INTRODUCTION TO MDSplus

by
Jeff Schachter

Presented at
General Atomics
San Diego, California

SEPTEMBER 21, 1999

DIII–D
NATIONAL FUSION FACILITY
SAN DIEGO

GENERAL ATOMICS

CLASS OUTLINE

Class 1: What is MDSplus and why should you care?

1. MDSplus — the philosophy
2. MDSplus at DIII–D
3. Basic concepts
4. MDSplus expressions
5. What you can do with MDSplus
6. Getting started

Class 2: How to use MDSplus

1. Overview
2. Examining tree structure
3. Reading data out of MDSplus
4. Writing data into MDSplus
5. Designing and building trees

CLASS 1: WHAT IS MDSplus AND WHY SHOULD YOU CARE?

- MDSplus is a data system from MIT, LANL, and IGI-Padova
  - Provides for acquisition, storage, and organization of data
  - At DIII–D: a centralized repository for analyzed data
- MDSplus provides many benefits for data storage
  - Simplifies data access
  - Flexibility to change and add results at any time
- Purpose of class: to encourage users to access MDSplus directly
  - Find out more than just the numbers
  - Store your own data (much faster than waiting for Jeff to do it)

(1) MDSplus — THE PHILOSOPHY

- **The four tenets of MDSplus:**

  1. All data is equivalent

  2. Store everything relevant

  3. Logical, not physical, user interaction

  4. Direct and uniform data access

TENET 1: ALL DATA IS EQUIVALENT

- System does not care about data type or origin
- All data stored in the same way
  - Raw digitizer bits
  - Calibrated data
  - Calibrations
  - Analyzed data
  - Text
  - Geometry
- Benefit: learn only one access method

TENET 2: STORE EVERYTHING RELEVANT

- Since all data is equivalent, store all information
  - Acquisition setup
  - Raw data
  - Analyzed data
  - Comments
  - Calibrations
  - Geometry
- Benefit: never need to look anywhere else

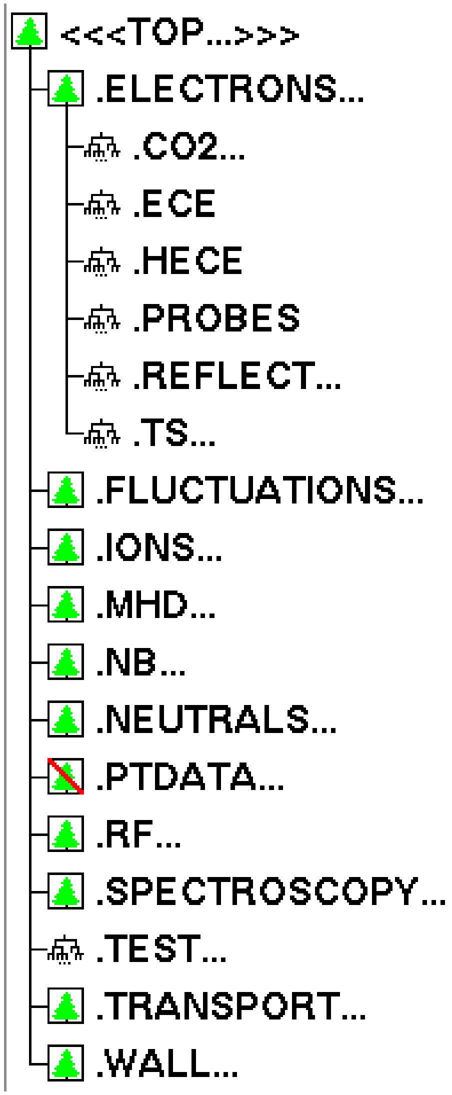
TENET 3: LOGICAL, NOT PHYSICAL, USER INTERACTION

- Organize stored information in hierarchical tree structure

- Benefits:
  - Structure provides context for data
  - Relationships made explicit
  - Easy to browse data for a shot

- Do not need to know:
  - Location of data files
  - Format of data files
  - Format of individual data records

- Generic operations on data based on structure
  - e.g., collect list of all EFIT a0 file outputs from tree



- <<<TOP...>>>
  - .ELECTRONS...
    - .CO2...
    - .ECE
    - .HECE
    - .PROBES
    - .REFLECT...
    - .TS...
  - .FLUCTUATIONS...
  - .IONS...
  - .MHD...
  - .NB...
  - .NEUTRALS...
  - .PTDATA...
  - .RF...
  - .SPECTROSCOPY...
  - .TEST...
  - .TRANSPORT...
  - .WALL...

TENET 4: DIRECT AND UNIFORM DATA ACCESS

- Can see exactly what is stored
  - Present/not present
  - Units
  - Size and shape
- Data access function calls independent of platform and language
- Benefits:
  - Know exactly what you are getting
  - Know how to get it no matter where you are

CONTRAST: USERS INSULATED FROM DATA WITH CODE

- Cannot see the data as it is — only as programmer allows

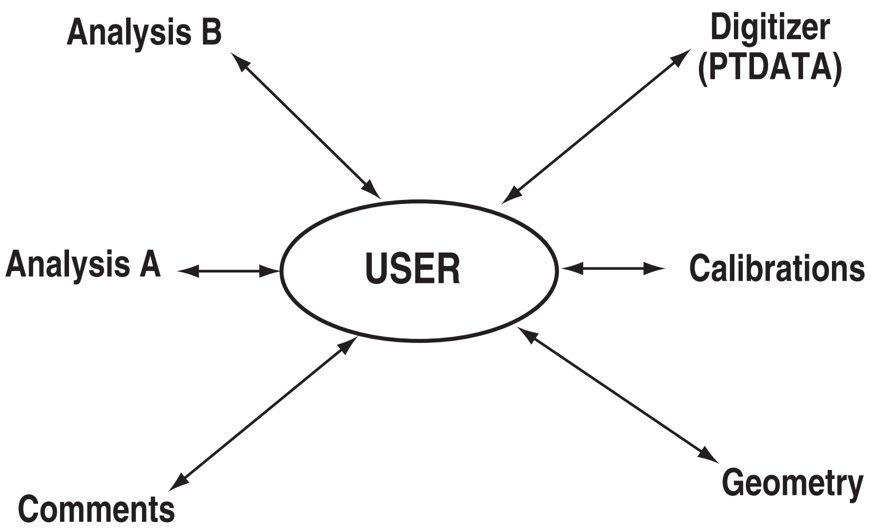
- Example: IDL “get” routines (get_cer, get_ts, etc.)
  - Return predetermined shape and structure
  - Must therefore have way to signify “no data”
  - Different return values and structures for each dataset

- Only run in IDL

- Example: EFITviewer rotation correction

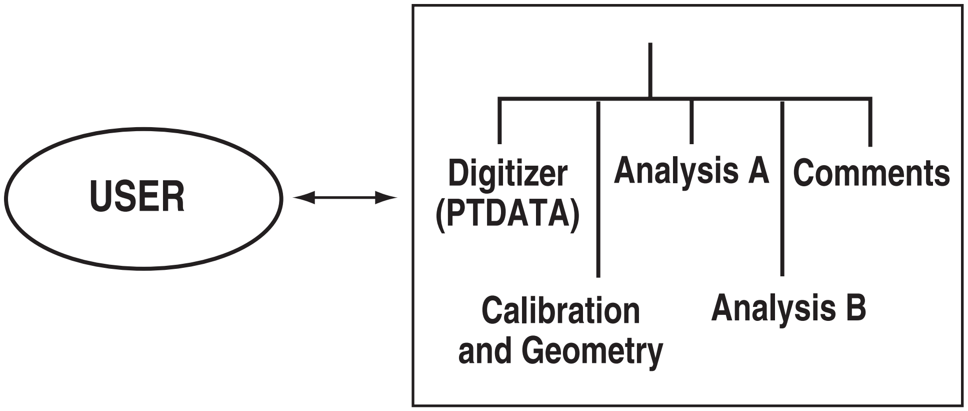
MDSplus SIMPLIFIES DATA ACCESS

Conventional Storage



- Separate interface for each data type
- Must know data format and file location
- Data and context stored separately
- Hard to share results

MDSplus



- One interface to many data types
- Only need location of data in tree
- Store <u>all</u> relevant information
- Remote exploration of data productive

DIII–D
NATIONAL FUSION FACILITY
SAN DIEGO

MDSplus IS NOT A RELATIONAL DATABASE

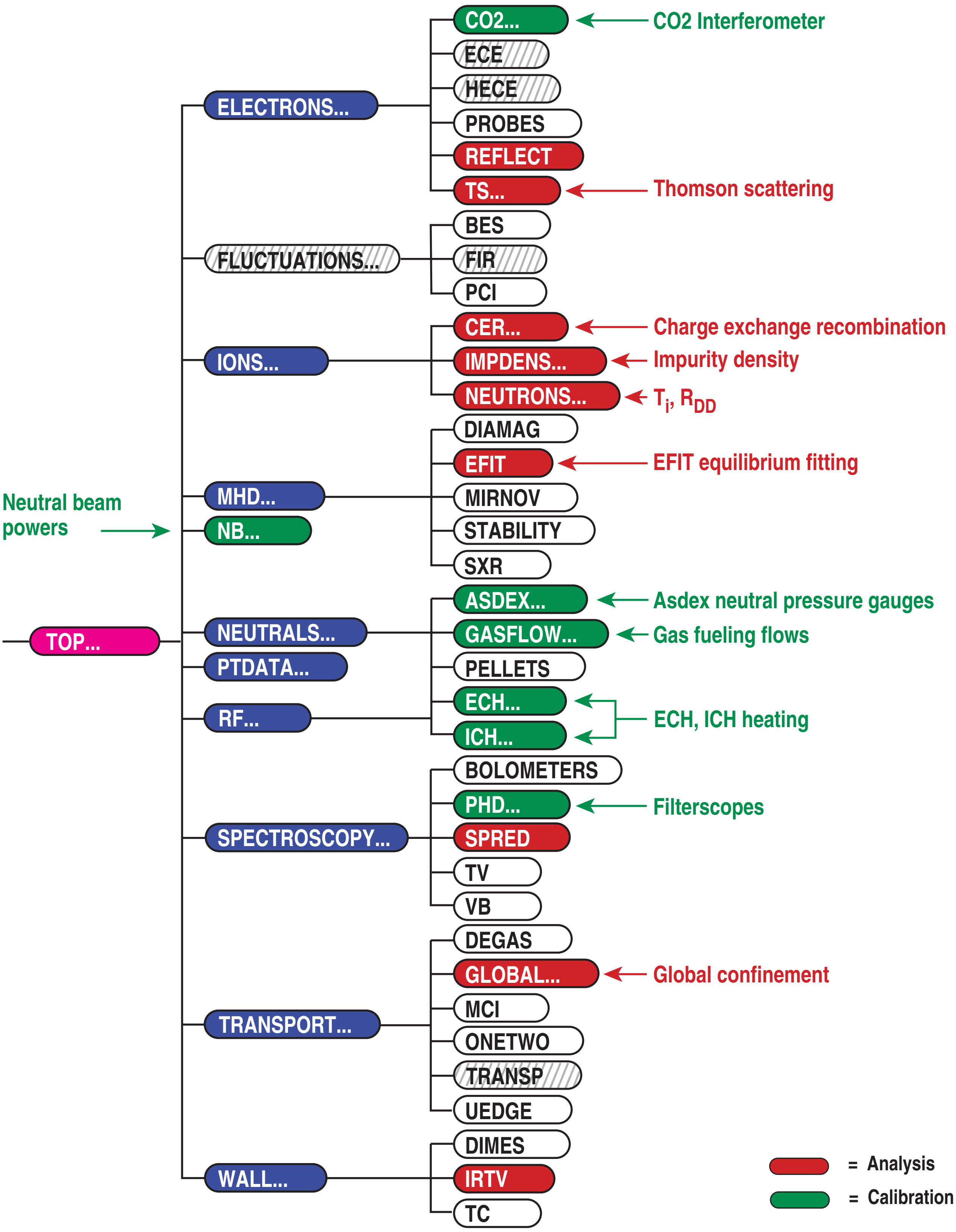
Query: Return a list of all H–mode shots from 1998 with $I_p > 1$ MA and $P_{NB} > 5$ MW

- MDSplus
  - Stores *all* data
  - Not optimized for queries across multiple shots
- Relational database
  - Stores highlights of data
  - Optimized for queries
- The two work together

(2) MDSplus AT DIII–D:
A CENTRALIZED REPOSITORY FOR ANALYZED DATA

- PTDATA handles raw data
- MDSplus stores data from >4700 shots so far
  - All shots since 1998 plus most popular old shots
  - 45 GB total disk space usage
  - Currently up to 18 datasets per shot (20 MB/shot)
  - More being added (add your own after this class)
- Access to MDSplus built into existing tools and routines
  - ReviewPlus, EFITtools, GAprofiles, REVIEW
  - getalldat, gadat, get_ts, get_cer, get_cerquick, etc.

ANALYZED DATASETS STORED IN MDSplus



- 🟪 TOP...
  - 🟦 ELECTRONS...
    - 🟩 CO2... ← CO2 Interferometer
    - ▨ ECE
    - ▨ HECE
    - PROBES
    - 🟥 REFLECT
    - 🟥 TS... ← Thomson scattering
  - ▨ FLUCTUATIONS...
    - BES
    - ▨ FIR
    - PCI
  - 🟦 IONS...
    - 🟥 CER... ← Charge exchange recombination
    - 🟥 IMPDENS... ← Impurity density
    - 🟥 NEUTRONS... ← T_i, R_{DD}
  - 🟦 MHD...
    - DIAMAG
    - 🟥 EFIT ← EFIT equilibrium fitting
    - MIRNOV
    - STABILITY
    - SXR
  - 🟩 NB... ← Neutral beam powers
  - 🟦 NEUTRALS...
    - 🟩 ASDEX... ← Asdex neutral pressure gauges
    - 🟩 GASFLOW... ← Gas fueling flows
    - PELLETS
  - 🟦 PTDATA...
  - 🟦 RF...
    - 🟩 ECH... ← ECH, ICH heating
    - 🟩 ICH... ← ECH, ICH heating
  - 🟦 SPECTROSCOPY...
    - BOLOMETERS
    - 🟩 PHD... ← Filterscopes
    - 🟥 SPRED
    - TV
    - VB
  - 🟦 TRANSPORT...
    - DEGAS
    - 🟥 GLOBAL... ← Global confinement
    - MCI
    - ONETWO
    - ▨ TRANSP
    - UEDGE
  - 🟦 WALL...
    - DIMES
    - 🟥 IRTV
    - TC

🟥 = Analysis

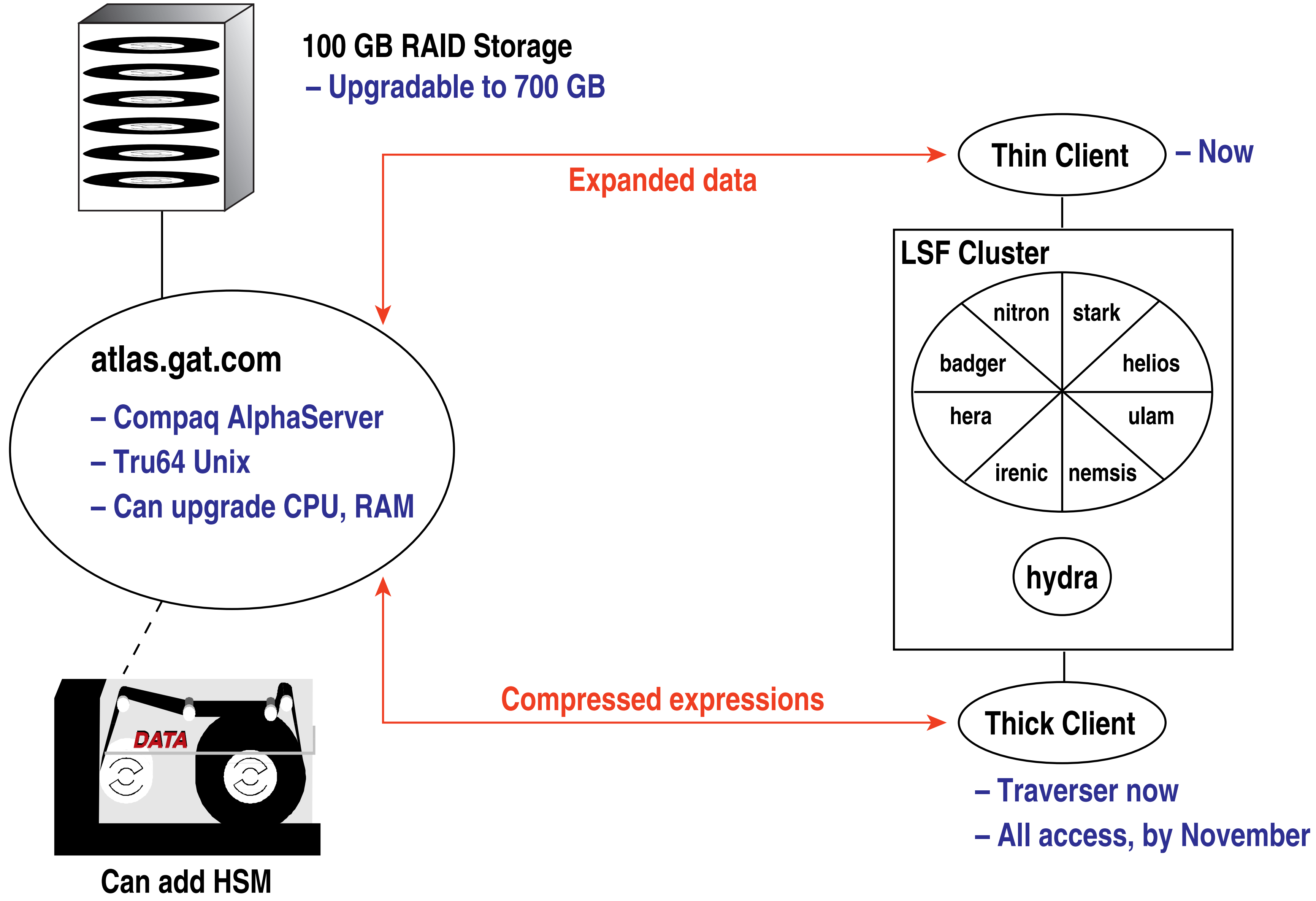
🟩 = Calibration

DIII–D
NATIONAL FUSION FACILITY
SAN DIEGO

GENERAL ATOMICS

252–99/df

DIII–D MDSplus DATA SUPPLIED BY CLIENT/SERVER SYSTEM



100 GB RAID Storage

– Upgradable to 700 GB

atlas.gat.com

– Compaq AlphaServer

– Tru64 Unix

– Can upgrade CPU, RAM

Can add HSM

Expanded data

Compressed expressions

Thin Client – Now

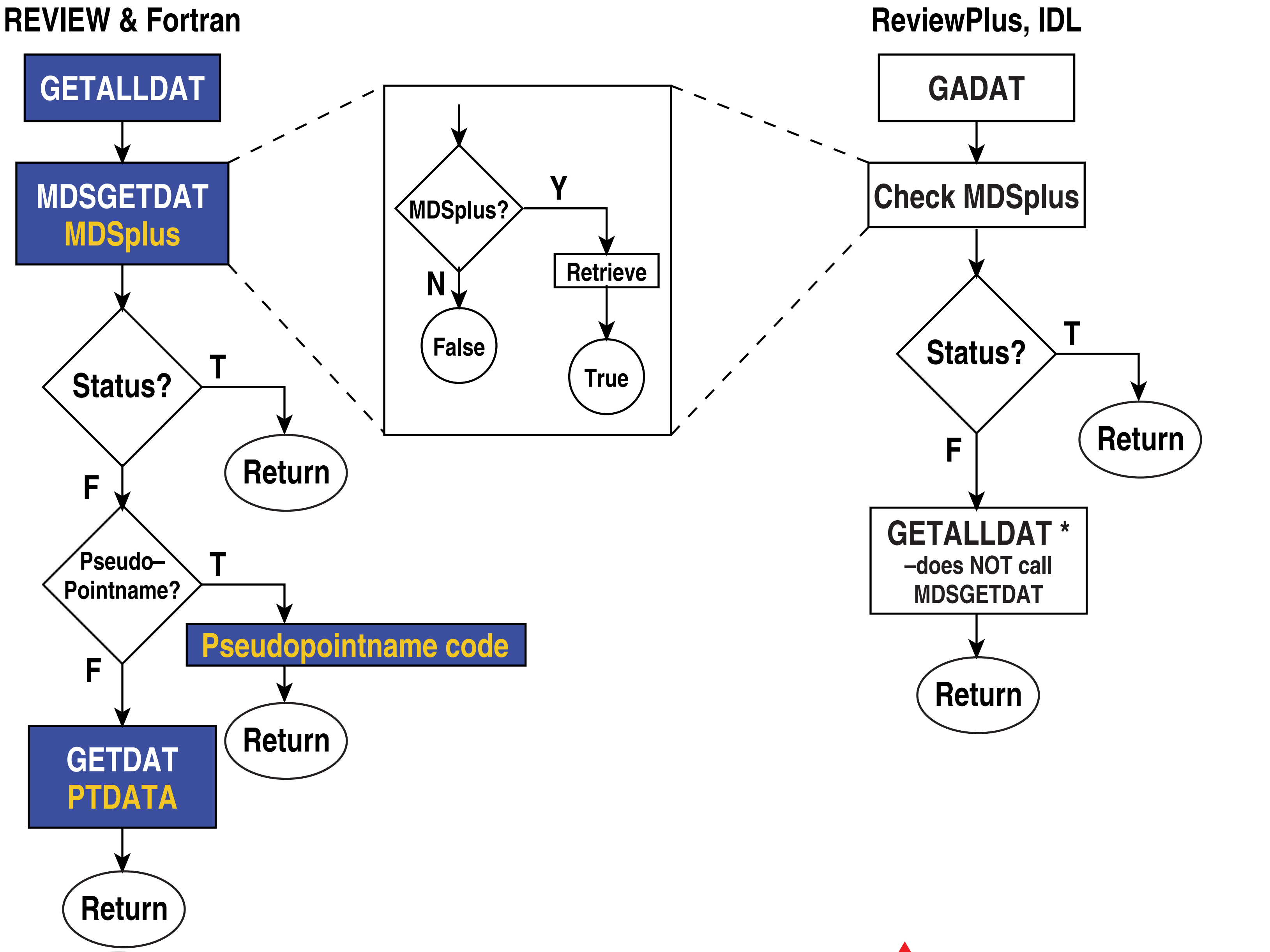
LSF Cluster

Thick Client

– Traverser now

– All access, by November

SO YOU WANT TO RETRIEVE SOME DATA?



DIII–D
NATIONAL FUSION FACILITY
SAN DIEGO

GENERAL ATOMICS

252-99 jy

(3) BASIC CONCEPTS

- Concepts to explain:
  - Data types (primitive and complex)
  - Trees and nodes
  - Locating data
  - Model and pulse trees
  - API and retrieving data

DATA TYPES

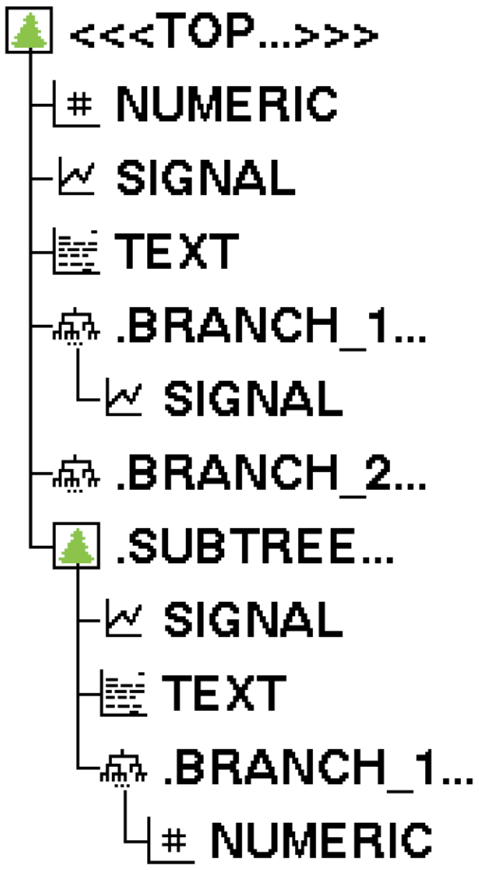
- **Primitive**
  - Integer (several precisions)
  - Float
  - Double
  - Complex numbers
  - String

- **Arrays and scalars**
  - Can have an array of any of the above types

- **Complex data types**
  - Signal: data and dimensions (dependent and independent values)
  - Units: data and units string
  - Many others: (see discussion of MDSplus expressions below)

TREES AND NODES

- All data for a given shot stored in a “tree”
  - Hierarchical organization of “nodes”
- Node type determines what is stored
  - Corresponds with data types: numeric, text, signal, etc.
  - Structure nodes: branches that do not contain data
- One tree can contain many “subtrees”
  - Subtree is basic unit for file storage
- Tree structure can be changed at any time (see later)



```
<<<TOP...>>>
  # NUMERIC
  SIGNAL
  TEXT
  .BRANCH_1...
    SIGNAL
  .BRANCH_2...
  .SUBTREE...
    SIGNAL
    TEXT
    .BRANCH_1...
      # NUMERIC
```

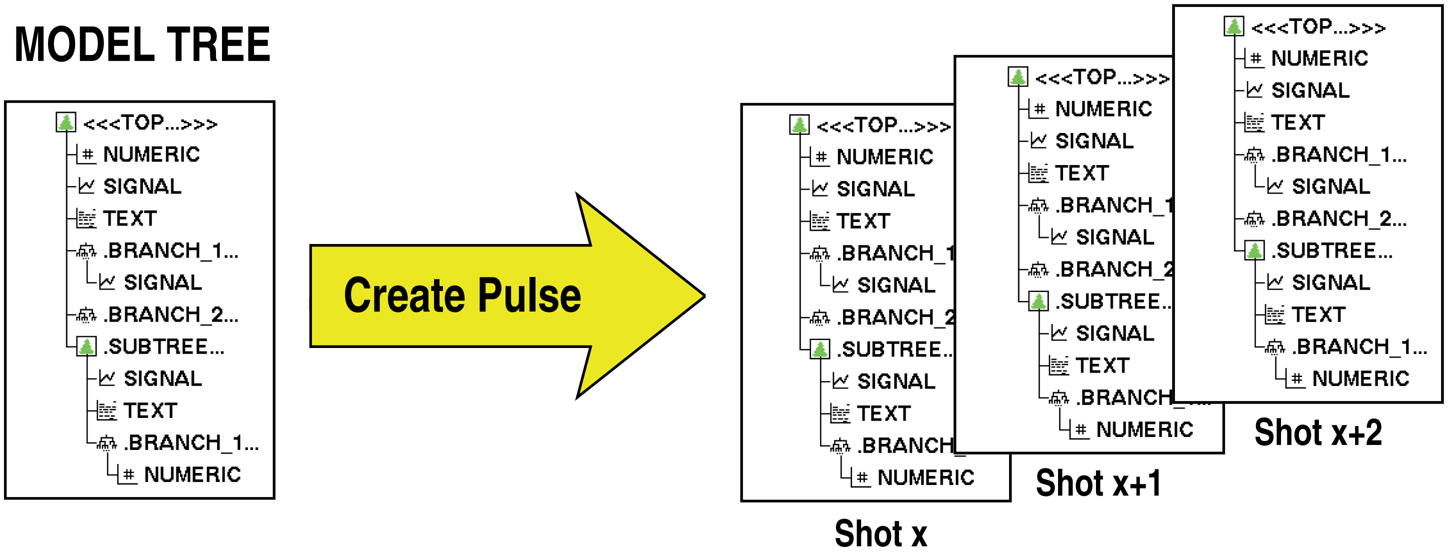
LOCATING DATA

- To retrieve data from MDSplus, must know subtree, and location of data within
- Absolute paths:
  - \TOP:NUMERIC
  - \TOP:SIGNAL
  - \TOP.BRANCH_1:SIGNAL
- Relative paths: depend on current location in tree
- Dots and colons
  - A pain
  - "." separates structure nodes
  - ":" separates all other nodes
  - IRTV branch of DIII–D's WALL subtree: no structure nodes → just use ":"
- Tags: shortcuts or aliases for a node
  - \TSTE_CORE, \WMHD
  - Tag must be unique within subtree

MODEL AND PULSE TREES

PULSE TREES

MODEL TREE



```
<<<TOP...>>>
├ NUMERIC
├ SIGNAL
├ TEXT
├ .BRANCH_1...
│ └ SIGNAL
├ .BRANCH_2...
└ .SUBTREE...
  ├ SIGNAL
  ├ TEXT
  └ .BRANCH_1...
    └ NUMERIC
```

Create Pulse

```
<<<TOP...>>>
├ NUMERIC
├ SIGNAL
├ TEXT
├ .BRANCH_1
│ └ SIGNAL
├ .BRANCH_2
└ .SUBTREE..
  ├ SIGNAL
  ├ TEXT
  └ .BRANCH_
    └ NUMERIC
```

Shot x

```
<<<TOP...>>>
├ NUMERIC
├ SIGNAL
├ TEXT
├ .BRANCH_1
│ └ SIGNAL
├ .BRANCH_2
└ .SUBTREE..
  ├ SIGNAL
  ├ TEXT
  └ .BRANCH_...
    └ NUMERIC
```

Shot x+1

```
<<<TOP...>>>
├ NUMERIC
├ SIGNAL
├ TEXT
├ .BRANCH_1...
│ └ SIGNAL
├ .BRANCH_2...
└ .SUBTREE...
  ├ SIGNAL
  ├ TEXT
  └ .BRANCH_1...
    └ NUMERIC
```

Shot x+2

- Template to create pulse trees
- Data nodes empty
- Modification affects only future shots

- Pulse tree holds data for given shot
- Data nodes full
- Pulse tree can be modified independent of other trees

MDSplus = “Model Driven System”

MDSplus API

- API = “application program interface”
- Fancy way of saying “how to access MDSplus”
- Nine basic routines:

mdsconnect	connect to server
mdsdisconnect	disconnect from server
mdsopen	open tree
mdsclose	close tree
mdsvalue	retrieve data (see next slide)
mdsput	put data into tree
mdssetdefault	change current location in open tree
mdsgetmsg	print MDSplus error message corresponding to status code
mdstcl	direct tree access/editing

- Documentation on our web site: http://fusion.ga.com/comp/analysis/mdsplus/

RETRIEVING DATA

- Basic process:
  - Locate data in tree → subtree, shot, path or tag
  - Open tree for shot of interest (mdsopen)
  - mdsvalue (path)
- But there is much more you can do
  - mdsvalue takes an MDSplus “expression” as an argument
  - Expressions form the heart of MDSplus data representation
  - Can be quite complex

(4) MDSplus EXPRESSIONS

- Language of expressions is “TDI” (tree-data interface)

Scalars	1. “string”
Arrays	[1.,2.] [“string 1”, “string 2”]
Node references	\D3D::TOP.MHD.EFIT.EFIT01.RESULTS.AEQDSK.WMHD
Signals	BUILD_SIGNAL(data_expression, raw_expression, dimension0_expression, . . .)
Units	BUILD_WITH_UNITS(data_expression, units_expression)
Error bars	BUILD_WITH_ERRORS(data_expression, error_expression)
etc.	

- Note that arguments to above expressions are also expressions
  - Compact storage (e.g., one timebase for many signals)
  - On-the-fly computation (e.g., calibration applied to raw data)

MORE TDI

- TDI has full expression evaluator
  - Mathematical combinations of data
  - Many built in (“intrinsic”) functions: math, matrix manipulation, etc.
- Can write functions in TDI
  - For more complicated retrievals
  - Syntax similar to C
  - Platform independent
  - Stored outside of tree in standard location
- TDI can call routines in shared libraries
  - C or Fortran
  - Use to extend power of TDI (e.g., CPU-intensive functions)
  - Or to wrap shared libraries so can call using standard MDSplus functions
  - This is how PTDATA is accessed through MDSplus
  - Shared libraries obviously not automatically platform independent
- Help on TDI: http://lithos.gat.com/comp/analysis/mdsplus/

GETNCI() — A WORKHORSE FUNCTION

- GETNCI() is a TDI function for getting “node characteristics information”

- Find out lots about a node in a tree
  - Full name of signal
  - Siblings
  - Node type
  - Length (0 if empty)
  - Data itself
  - etc.

- Example: full path and length of EFIT signal \WMHD

  -
    ```
    mdsopen,‘EFIT01’,97979
    ```

  -
    ```
    print,mdsvalue(‘GETNCI(“\\WMHD”,“FULLPATH”)’)
    \EFIT01::TOP.RESULTS.AEQDSK:WMHD
    ```

  -
    ```
    print,mdsvalue(‘GETNCI(“\\WMHD”,“LENGTH”)’)
    562 (bytes)
    ```

- Can use on many nodes at once — wildcards

- Example: get list of EFIT AEQDSK signal names

  -
    ```
    names=mdsvalue(‘GETNCI(“\\TOP.RESULTS.AEQDSK:*”,
                              “NODE_NAME”,“SIGNAL”)’)
    NAMES       STRING = Array[121]
    ```

(5) WHAT YOU CAN DO WITH MDSplus

- Illustrative examples only — “how to” in the next class

- Simple data fetching

mdsconnect,‘atlas’	connect to atlas, the MDSplus server
mdsopen,‘ELECTRONS’,97979	open ELECTRONS subtree, shot 97979
data = mdsvalue(‘\TSTE_CORE’)	use tag for Thomson T_e
d0 = mdsvalue(‘DIM_OF(\TSTE_CORE,0)’)	get 1st dimension of T_e
u0 = mdsvalue(‘UNITS(DIM_OF(\TSTE_CORE,0))’)	get 1st dimension units

EXAMPLE: EFIT DATA

- With MDSplus, can fetch only the EFIT information you need
- Time history of magnetic axis
  - mdsopen,‘EFIT01’,97979
  - r0 = mdsvalue(‘\R0’)
  - t = mdsvalue(‘DIM_OF(\R0)’)
- This contrasts with READA in IDL
  - READA gets *all* the EFIT quantities at a given time
- Each serves a purpose

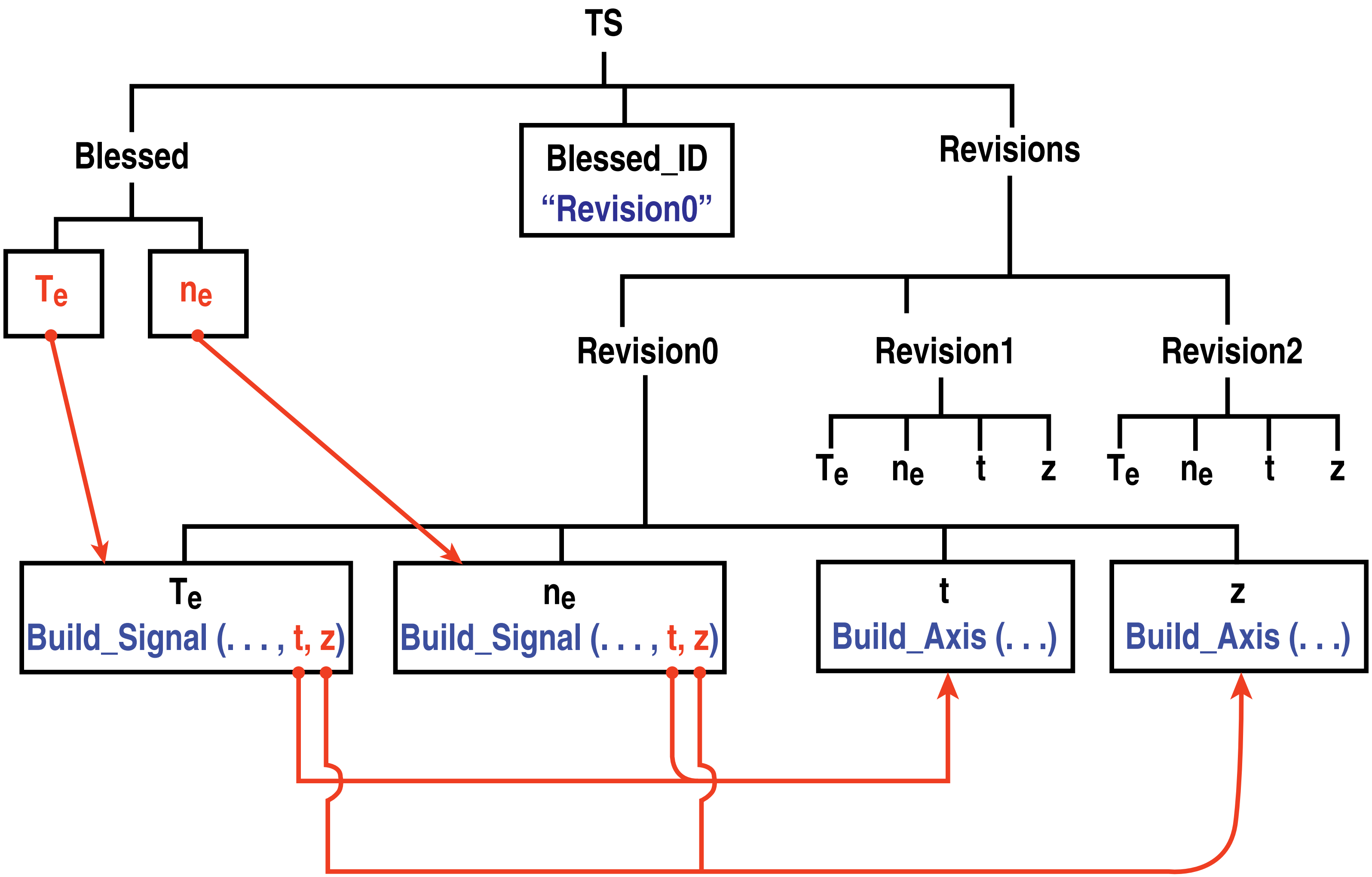
EXAMPLE: FILTERSCOPE CALIBRATIONS

- Filterscope calibration turns PHDnn signal into FSnnX signal
  - nn is channel number
  - X is species (D_α, D_β, He II, C III)

- Calibration is simple: FS00 = a(PHD00) + b
  - a, b stored (calculated) by MDSplus
  - PHD00 obtained from PTDATA

- Changing a and b in MDSplus changes FS00 without having to change raw data

\TSTE_CORE DEMONSTRATES MANY ASPECTS OF TDI



DIII–D NATIONAL FUSION FACILITY SAN DIEGO

GENERAL ATOMICS

252-99 jy

(6) GETTING STARTED

- Use IDL, check the web for help on MDSplus IDL routines
- Online help on MDSplus good as reference material
  - TDI functions
  - Commands to build tree (TCL)
- Get your own data into MDSplus!
  - Come to next class (tree building)
  - Look at other subtrees and branches with traverser
- Ask!