Software for Detectors
Detectors for Software

Maria Grazia Pia
INFN Genova, Italy

http://www.ge.infn.it/geant4/talks

Topical Seminar on Innovative Particle and Radiation Detectors
Siena, 1-5 October 2006
Storage
raw recording rate 0.1–1 GByte/s
accumulating at 12-14 PBytes/year

Processing
70,000 of today’s fastest PCs
(~6 hours’ Intel CPU production today)

1000 person-years
“offline” software effort per experiment

~5000 Physicists
around the world, around the clock

20 years software life-span

(+30 minimum bias events)
All charged tracks with pt > 2 GeV
From deep underground...

Dark matter and $\nu$ experiments

...to space

X and $\gamma$ astronomy, gravitational waves etc.

Cosmic ray experiments

Variety of requirements from diverse applications

Physics from the eV to the PeV scale

Detectors, spacecrafts and environment

For such experiments software is often mission critical

Require reliability, rigorous software engineering standards

Maria Grazia Pia
Accurate modelling of radiation sources, devices and human body
Precision of physics
Reliability
Easy configuration and friendly interface
Speed

From hospitals...
...to Mars

Medical Physics, Radiation Protection

Maria Grazia Pia

Courtesy of ESA
Software for detectors

Software permeates HEP experimental life

- Detector design and optimisation
- Trigger
- Data acquisition
- Reconstruction
- Physics analysis
- Radiation protection
- Accelerator design and operation...

- Spacecrafts, mission trajectory calculation, patients’ treatment planning, medical imaging etc.

- Software is as essential as hardware in our experiments and requires a large investment of resources

LHC offline software and computing investments exceed the cost of a LHC experiment like ATLAS or CMS!

Maria Grazia Pia
Challenges

Performance challenge
- Increased needs of computing resources
- New paradigms emerging: the grid

Programming challenge
- Large scale software systems
- Long timescale
- Complexity of physics and detectors

Prediction challenge
- Complexity of software systems
- Large and geographically distributed development teams

Do we trust software?
The primary cause was found to be a piece of software which had been retained from the previous launchers systems and which was not required during the flight of Ariane 5. As well, the software contained implicit assumptions about the parameters, in particular the horizontal velocity that were safe for Ariane 4 but not Ariane 5.

Therac-25 accidents

6 known massive overdose cases (June 1985 – January 1987)
Software coding error found
Poor software engineering practices
Safety analysis excluded software

10 November 1999

Failure to convert English measures to metric values caused the loss of the Mars Climate Orbiter, a spacecraft that smashed into the planet instead of reaching a safe orbit
Verification & Validation

Codes are not reality, only a model of reality

Software verification
Confirmation that the software correctly implements the specifications
- The software does what is supposed to do
- Software Quality Assurance

Software validation
Confirmation that the physical models implemented are correct representations of the physical phenomena of interest
- Ensure models reflect nature
- Check against experimental data
ASCI experience

Accelerated Strategic Computing Initiative
to certify US nuclear stockpile without testing

Lessons learned from 6 software projects
- 2 succeeded
- 4 missed initial milestone
- 2 cancelled

Major lesson is that one needs to improve:
- Verification
- Validation
- Software Project Management and Software Quality

V&V emphasized since the beginning in ASCI
Difficulties encountered by software project teams
→ Need better V&V methods

Test coverage

- More complex issue for scientific software than for most business software
  - Huge number of variants and controls

- Are conventional test-coverage tools adequate for scientific software?
  - Probably not...
  - Show which lines of code we are not testing, but this does not say much about physical validity

The issue of test coverage is an unsolved problem
Validation is holistic

One has to validate the entire calculation system

Including:
- User
- Computer system
- Problem setup
- Running
- Results analysis

An inexperienced user can easily get wrong answers out of a good code in a valid régime

Experimental evidence in Geant4 Problem Report System
...but also in my personal editorial experience in a major scientific journal

Maria Grazia Pia
The Columbia Space Shuttle wing failed during re-entry due to hot gases entering a portion of the wing damaged by a piece of foam that broke off during launch.

Shortly after launch, Boeing did an analysis using the code CRATER to evaluate the likelihood that the wing was seriously damaged.

Problems with analysis:
- The analysis was carried out by an inexperienced user.
- CRATER was designed to study the effects of micrometeorite impacts, and had been validated only for projectiles less that 1/400 the size and mass of the piece of foam that struck the wing.
- Didn’t use a code like LS-DYNA that was the industry standard for assessing impact damage.

The prior CRATER validation results indicated that the code gave conservative predictions.

Analysis indicated that there might be some damage, but probably not at a level to warrant concern. (Maybe there was no way to fix the problem and avoid the accident.)

— NASA Columbia Shuttle Accident Report
## Data for software validation

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passive observations of physical events</td>
<td>(e.g. supernovae explosions or the weather)</td>
</tr>
<tr>
<td>Experiments designed to elucidate a general physics principle or law</td>
<td>(e.g. typical HEP experiments)</td>
</tr>
<tr>
<td>Experiments designed to certify a detector</td>
<td>(e.g. test beams)</td>
</tr>
<tr>
<td>Experiments <strong>specifically designed</strong> to validate a software system/component</td>
<td></td>
</tr>
</tbody>
</table>
Validation: finding reliable data

backscattering for e-
- energy range: 0.1 keV → 102 keV

... or there are no data, or the details of the experimental setup are not documented, or the systematic is not analyzed, or experimenters are not keen to share their data, or…
We need a paradigm shift...

- Experimenters and funding agencies understand the value of experiments designed
  - to explore new scientific phenomena
  - to test theories
  - to examine the performance of design components

- Few appreciate the value of experiments explicitly conducted for software validation

- Gain of consciousness in some fields
  - NASA: experiments at accelerators
  - Military projects: e.g. weapon simulations

- What about particle physics?

One may say that few appreciate the need for funding software…

Maria Grazia Pia
Scientific codes are frequently changed throughout their lifetimes, not just when they are young. The more frequently a code is changed, the more difficult it is to maintain its correctness.

**Techniques that help**
- Protocol for change control
- Design methods
- Unit testing
- Integration testing
- Regression testing
- Automated testing
- Release management
- Bug tracking

Are they established in the scientific software development process? …not just a problem with scientific software (e.g. recent incident with a famous DB software)

Maria Grazia Pia  
*P.F. Dubois, Maintaining correctness in scientific programs, Comp. Sci. & Eng., No. 5, p. 80, 2005*
Key elements for software quality

- Highly competent and motivated people in a good team
- Software project management
- Support from management and stakeholders
- Schedule and resources determined by requirements
- Software engineering: established best practices
- Project leader must control resources (otherwise “leader” is just a “cheerleader”)

D. Post, The Coming Crisis in Computational Science, LA-UR-04-0388
T. DeMarco, 2000; DeMarco and Lister, 1999; Cockburn and Highsmith, 2001; Thomsett, 2002; McBreen, 2001
Brooks, 1987; Remer, 2000; Rifkin, 2002; Thomsett, 2002; Highsmith, 2001
What happens in scientific journals?

Computational results without documented validation, but also:

**FIG. 4.** Depth-dose curves for a water phantom irradiated by a parallel electron beam. The GEANT4 predictions are compared with the MCNP (Ref. 18) and EGSmc (Ref. 16) predictions.

**M. Zaider et al., Rad. Res. 95, 231-247 (1983)**

![Graph showing depth-dose curves for electron beam](image)

**FIG. 5.** Double differential electron production cross sections for 0.3 MeV protons incident on water vapor. The curves are as used in the code for production of (from the top) 9.6, 47, 94.1, 235.2, and 800.6 eV electrons. The experimental data (points) are from Toburen and Wilson (34).


![Graph showing depth-dose curves for electron beam](image)

How well does the peer-review system work with software-oriented papers?

Many things that a referee cannot detect could be wrong with a computational paper …

Maria Grazia Pia
To make progress on both fronts simultaneously will require collaboration among the hardware-oriented and software-oriented physics communities.

Methods

A strong intellectual basis for V&V
We need research in methodology to test scientific code

Experiments

We need specific, adequate, reliable data taken in controlled conditions to test scientific code

Maria Grazia Pia
We must be still and still moving
Into another intensity
For a further union, a deeper communion
...
In my end is my beginning.

_Four Quartets – East Coker_
_Thomas S. Eliot, 1940_
Does software have predictive capabilities?

Prediction: use of a computational model to foretell the state of a physical system under conditions for which the computational model has not been validated.