Ian Bird

Grid Deployment Board

CERN, 12th February 2014

HEP Software Collaboration

Thanks to John Harvey, Pere Mato, Federico Carminati

Some background

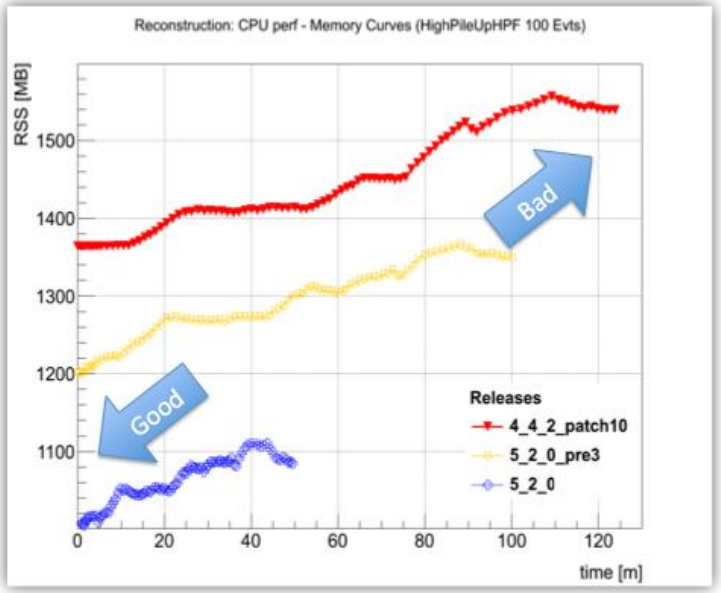
Main Software Requirements

(as understood in 2004)

- The software being developed by the LHC experiments must cope with the unprecedented conditions and challenges that characterizes these experiments (trigger rate, data volumes, etc.)
  - The **software should not become the limiting factor** for the trigger, detector performance and physics reach for these experiments
- In spite of its complexity it should be easy-to-use
  - Each one of the ~ 4000 LHC physicists (including people from remote/isolated countries, physicists who have built the detectors, software-old-fashioned senior physicists) **should be able to run the software, modify part of it** (reconstruction, ...), **analyze the data, extract physics results**
- Users demand simplicity (i.e. hiding complexity) and stability
- **Performance** and **Efficiency** came much later

Performance is now a Limiting Factor

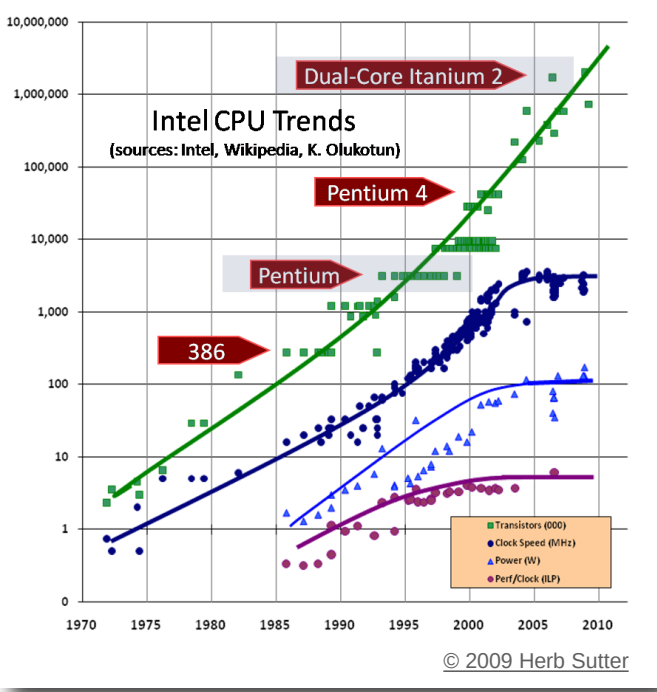
- The LHC software (trigger, simulation, reconstruction, analysis) was ready at the start of the LHC operation being **able to deliver the exceptional physics results** that we know
- Experiments have invested very heavily to improve algorithms and keep memory size and CPU performance matching existing resources
  - E.g. CMS reconstruction
- There is the general opinion that with more computing resources or more optimal software **we could do even more and better physics**
- Computing resources will not scale to a post-LS1 scenario
  - 140 Pileup Events, 25ns
- We need to radically evolve experimental software



CMS Reconstruction

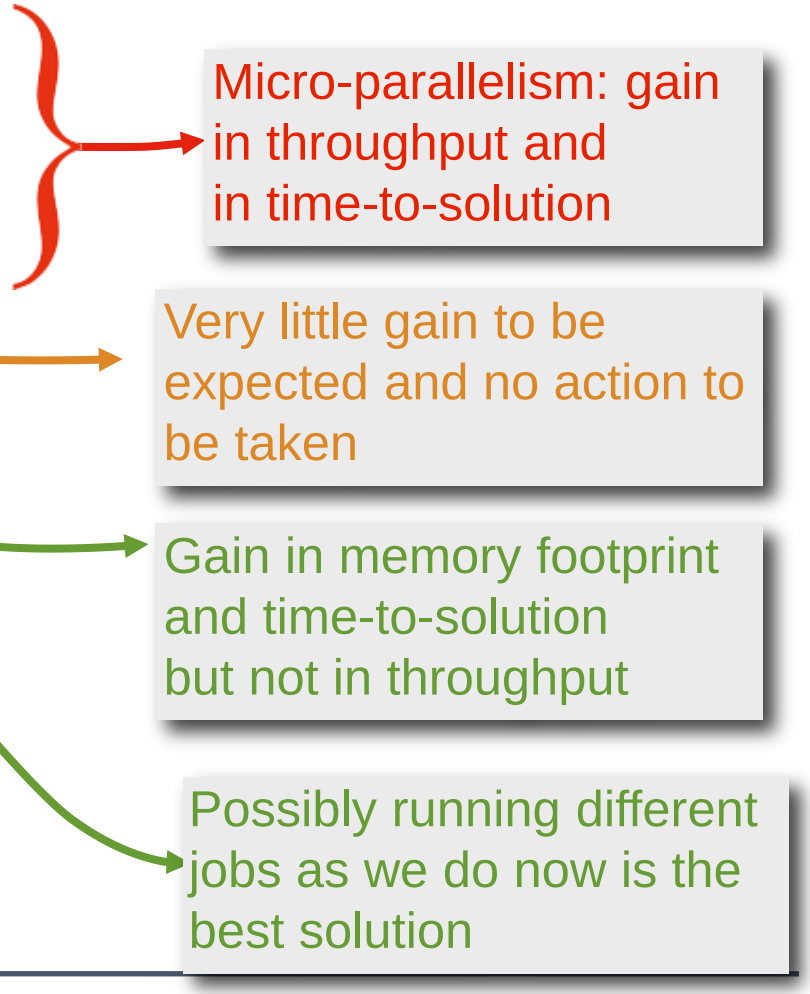
CPU Technology Trends

- Until ~2004 we have had an easy life in HEP software and computing
  - Year after year up to 2x increase in computing capacity thanks to the #transistor/chip (Moore law) and higher clock frequencies
  - The same program that in year 1995 needed 10 seconds, would need 1 second in 2002
- The "easy life" is now over
  - The available transistors are used for adding new CPU cores while keeping the clock frequency basically constant thus limiting the power consumption
- We need to **introduce parallelism** into applications to fully exploit the continuing exponential CPU throughput gains



The Eight dimensions

- The “dimensions of performance”
  - Vectors
  - Instruction Pipelining
  - Instruction Level Parallelism (ILP)
  - Hardware threading
  - Clock frequency
  - Multi-core
  - Multi-socket
  - Multi-node



Prospects for HEP Software

- **Potential gains** can be made by exploiting features of today's CPUs' micro architecture
  - by making use of vector registers, instruction pipelining, multiple instructions per cycle
  - by improving data and code locality and making use of hardware threading
- New architectures to **off-load large computations** to accelerators (GPGPUs, Intel MIC) or the new integrated architectures with heterogenous processors (AMD)
  - specific memory models will force explicit memory programming
  - new programming languages (Cuda, OpenCL, etc.)

Why Micro-Parallelism?

- To exploit full potential of new architectures will have to make use of the other performance dimensions, such as vector instructions (SIMD), instruction level parallelism (ILP)
  - the only realistic potential gain in throughput
  - effective use of vector instructions is essential in order to make full use of current CPUs
  - if we can gain from SIMD instructions, code is also better prepared for accelerators (GPUs, Intel-Phi, ...)
- Independent of Multi-Threading
  - a multi-threaded application may or not use vector instructions
- Optimization suited for individual algorithms or libraries
  - e.g. VDT (math functions)
- Much stronger requirements on properly designed data structures and uniformity of computations

Paradigm Shift

- Most of the scientific software and algorithms was designed for sequential processor in use for many decades and **will require significant re-engineering**
- Migrating sequential applications to multi-threaded is highly non-trivial
  - Difficult to **develop**: we not only need to code what needs to be done but also how this is done in parallel
  - Difficult to **debug**: nasty data race conditions will be difficult to reproduce, and so to fix.
  - Difficult to **maintain**: latent threading bugs may take years to be visible.
- The community needs to **develop expertise in concurrent programming**
  - Similarly to the OOP migration, training will be eagerly needed
- As for the case of major software evolutions such as the migration to OOP, the main final benefit is often not the initial naive selling argument
  - OOP was sold as faster computing while in fact is not, but is the only way we can cope with the unprecedented complexity of today's applications
  - Multiple threads is not an optimization. Using multiple threads will make your workload complete faster only if you have multiple processors and they are underused. The total amount of work typically grows because of the threading overhead.

The Challenge for HEP

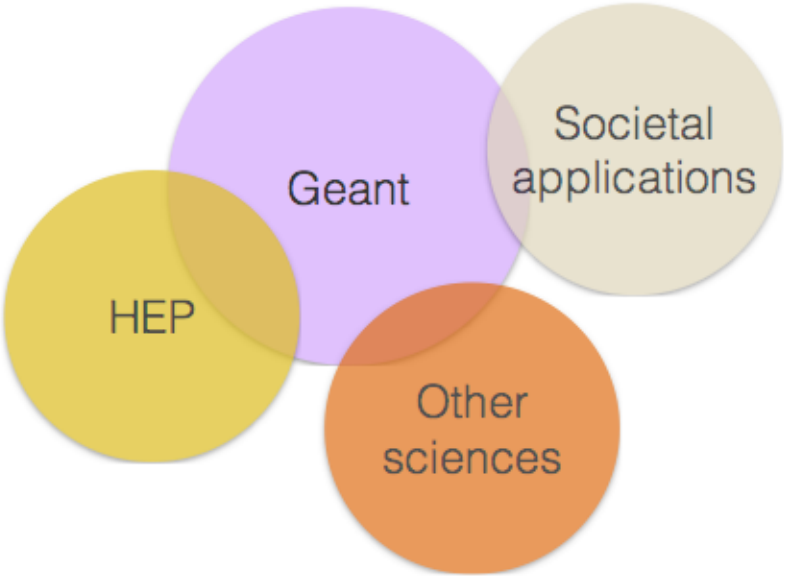
- Concrete algorithms can be run in parallel by making use of threads but integrating them to run in a single application is highly non-trivial
- It will require **new levels of expertise** that need to be acquired by the community
- Tasks will involve a **major re-engineering of frameworks, data structures and algorithms** e.g.
  - making code thread-safe to exploit multi-threading
  - developing new framework services for scheduling parallel execution of threads on all available cores
  - targeting libraries/toolkits where the biggest impact can be achieved e.g. GEANT
- Making changes in the code of running experiments **must be done gradually** whilst preserving the correctness of the physics output
- A fast response is needed since technology is evolving quickly
  - At the LHC, LS1 provides a window of opportunity for introducing first changes
- The scale of the changes requires a very large effort to be invested

Initiatives taken so far

- The adoption of a **collective response** will help to meet the challenges using available expertise and resources and within the required timescale
  - Initial workshop at Fermilab with many experiments represented (LHC and others)
- A Concurrency Forum was established 2 years ago, with the aim of :
  - sharing knowledge amongst the whole community
  - forming a consensus on the best concurrent programming models and on technology choices
  - developing and adopting common solutions
  - The forum meets bi-weekly and there has been an active and growing participation involving many different laboratories and experiment collaborations
- An R&D programme of work was started on a number of **demonstrators** for exercising different capabilities, with clear deliverables and goals

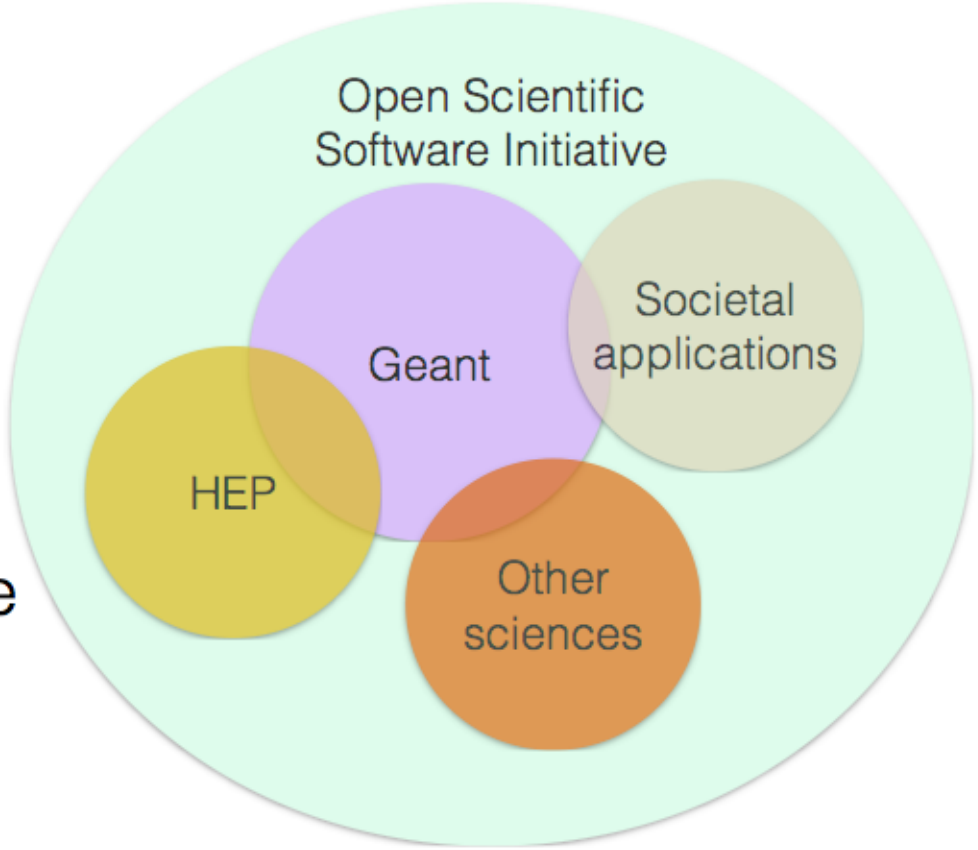
Geant & the brave new world

- The HEP software initiative will be discussed on April 3-4
- The SFT simulation activity is uniquely poised to take advantage and contribute to this initiative
- But this will not be leveraged without change



Geant & the brave new world

- The HEP software initiative will be discussed on April 3-4
- The SFT simulation activity is uniquely poised to take advantage and contribute to this initiative
- But this will not be leveraged without change

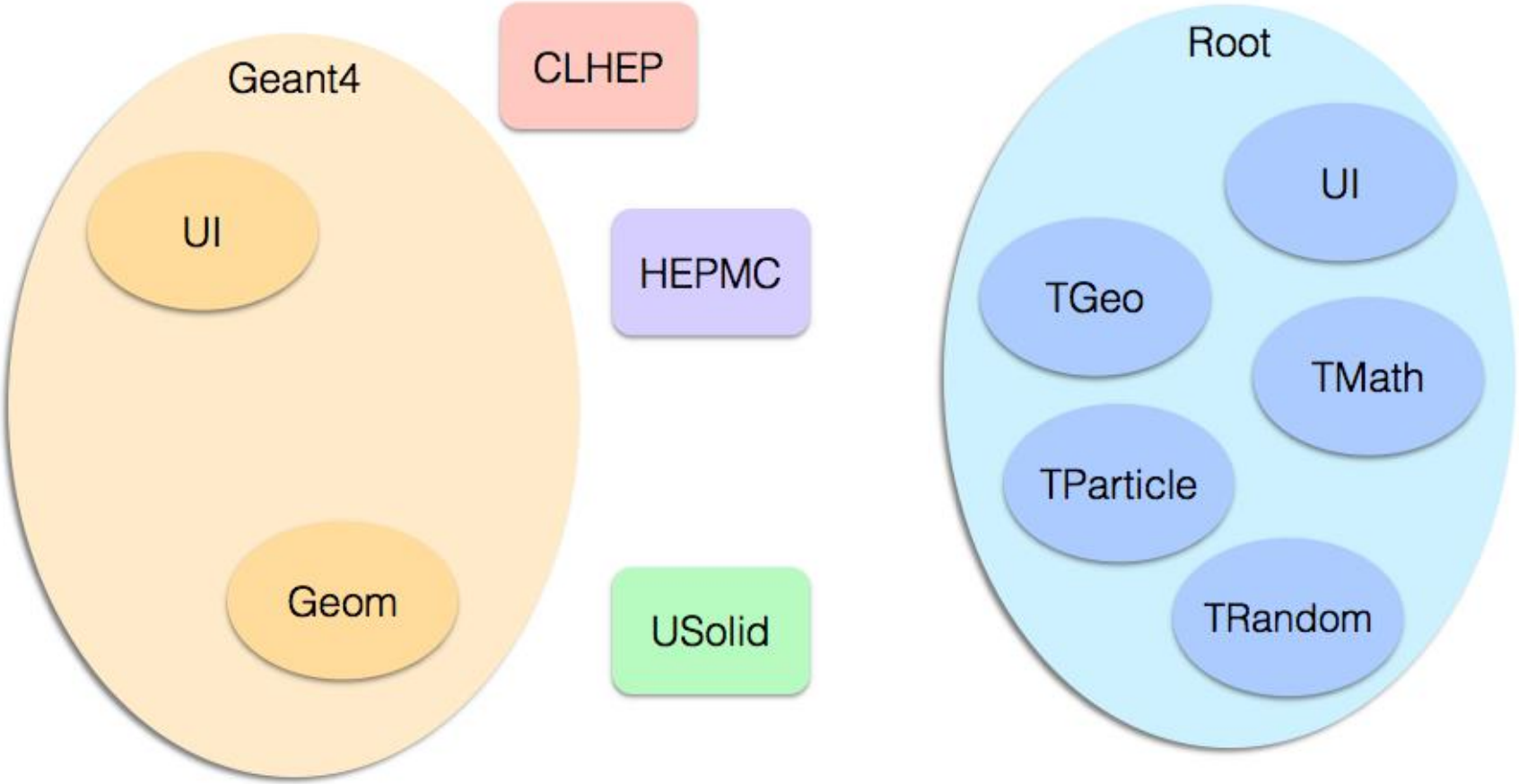


Changing times

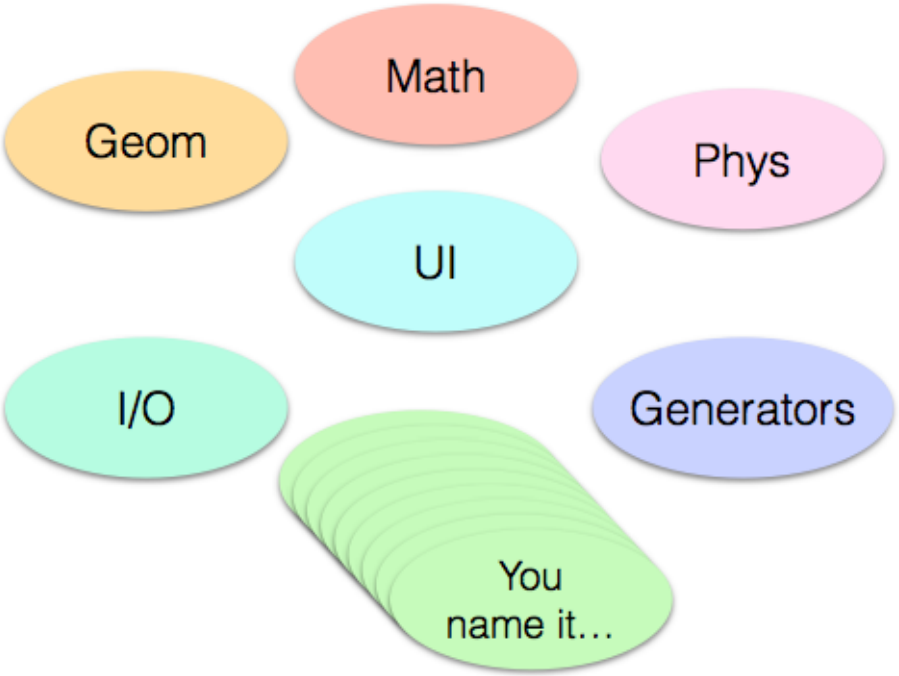
- A very successful system
  - This is, after all, the "Higgs code suite"
- But some sore points
  - Two separate systems – no synergy
  - Large duplications
  - Hardly compatible (and not streamlined) installation systems
  - Different coding conventions, release strategy, user support, code maintenance, documentation, web site, licensing, inline doc, bug reporting…
  - No coherent planning & governance
  - No configuration manager (apt-get, fink, yum…)
- We should (at least have a plan to) "clean the house" before inviting guests
- We should anticipate change, not suffer it!

However we must be ready for change…

Once upon a time… (or rather now…)



A more modular infrastructure



- Separated interacting software packages
- Good idea in principle
  - We have been there and failed more than once
- It seems to be the only way ahead now in a truly distributed and multi-disciplinal environment
  - Both from the maintenance and the governance viewpoint
- We should start planning in this direction
  - Governance, interface definition, life cycle, packaging…

HEP Software Collaboration

New Initiative - Software Collaboration

- Ideas were been discussed in the preparation of the Computing Model Update
- Discussions took place in December with CERN management
  - Director of Research and IT and PH departments
- Agreed that initially should focus on HEP community
  - But always with the intention to work with other scientific and industrial partners where interested

HEP Software Collaboration

- Goal
  - To build and maintain advanced scientific libraries and tools of general interest
  - Facilitate engagement with other scientific and industrial partners
- Initially the HEP community
  - Discussing with all HEP labs, software projects, experiments, etc.
- Make it Open – to enable engagement with others

Proposal

- Establish a collaboration to develop open scientific software packages guaranteed to work together
  - Includes frameworks to help assemble full applications in various domains
- Features
  - Build on work in concurrency forum
  - Formality provides framework for increasing collaboration, attracting experts from HEP and elsewhere
  - A means to provide recognition, give credit
  - Can build coherent proposals for funding from national and international funding agencies; engagement with industry
  - Publish roadmaps and priorities – helps potential collaborators see where they may contribute, but also ensures delivery of important developments
  - Establish a team of experts to provide consultancy and help in optimisation, debugging, etc.
  - Assumes high degree of commonality in the core software of the experiments – libraries, frameworks, tools

Structure

- Formal collaboration with an agreed governance model
  - has worked successfully in other domains (WLCG)
  - equal partners who all can receive recognition
  - can provide a framework for elaborating proposals for acquiring resources
    - EU – Horizon 2020
    - US - NSF/DOE (increase scope of core software activities)
- Organise activities in work packages
  - R&D studies
  - development infrastructure (tools etc.)
  - computing infrastructure (TechLab)
  - task forces to assist collaborations in ‘software campaigns’

Software Collaboration – 3

- Currently discussing with partners in other HEP labs with core software teams
  - inviting comments, suggestions and to share ownership of initiative
- Organised a 1½ day meeting to discuss details
  - Attached to the Concurrency Forum Workshop (April 3,4)
- Suggestion to hold follow-up meeting in US later in year
  - finalise governance model
  - possibly invite US funding agencies to send representatives