU-intensive with modest I/O.

Event level (coarse-grained) parallelism

N = 15 - 210 reconstruction processes evenly distributed across the Exemplar system.

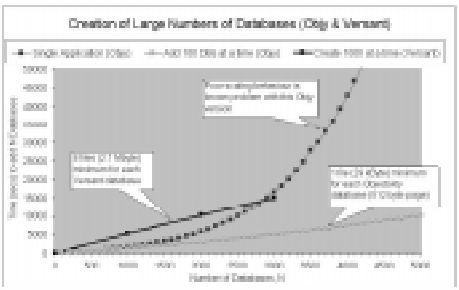
Data in an Objectivity/DB database federation hosted on the Exemplar.

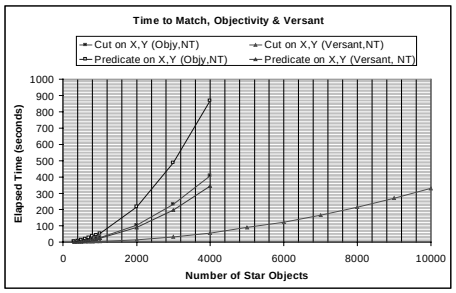
Objects read with simple read-ahead optimisation layer (gains a factor 2)

Results: Exemplar very well suited to this workload. With four node filesystems it was possible to utilise 240 processors in parallel with very high efficiency.

Versant tests

- Evaluated usability and performance of Versant ODBMS
- Converted the test application without problem, then measured performance in same way as we had tested Objy





- Conclusion: Versant a decent "fall-back" solution for us
- Following these tests we concentrated solely on Objectivity

03-Nov-98 LIGO workshop. J.J.Bunn

Evaluated usability and performance of Versant ODBMS, Objy's main competitor.

Converted the test application without problem, then measured performance in same way as we had tested Objy

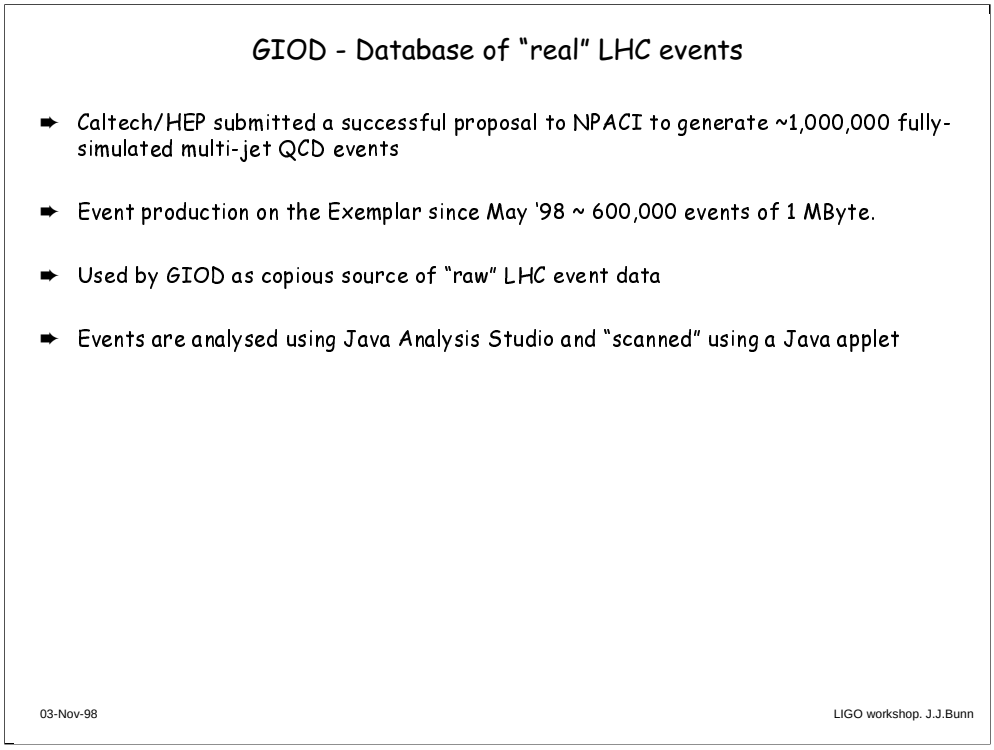
Conversion took ~1/2 a day: minor changes to schema and source code

Systems operation of Versant more cumbersome and time consuming (e.g. applying schema to each and every database)

API easier to use and offered some convenient built-in features (e.g. LinkVstr - Objy equivalent is user-constructed ooVArray).

"Predicate" queries seemed better implemented: certainly faster.

Versant Java binding worked well: at the time we tested, Objy did not offer Java.



Caltech/HEP submitted a proposal to NPACI to generate ~1,000,000 fully-simulated multi-jet QCD events

- Using CMSIM on the Exemplar
- Directly study **Higgs** ⇨ $\gamma\gamma$ backgrounds for first time
- Computing power of Exemplar makes this possible in ~few months
- Accepted in March '98.

Event production on the Exemplar since May 24th.

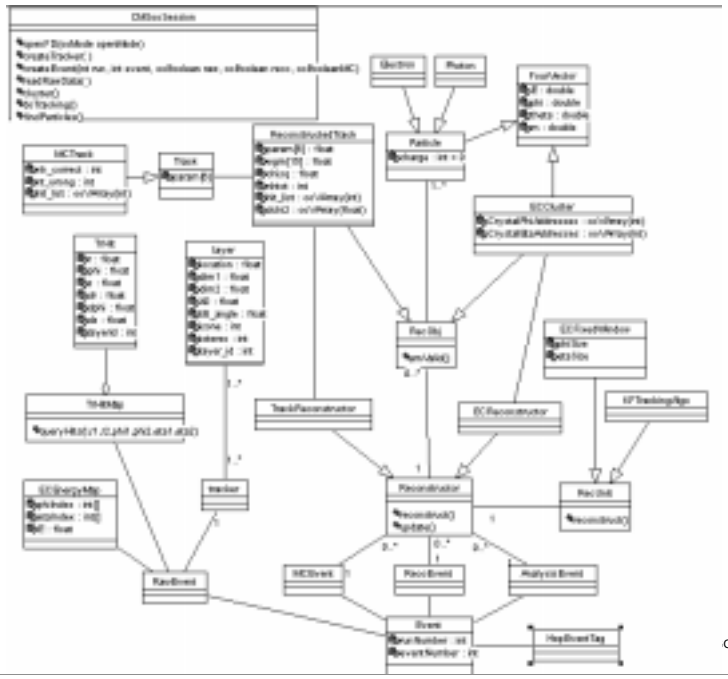
- ~0.6 TBytes of FZ files (~600,000 events) in HPSS

Used by GIOD as copious source of "raw" LHC event data

- Files are read using the "ooZebra" utility developed in CMS
- Raw data objects (tracker hit maps, ECAL energy maps) created for each event and stored in Objy federated database
- Tracks and energy clusters reconstructed: new objects stored in the database
- Pattern matching creates "physics" objects like Photons and Electrons: stored in the database

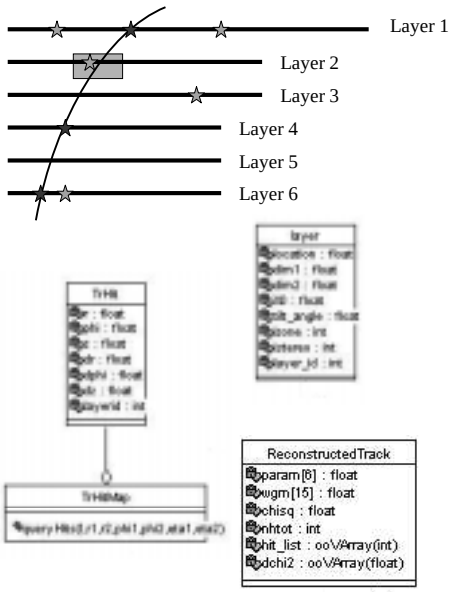
Events in the database are analysed using Java Analysis Studio and "scanned" using a Java applet

CMSOO Class Diagram

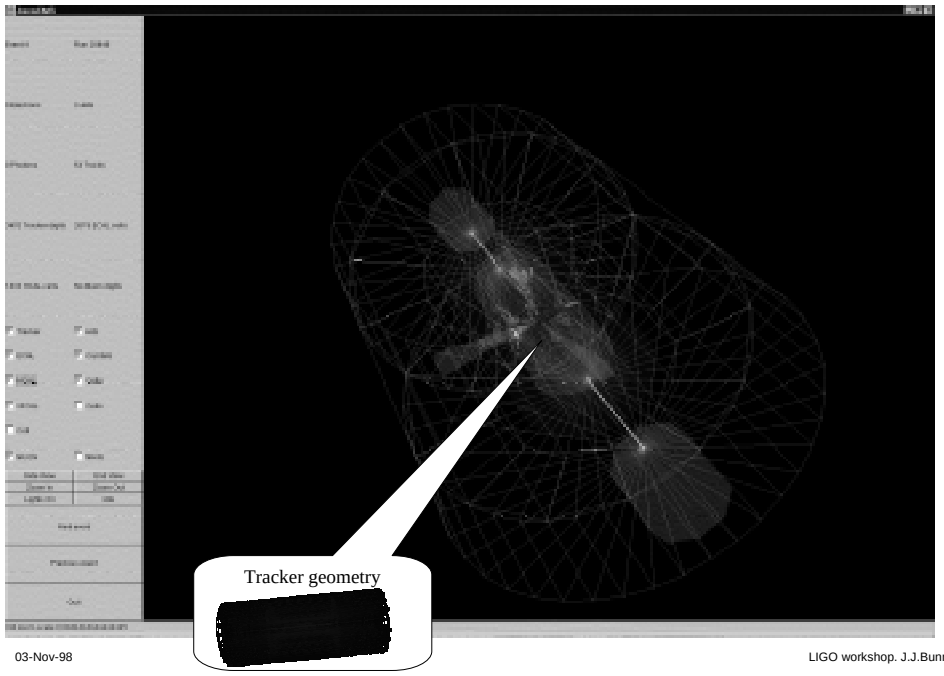


CMSOO - Track Reconstruction Procedure

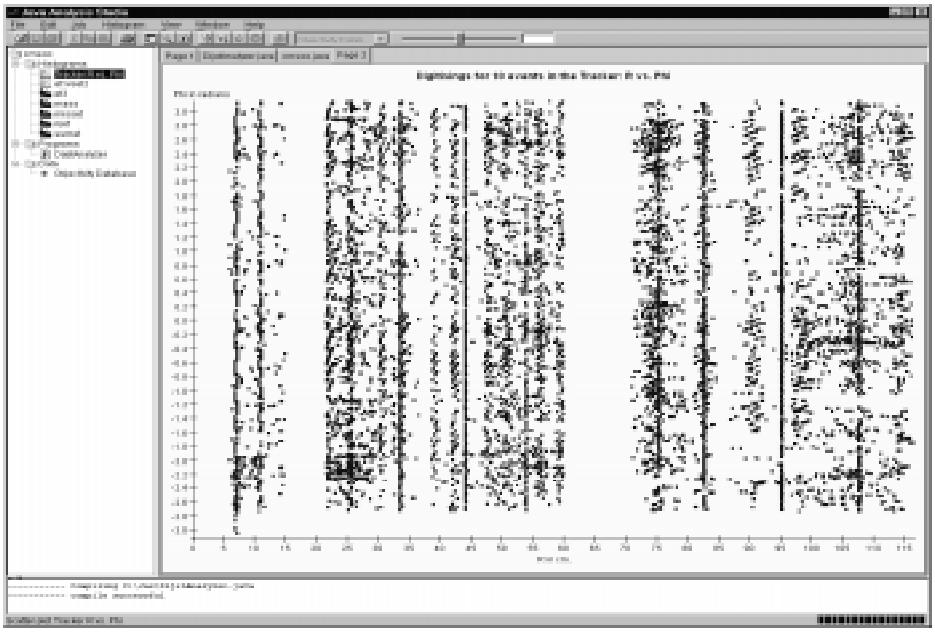
- The detector is arranged in layers: kept as persistent objects in the DB
- Each layer contains a number of TrHits
- These TrHits are kept in a persistent class "TrHitMap" in the DB
- Reconstruction begins by selecting a trio of TrHits (★) on three layers working from the outside in.
- A transient track is generated from the trio of candidate hits and "grown" if its χ^2 is small enough
- By swimming the track, and using windows (☆) on the intervening layers, unused TrHits are added to the track.
- If the track has sufficient hits, and a low enough χ^2 ,then it is added to a transient vector of good tracks, and the process begins again.
- Finally, an update lock is obtained on the DB, and all tracks are made persistent as ReconstructedTracks



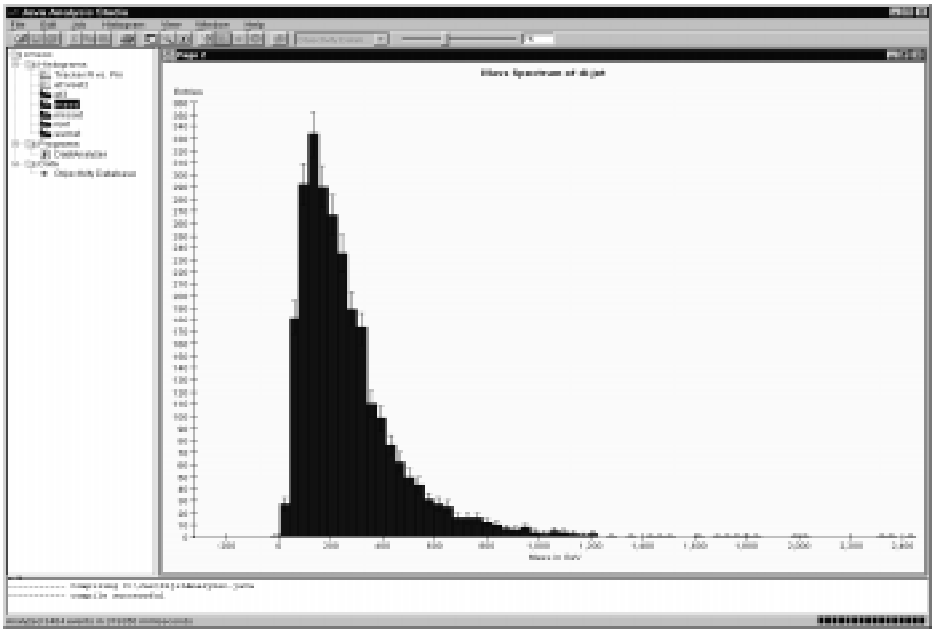
CMSOO - Java 3D Applet



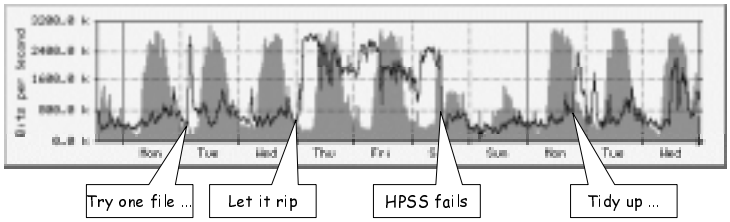
Java Analysis Studio



Java Analysis Studio



CERN-USA Traffic (week to 14/10/98)



- Transfer of ~31 GBytes of Objectivity databases from Shift20/CERN to HPSS/Caltech
- Achieved ~11 GBytes/day (equivalent to ~4 Tbytes/year, or 1 Pbyte/year on a 622 Mbits/sec link)
- HPSS hardware problem at Caltech , not network, caused transfer to abort

GIOD - Summary

- LHC Computing models specify
  - Massive quantities of raw, reconstructed and analysis data in ODBMS
  - Distributed data analysis at CERN, Regional Centres and Institutes
  - Location transparency for the end user
- GIOD is investigating
  - Usability, scalability, portability of prototype OO LHC codes coupled with ODBMS
  - In a hierarchy of large-servers, and medium/small client machines
  - With fast LAN and WAN connections
  - Using realistic LHC raw and reconstructed event data
- Results are encouraging … next steps are
  - To expand the CMSOO tests to the WAN and pseudo-production mode
  - To test integration of the HPSS with the CMSOO database

Presentation by
Fons Rademakers, CERN

ROOT

an Object-Oriented Framework for Large Scale Data Handling

Fons Rademakers

GSI Darmstadt / Hewlett-Packard

The ROOT Team:

Rene Brun (CERN)
Fons Rademakers (GSI/HP)
Masaharu Goto (HP)

Prehistory

In the beginning there was PAW

- **HBOOK** (histogramming package)
- **ZEBRA** (memory manager and I/O package)
- **KUIP** (command line and macro processor)
- **COMIS** (Fortran interpreter)
- **SIGMA** (array manipulation package)

Mini/Micro-DST analysis was done using Ntuples (RWN and CWN)

- Ntuples are basically simple tables
- Only basic types (RWN only floats)
- No data structures (except arrays in CWN's)
- No cross reference between Ntuples
- Successful because simple and efficient

Dead-end

- No way to grow to more complex data structures
- Difficult to extend
- Expensive to maintain
- Written in soon to be obsolete language -:)

Main Goals for New System

Being able to support full data analysis chain

- Raw data, DSTs, mini-DSTs, micro-DSTs

Being able to handle (store/retrieve) complex structures

- Complete objects
- Object (inheritance and containment) hierarchies

Support at least the PAW data analysis functionality

- Histogramming
- Fitting
- Visualization

Support for advanced features

- Parallelism, shared memory, remote database access, networking, ...

Only one language

- C++

Better maintainability

- Use OOP

Better extensibility

- Use OO framework technology

OO Frameworks

A *framework* is a collection of cooperating classes that make up a reusable design solution for a given problem domain.

There are three main differences between frameworks and class libraries:

Behaviour versus protocol. Class libraries are essentially collections of behaviours that you can call when you want those individual behaviours in your program. A framework, on the other hand, provides not only behaviour but also the protocol or set of rules that govern the ways in which behaviours can be combined.

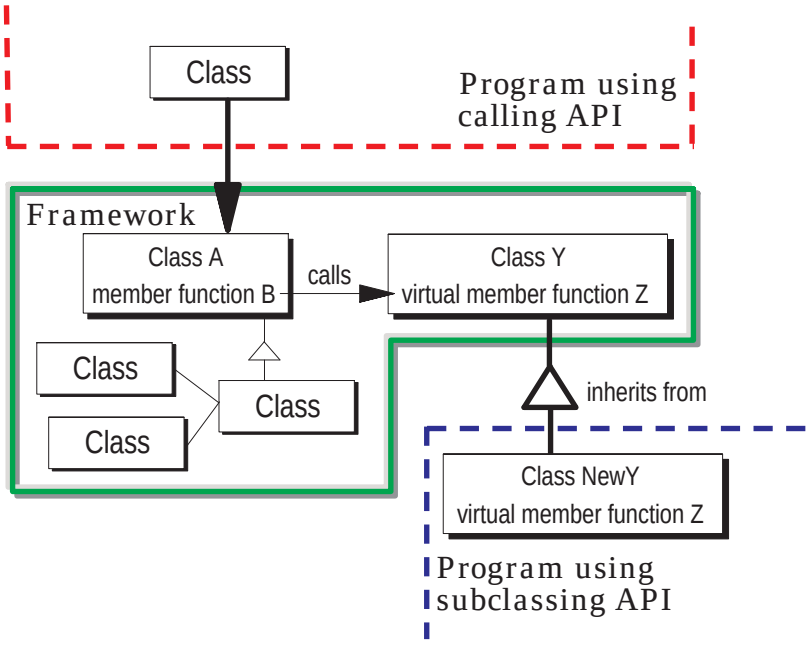
Don't call us, we'll call you. With a class library, the code the programmer writes instantiates objects and calls their member functions. With a framework a programmer writes code that overrides and is called by the framework. The framework manages the flow of control among its objects. This relationship is expressed by the principle: "Don't call us, we'll call you".

Implementation versus design. With class libraries programmers reuse only implementations, whereas with frameworks they reuse design. A framework embodies the way a family of related classes work together.

Calling API Versus Subclassing API

Frameworks can be thought of as having two Application Programmer Interfaces (APIs): a calling API, which resembles a class library, and a subclassing API, which is used for overriding framework member functions.

The calling/subclassing distinction describes the mechanism by which functions are invoked: call or be called.



Calling API versus subclassing API

Advantages of Frameworks

The benefits of frameworks can be summarized as follows:

- **Less code to write.** Much of the program's design and structure, as well as its code, already exist in the framework.

- **More reliable and robust code.** Code inherited from a framework has already been tested and integrated with the rest of the framework.

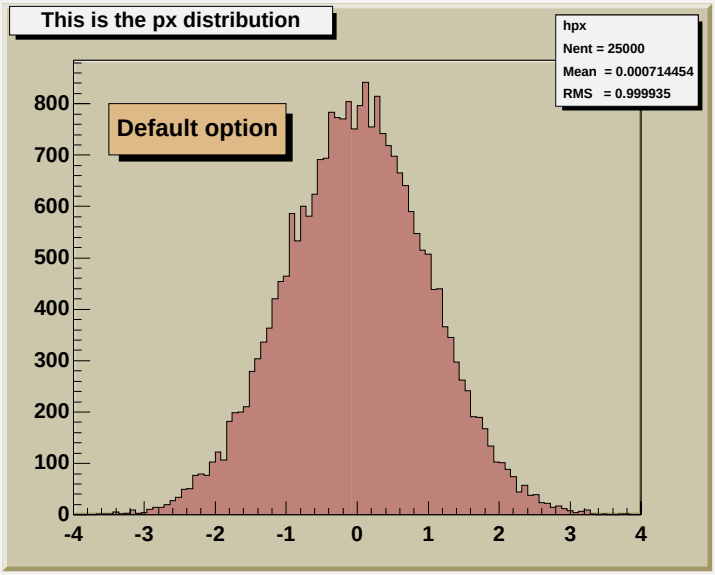
- **More consistent and modular code.** Code reuse provides consistency and common capabilities between programs, no matter who writes them. Frameworks also make it easier to break programs into smaller pieces.

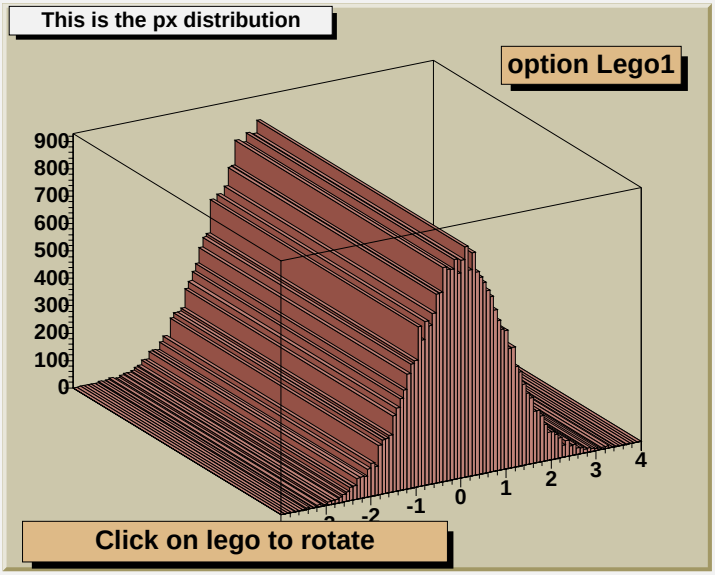
- **More focus on areas of expertise.** Users can concentrate on their particular problem domain. They don't have to be experts at writing user interfaces, graphics, or networking to use the frameworks that provide those services.

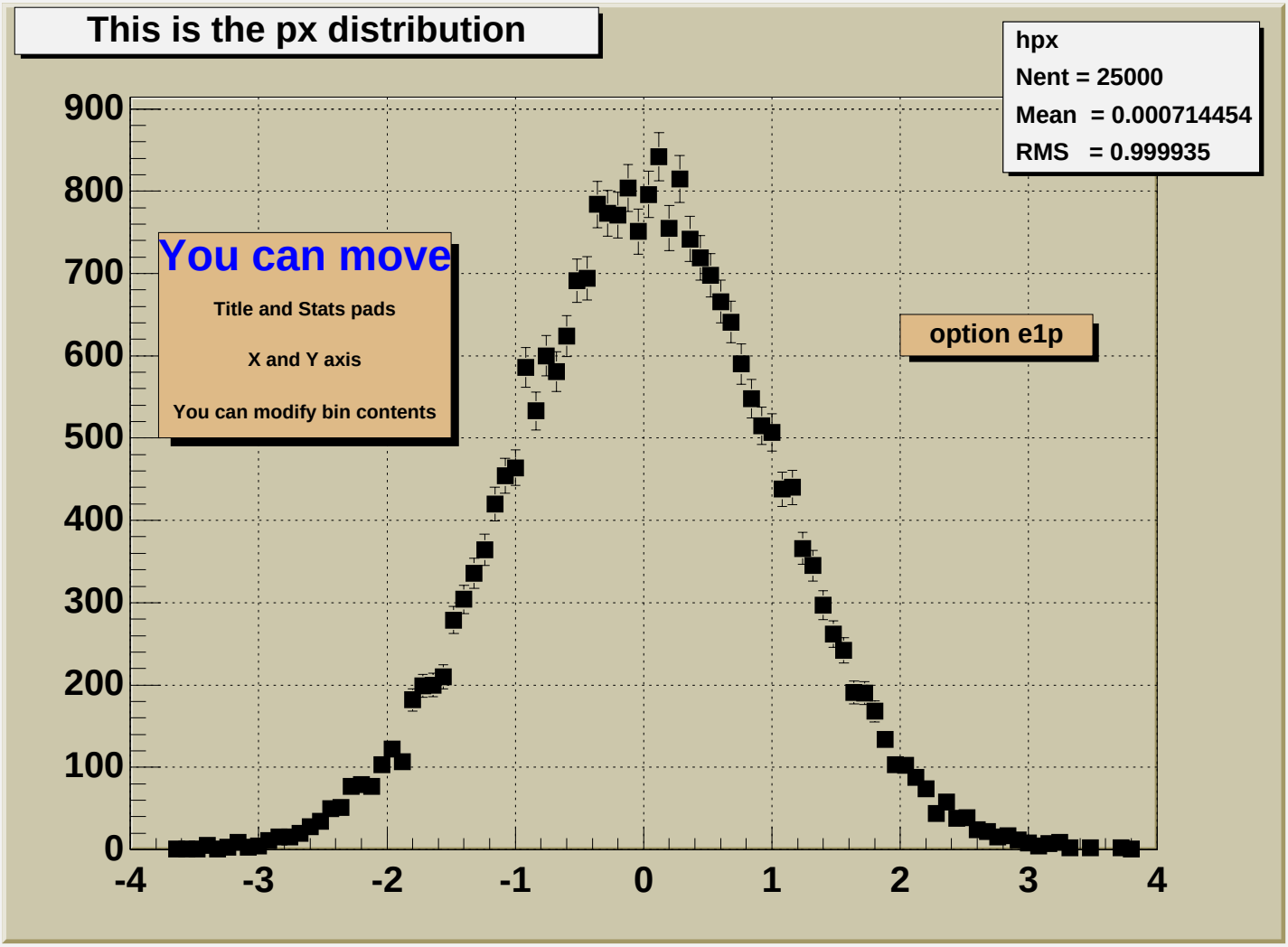
Main Features of ROOT

- A large subset of CERNLIB redesigned using OO technology
- Assumes standard C++ environment (no other special tools)
- The same language, C++, for batch, macros and interactive work
- A very powerful and mature C++ interpreter
- User describes the data model in C++ header files (no ddl, idl or odl)
- A rich set of fully integrated container classes
- Extensible via dynamic loading of shared libraries
- Histogramming and Ntuples
- Minimization (C++ version of Minuit)
- 2D and 3D graphics + OpenGL interface
- Detector description and geometric rendering
- Automatic user interface and graphics (using interpreter dictionary)
- Automatic HTML documentation generation (using dictionary)
- ➳ Automatic code generation for object serialization and inspection
- ➳ Machine independent OODB
- ➳ Trees (super Ntuples generalized to complete objects)
- ➳ Parallel processing of Trees (PROOF)
- ➳ Access to remote DB files via rootd daemon or web server
- Facilities for object storage in shared memory and multi-threading
- h2root: to convert legacy PAW/HBOOK files to ROOT files
- g2root: to convert Geant3.21 geometry files to ROOT files

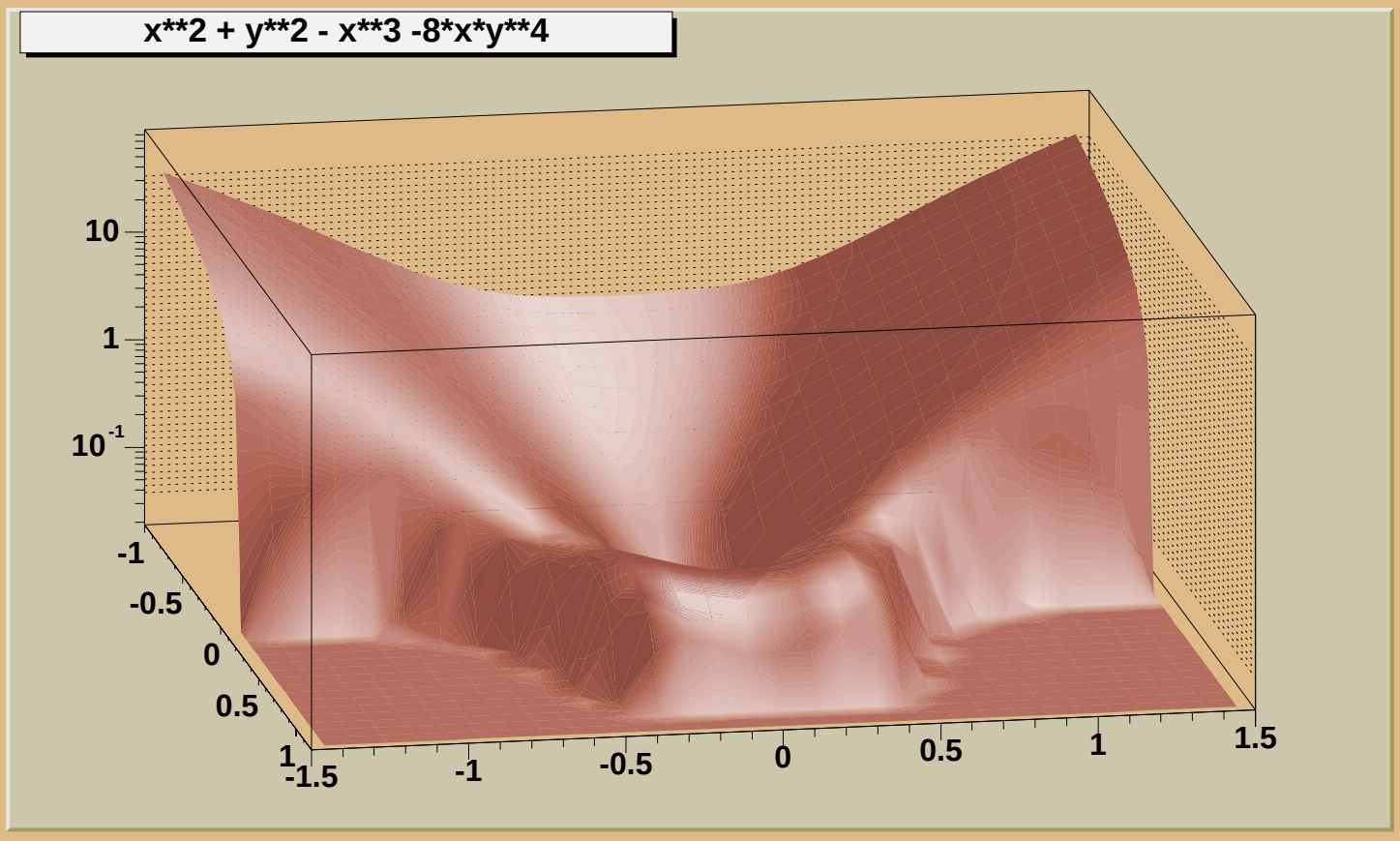
Drawing options for one dimensional histograms

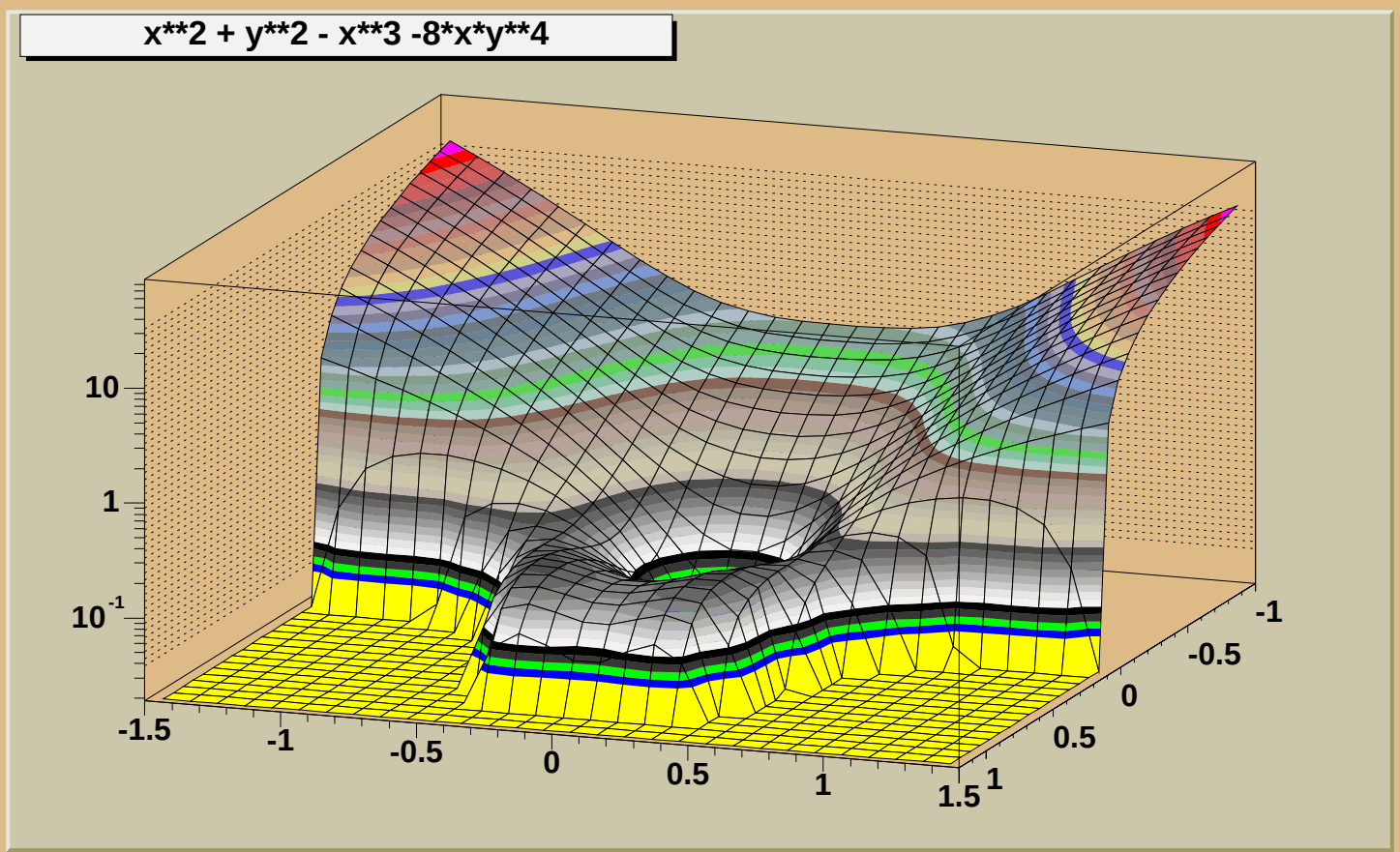


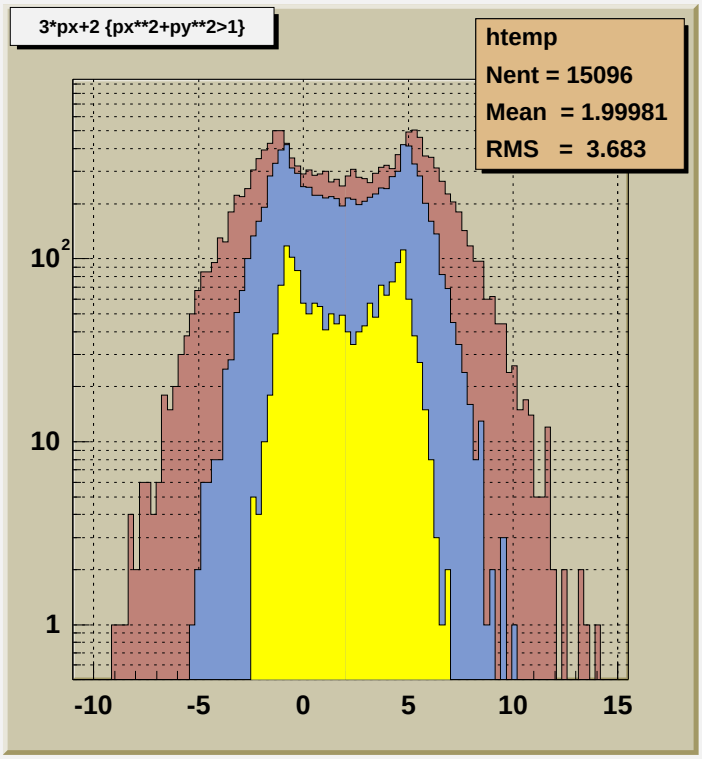


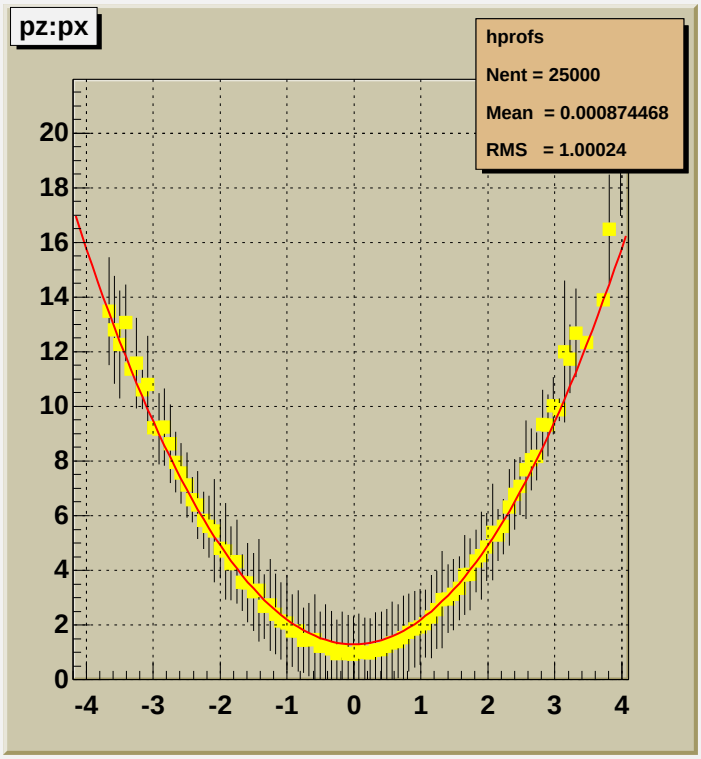


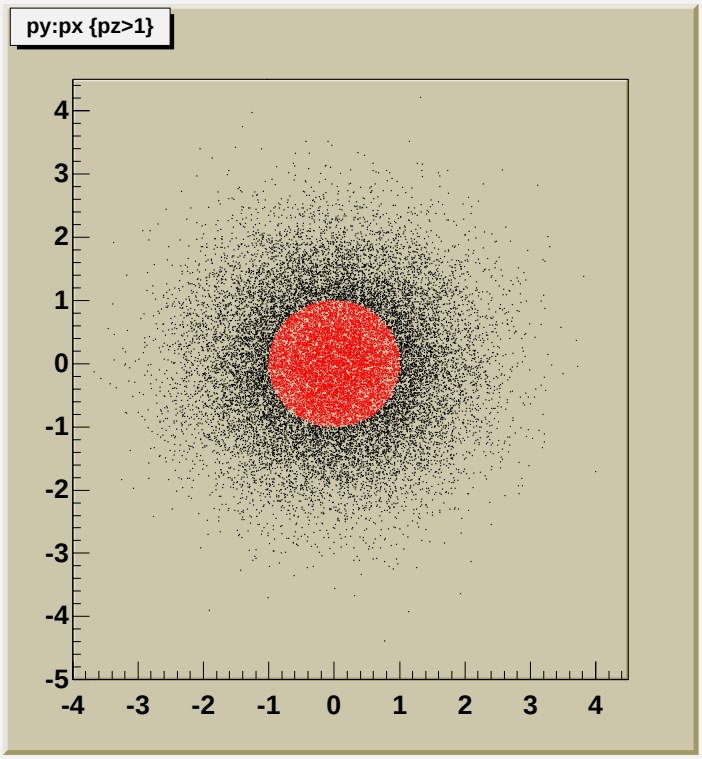
Examples of Surface options

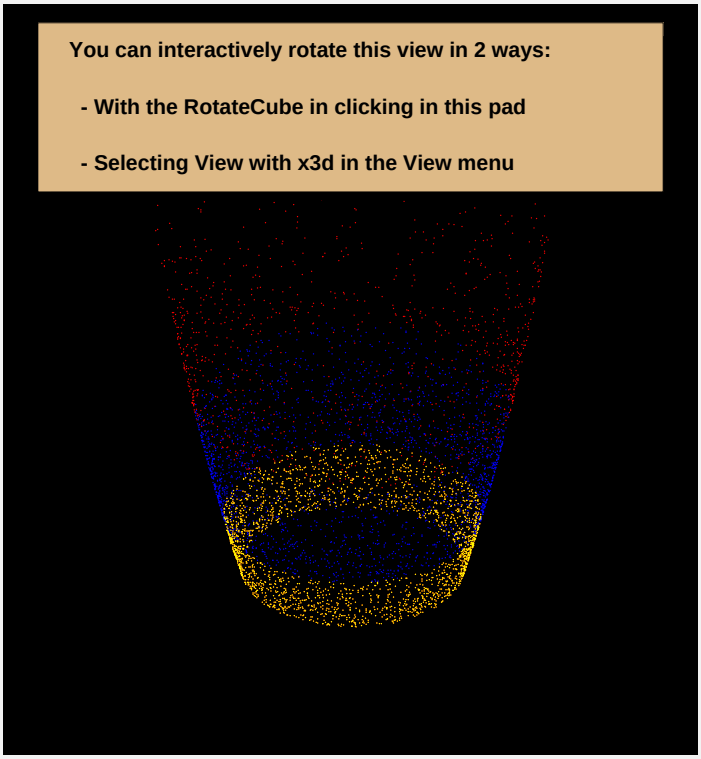





3*px+2 {px**2+py**2>1}
htemp
Nent = 15096
Mean = 1.99981
RMS = 3.683
10^2
10
1
-10
-5
0
5
10
15

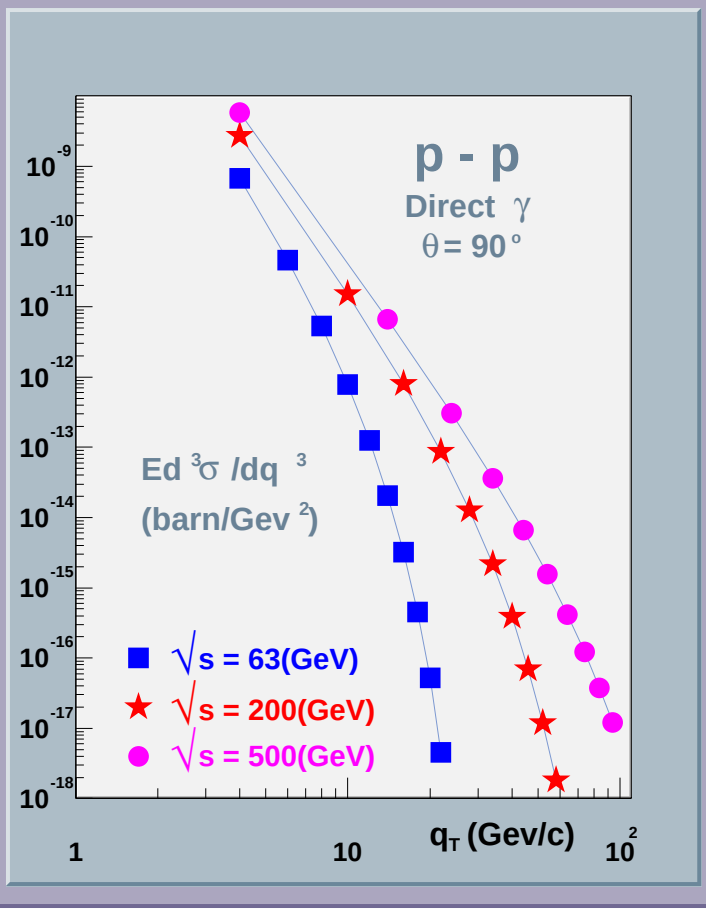

pz:px
hprofs
Nent = 25000
Mean = 0.000874468
RMS = 1.00024
20
18
16
14
12
10
8
6
4
2
0
-4
-3
-2
-1
0
1
2
3
4

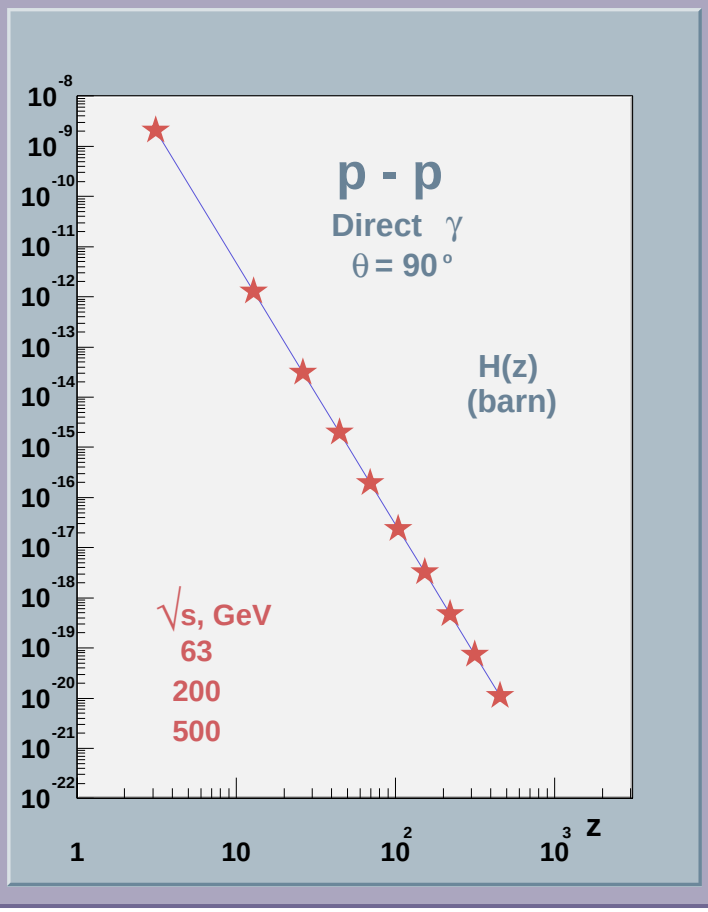

py:px {pz>1}
4
3
2
1
0
-1
-2
-3
-4
-5
-4
-3
-2
-1
0
1
2
3
4


You can interactively rotate this view in 2 ways:
- With the RotateCube in clicking in this pad
- Selecting View with x3d in the View menu

Z-scaling of Direct Photon Productions in pp Collisions at RHIC Energies

M.Tokarev, E.Potrebenikova ***JINR preprint E2-98-64, Dubna, 1998***





Interactive ROOT Sessions

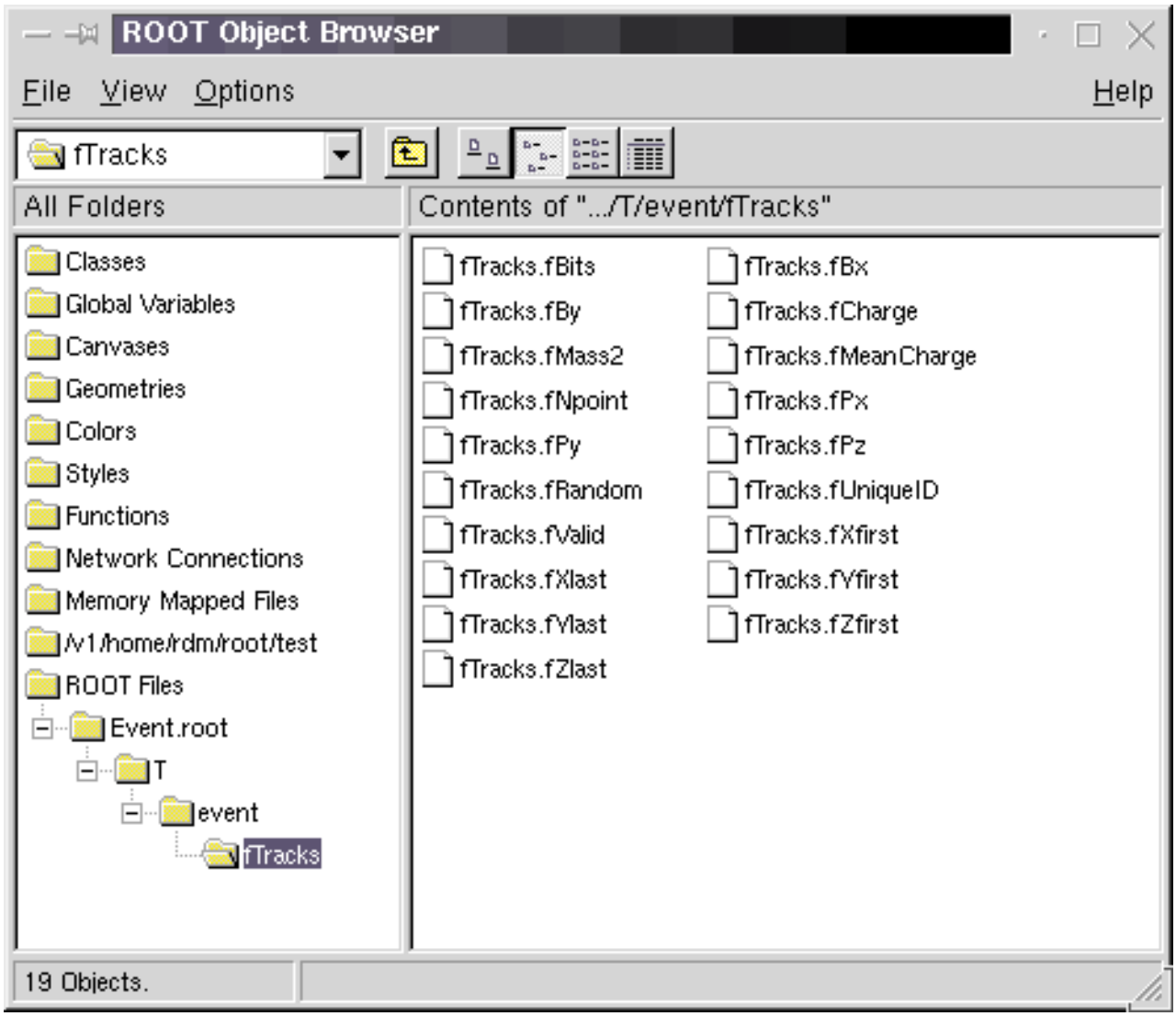
```
root> TFile f("demo.root")
root> tree.Draw("fPx")
root> tree.Draw("fCalor.GetTotalEnergy()",fNtrack<500")
root> TEvent *event
root> tree.GetBranch("event")->SetAddress(&event)
root> tree.GetEvent(4567)
root> event.Draw()
```

```
root> .x analysis.C(645)
```

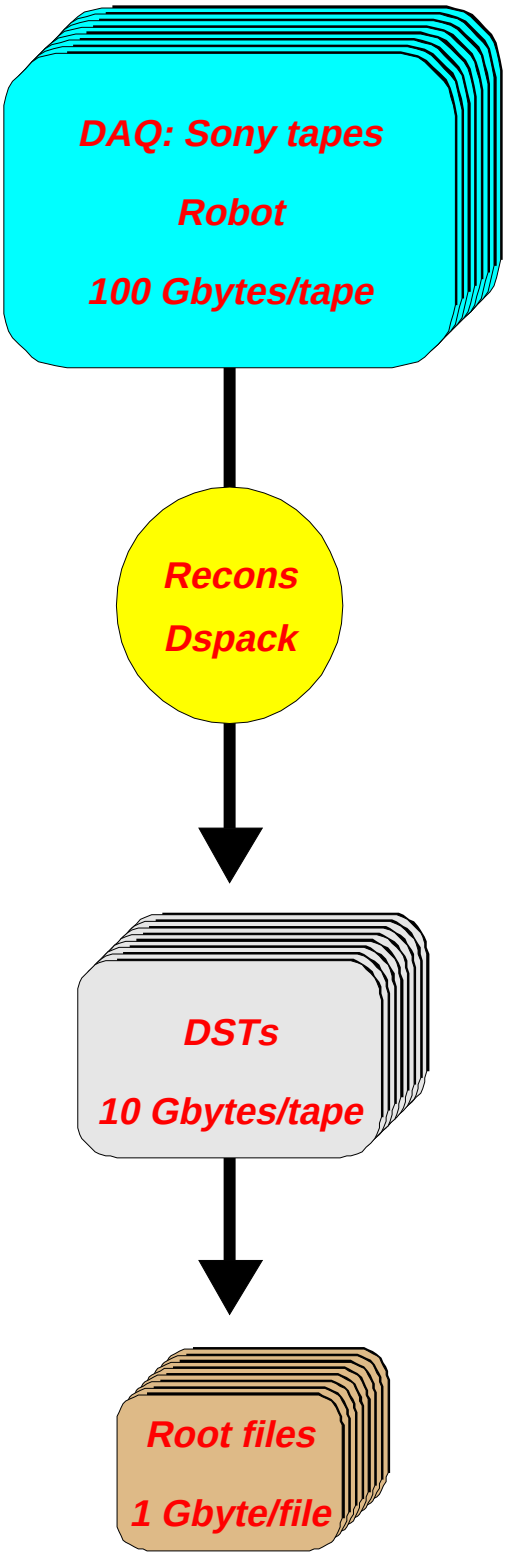
```
//-- analysis.C
void analysis(Int_t maxtracks = 500)
{
   TFile  *f = new TFile("demo.root");
   TH1F   *hpt = new TH1F("hpt","pt distrib",100,0,50);
   TTree  *tree = (TTree *) f->Get("tree");
   TEvent *event;
   tree->GetBranch("event")->SetAddress(&event);
   Int_t nevents = tree->GetEntries();

   for (int ev = 0; ev < nevents; ev++) {
      tree->GetEvent(ev);
      Int_t ntracks = event->GetNtracks();
      if (ntracks < maxtracks) continue;
      for (int t = 0; t < ntracks; t++) {
         TTrack *track = event->GetTrack(t);
         hpt->Fill(track->GetPt());
      }
   }
   hpt->Draw();
}
```

Or by Using the ROOT Browser

root> **TBrowser b**



NA49 Data Flow



Pb+Pb at 160 GeV/nucleon

October 96 run

12 Terabytes in 5 weeks

1.2 events/second

Event Size = 10 Mbytes

Reconstruction farm on SHIFT

20 HP PA8000

1 Mbyte per event on DST

1800 reconstructed tracks/event

10 minutes/event

Physics Analysis farm

Linux PCs with parallel ROOT

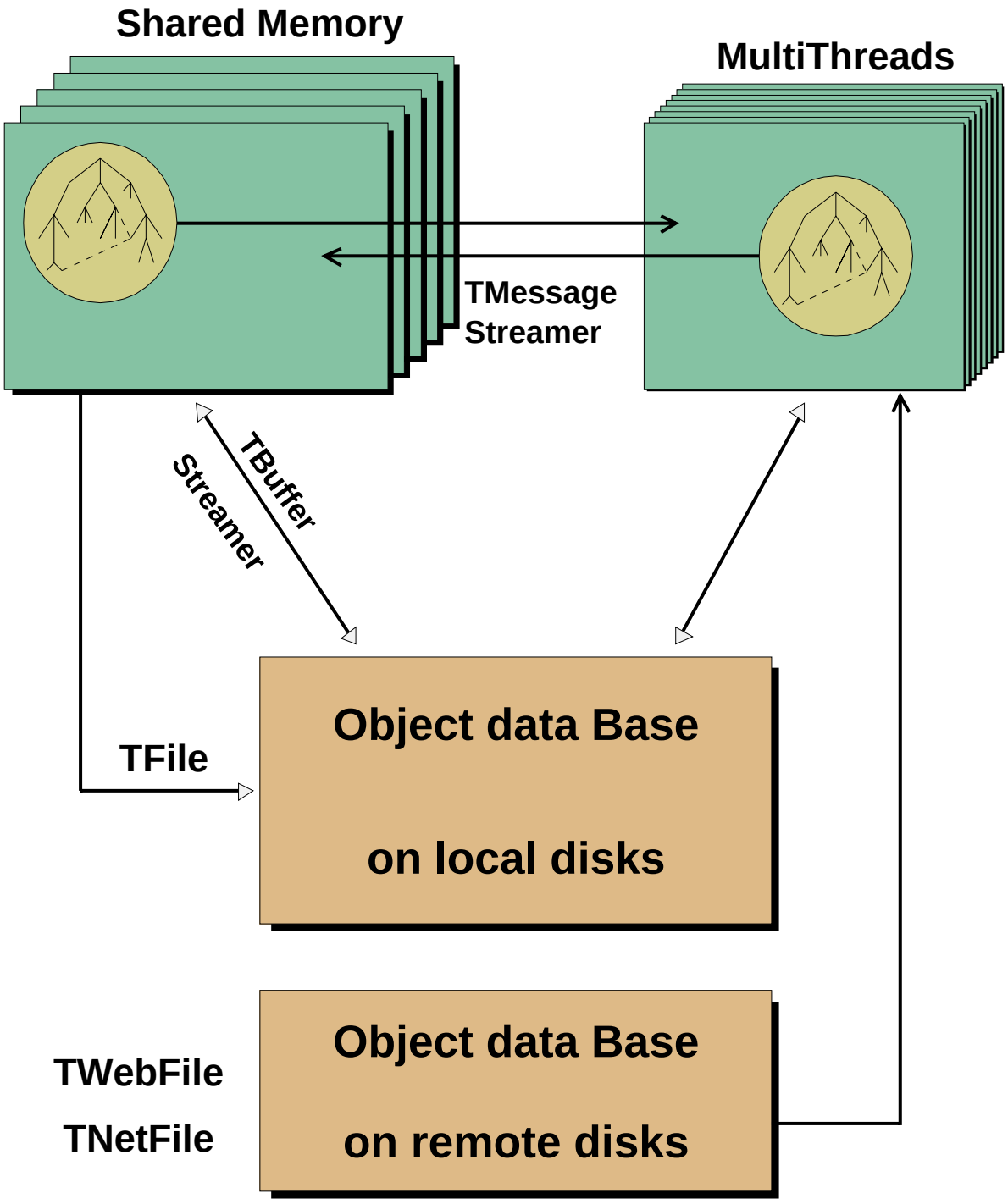
100 Kbytes/event

50 events/second

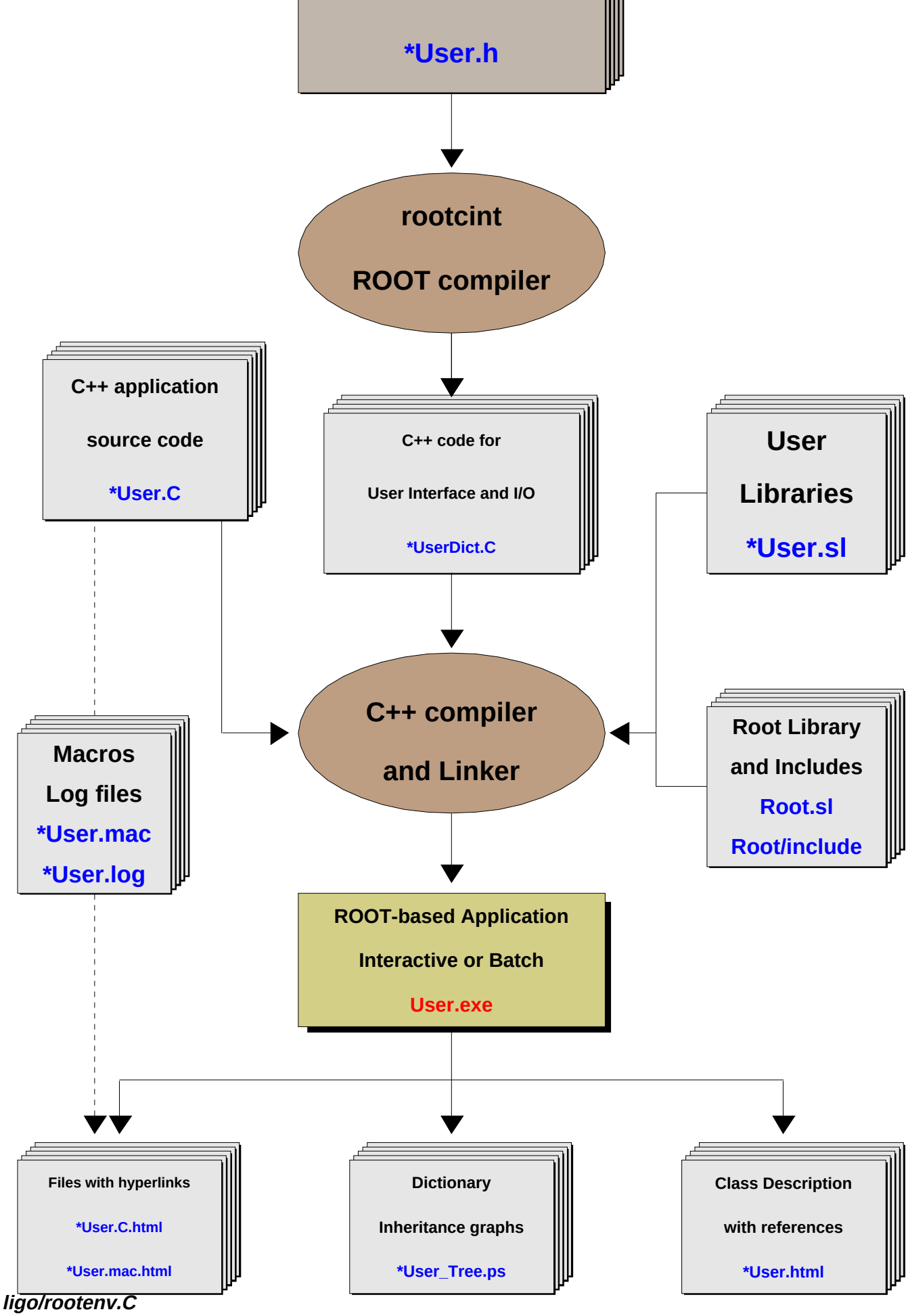
ligo/dataflow.C

Object I/O: Variety of problems: Common solutions

Multi Threading:	**Synchronization and re-entrancy**
Shared Memory:	**Synchronization**
InterProcess:	**Messaging and Client/Server**
Local File:	**Large Containers, Rapid Access**
Remote File:	**Web and Remote servers**



ligo/iogen.C


*User.h
rootcint
ROOT compiler
C++ application
source code
*User.C
C++ code for
User Interface and I/O
*UserDict.C
User
Libraries
*User.sl
C++ compiler
and Linker
Macros
Log files
*User.mac
*User.log
Root Library
and Includes
Root.sl
Root/include
ROOT-based Application
Interactive or Batch
User.exe
Files with hyperlinks
*User.C.html
*User.mac.html
Dictionary
Inheritance graphs
*User_Tree.ps
Class Description
with references
*User.html
ligo/rootenv.C

rootcint - The Dictionary Generator

```
shell> rootcint myDict.cc -c myClass.h
```

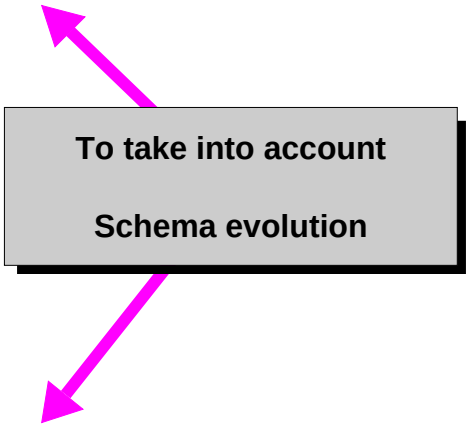
rootcint parse your header files and generates the C++ code necessary to:

- **Stream objects** (that can be stored in the DB or send over the network)
- **Access the public class members via the interpreter**
- **Access the class member functions via the GUI**
- **Get extended Run Time Type Information**
- **Generate the HTML documentation**

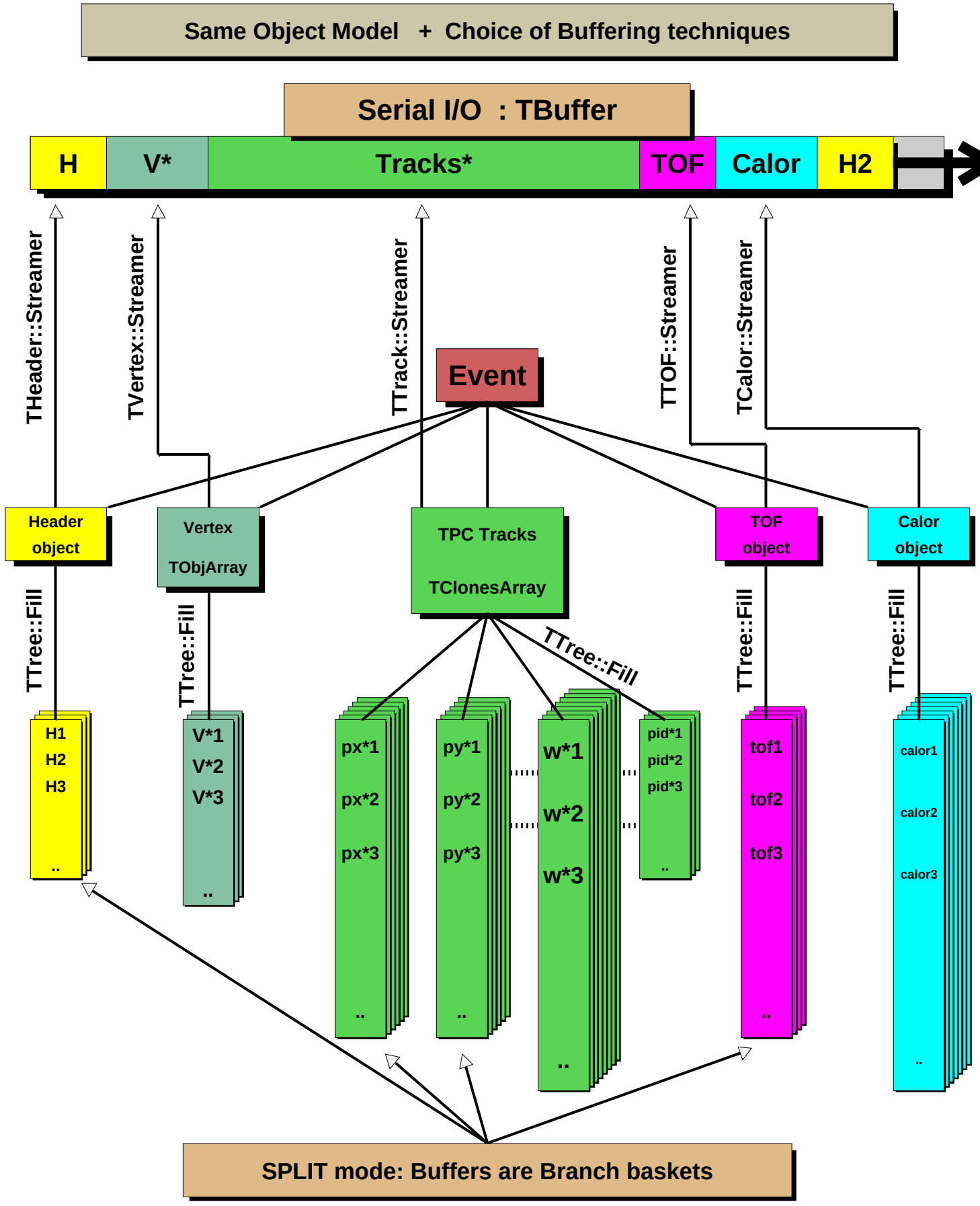
Example of Streamer function

```
class TShape: public TNamed, public TAttLine, public TAttFill
   Int_t      fNumber;      //Shape category
   Bool_t     fVisibility;  //Visibility flag
   TMaterial  *fMaterial    //Pointer to material object
```

```
void TShape::Streamer(TBuffer &buf) {
  // Stream a TShape object
  // Code automatically generated by rootcint from Shape.h
   if (buf.IsReading()) {
      Version_t v = buf.ReadVersion();
      TNamed::Streamer(buf);
      TAttLine::Streamer(buf);
      TAttFill::Streamer(buf);
      buf >> fNumber;
      buf >> fVisibility;
      buf >> fMaterial;
   } else {
      buf.WriteVersion(TShape::IsA());
      TNamed::Streamer(buf);
      TAttLine::Streamer(buf);
      TAttFill::Streamer(buf);
      buf << fNumber;
      buf << fVisibility;
      buf << fMaterial;
   }
}
```



ligo/streamer.C



ligo/iomode.C

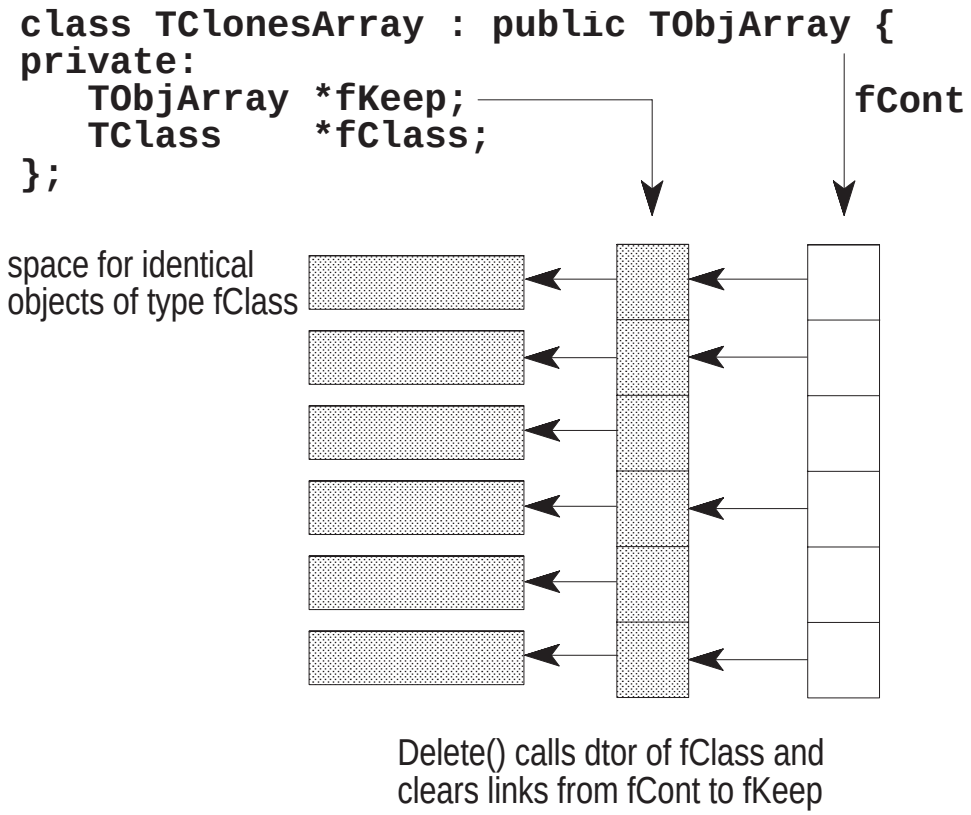
Example of a Tree

```
class TEvent : public TObject {
private:
   THeader        *fHeader;  //Event header
   TObjArray      *fVertex;  //Array of vertices
   TClonesArray   *fTracks;  //Arrays of tracks
   TTOF           *fTOF;     //Time Of Flight object
   TCalor         *fCalor;   //Calorimeter object
public:
...
...
};


main() {
   TEvent *event;
   TFile   dst("demo.root", "new");
   TTree   tree("tree", "ExampleOfTree");
   Int_t split = 1;   // or split = 0;
   tree.Branch("event", "TEvent", &event, split);

   for (int ev = 0; ev < 10000; ev++) {
      event = new TEvent(ev);
      ...
      ...
      tree.Fill();
      delete event;
   }
   dst.Close();
}
```

TClonesArray - Array of Identical Objects

Why do we need special collection classes and not just STL?

An array of clone (identical) objects. The memory for the objects stored in the array is allocated only once in the lifetime of the clones array. All objects must be of the same class. For the rest this class has the same properties as TObjArray.



The class is specially designed for repetitive data analysis tasks, where in a loop many times the same objects are created and deleted.

Theory of the TClonesArray

To reduce the very large number of new and delete calls in large loops like this (O(100000) x O(10000) times new/delete):

```
TObjArray a(10000);
while (TEvent *ev = (TEvent *)next()) {    // O(100000)
   for (int i = 0; i < ev->Ntracks; i++) { // O(10000)
      a[i] = new TTrack(x,y,z,...);
      ...
      ...
   }
   ...
   a.Delete();
}
```

One better uses a TClonesArray which reduces the number of new/delete calls to only O(10000):

```
TClonesArray a("TTrack", 10000);
while (TEvent *ev = (TEvent *)next()) {    // O(100000)
   for (int i = 0; i < ev->Ntracks; i++) { // O(10000)
      new(a[i]) TTrack(x,y,z,...);
      ...
      ...
   }
   ...
   a.Delete();
}
```

Considering that a new/delete costs about 70 μs, O(10^9) new/deletes will save about 19 hours.

Where is ROOT Being Used?

High Energy Physics:

Offline: NA49, Phobos, Phenix, Star, Zeus, H1, Hera-B, Delphi, Atlas, Alice, CMS, LHCb, D0, CDF, SLD, Hermes, E917, E877, Belle, Opal, AMS, E614, Cleo, ...

Online: Finuda, Kloe, Compass, LHCb, ...

Theory: S. Jadach, ...

Nuclear Physics:

Offline: Hades (GSI), ...

Online: Munchen, ...

Astronomy:

Offline: Integral Gamma Ray Satelite, Celeste

Miscelleneous:

Plasma Physics, Biology, Genetics, Insurance, Finance, Pharmaceutical, ...

More than 20000 downloads of ROOT since January 1997

More than 200000 hits on the web site per month

More than 720 registered users

ROOT Documentation and Availability

Everything about ROOT can be found at:

The ROOT web site: **http://root.cern.ch/**

From:

http://root.cern.ch/root/Availability.html

You can download

- ☞ **The complete source**
- ☞ **The test programs and macros**
- ☞ **The PostScript version of the documentation**
- ☞ **The HTML version of the documentation**
- ☞ **The executable modules and shared libraries for:**
- ❒ **Linux (Intel, PowerPC)**
- ❒ **HP-UX**
- ❒ **Solaris (Sun, Intel)**
- ❒ **Alpha OSF**
- ❒ **AIX**
- ❒ **SGI**
- ❒ **Windows NT/95**

ROOT does not require any commercial components!

Bottomline: the system is free and runs everywhere