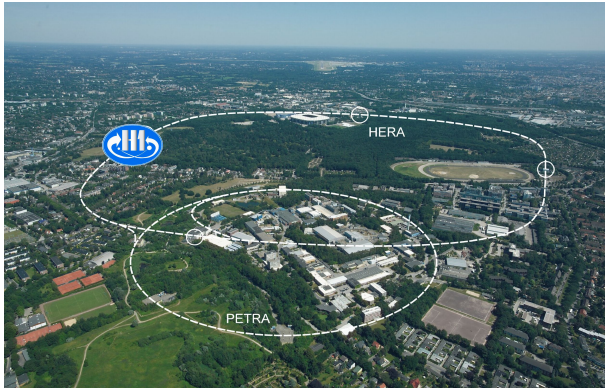
H1 Data Preservation Status and Perspectives

DPHEP Preparatory Meeting, 2nd March 2021

Daniel Britzger (MPP), David South(DESY)

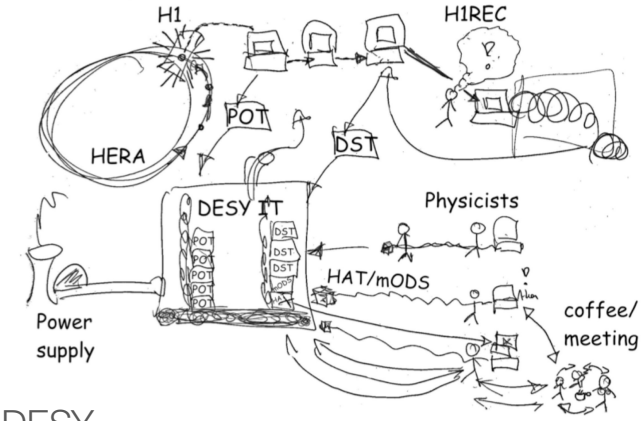
H1 and DPHEP

- H1 was an experiment on the HERA lepton-proton collider at DESY, Hamburg
  - Data taking in the years 1992-2007, 1 billion *ep* collisions
  - Represents a unique dataset in high energy physics
- Significant involvement in the formative years of the DPHEP Study Group and the six workshops 2009-2012
  - Editorial contributions to blueprint publication at CHEP 2012, including the now universally adopted “preservation levels”
- In order to have meaningful and full access to the data, the software must also be considered; H1 adopted a “level 4” preservation model:
  - Preserve not only analysis level data, but also reconstruction and simulation software as well as the basic level data
  - Retain the full flexibility and potential of the experimental data



DPHEP Study Group for Data Preservation and Long Term Analysis in High Energy Physics

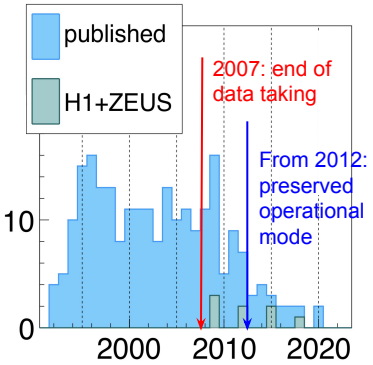
H1 data and software stack



- The H1 data themselves are, as we suspected in 2009, not going to be the problem by today's standards
  - Total RAW data: 75TB; DST: 20TB; Analysis "H1OO": 4TB
  - Considering other special data, full set of MC samples, conservative estimate for a total data volume: 0.5PB
- Final versions of full 1996-2007 data ready by summer 2012
  - Data organised in a dedicated, **dCache** based DPHEP storage at DESY
  - Two copies of RAW data on different tape media. Additional copy of data at MPP
- H1 core software written in **FORTRAN 77**, first developed in 1988, highly modular, **BOS/FPACK**
  - Series of machine independent packages: reconstruction, simulation (based on GEANT3), visualisation..
  - External dependencies: *CERNLIB, GEANT3, GKS, oracle-instant*
- H1OO analysis framework written in **C++98**, and until recently based on **ROOT 5.34**
  - About 50 packages and 600 classes; analysis environment and data formats for analysis
  - External dependencies: *ROOT, fastjet, neurobayes-expert*

Preserved Operational Mode: 2012-2020

- Significant “level 1” effort made on documentation
- New collaboration model: H1 Physics Board
- H1 physics publications continued
  - 28% of 233 papers after data taking ended
  - 66 after 2007, including 18 after 2012



- Level 4 preservation model includes recompilation of software and migration to newer OS
  - Main OS used in 2012: 32-bit Scientific Linux (DESY) 5, based on **RHEL5** (EoL: March 2017)
- Migration to 64-bit **SLD5** from 2012, careful work requiring detailed validation (*sp-system*, Vitaliy)
  - Successful move to 64-bit **SLD6** (EoL: Nov 2020) and later **CentOS7** made possible due to this effort
- Software remained static during this period
  - H1OO effectively frozen at **ROOT 5.34**
  - External dependencies reliant on H1 action (and experts) for updates

Component	Responsible	Maintained packages	Discontinued packages
H1 software	H1	H1 core software, H1OO	–
OS dependencies (continuous updates)	DESY–IT	Oracle, dCache, web–services, compilers, GNU utilities, gmake, system libraries	CVS
External dependencies (selected fixed releases)	H1	fastjet, neurobayes–expert, MC generators	CERNLIB, GKS, GEANT3, ROOT5, LHAPDF5, MC generators

Modernisation of the H1 software

- 2020: Successful migration to Centos7, but a few shortcomings now evident in the H1 software
  - The programming languages (C++98) and standards are unattractive for new (young) people to learn
  - Outdated dependencies, such as ROOT 5, complicate the usage of modern data analysis techniques
  - New dependencies may be incompatible, different compilers standards or generator formats
  - Modern tools cannot or have not in general been introduced

- Restructuring the software: Introduction of dependence on the LCG package repository
  - Previously no externally maintained package repository: packages provided manually
  - Two effects: reduction of H1 maintenance and bring in newer versions of existing software dependencies, new compilers (huge jump in GNU Compiler collection 4.85 to 9.2.0)

Component	Responsible	Maintained packages	Discontinued packages
H1 software	H1	H1 core software, H1OO	–
OS dependencies (continuous updates)	DESY–IT	Oracle, dCache, web–services, GNU utilities, git, gmake, system libraries	–
External dependencies (selected fixed releases)	H1	–	CERNLIB, GKS, GEANT3 (selected) MC generators
External dependencies (selected regular updates)	LCG	LHADPF6, ROOT6, compilers, fastjet, neurobayes–expert, MC generators, (and as back up option: Oracle, dCache, git)	–

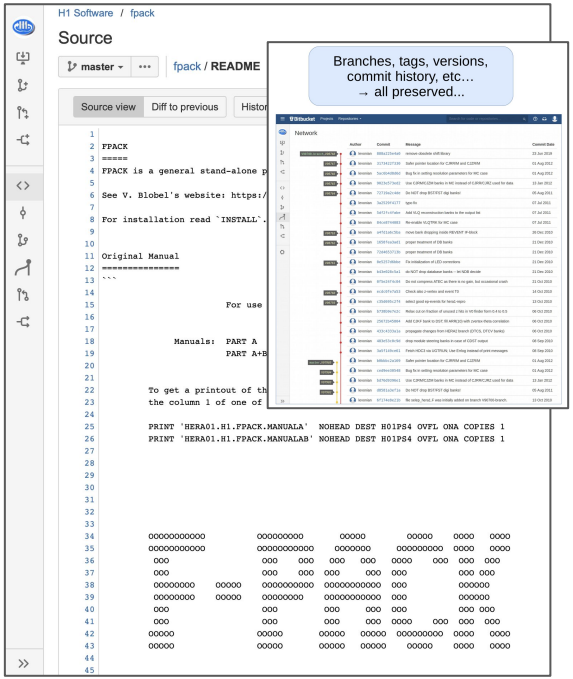
- Core (f77) software re-compiled without problems

Modernisation of the H1 software (continued)

- H1OO analysis framework updated to **ROOT 6** and **C++17**; **CLING** replaces **CINT**
  - Original production of data and MC files remain compatible
  - New **C++** standard allowed s/w improvements, for example range-based for loops in **H1Arrays**
  - Another benefit of **ROOT 6** is **PyROOT**: Fully pythonic analysis of H1 data now possible, incl. interactive
- Complete release of all H1 software on **/afs** and **/nfs** at DESY, soon to be distributed on **/cvmfs**
  - H1 core packages were previously bound to the DESY–IT infrastructure and could not be relocated
  - H1 s/w now runs without problems e.g on **CentOS7 lxplus** at CERN
- All code repositories migrated to **git**
  - H1 used **CMZ** and **CVS**, did not get to **SVN**
- Build instructions for entire H1 s/w stack available
  - Less reliance on former knowledge of structure or historic development of the H1 software
- Bonus: **SLD5**, **SLD6** container builds using **Singularity** as retrospective "level 3" preservation

Component	Responsible	Maintained packages	Discontinued packages
H1 software	H1	H1 core software, H1OO	–
OS dependencies (continuous updates)	DESY–IT	Oracle, dCache, web–services, GNU utilities, git, gmake, system libraries	–
External dependencies (selected fixed releases)	H1	–	CERNLIB, GKS, GEANT3 (selected) MC generators
External dependencies (selected regular updates)	LCG	LHADPF6, ROOT6, compilers, fastjet, neurobayes–expert, MC generators, (and as back up option: Oracle, dCache, git)	–

Summary



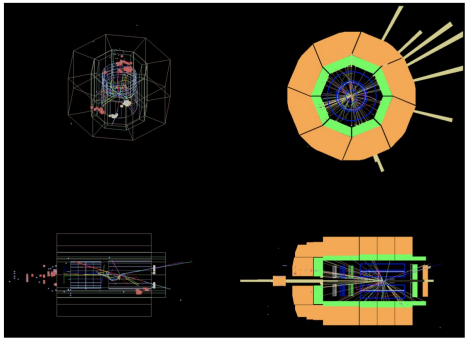
Branches, tags, versions, commit history, etc… → all preserved...


```
# A minimum example to read H1 MOD files
# with python and write the results into
# a ROOT.TH1D histogram and plot it
def minimum_example():
    tree = H1.H1Tree.Instance()
    tree.AddFile("/pnfs/desy.de/dphep/online/h1/mc2/oo-4.0/djang
    tree.AddFile("/pnfs/desy.de/dphep/online/h1/mc2/oo-4.0/djang

    # direct access to MOD quantities using H1 Pointers
    Q2_ptr  = H1.H1FloatPtr("Q2e")          # virtuality
    Wgt_ptr = H1.H1FloatPtr("Weight1")      # Event weight
    PartCands_ptr = H1.H1PartCandArrayPtr()  # Array pointer to

    # book some ROOT-histograms
    hist_Q2 = ROOT.TH1F("Q2", "Q^{2} Detector level;Q^{2} [GeV^{
    hist_Pt = ROOT.TH1F("pt", "P_{T} Particles;P_{T} of all Part
    hist_VtxZ = ROOT.TH1F("VtxZ", "Vertex z-position; Vertex z-position [cm];events",  40, -55, 55);
    hist_Empz = ROOT.TH1F("Empz", "Empz; E - P_{Z} of FS [GeV];events",  40, -0, 100);
    # --- H1Calculator, if requested
    gH1Calc = H1.H1Calculator.Instance()

    # --- event loop
    events = 0;
    while tree.Next() and events < 10000:

        # acces Q2, Wgt, etc...
        Q2  = Q2_ptr[0]
        Wgt = Wgt_ptr[0]
        # fill histogram
        hist_Q2.Fill(Q2,Wgt);

        # loop over H1 particle candidates
        for  part  in PartCands_ptr :
            hist_Pt.Fill(part.GetPt(),Wgt);
```

Python interface

- Through ROOT::TNamed all H1oo classes are automatically accessible with python
- Python-like for-loops….
- H1Calculator….
- Access to MODs, etc…

'H1Red' for simulated Pythia8.3 event



Python interactive analysis at LXPLUS

```
[lxplus7106] ~/test $ python
Python 2.7.16 (default, Jul 12 2019, 12:27:03)
[GCC 9.1.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import ROOT
>>> execfile( "LoadH1.py" )
>>> tree = ROOT.H1Tree.Instance()
>>> tree.AddFile("hat.4.0.6.DJANGOH14.NC.EPLUSP.0607.RAD.Q2GT60.CTEQ6L.A.7091.S39200.R97800.ftt.DST.0000.root")
True
>>> tree.AddFile("mods.4.0.6.DJANGOH14.NC.EPLUSP.0607.RAD.Q2GT60.CTEQ6L.A.7091.S39200.R97800.ftt.DST.0000.root")
True
>>> Q2_ptr  = ROOT.H1FloatPtr("Q2e")
>>> PartCands_ptr = ROOT.H1PartCandArrayPtr()
>>> tree.Open()
Consistency checks for list H1TreeEventList

====== H1SteerManager: Using default values for class 'H1SteerOdsEvent'  ======
  H1SteerOdsEvent bank names:
HEAD CRME FRME TOFT TOFS CRPE DBPC DMIS DELE DBFC HRDE DTNY DTRY TLV2 DFPS JDTX
JDPX GHD  GKI  GTR  GVX  GEVC STR  SVX  SIPA TL23 NSPT FNHT FNCR DPCC DCLH BRSE
BRUE DTAG DRP1 DRP2 DRP3 DL5W DTZS CSKH CSHY DFTS Y4TT YECL VTCL VRCE YRCL YRL5
DISM JMFI JMXY HDOO TEL1 HEAR FNCE FNCL FRSE FRXT DRME BPCR JTGT JTST CRM2 TLV3
TT10 TT20 TT30 TT1T TT2T TT2K TTEV BJKR
H1BankEvent uses all 3/4 letter branches in SetBranchAddresses
H1SteerOdsEvent: Call to cstrec disabled
====== H1SteerManager: End of defaults for class 'H1SteerOdsEvent'  ==========
```

- H1 has unique data and continues to produce physics publications, long after data taking ended
- A recent software modernisation program has been performed to allow this to continue using modern analysis tools, recent programming languages and on state-of-the-art platforms