Report on
Workshop on Concurrency in the many-Cores Era – @FNAL
Impact for SuperB

Francesco Giacomini – INFN

2nd SuperB Collaboration Meeting
LNF – 13-16/12/2011

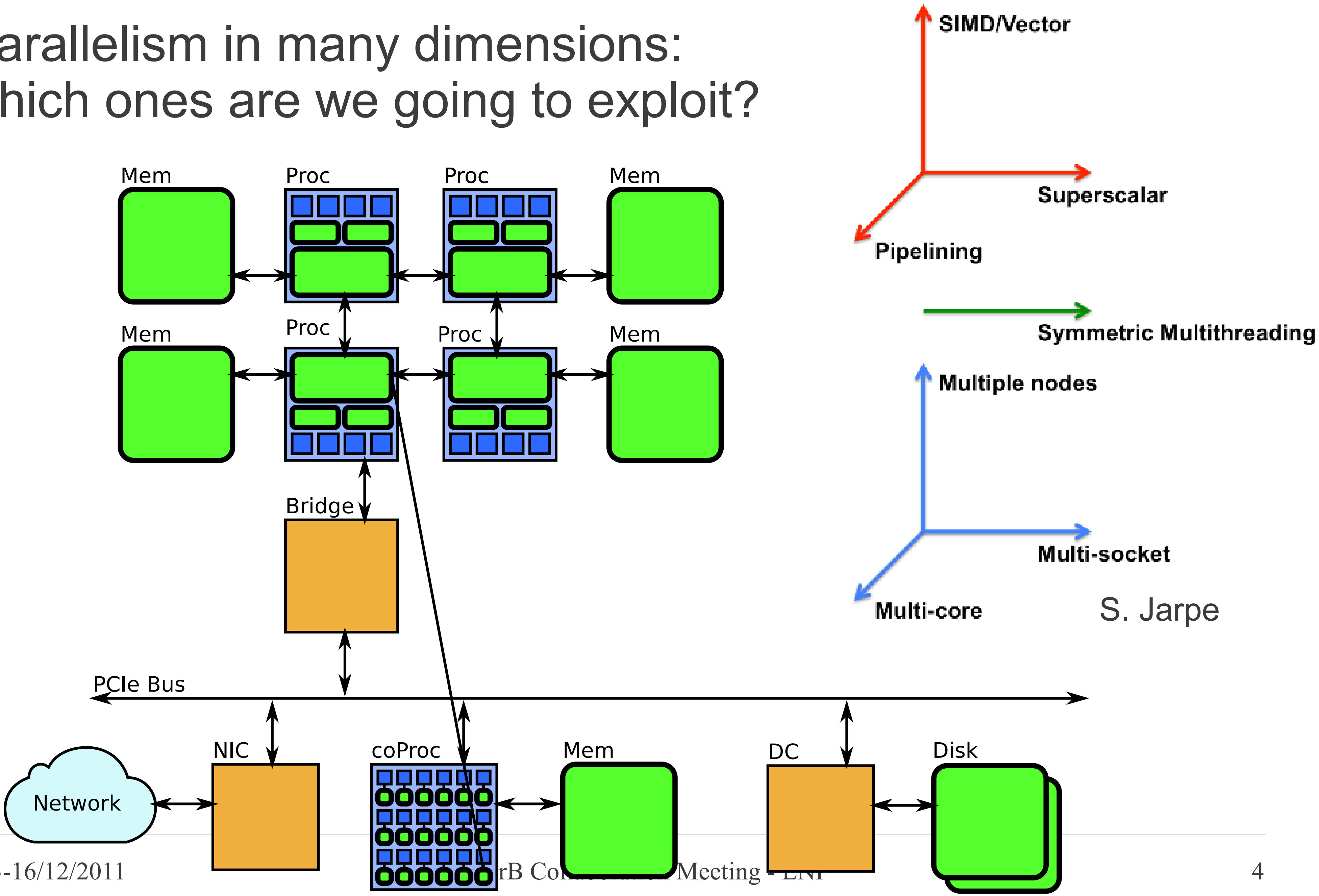
Report

WS Overview

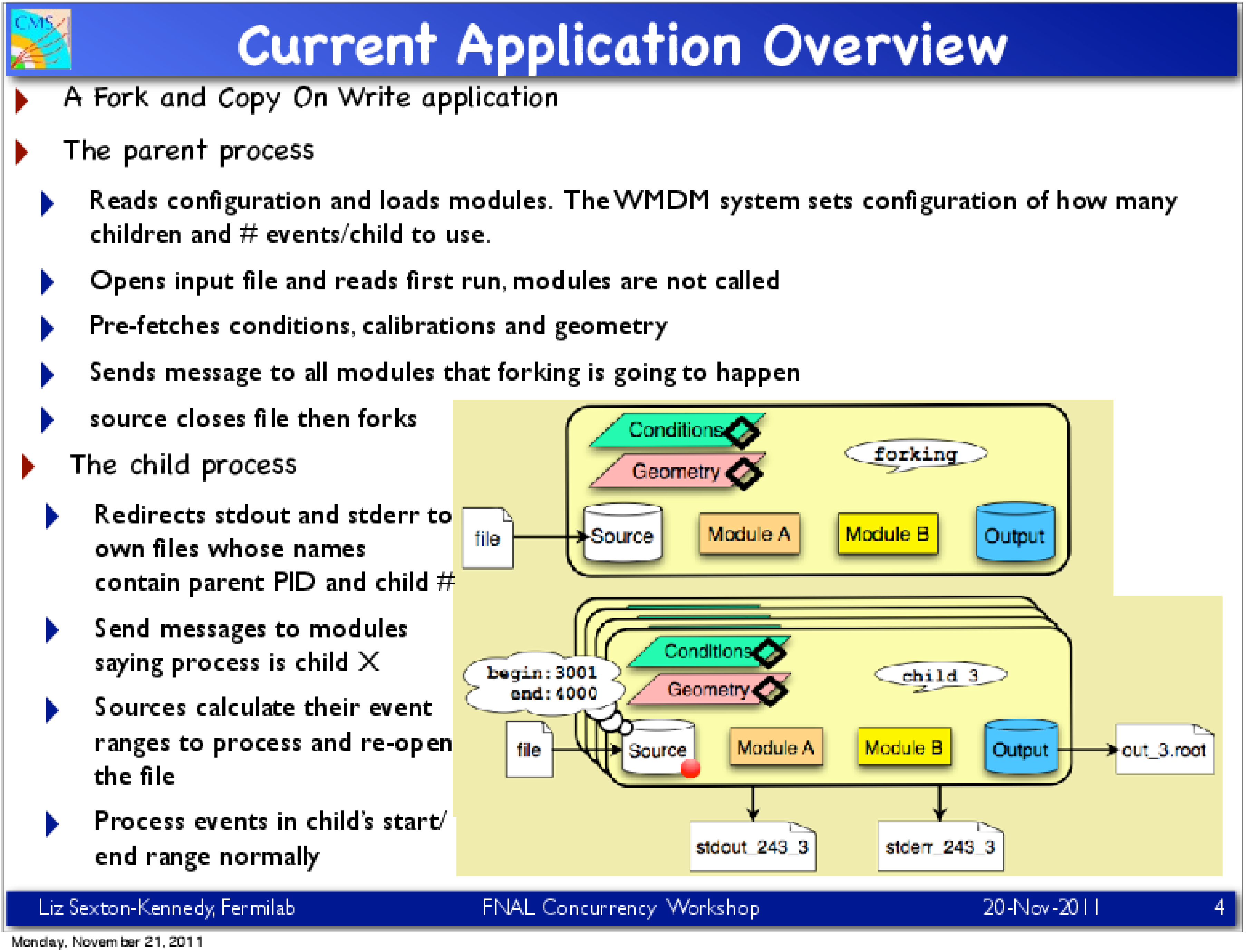
- Main goal: explore the possibility that interested HEP institutions/projects collaborate on R&D of concurrent frameworks
  - Understand requirements and constraints of current and future experiments to efficiently exploit parallel processor architectures
  - Identify potential technologies worth exploring
  - Identify commonalities, synergies and communication model
- Participants from FNAL, CERN, LBL, DESY, (INFN)
- All material available at https://indico.fnal.gov/conferenceDisplay.py?confId=4986

The problem: Heterogeneous and Parallel Systems

Parallelism in many dimensions: which ones are we going to exploit?



S. Jarpe



Current Application Overview

- A Fork and Copy On Write application
- The parent process
  - Reads configuration and loads modules. The WMDM system sets configuration of how many children and # events/child to use.
  - Opens input file and reads first run, modules are not called
  - Pre-fetches conditions, calibrations and geometry
  - Sends message to all modules that forking is going to happen
  - source closes file then forks
- The child process
  - Redirects stdout and stderr to own files whose names contain parent PID and child #
  - Send messages to modules saying process is child X
  - Sources calculate their event ranges to process and re-open the file
  - Process events in child's start/end range normally

Liz Sexton-Kennedy, Fermilab FNAL Concurrency Workshop 20-Nov-2011 4

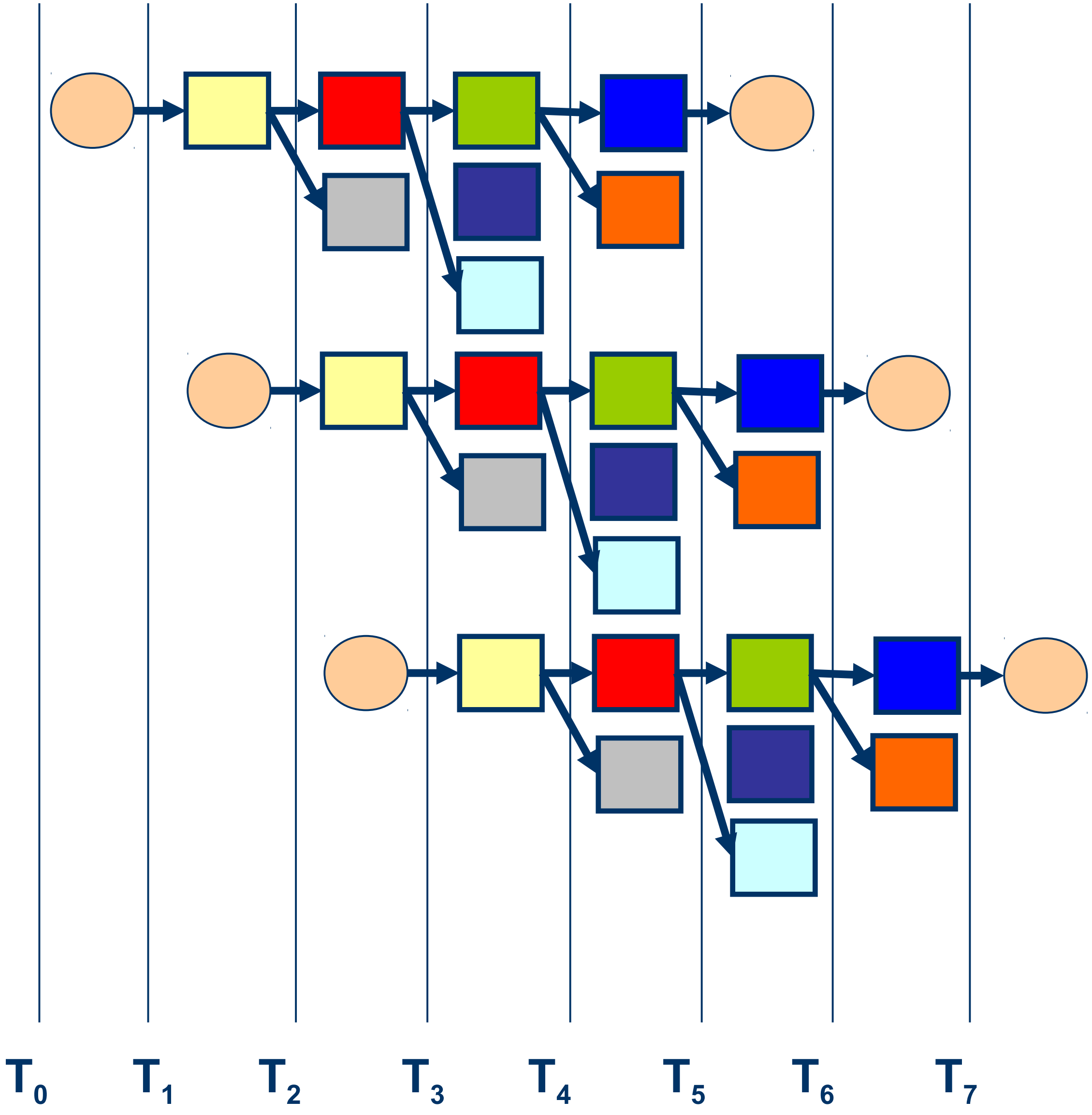
Monday, November 21, 2011

To be combined with whole-node scheduling

Fork & COW

- Helps significantly in reducing memory usage
- Issues
  - When to fork?
  - Unshared memory
    - Beware of finalization/destruction
  - Merging of output files
  - Parallel I/O
  - Inter Process Communication
    - If needed...
    - Passing objects via shared memory is tricky
  - Legacy interfaces
    - Big changes are possible only within the framework or in isolated places (e.g. a specific algorithm)

What next?



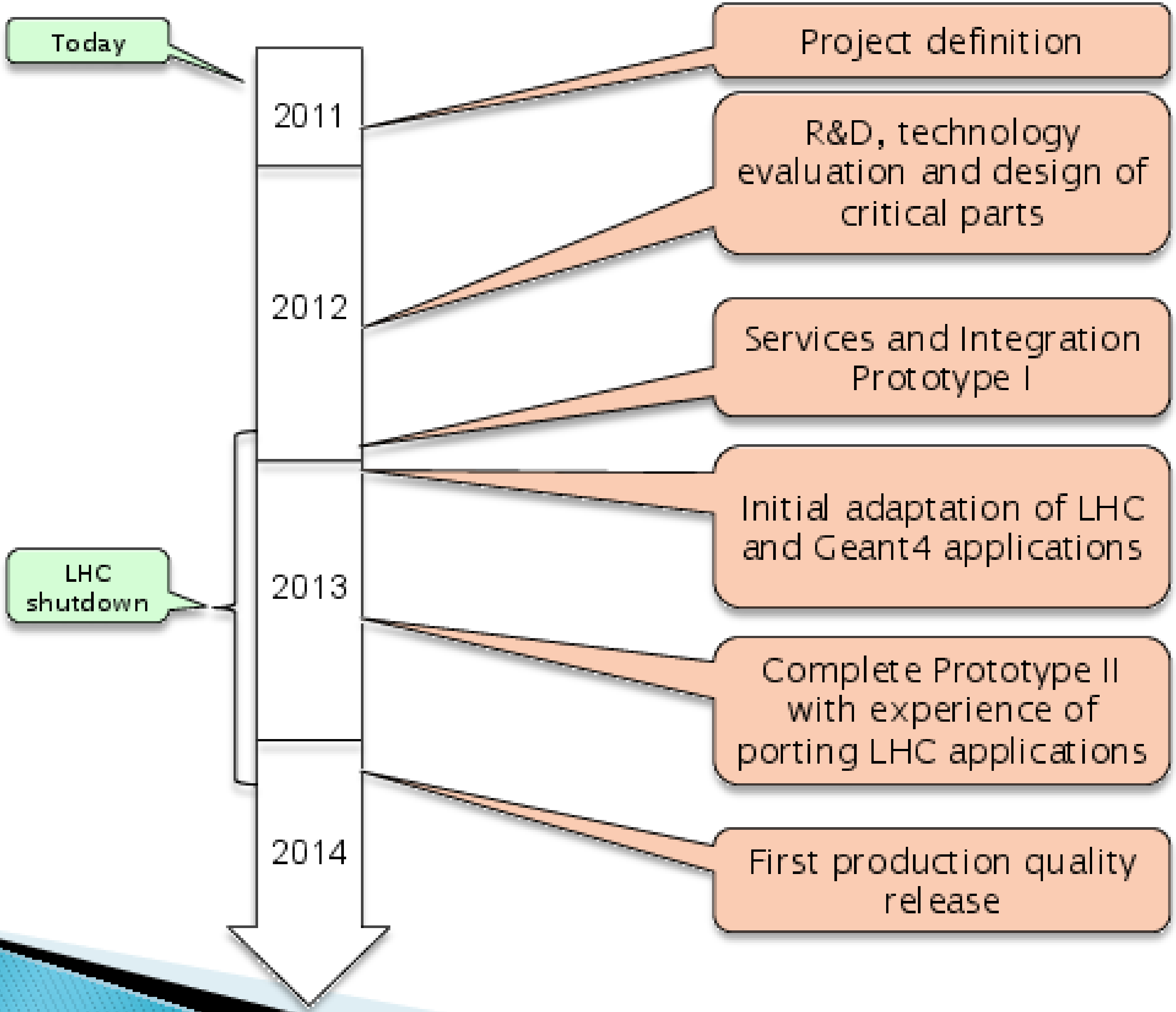
M. Franck / LHCb

- Multiple events, multiple stages per event at once
  - Multi-threaded
- Each stage/algorithm can be parallelized as well
  - Can be run on an accelerator too
- Various technologies
  - GCD, TBB, CnC, OpenMP, ...

R&D - Demonstrators

- Multiple process management
- Performance tools
- Improved data locality
- Scheduling work
- Parallelization within modules
- “Whiteboard” services (geometry, magnetic field, random number generators, …)
- Concurrent building of histograms
- Define performance metrics to evaluate frameworks
- Multi-threaded I/O
- Impact of virtualisation technology
- Use of the Go language

Straw man Project Timeline



Today

LHC shutdown

2011

2012

2013

2014

Project definition

R&D, technology evaluation and design of critical parts

Services and Integration Prototype I

Initial adaptation of LHC and Geant4 applications

Complete Prototype II with experience of porting LHC applications

First production quality release

B. Hegner, P. Mato/CERN

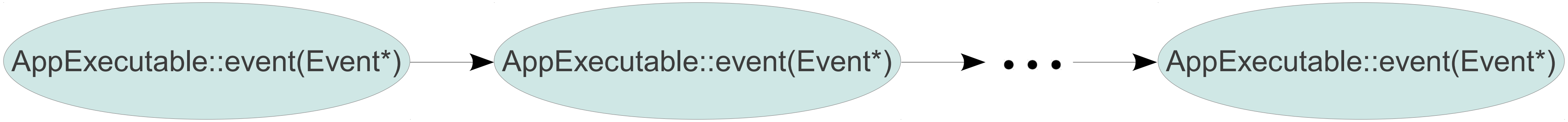
Impact for SuperB

What shall we do?

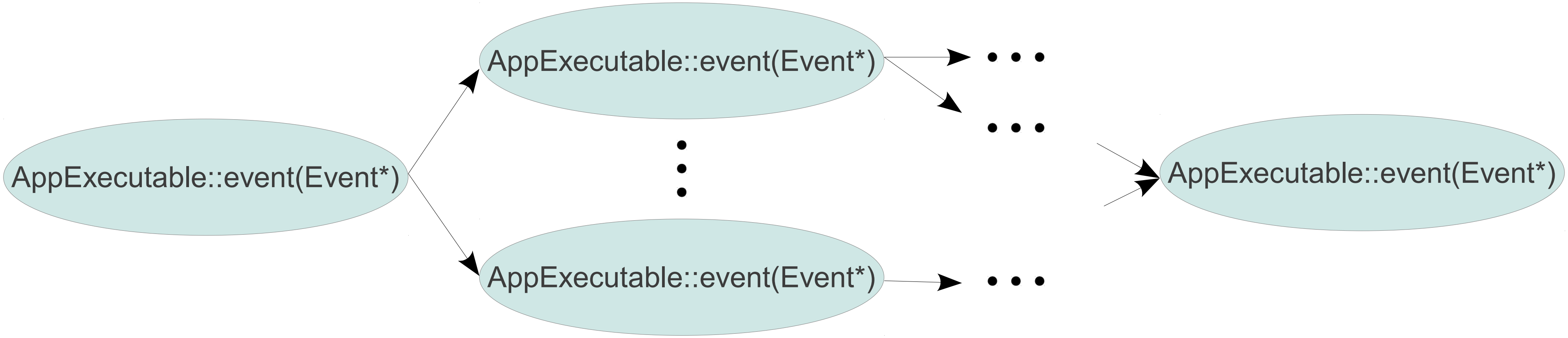
- Contribute to the R&D joint effort
- One possible activity is the investigation of Intel Threading Building Blocks (TBB)
  - Library offering a rich approach to expressing parallelism in a C++ program
  - It represents a high-level, task-based parallelism that abstracts platform details and threading mechanisms
- Adopt solutions and follow recommendations coming from that joint effort

TBB Usage Example

- Modified the framework so that the loop executing the modules in a sequence has been replaced by a graph with the same modules


AppExecutable::event(Event*)
AppExecutable::event(Event*)
...
AppExecutable::event(Event*)

- Inefficient way of doing the same thing
  - Proof of concept
- But what about the following?


AppExecutable::event(Event*)
AppExecutable::event(Event*)
...
...
AppExecutable::event(Event*)
AppExecutable::event(Event*)
...
AppExecutable::event(Event*)

Challenges

- Thinking parallel
  - Probably more natural than thinking sequential
- Express explicitly a dependency on (products of) other modules
- Syntax friendly to the framework user
- Leave enough flexibility to the framework developer
- Find right abstractions to express parallelilsm
- Tool support