Report from

HEP Software Collaboration meeting

http://indico.cern.ch/event/297652

chaired by Ian Bird (CERN), John Harvey (CERN)

from Thursday, 3 April 2014 at **08:00** to Friday, 4 April 2014 at **18:00** (Europe/Zurich)
at **CERN (503-1-001 - Council Chamber)**

Pere Mato/CERN

GDB Meeting, 9 April 2014

Disclaimer

- This is a personal report
  - not the official summary of the meeting
  - not the CERN report
  - ...

Agenda

- Thursday, 3 April 2014
  - 09:00 Welcome from CERN Director of Research (Sergio Bertolucci)
  - 09:15 Vision behind the HEP Software Collaboration Initiative initiative (Richard Mount)
  - 10:00 Recent HEP experience developing common software packages (Peter Elmer)
  - 11:15 Model for organising and supporting development activities (Pere Mato)
  - 14:00 Governance Model (Panagiotis Spentzouris)
  - 15:30 Funding Opportunities for new software proposals
    - NSF Funding opportunities (Peter Elmer)
    - DOE Funding Opportunities (Panagiotis Spentzouris)
    - EU -Horizon2020 funding opportunities (Mauro Morandin)
    - EU - T0 proposal Funding Opportunities (Giovanni Lamanna)
    - LPaSo project (David Rousseau)
    - Japan Funding opportunities (Takashi Sasaki)
- Friday, 4 April 2014
  - 09:00 Discussion Session (Federico Carminati)
    - Software Ventures (Jeffrey Tseng)
    - Comments and Suggestions (Rene Brun)
    - Comments from Budapest (Gyoergy Vesztergombi)
  - 11:00 Wrap-up and conclusions (Federico Carminati)

Motivations

- Much of our HEP software is now old (> 20 years)
  - it needs to be adapted to more modern standards
- Paradigm-shift resulting from the evolution of CPU architectures
  - our code has to be re-engineered to make use of the full capabilities
- Make use of all resources available to our community
  - HPC facilities, commercial clouds, volunteer resources
- Must attract people with the required advanced skills and experience
- Ensure interoperability with software developed by other scientific communities
- Opportunity for sharing software between different experimental programs

Stakeholders

Software: Who are the Collaborators/Stakeholders?

SLAC

Regions?

- Asia, Europe, North America, …

Funding Agencies?

- Europe-National, EU, US-DOE, US-NSF, …, NASA, …

Institutions?

- Universities, CERN, Fermilab, KEK, …, ESA, …, hp.com, intel.com…

Experiments?

- ATLAS, CMS, …, LBNE, …, IceCube, …

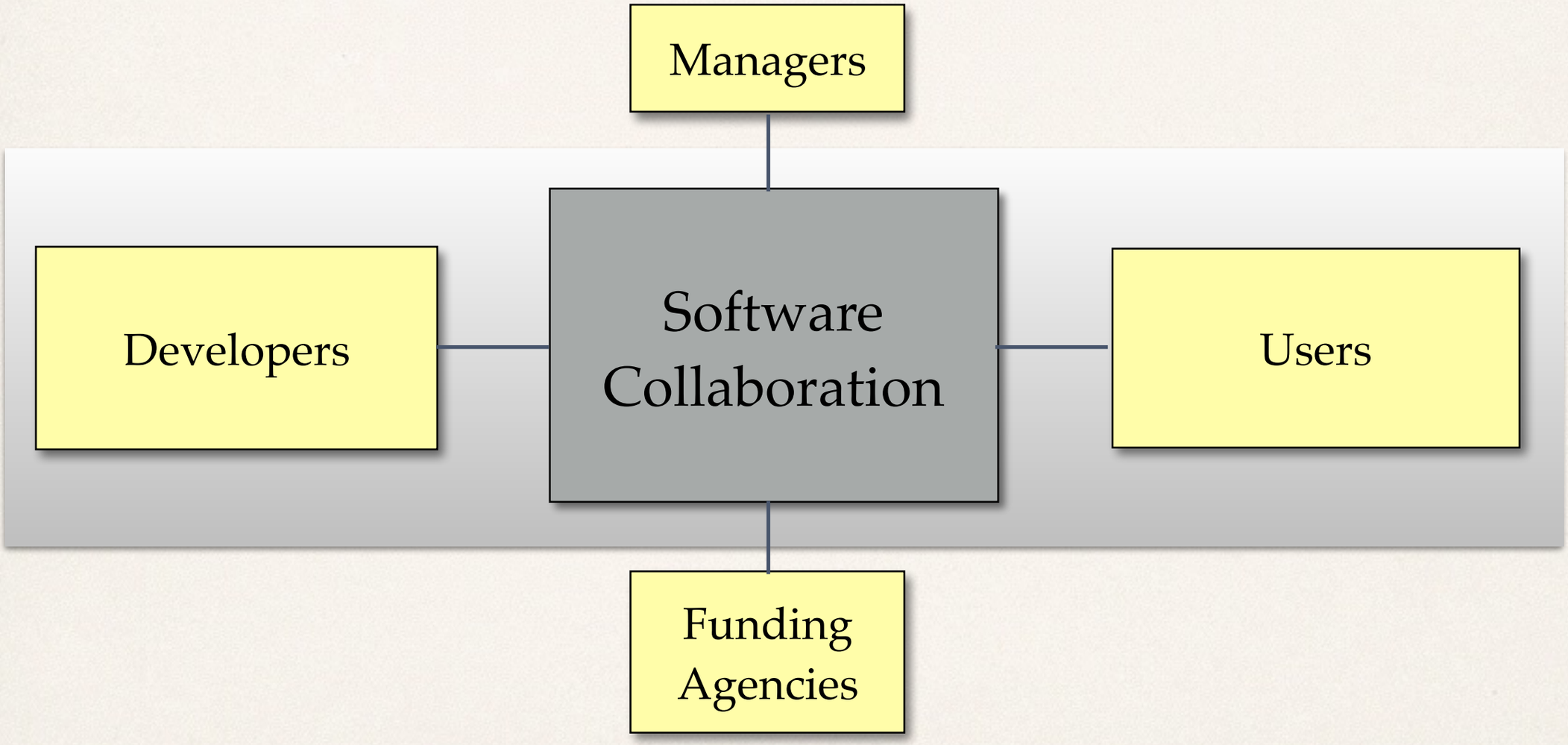
Experimental programs?

- X-ray laser science at DESY, SLAC …

People?

- Wizards, …, Workers

Main Stakeholders



Software Challenges

HEP Software Challenges – LHC Focus

SLAC

- LHC has an increasingly data-intensive future
- Technology evolution will not meet our needs (at least not without a lot of work)
- We could 'survive' by tightening our focus and our triggers → ensure that we ignore the truly unexpected
- Much better to evolve our software process to exploit the complex shifting landscape of computing in the next 20 years (massive parallelism; changing cost balance between computation, memory, storage and networks)

HEP Software Challenges – Non-LHC

SLAC

Smaller (but not necessarily small) experiments look hungrily at the rich LHC software and distributed computing environment

- They get little benefit from the LHC software successes
- LHC experiments are [almost] not funded to help smaller experiments
- Funding agencies search for a way to fund and manage cross-cutting software development and support
- We have to show the way (in our own self interest).

The Promise

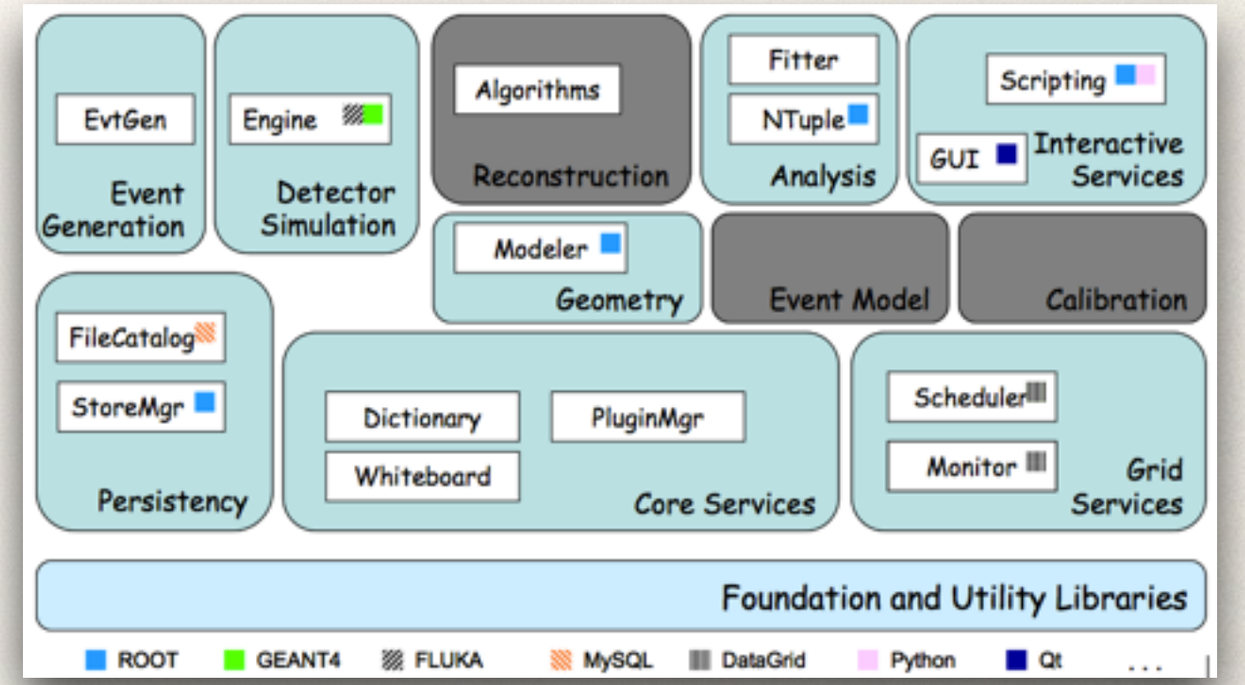
HEP Software Collaboration – the Promise

SLAC

- Recognition of HEP as a collaborative software powerhouse
- Non-HEP agencies eager to support software projects that have HEP involvement
- Affordable high-quality software for smaller-than-LHC experiments
- Uniquely capable software for our data-intensive future.

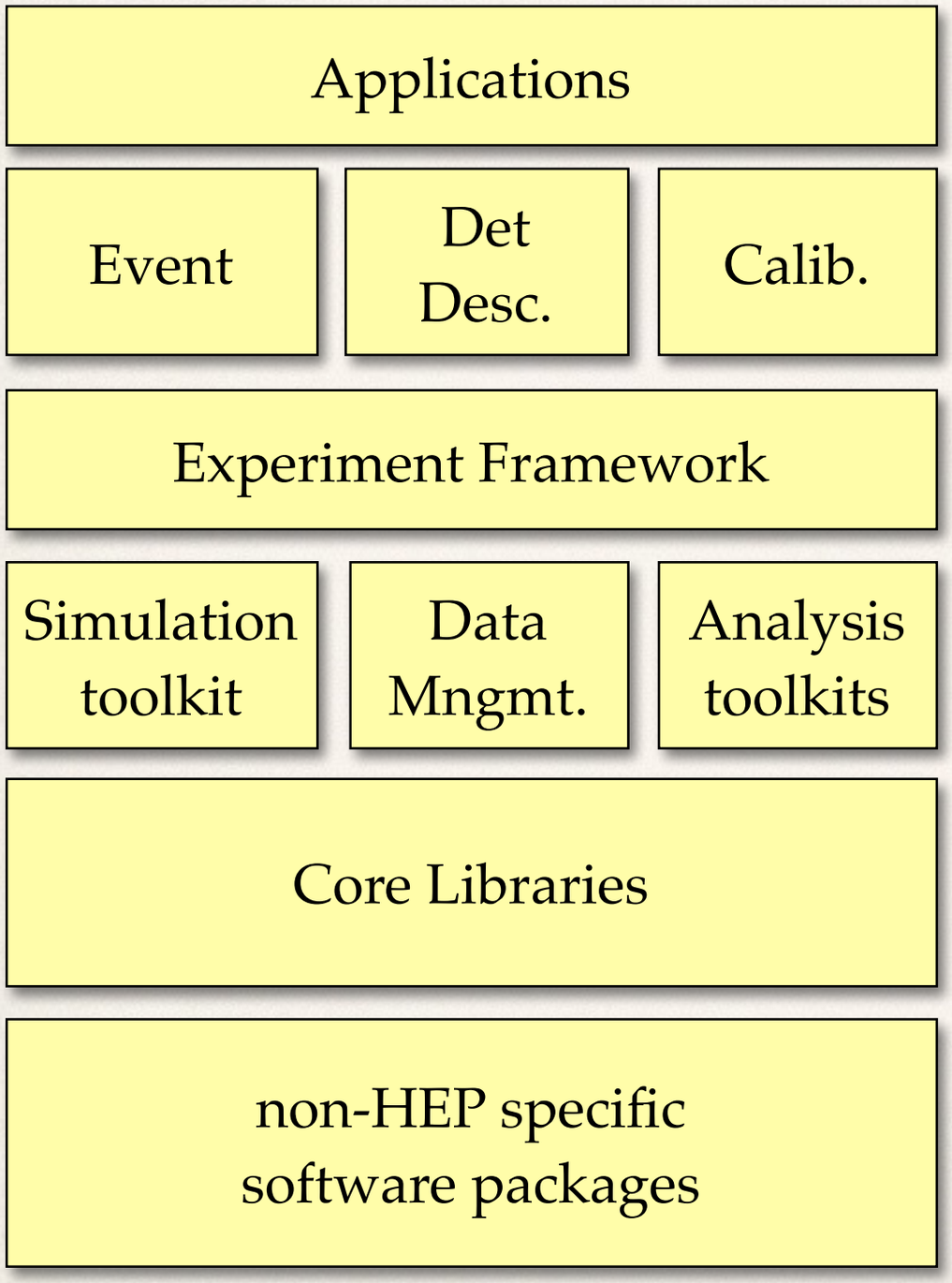
Central Goal: Software Packages

- The goal is to **develop** software components [packages] in collaboration and make them available to the HEP [scientific] community **to use** them
  - Facilitate package development
  - Facilitate package usage
- We expect different sort of packages of different **levels** (foundation, core, generic, specialized) in different software **domains** (simulation, statistics, math, graphics, etc.)
- Nothing very new so far (same ideas 12 years ago)
  - A good moment to re-think about modularity



Domain decomposition from *LHC Computing Grid Project Architecture Blueprint RTAG*, 2002

Typical HEP Software Stack



Applications are built on top of frameworks and implementing the required algorithms (e.g. simulation, reconstruction, analysis, trigger)

Every experiment has a framework for basic services and various specialized frameworks: event model, detector description, visualization, persistency, interactivity, calibration, etc.

Specialized domains that are common among the experiments

Core libraries and services that are widely used in HEP and provide basic functionality

General purpose non-HEP libraries

Common Software Packages

- Pete Elmer inventory of packages, how they were conceived and how they became ‘common’

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

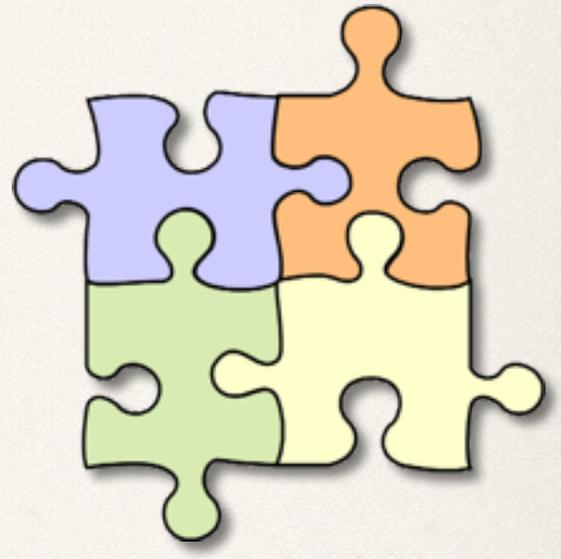
- alpgen (187k), cascade (35.9k) charybdis (2.9k) jimmy (5.4k) LHAPDF (79.6k) Rivet (77.9k) Pythia6 (78k) Pythia8 (75k) Tauola (21.8k) Tauola++ (58.4k) ThePEG (69k) toprex (33k) Sherpa (297.5k) MCDB (1.2k) libHepML (2k) HepMC (9.3k) HepPDT (13.1k) Herwig (120k) Herwig++ (189.9k) Photos (69.k) Professor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

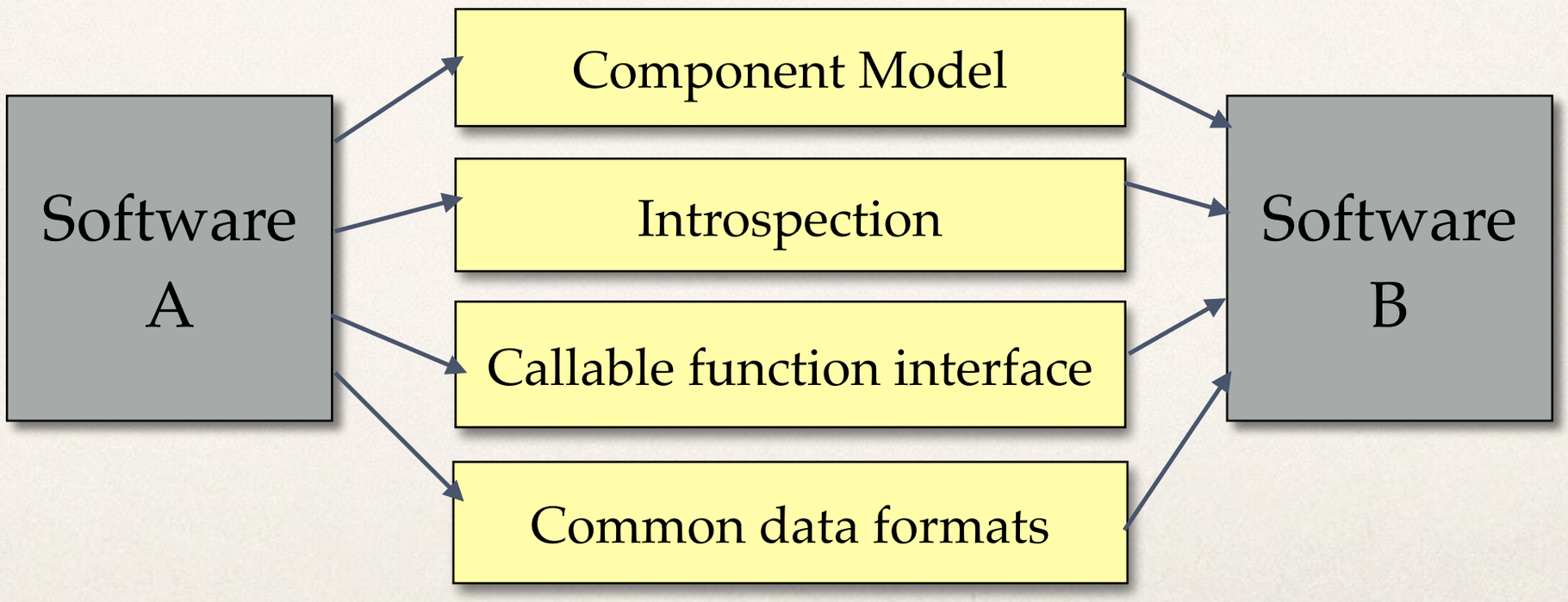
AIDA-WP2 Example

- The common software work package of EU AIDA project is delivering a set of generic software toolkits for geometry and reconstruction (9 partners)
  - Some of them developed in the context of one experiment but abstracted and packaged in experiment-independent manner
- Some of the packages are:
  - USolids - Unified 3D shapes library
  - DD4hep - Toolkit for describing detectors
  - aidaTT - Tracking toolkit
  - PandoraPFA - particle flow algorithms
  - tkLayout - track trigger simulation
  - Bach - telescope reconstruction and alignment
  - Arbor - Topological clustering

Interoperability

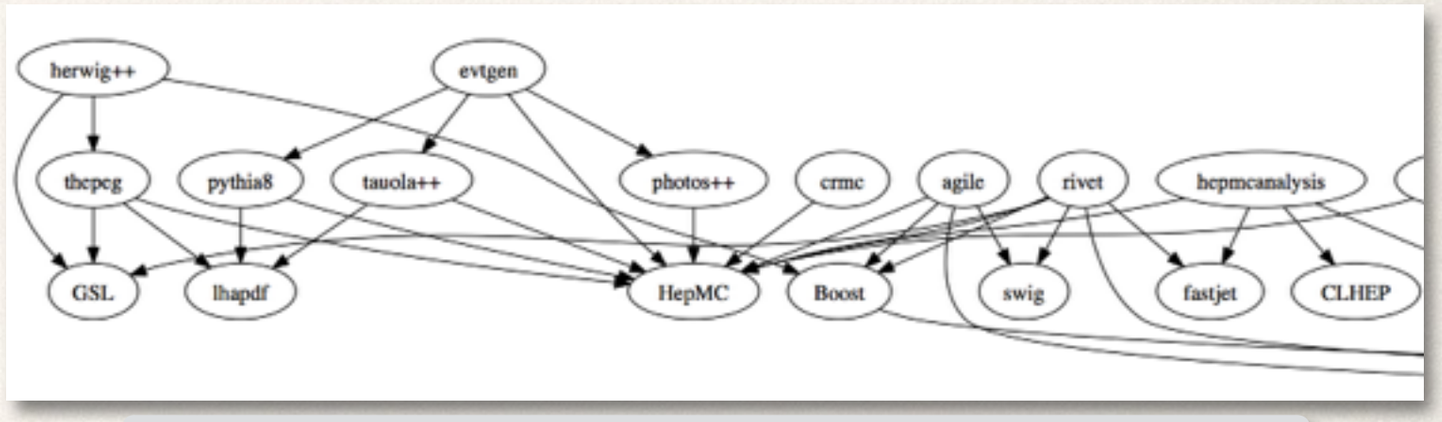
- Capability of different software components to exchange data via a common set of exchange formats or interfaces
- There are several levels of interoperability
  - Level 0 - Common Data Formats
  - Level 1 - Callable Interface
  - Level 2 - Introspection Capabilities (generic callable interface)
  - Level 3 - Component Model (common framework)





Package Dependencies

- Very few packages are truly standalone (not having any dependency)
  - Very often packages depend of other packages
- Package dependencies are difficult to manage
  - Complicates the configuration, the build process, the distribution and the deployment
- Avoiding dependencies is not a good solution in general
  - Adds code duplication
  - Reduces code re-use
- **Managing dependencies is essential**
  - 'Standards' and tools are required



Fragment of the dependency graph of some MC generator packages

Packaging and Distribution

- Tar-balls are great, but difficult (up to impossible) to upgrade, uninstall, and simply keep track of them
- We will need to develop or adopt some easy-to-use system for compiling, installing, and upgrading HEP software
  - Examples: MacPorts, Fink, APT, etc.
  - Multi-platform (Unix, Mac, Win) support is required
- Perhaps even better to distribute the software using CernVM-FS
  - All HEP software will be 'virtually' installed automatically

Key to the success of the HEP Software Collaboration

Project Independency

- Each development team should **keep its autonomy**
  - The Collaboration should not enforce any particular software process, project management or methodology
  - Each team may use its own repository, bug tracker, web project site, user forum, etc.
  - But the Collaboration may provide them if needed
- 'Ownership' of the package resides with its developers
  - A clear way of recognition and proper credits
  - At the same time the developers need to ensure support and maintenance

Added Value for Developers

- Channel for advertising quality software to a large scientific community
  - Make good packages known to potential users
- Ensuring the coherency with the packages from other colleagues
  - This facilitate their integration into systems
- Providing integration builds and integration tests
  - Validation on a extended set of platforms (architectures, OS and compilers)
- Providing distribution repositories to the community
  - Easy to locate, select and install the required package

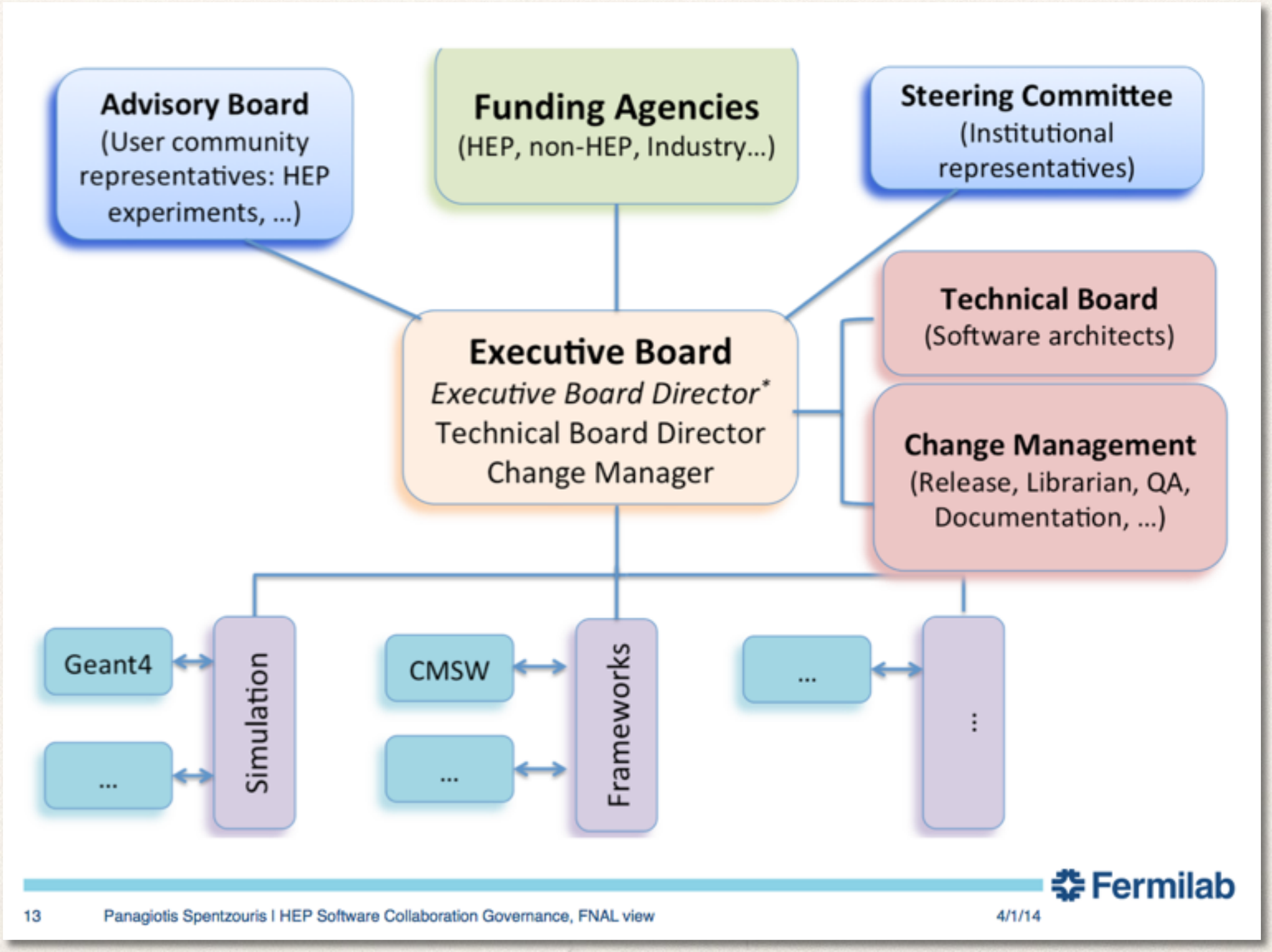
Added Value for Users

- All HEP packages in a single repository with standard documentation, build procedures, file structures, coherent interfaces, etc.
  - Even better, all pre-installed in the CernVM-FS system!
- Users will no need to re-code functionality that is available in repository
  - Assuming that the package is working, high quality, performant and well integrated to the rest of the software packages
- No fear to add the needed extra dependencies because building and installation will be complete automatic (transparent)
- Easy and centralized channel for providing feedback and bug reports

Summary of Development Model

- Expect to populate a repository with a variety of software packages
  - Different levels (foundation, core, generic, specialized) and in different software domains
- These components will need to interoperate with other components to provide the required functionality
  - Definition of data formats, interfaces, API, component models required
- Standard software process should not be enforced
  - Encourage a common set of common practices and tools to facilitate integration and testing
- Each development team should keep its autonomy and way of working
  - Code repository, bug tracker, forum, web site, etc.
- Lowering the barriers for users to use any package in the repository
  - Easy-to-use installation system

Governance Model



Funding Opportunities

- Potential of enhancing the opportunities in the EU and US

The HEP software collaboration

- If successful, the HEP software collaboration will enhance the probability of collaborating projects to secure R&D funding from different offices (direct or partnership).
- The collaboration should be able to produce appealing proposals to relevant programs that focus on infrastructure and collaboratories.

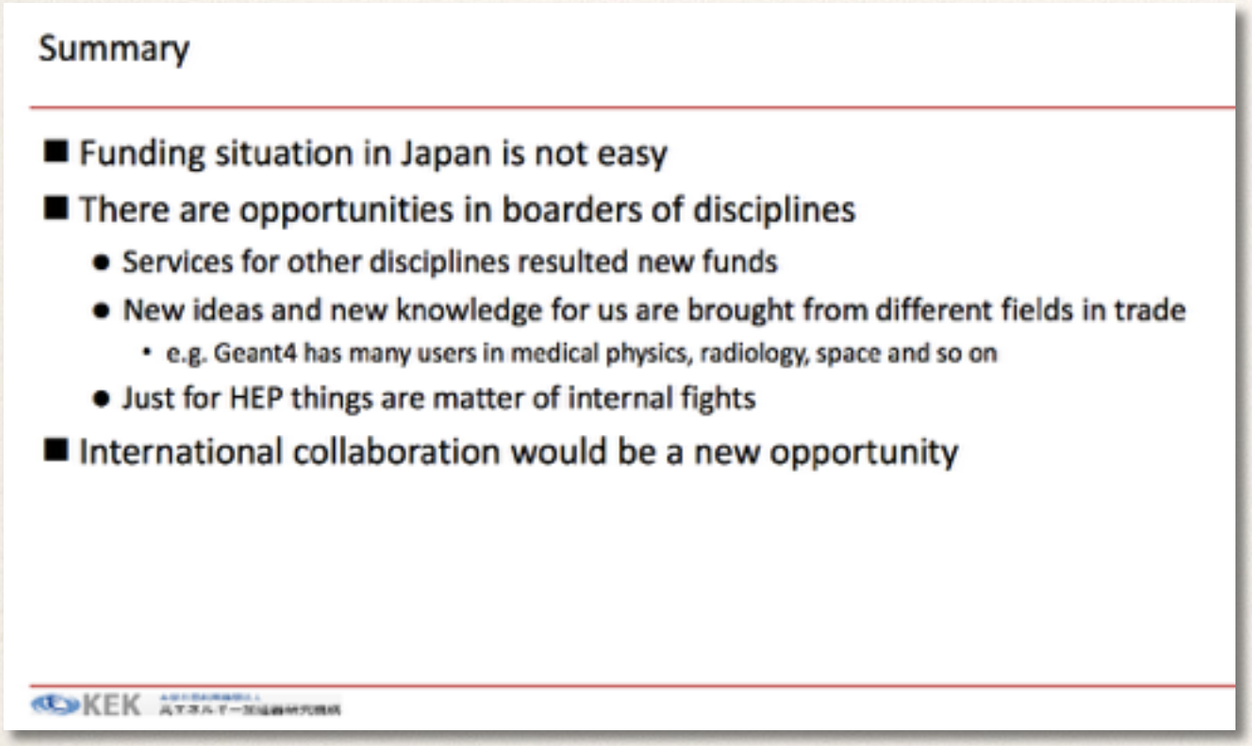
Conclusions

INFN Istituto Nazionale di Fisica Nucleare Sezione di Padova

- Securing support for our software evolution needs from the H2020 program is not going to be trivial, but it may represent anyway an **important ingredient** for the sustainability of the process; it requires:
  - careful analysis of the opportunities
  - evaluation of the trade off between possible benefits and the efforts needed to submit and to manage projects
- the Software Collaboration can help in this respect by:
  - **elucidate what are the priorities** for the common software our community relies on
  - **enhance possible synergies** between the opportunities of the various programs
  - **catalyze and harmonize initiatives** aimed at submitting proposal both at the National and the International level
  - **provide advice** based on past experience
  - **provide feedback** to the EC by presenting a coherent vision of the needs and the plans of our community

Funding Opportunities

- Similarly for Japan and National funding such as France


Summary

- Funding situation in Japan is not easy
- There are opportunities in boarders of disciplines
  - Services for other disciplines resulted new funds
  - New ideas and new knowledge for us are brought from different fields in trade
    - e.g. Geant4 has many users in medical physics, radiology, space and so on
  - Just for HEP things are matter of internal fights
- International collaboration would be a new opportunity

KEK


LPaSo

LHC Parallel Software

french project

David Rousseau
LAL-Orsay
on behalf of the LPaSo partners
3rd April 2014

HEP software collaboration kick-off meeting 1

Conclusions

- Some criticism to the development model
  - What's the role of LHC experiments? How this will affect them?
- Building the Collaboration bottom-up
  - The governance model was a bit scaring
  - What is the added value of the collaboration for developers?
- 'Software Foundation' was somehow a preferred term
  - Should learn form existing similar foundations
- Invitation to the groups to write a 'white paper' with the collaboration goals and organization
  - Short (5 pages), in 4 weeks, participants as initial mailing list