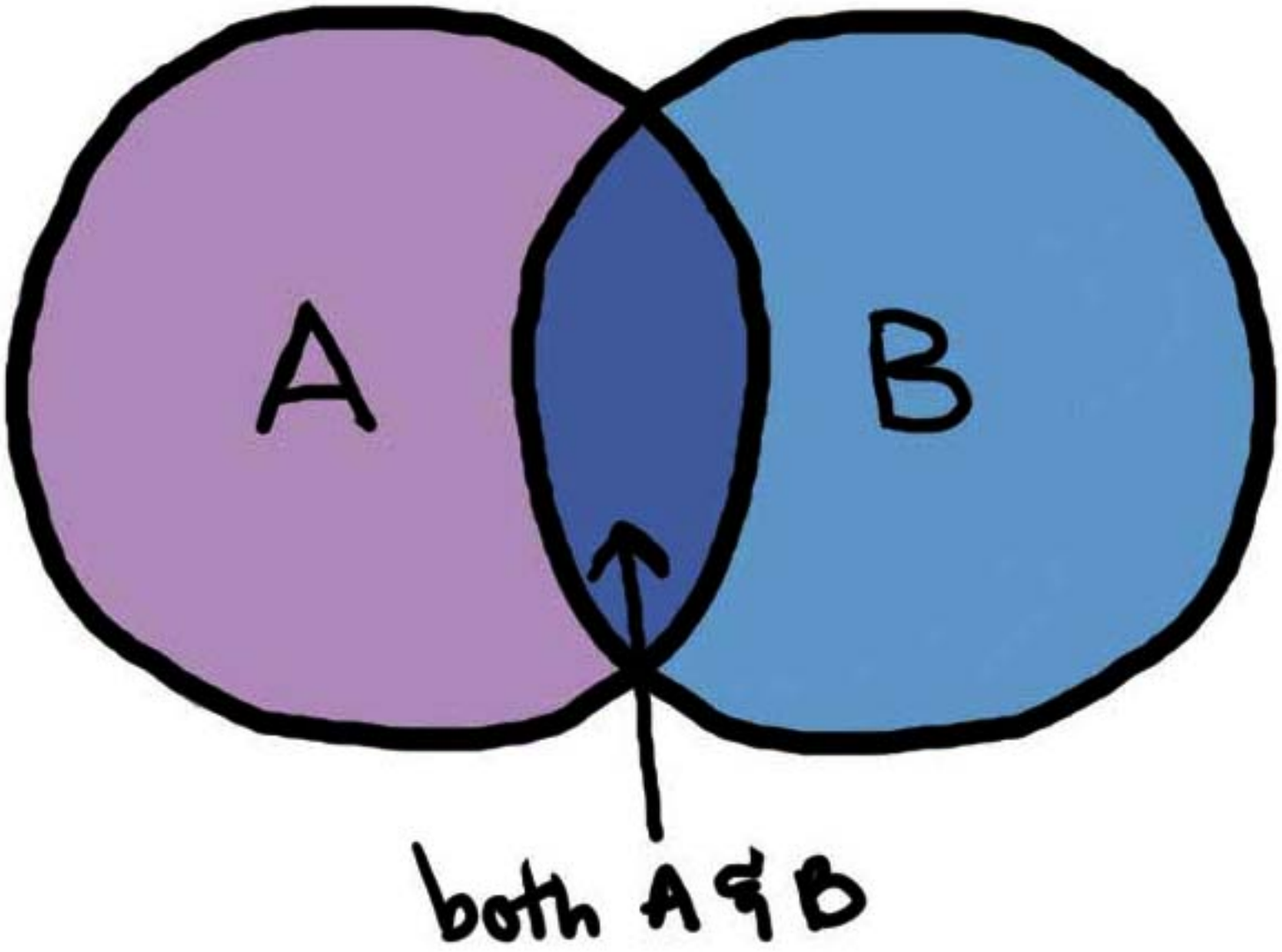


Recent HEP Experience with Common Software

Peter Elmer - Princeton University

HEP Software Collaboration

- In this talk I am primarily interested in:

- Understanding better what our common software projects look like, and how they get adopted by others and evolve.

- Understanding better how and when projects succeed in "leaving the ghetto" of a particular group, experiment or the field is potentially interesting.

Application software code bases

- The CMS software (CMSSW) release consists of ~3.6M SLOC of C++ (SLOC = source lines of code)

- Atlas has reported even larger code base (~5M SLOC?)

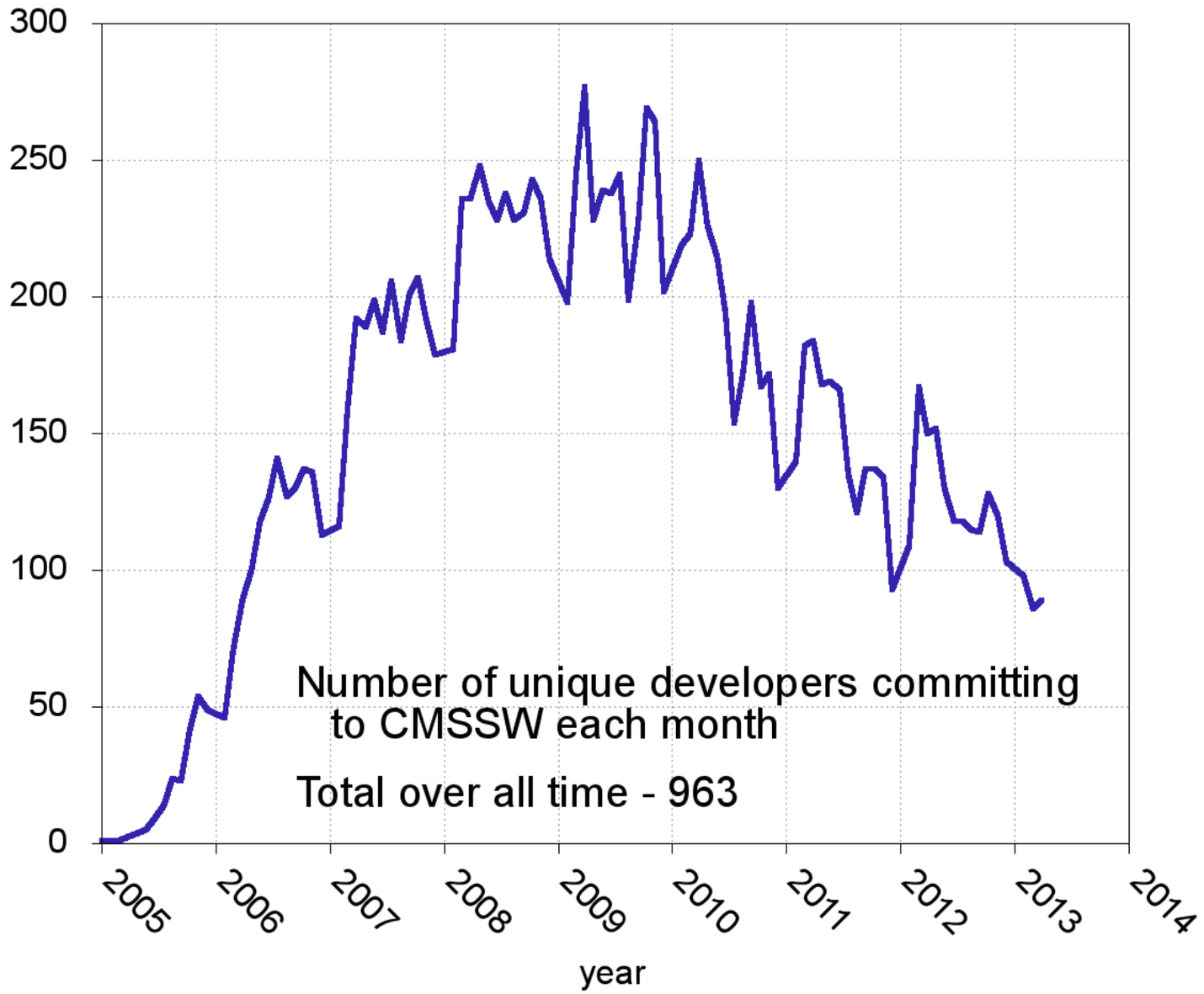
- BaBar had ~3M SLOC in 2007 in the software release

- CDF (Run 2) had ~2.7M SLOC in 2007 in their software release

- These numbers are for code in releases, analysis code is not counted here

- ROOT is ~1.7 MSLOC, Geant4 is ~1.2 MSLOC, all generators used by CMS sum to ~1.4M SLOC, gcc is 7M SLOC

People committing code to CMSSW per month



See also "The Life Cycle of HEP Offline Software", P.Elmer, L. Sexton-Kennedy, C.Jones, CHEP 2007

Common Software Packages

- ROOT and Geant4 are the obvious common packages, which will in one form or another be part of any HEP collaboration.

- An important question is: what else could be?

- Another important question is: what else is common today and how/why did the packages become "common code"?

- How do we develop more common software?

Physics Generators

- alpgen (187k), cascade (35.9k) charybdis (2.9k) jimmy (5.4k) LHAPDF (79.6k) Rivet (77.9k) Pythia6 (78k) Pythia8 (75k) Tauola (21.8k) Tauola++ (58.4k) ThePEG (69k) toprex (33k) Sherpa (297.5k) MCDB (1.2k) libHepML (2k) HepMC (9.3k) HepPDT (13.1k) Herwig (120k) Herwig++ (189.9k) Photos (69.k) Professor (14.5k) EvtGenLHC (38.7k)

- Total of about 1.4 MSLOC, approx. half C++, half Fortran, clearly HEP-specific, not of interest to others.

- Starting to become more computationally intensive. (And incentives for theorists are different...)

System and Software Engineering

- General open source: boost bz2lib ccache curl cppunit distcc DMTCP doxygen expat jemalloc gcc gccxml gdb git gmake llvm libjpg libpng libtiff libungif libuuid libxml2 opengl openldap openssl oracle SLOCCount TBB Google-Perftools Valgrind xerces-c xz zlib libSigC++ python python-ldap protobuf ipython sqlite gdbm lcov cvs2git pacparser pcre rpm apt glimpse

- Developed in HEP: Castor classlib dcap dpm xrootd lgProf

Data Analysis, Math and Graphics

- General open source: gnuplot matplotlib numpy scipy GSL meschach lapack fftw3 CGAL graphviz Qt PyQt SIP

- Developed in HEP: CLHEP VDT ROOT (RooFit RooStats) PyMinuit2

- Several potentially interesting tools for the rest of the world.

HEP simulation and misc

- CORAL FastJet fftjet frontier-client Geant4 KtJet Hector TKOnlineSW

Inventory of HEP Common Packages

- I looked through the CMS software externals, the LCG AA distribution and added a few more examples picked randomly from outside that

- I will mostly ignore generators here (many with theorist collaborations) and focus on code developed by or for the experiments

- I will ignore (mostly) "Grid Tools" in favor of application software

See accompanying draft note "Case Studies of Scientific Software Collaborations in High Energy Physics and Beyond"

Keywords/Questions

- Champions - How do we create them?

- Collaborations and contributors - common aspect of how things grew (perhaps more important than acquiring new user communities at the beginning)

- User driven/experiment driven - known to be critical

- Facilitating adoption and distribution

Keywords/Questions

- Longevity/Incentives
- R&D and Novel Ideas - Unique and/or new functionalities, how can we encourage their creation as common software and or migration into common software
- Standards and the rest of the world

Glast/Fermi Comments

- Limited effort for in-house software development
- "Beg, Borrow and Steal"
- Gaudi, ROOT, CLHEP, Xrootd, CFITSIO and related tools
- Adopted CMT, but later had to drop it when it seemed support had vanished
- Gaudi had more dependencies than they needed, "separation of xrootd from ROOT a godsend"
- Grid not ready for them, they

Other projects - HepForge

- A lightweight development environment for HEP software
- Mailing lists, shell accounts, archiving of releases and low maintenance web sites
- Perhaps the rest of the Web has caught up in some aspects (e.g. github), but a logical next step might be hosted build and runtime tests (with both global and project specific test configurations and versions?)

An aside: Scientific Software Production, Cyberinfrastructure, Ecosystems

- While my aims are actually very pragmatic, while looking into this I see that "Scientific Software Production" is actually an academic area of research. See, for example, James Howison (http://howison.name/) and follow references from there. For example:

- http://howison.name/pubs/IncentivesAndIntegration-p459-howison.pdf

- See also (another aside): "When Authorship Isn't Enough: Lessons from CERN on the Implications of Formal and Informal Credit Attribution Mechanisms in Collaborative Research ", J. Birnholtz

- http://quod.lib.umich.edu/j/jep/3336451.0011.105?rgn=main;view=fulltext

Conclusions

- Common packages succeed for a number of reasons, but the themes are well known

- In addition to specifying a governance, perhaps we should document precisely how the collaboration will foster the creation of software champions, how we improve incentives (beyond top-down control of resources) and ultimately how we foster an ecosystem of software, not a single toolkit