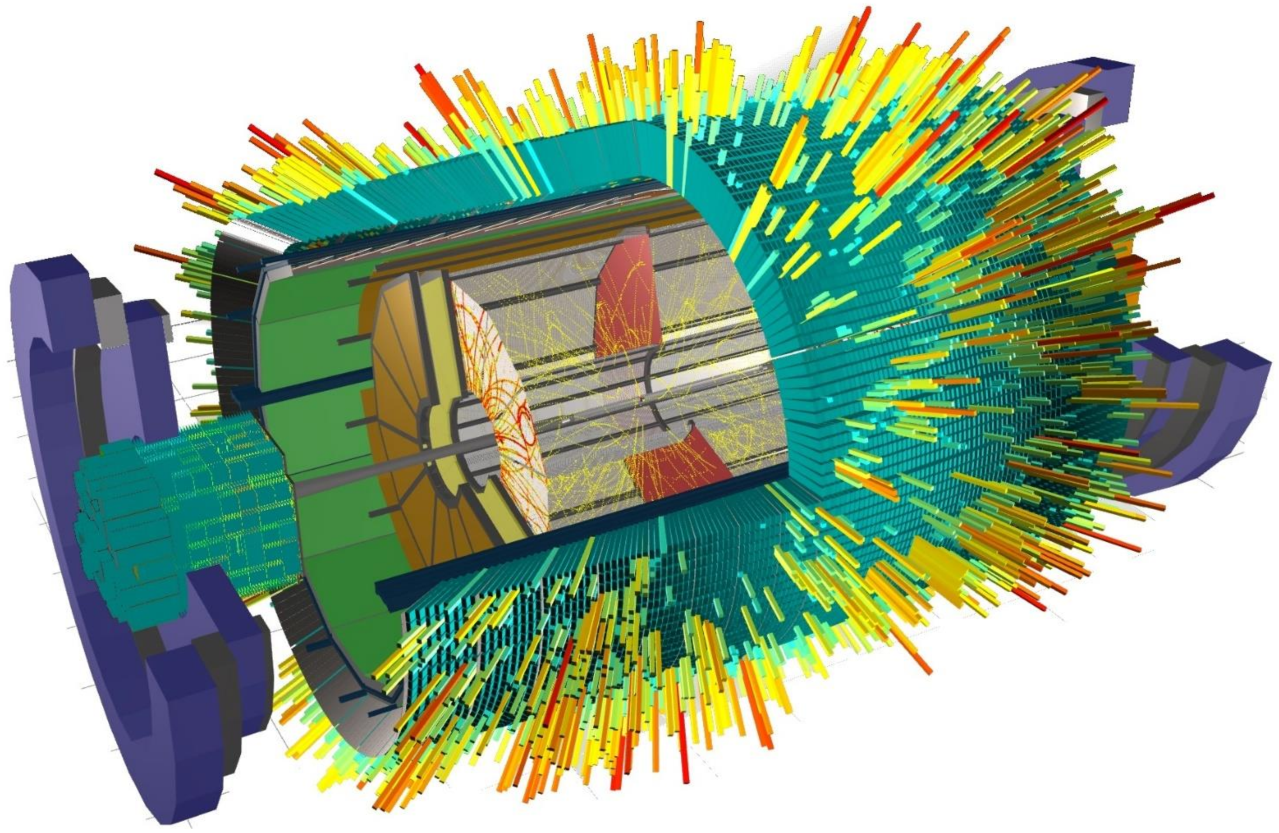
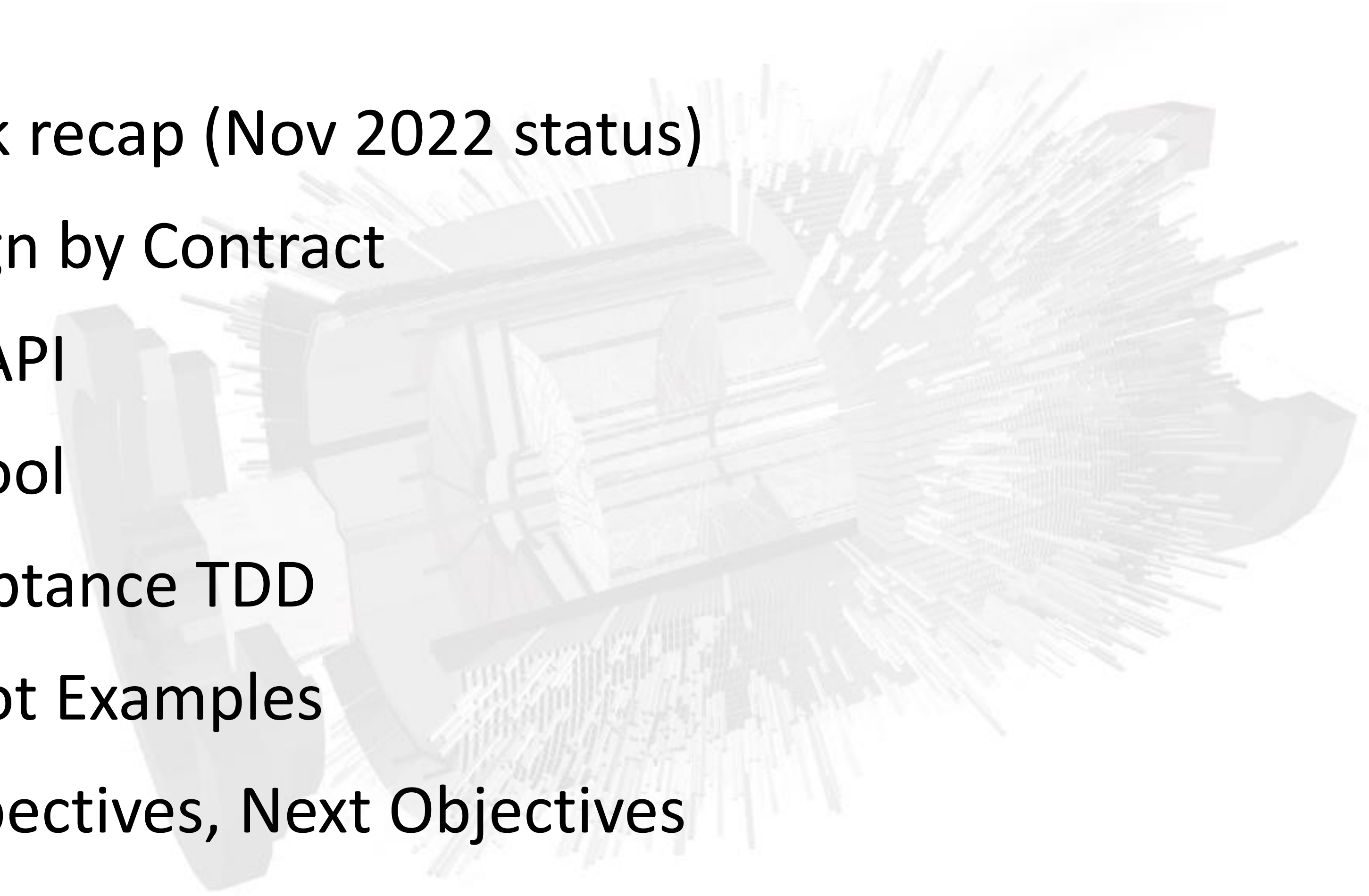
Design by Contract & Acceptance Test Driven Development in MPDRoot

HNATIC Slavomir

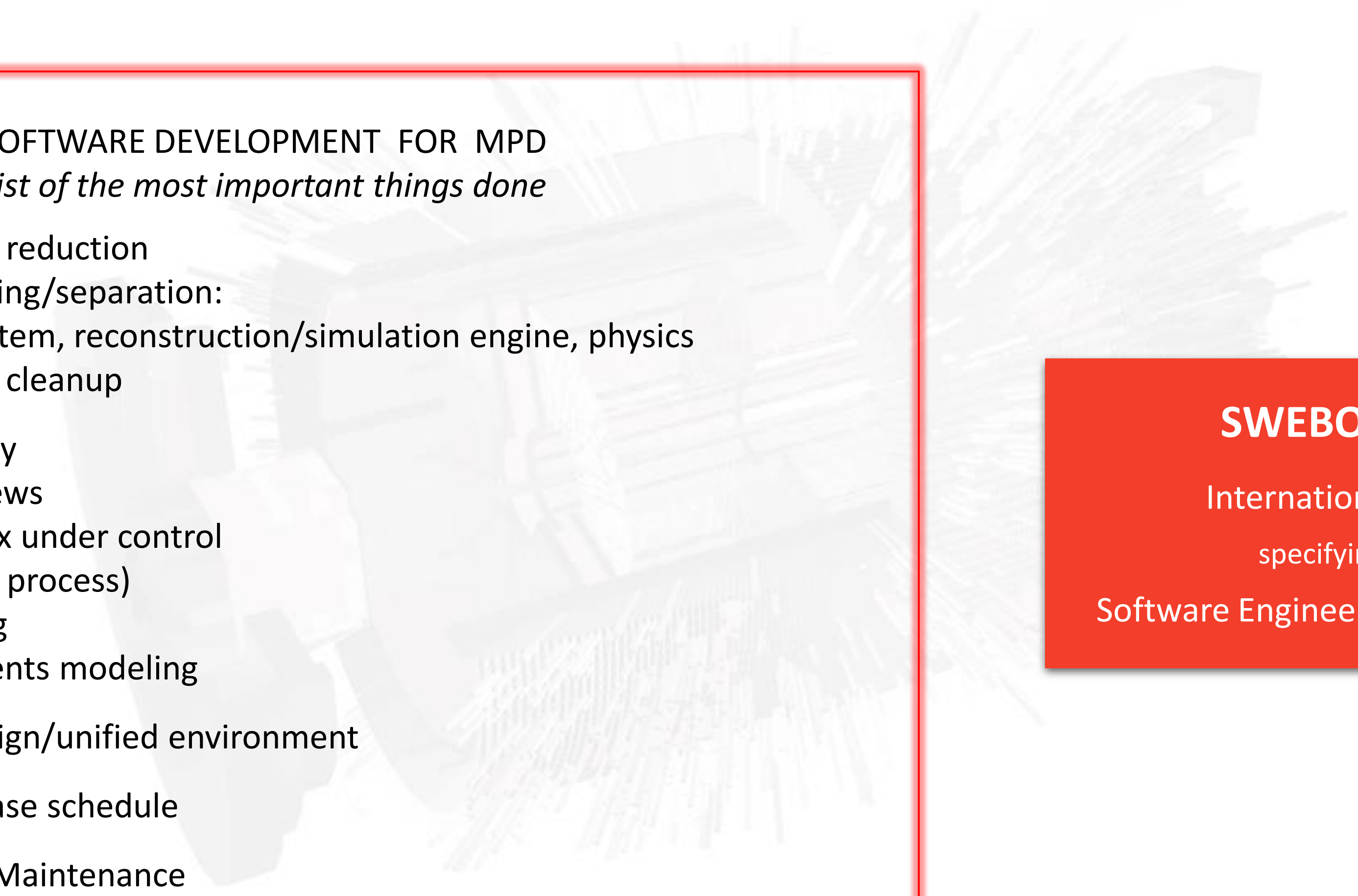


OUTLINE



- Quick recap (Nov 2022 status)
- Design by Contract
- TPC API
- QA tool
- Acceptance TDD
- JSRoot Examples
- Perspectives, Next Objectives
- Final Remarks

QUICK RECAP



SOFTWARE DEVELOPMENT FOR MPD
List of the most important things done

- Complexity reduction
  - downscaling/separation:
    build system, reconstruction/simulation engine, physics
  - codebase cleanup
- Code quality
  - code reviews
  - code influx under control
  - testing (in process)
  - formatting
  - requirements modeling
- Build redesign/unified environment
- Stable release schedule
- Support & Maintenance
  - service desk
  - website
  - telegram support chat

SWEBOK v3 (2015)

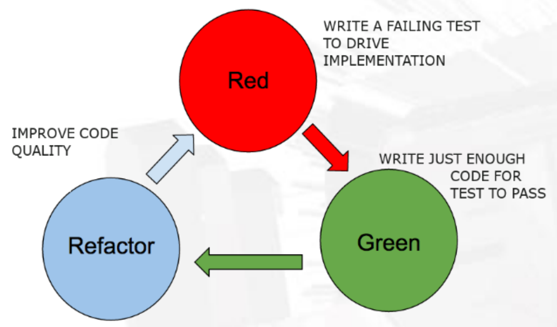
International ISO Standard

specifying the guide to

Software Engineering Body of Knowledge

TDD: ALGORITHM DEVELOPMENT

Status & Objectives as of November 2022


Red
WRITE A FAILING TEST TO DRIVE IMPLEMENTATION
Green
WRITE JUST ENOUGH CODE FOR TEST TO PASS
Refactor
IMPROVE CODE QUALITY

DESIGNING TESTS ON MULTIPLE ABSTRACTION LEVELS

Test level hierarchy “system / component / unit“ adapted for MPDRoot’s backend:

- Top level…………system (bench) tests…….QA
- Middle level……….component tests……….reconstruction FairTasks (invariant interfaces)
- Bottom level…………….unit tests……………..interface units (invariant pure virtual methods)

Cluster Hit Finder

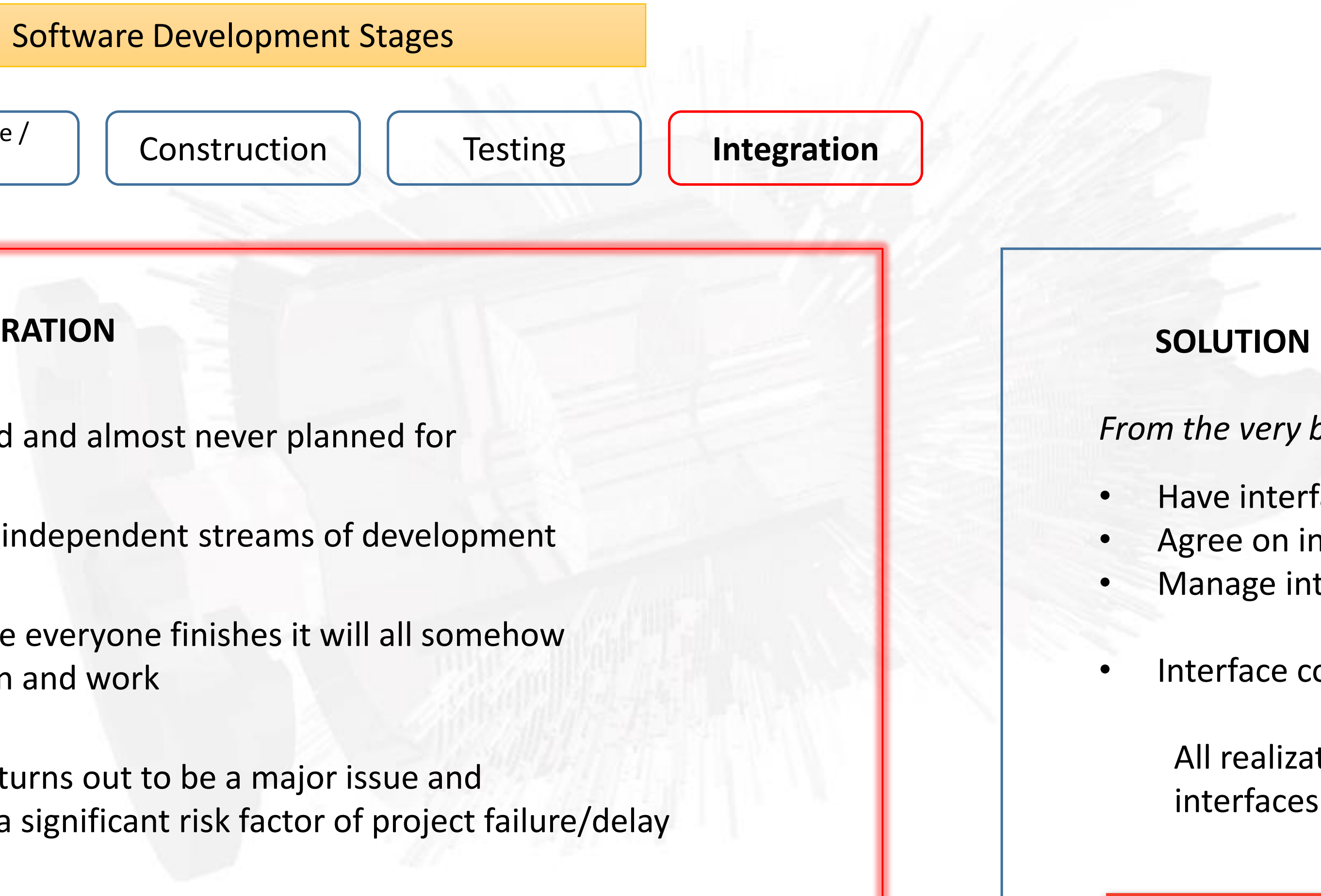
Preparatory work

- create **invariant** Base class for geometry
- interface for clusterhitfinder
- port mlem & fast implementations to it
- getting rid of singletons
- test-friendly design – dependency injection

TDD

- multilevel analysis
- multi-module analysis
- performance & accuracy criteria
- data-driven tests
- hybrid algorithms

DESIGN BY CONTRACT



Software Development Stages

Requirements | Architecture / Design | Construction | Testing | **Integration**

INTEGRATION

- Rarely mentioned and almost never planned for
- Reality: multiple independent streams of development
- Assumption: once everyone finishes it will all somehow
  fit in and work
- Common result: turns out to be a major issue and
  a significant risk factor of project failure/delay
- Last resort fixes: redesign at late project stages,
  writing of unnecessary modules

SOLUTION

From the very beginning do:

- Have interfaces
- Agree on interfaces
- Manage interfaces

- Interface control document

All realizations must implement interfaces that are agreed upon

Ensures software fitness, compactness and TESTABILITY

TPC API

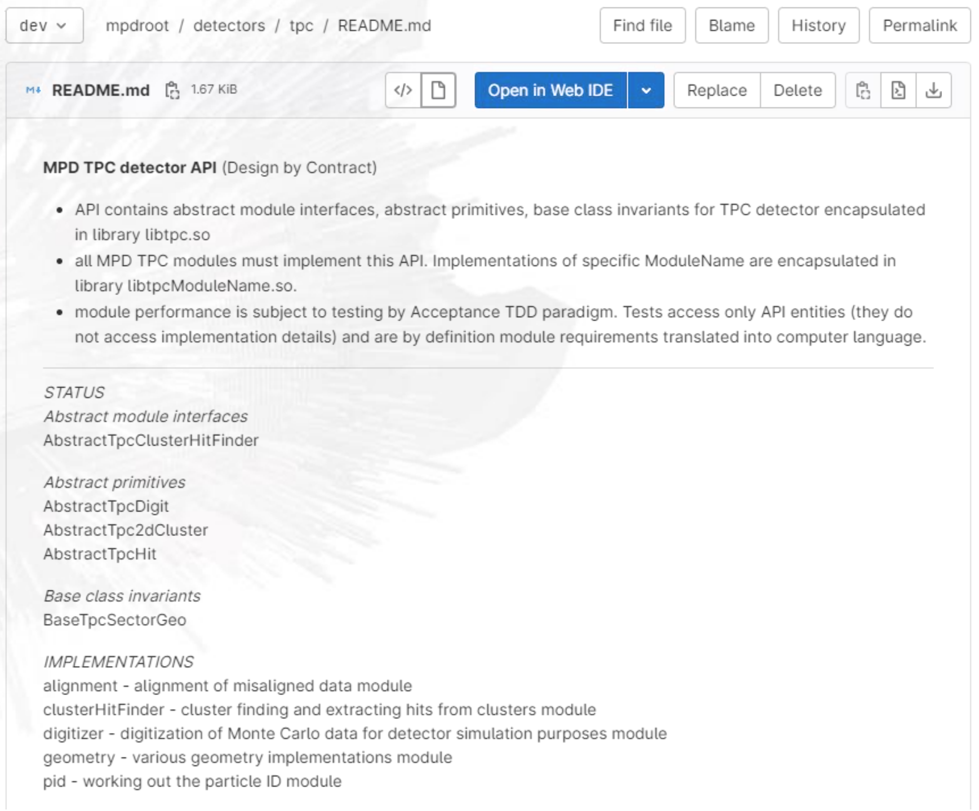
API – set of signatures that are exported and available to the users of a library or framework to write their applications.

Key API design notes

- Lead to readable code
- Easy to learn and memorize
- Be complete & stable for proper development and maintenance (be model based)
- Outlast its implementations (invariants)
- Be hard to misuse
- Be easy to extend
- Lead to backward compatibility

Source: *SWEBOK (Software Engineering Body of Knowledge), 2015*

$$\begin{tikzcd}\boxed{\text{Testing}} \arrow[r, leftrightarrow] & \boxed{\textbf{API}} \arrow[r, leftrightarrow] & \boxed{\text{Implementation}}\end{tikzcd}$$



MPD TPC detector API (Design by Contract)

- API contains abstract module interfaces, abstract primitives, base class invariants for TPC detector encapsulated in library libtpc.so
- all MPD TPC modules must implement this API. Implementations of specific ModuleName are encapsulated in library libtpcModuleName.so.
- module performance is subject to testing by Acceptance TDD paradigm. Tests access only API entities (they do not access implementation details) and are by definition module requirements translated into computer language.

STATUS
Abstract module interfaces
AbstractTpcClusterHitFinder

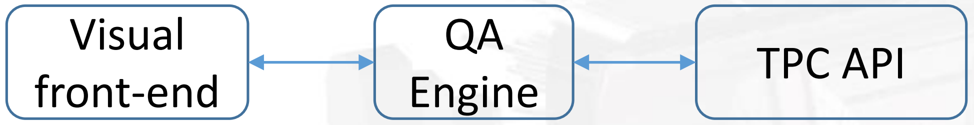
Abstract primitives
AbstractTpcDigit
AbstractTpc2dCluster
AbstractTpcHit

Base class invariants
BaseTpcSectorGeo

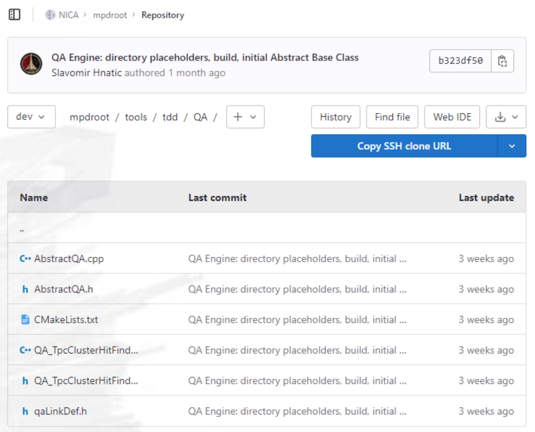
IMPLEMENTATIONS
alignment - alignment of misaligned data module
clusterHitFinder - cluster finding and extracting hits from clusters module
digitizer - digitization of Monte Carlo data for detector simulation purposes module
geometry - various geometry implementations module
pid - working out the particle ID module

QA TOOL

Architecture



- QA Engine is a separated entity on its own
- interacts through API with reconstruction/simulation backend and generates output for visual front-end
- work of testers and algorithm developers is separated



QA Engine: directory placeholders, build, initial Abstract Base Class
Slavomir Hnatic authored 1 month ago
b323df50

mpdroot / tools / tdd / QA /

Name	Last commit	Last update
..		
AbstractQA.cpp	QA Engine: directory placeholders, build, initial ...	3 weeks ago
AbstractQA.h	QA Engine: directory placeholders, build, initial ...	3 weeks ago
CMakeLists.txt	QA Engine: directory placeholders, build, initial ...	3 weeks ago
QA_TpcClusterHitFind...	QA Engine: directory placeholders, build, initial ...	3 weeks ago
QA_TpcClusterHitFind...	QA Engine: directory placeholders, build, initial ...	3 weeks ago
qaLinkDef.h	QA Engine: directory placeholders, build, initial ...	3 weeks ago

Implementation

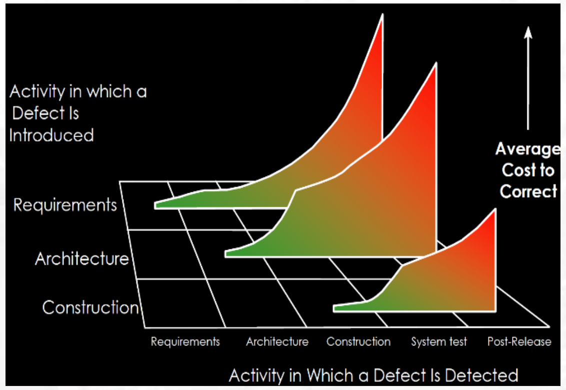
- Modular design, lives in backend interfaces, operates with abstractions
- QA engine turned off by default, option to turn on QA for separate modules
- output QA information stored into .root files for use in later processing

ACCEPTANCE TDD

Fundamental Rule

The more systematic we are in testing, the more efficient/effective we are in building/supporting/maintaining our software.

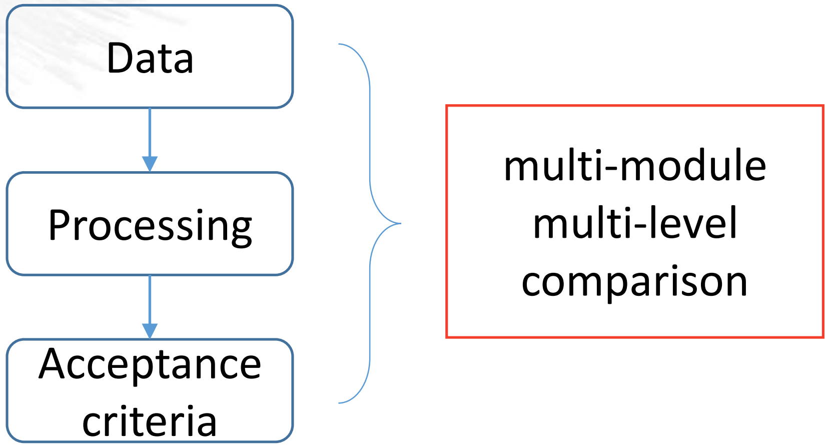
Software Defects



- the later the defect is fixed, the more it costs to correct
- detect defects early
- fix defects asap, avoid technical debt

ACCEPTANCE TESTS = REQUIREMENTS

- development driven by multi-level acceptance tests
- requirements written in precise test case language
- acceptance criteria/their fulfillment is data-driven



! data are customized for acceptance criteria !

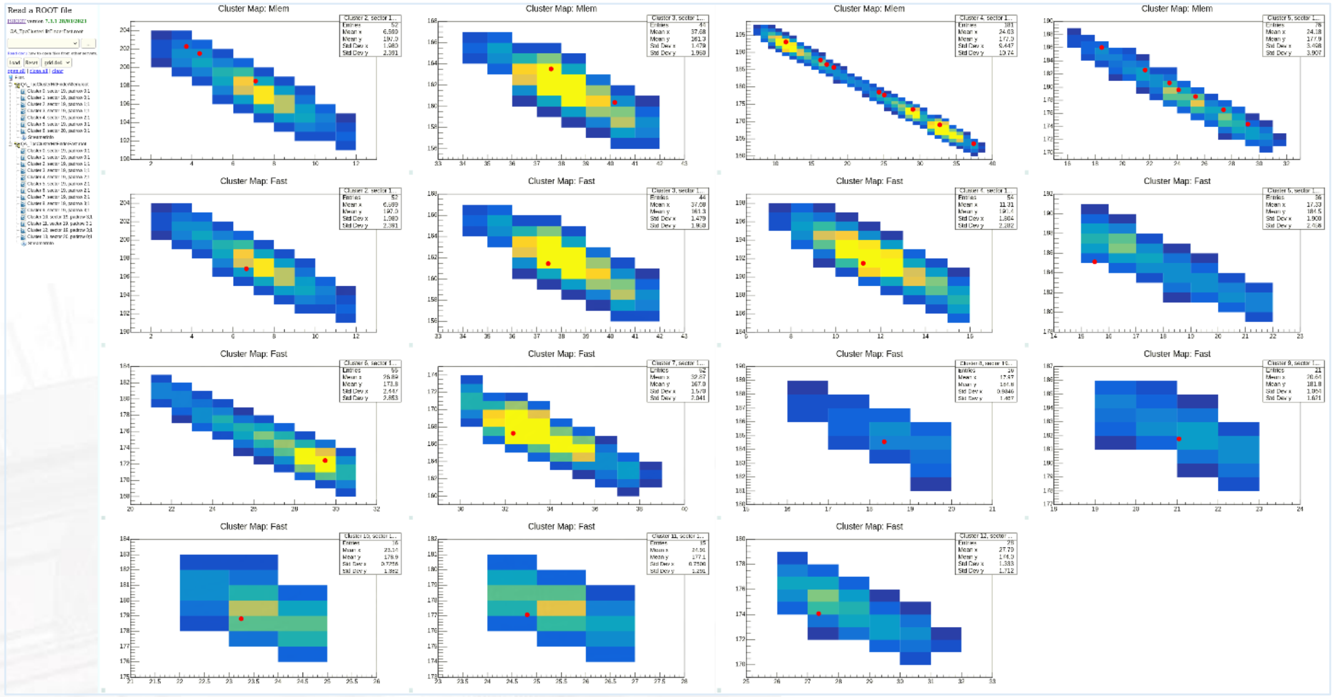
EXAMPLE IN JSROOT

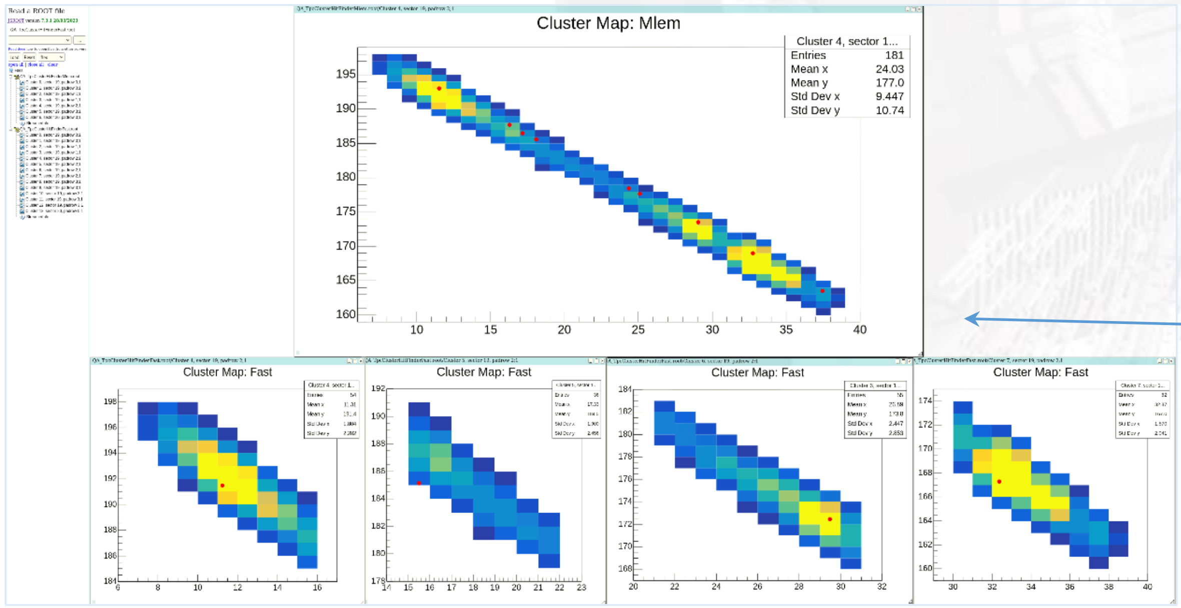
CLUSTERHITFINDER COMPARISON

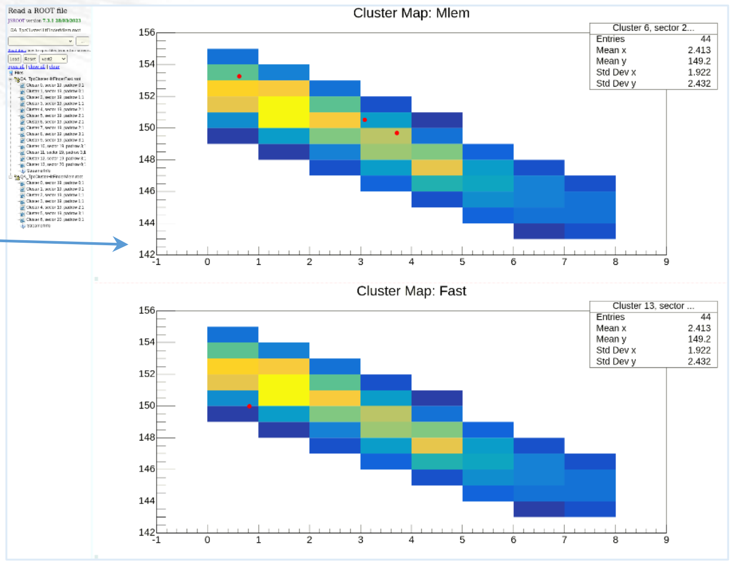
- Mlem
- Fast

ABSTRACTION LEVELS

- Top …………..bench………Reconstruction
- Middle…..component….ClusterHitFinder
- Bottom ………units……….Clustering, Topology, Hit extraction







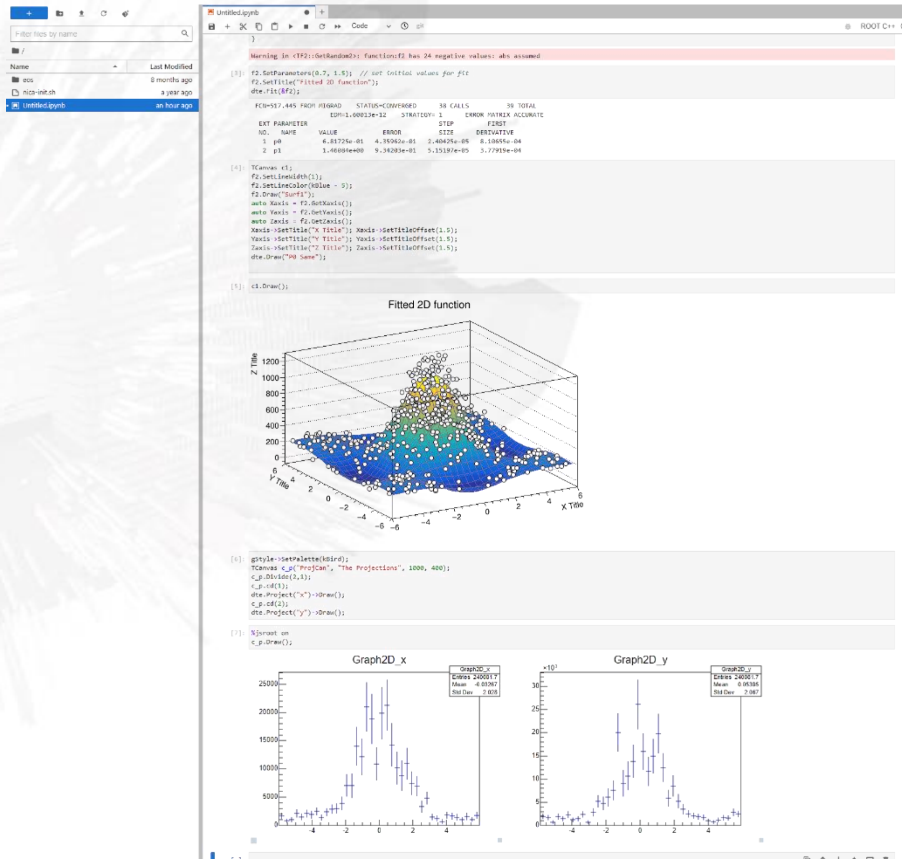
jsRoot server
(remote front-end)

PERSPECTIVES, FUTURE PLAN

ENVIRONMENT for ALGORITHM IMPROVEMENT

Automation - QA Gallery / Interactive Development
using the existing JINR infrastructure

- JupyterHUB
- EOS filesystem
- Sets of QA plots automatically displayed
- Custom code injection
- Cell structure with reprocess functionality
- Improvements integrated into
  main C++ codebase



```
}
Warning in <TF2::GetRandom2>: function:f2 has 24 negative values: abs assumed

[3]: f2.SetParameters(0.7, 1.5);  // set initial values for fit
     f2.SetTitle("Fitted 2D function");
     dte.Fit(&f2);

 FCN=517.445 FROM MIGRAD    STATUS=CONVERGED      38 CALLS          39 TOTAL
                     EDM=1.60013e-12    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           6.81725e-01   4.35962e-01   2.40425e-05   8.10655e-04
   2  p1           1.46084e+00   9.34203e-01   5.15197e-05   3.77919e-04

[4]: TCanvas c1;
     f2.SetLineWidth(1);
     f2.SetLineColor(kBlue - 5);
     f2.Draw("Surf1");
     auto Xaxis = f2.GetXaxis();
     auto Yaxis = f2.GetYaxis();
     auto Zaxis = f2.GetZaxis();
     Xaxis->SetTitle("X Title"); Xaxis->SetTitleOffset(1.5);
     Yaxis->SetTitle("Y Title"); Yaxis->SetTitleOffset(1.5);
     Zaxis->SetTitle("Z Title"); Zaxis->SetTitleOffset(1.5);
     dte.Draw("P0 Same");

[5]: c1.Draw();

[6]: gStyle->SetPalette(kBird);
     TCanvas c_p("ProjCan", "The Projections", 1000, 400);
     c_p.Divide(2,1);
     c_p.cd(1);
     dte.Project("x")->Draw();
     c_p.cd(2);
     dte.Project("y")->Draw();

[7]: %jsroot on
     c_p.Draw();
```

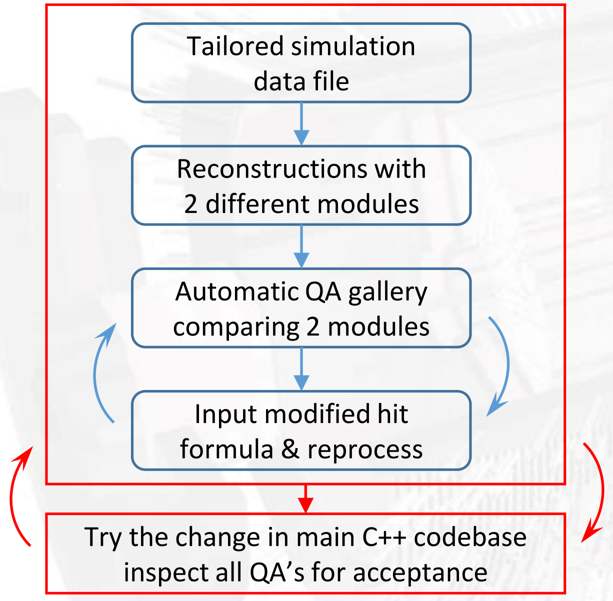
Graph2D_x	
Entries	240081.7
Mean	-0.03267
Std Dev	2.028

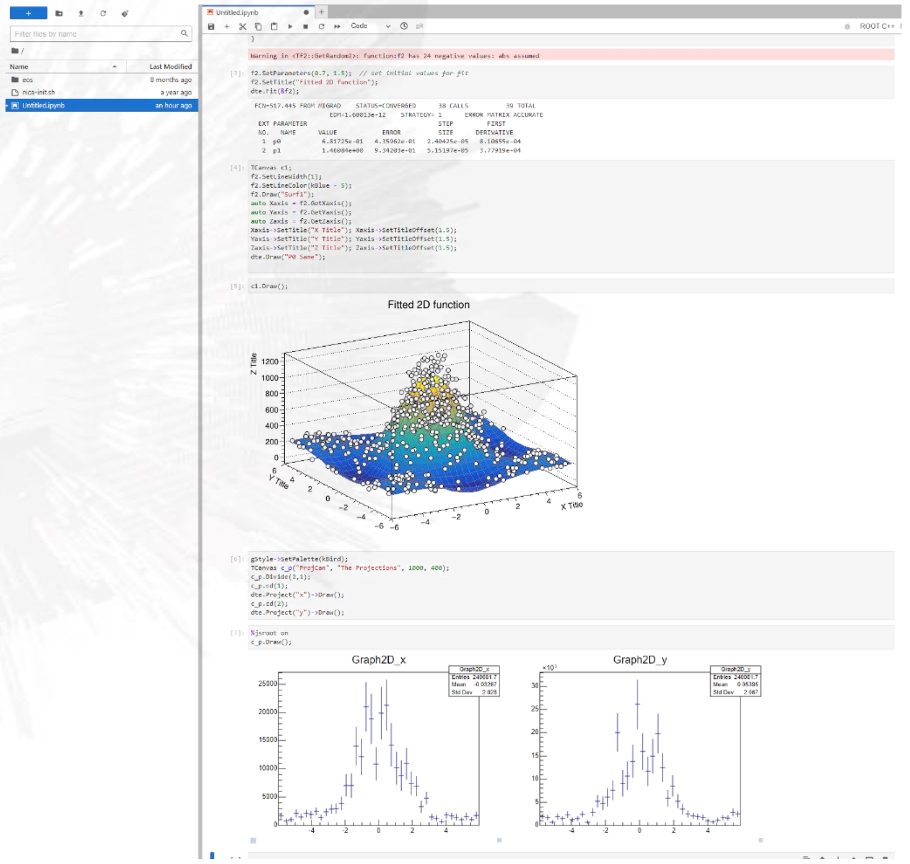
Graph2D_y	
Entries	240081.7
Mean	0.05385
Std Dev	2.067

PERSPECTIVES, FUTURE PLAN

ENVIRONMENT for ALGORITHM IMPROVEMENT

Interactive Development Workflow Example


Tailored simulation data file
Reconstructions with 2 different modules
Automatic QA gallery comparing 2 modules
Input modified hit formula & reprocess
Try the change in main C++ codebase inspect all QA's for acceptance



```
Warning in <TF2::GetRandom2>: function:f2 has 24 negative values: abs assumed

[3]: f2.SetParameters(0.7, 1.5);  // set initial values for fit
     f2.SetTitle("Fitted 2D function");
     dte.Fit(&f2);

 FCN=517.445 FROM MIGRAD    STATUS=CONVERGED      38 CALLS          39 TOTAL
                     EDM=1.60013e-12    STRATEGY= 1      ERROR MATRIX ACCURATE
  EXT PARAMETER                                   STEP         FIRST
  NO.   NAME      VALUE            ERROR          SIZE      DERIVATIVE
   1  p0           6.81725e-01   4.35962e-01   2.40425e-05   8.10655e-04
   2  p1           1.46084e+00   9.34203e-01   5.15197e-05   3.77919e-04

[4]: TCanvas c1;
     f2.SetLineWidth(1);
     f2.SetLineColor(kBlue - 5);
     f2.Draw("Surf1");
     auto Xaxis = f2.GetXaxis();
     auto Yaxis = f2.GetYaxis();
     auto Zaxis = f2.GetZaxis();
     Xaxis->SetTitle("X Title"); Xaxis->SetTitleOffset(1.5);
     Yaxis->SetTitle("Y Title"); Yaxis->SetTitleOffset(1.5);
     Zaxis->SetTitle("Z Title"); Zaxis->SetTitleOffset(1.5);
     dte.Draw("P0 Same");

[5]: c1.Draw();

[6]: gStyle->SetPalette(kBird);
     TCanvas c_p("ProjCan", "The Projections", 1000, 400);
     c_p.Divide(2,1);
     c_p.cd(1);
     dte.Project("x")->Draw();
     c_p.cd(2);
     dte.Project("y")->Draw();

[7]: %jsroot on
     c_p.Draw();
```

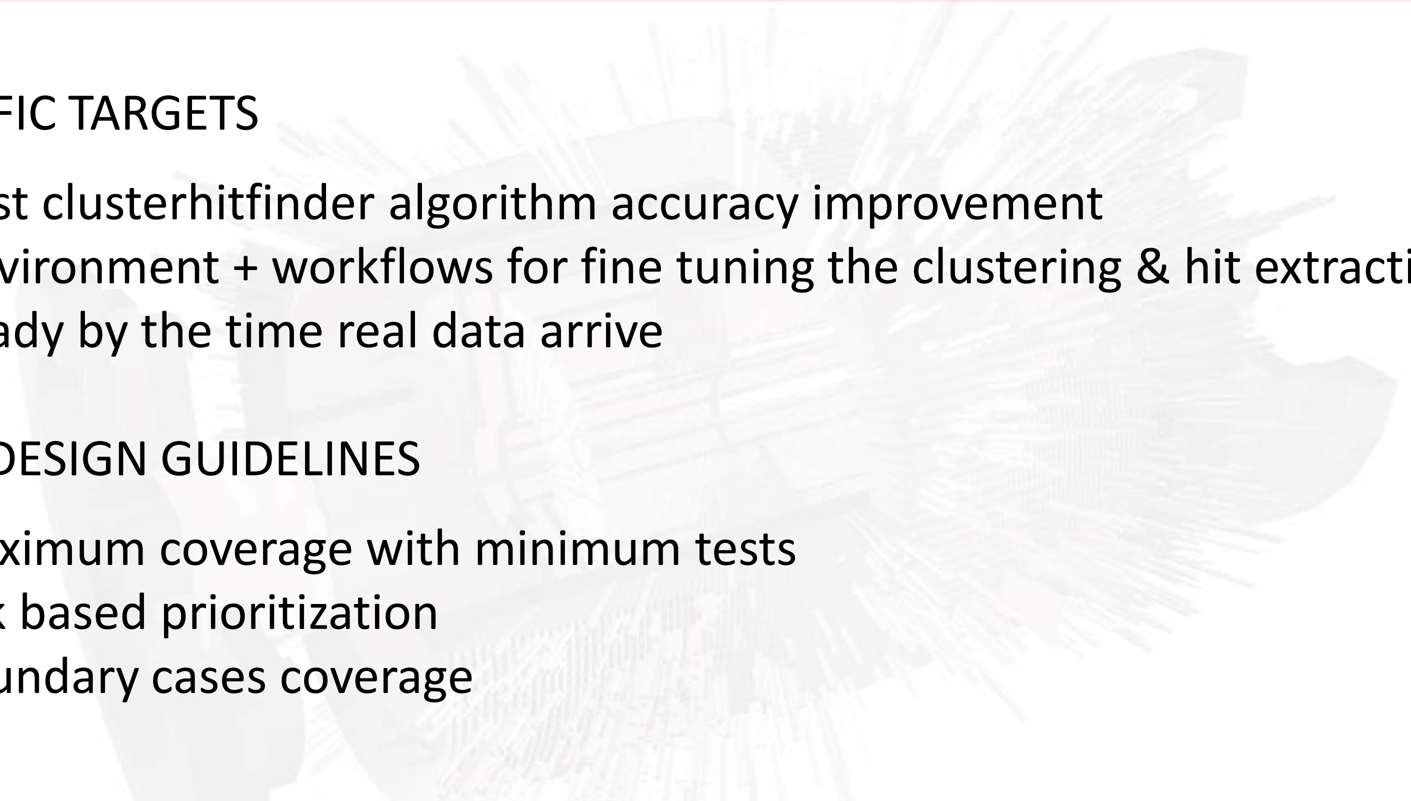
Graph2D_x	
Entries	240081.7
Mean	-0.03267
Std Dev	2.028

Graph2D_y	
Entries	240081.7
Mean	0.05395
Std Dev	2.067

MAJOR BENEFIT

On arrival of the data from real experiment, the optimized algorithm improvement workflow with required infrastructure/environment is in place

FINAL REMARKS



SPECIFIC TARGETS

- Fast clusterhitfinder algorithm accuracy improvement
- Environment + workflows for fine tuning the clustering & hit extraction ready by the time real data arrive

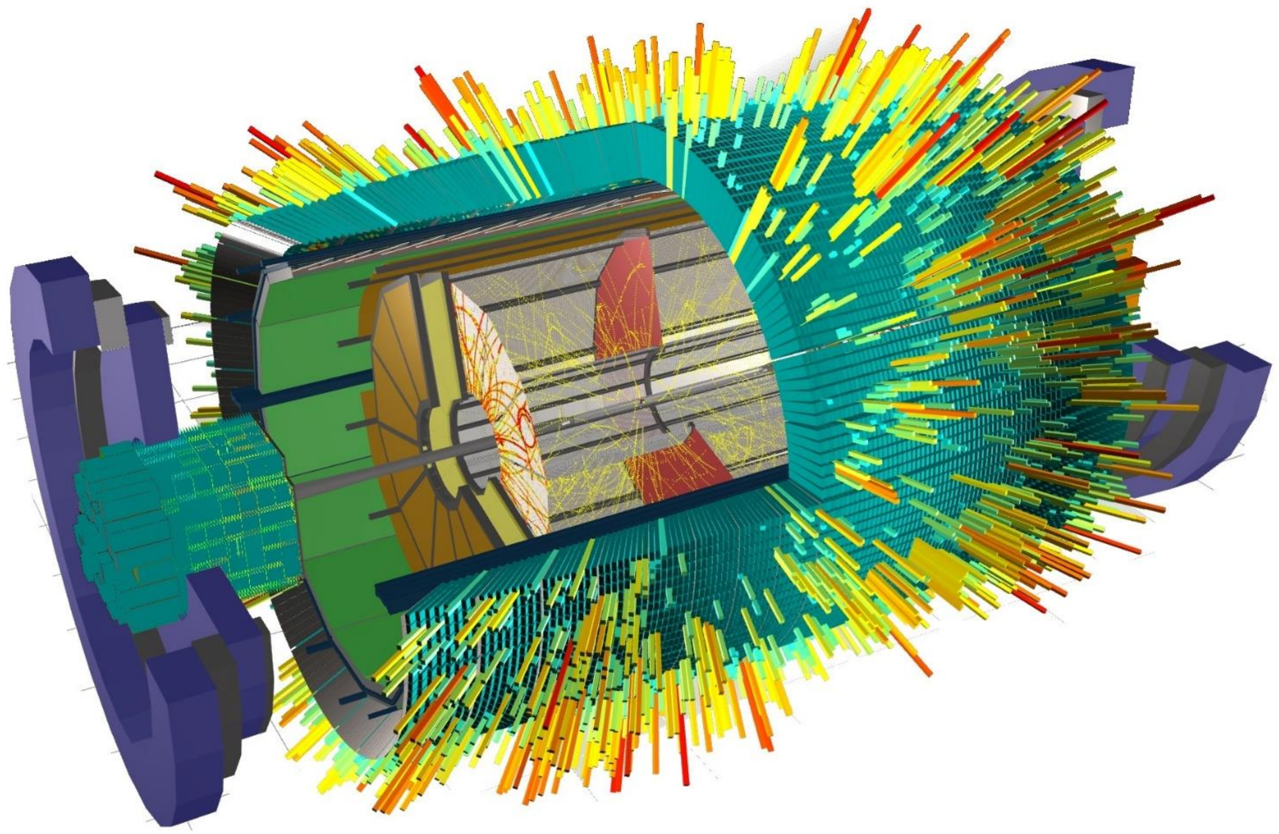
TEST DESIGN GUIDELINES

- maximum coverage with minimum tests
- risk based prioritization
- boundary cases coverage

Test environment is effective when absolute majority of defects is caught by developers, not by users.

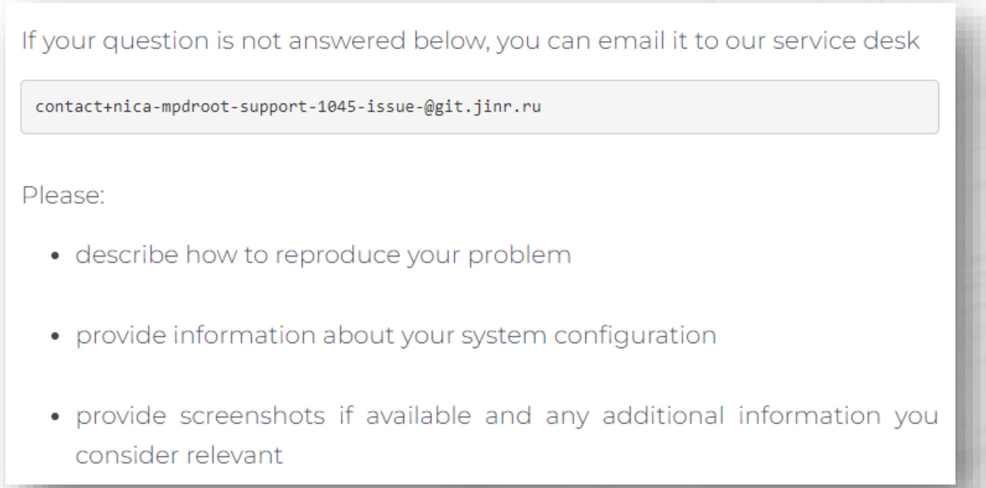
Thank You !

Q & A



SERVICE DESK for Questions

http://mpdroot.jinr.ru/q-a/

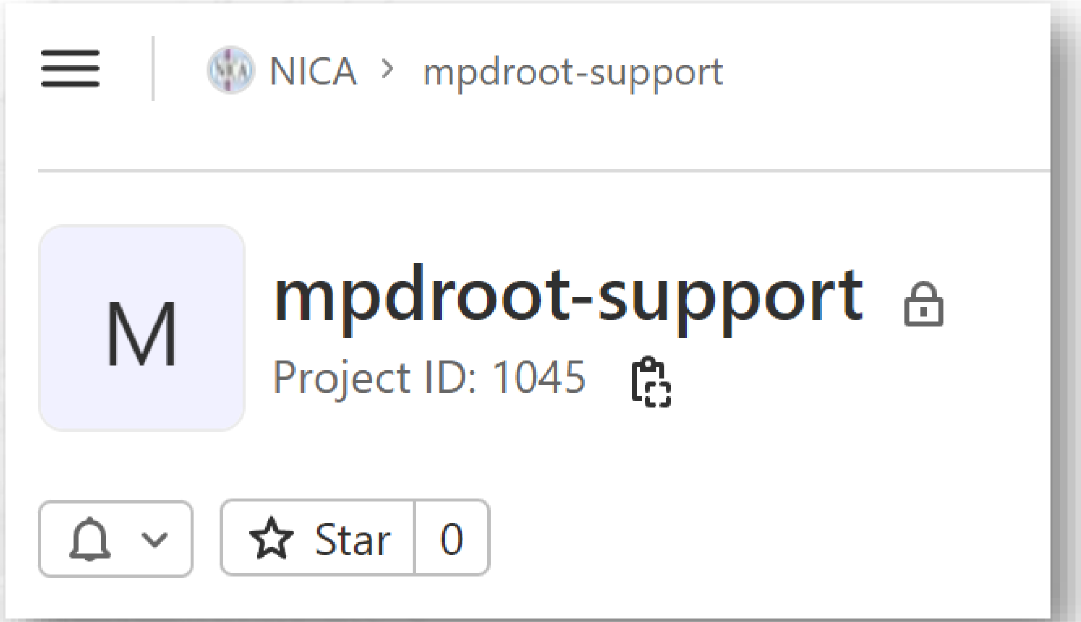


If your question is not answered below, you can email it to our service desk

```
contact+nica-mpdroot-support-1045-issue-@git.jinr.ru
```

Please:

- describe how to reproduce your problem
- provide information about your system configuration
- provide screenshots if available and any additional information you consider relevant



NICA › mpdroot-support

mpdroot-support

Project ID: 1045

“User Involvement – **critical** project success factor”
CHAOS Report 2015, Standish Group