C++ Software Quality in the ATLAS Experiment

Stefan Kluth, Stewart Martin-Haugh, Emil Obreshkov, *Shaun Roe*, Rolf Seuster, Scott Snyder, **Graeme Stewart**

Tools in use

Experience

Progress, Pleasures and Pitfalls

TOOLS

Gcc-plugin Iwyu tctoolkit PRQA

Coverity

Cppcheck

Clang Ubsan

Compilation * Static Analysis * run-time behaviour
continuous testing

ATLAS code

2

6

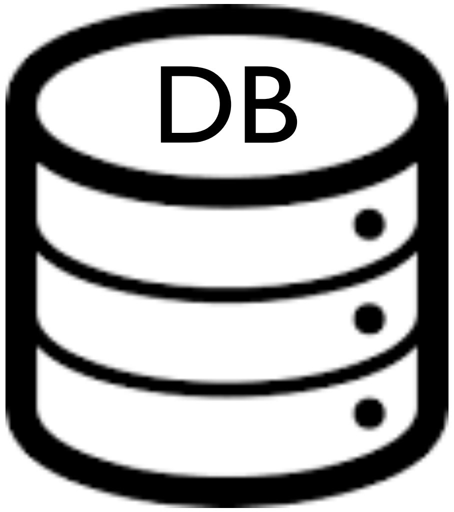
0

10

loc (millions)

~420

developers



correspondence →

~140 'domains'

e.g. Tracking-Fitting

2400 individual packages

(gcc 4.9)

Compiler checks

Code should always, where possible, compile without warnings.
It is beneficial to expose the code to different compilers; e.g. **Clang builds** give additional info:

- Mismatched struct/class; Unused private class members; Mismatched header guard.
- `std::abs(eta < 2.5)`
- `if ((bitfield && 0x0011) != 0)` ...
- `!val==10`

gcc plug-ins

- Check for inheritance structure already in place
- Coming soon: naming convention checks

Static Analysis

Attempts to follow your program and points out possible errors; some examples:

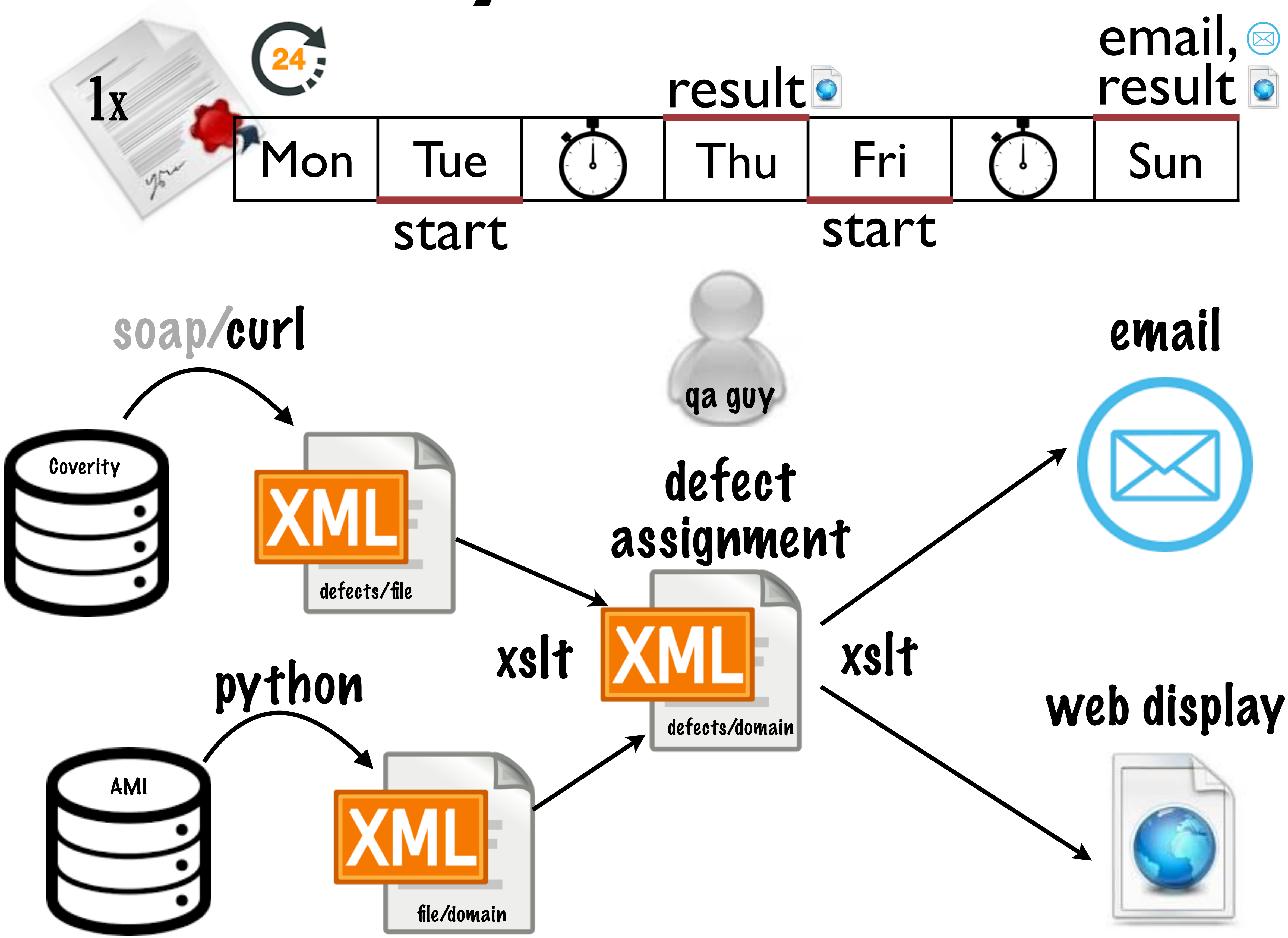
- ‘new’ without delete
- scalar ‘delete’ on array object
- array indices out-of-bounds
- possible cut-and-paste errors
- suspect indentation

```
if (m_minNUsedHitsdEdx > 0) {
  original: this->m_minNUsedHitsdEdx looks like the original copy.
  ATH_MSG_INFO( "  Minimum used hits for dEdx: " << m_minNUsedHitsdEdx );
  auto dEdx = std::make_shared<MinUsedHitsdEdxCut>(this, m_minNUsedHitsdEdx);
  ATH_CHECK( dEdx->initialize() );
  m_trackCuts["dEdxHits"].push_back(dEdx);
}
if (m_minNOverflowHitsdEdx > 0) {
  CID 28963 (#1 of 1): Copy-paste error (COPY_PASTE_ERROR)
  copy_paste_error: m_minNUsedHitsdEdx in this->m_minNUsedHitsdEdx looks like a copy-paste error.
  Should it say m_minNOverflowHitsdEdx instead?
  ATH_MSG_INFO( "  Minimum IBL overflow hits for dEdx: " << m_minNUsedHitsdEdx );
```

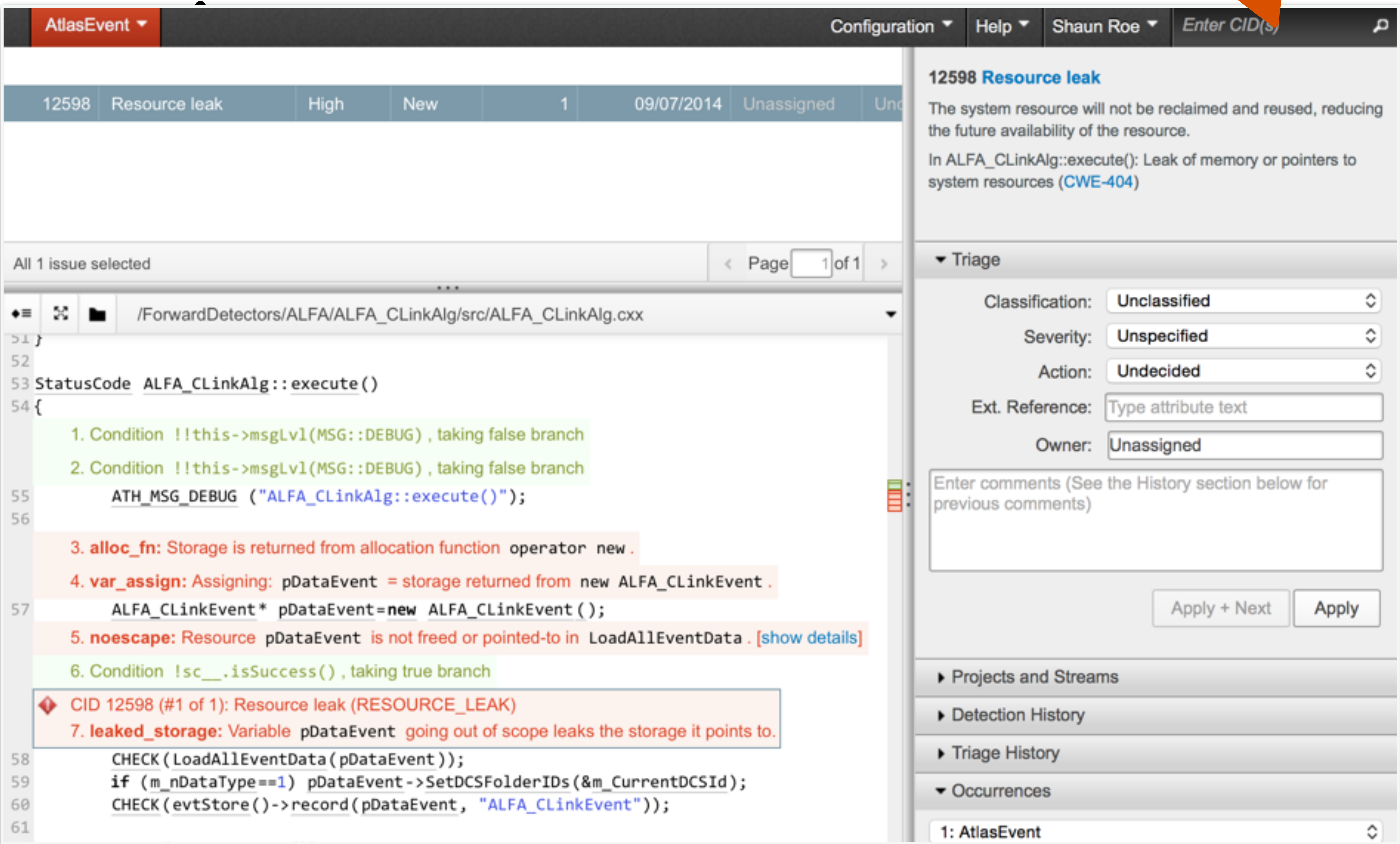
```
  parent: This 'if' statement is the parent, indented to column 3.
if (m_fullMaterial)
  nephew: This statement is nested within its parent, indented to column 8.
       sl << "    - fullMaterial          : " << *m_fullMaterial << std::endl;
  CID 11566 (#1 of 1): Nesting level does not match indentation (NESTING_INDENT_MISMATCH)
  uncle: This statement is indented to column 8, as if it were nested within the preceding parent statement, but it is not.
       sl << "    - split factor          : " << m_splitFactor << std::endl;
return sl;
```

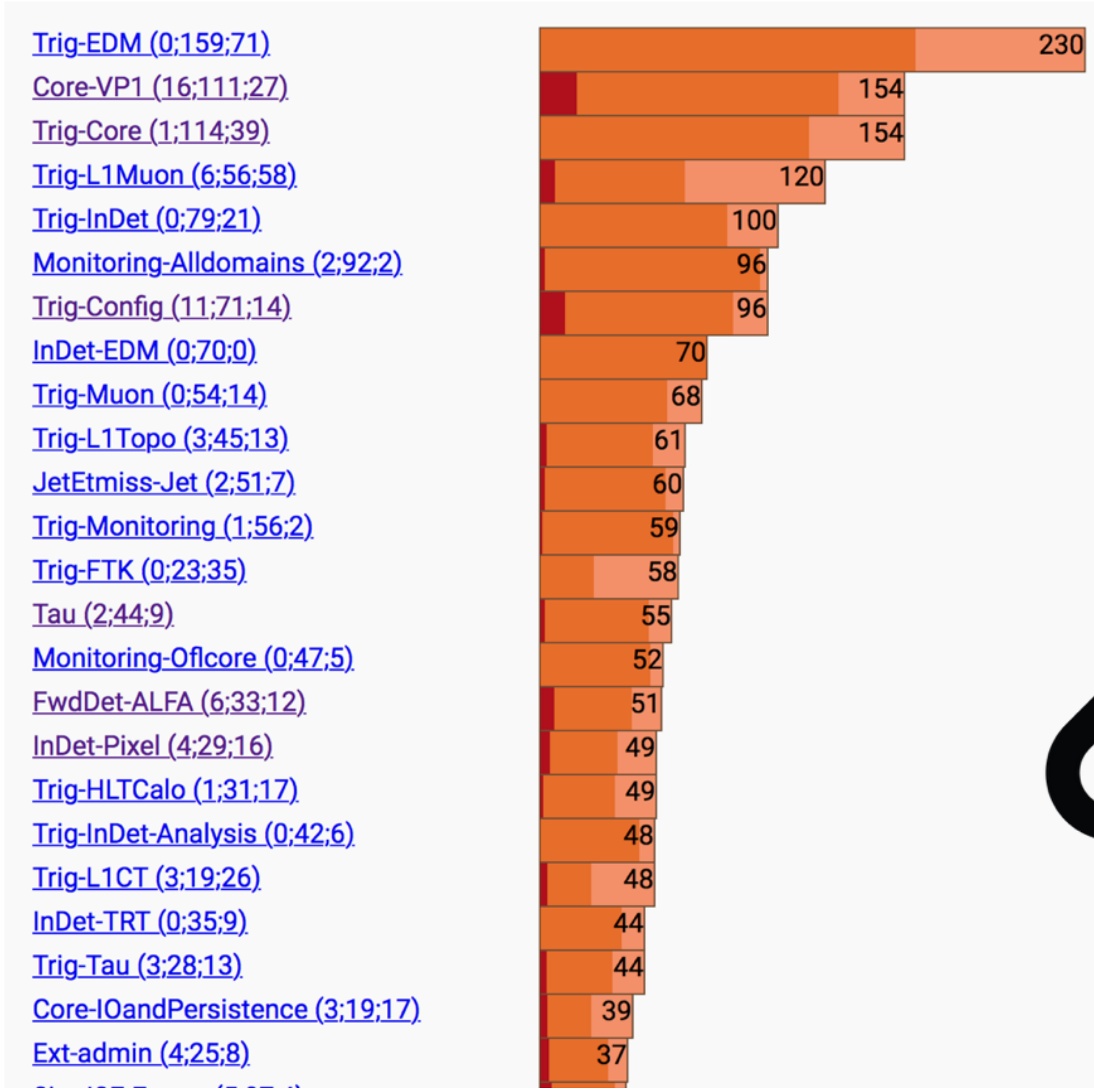
The examples are drawn from Coverity®, one of the most widely used static analysers.

Coverity®


1x
Mon
Tue
Thu
Fri
Sun
result
email,
result
start
start
soap/curl
Coverity
defects/file
qa guy
defect
assignment
xslt
xslt
defects/domain
email
web display
python
AMI
file/domain

- 12598 09/07/2014 (High) Resource leak :/ ForwardDetectors/ALFA/ALFA_CLinkAlg/src/ALFA_CLinkAlg.cxx in function "execute"

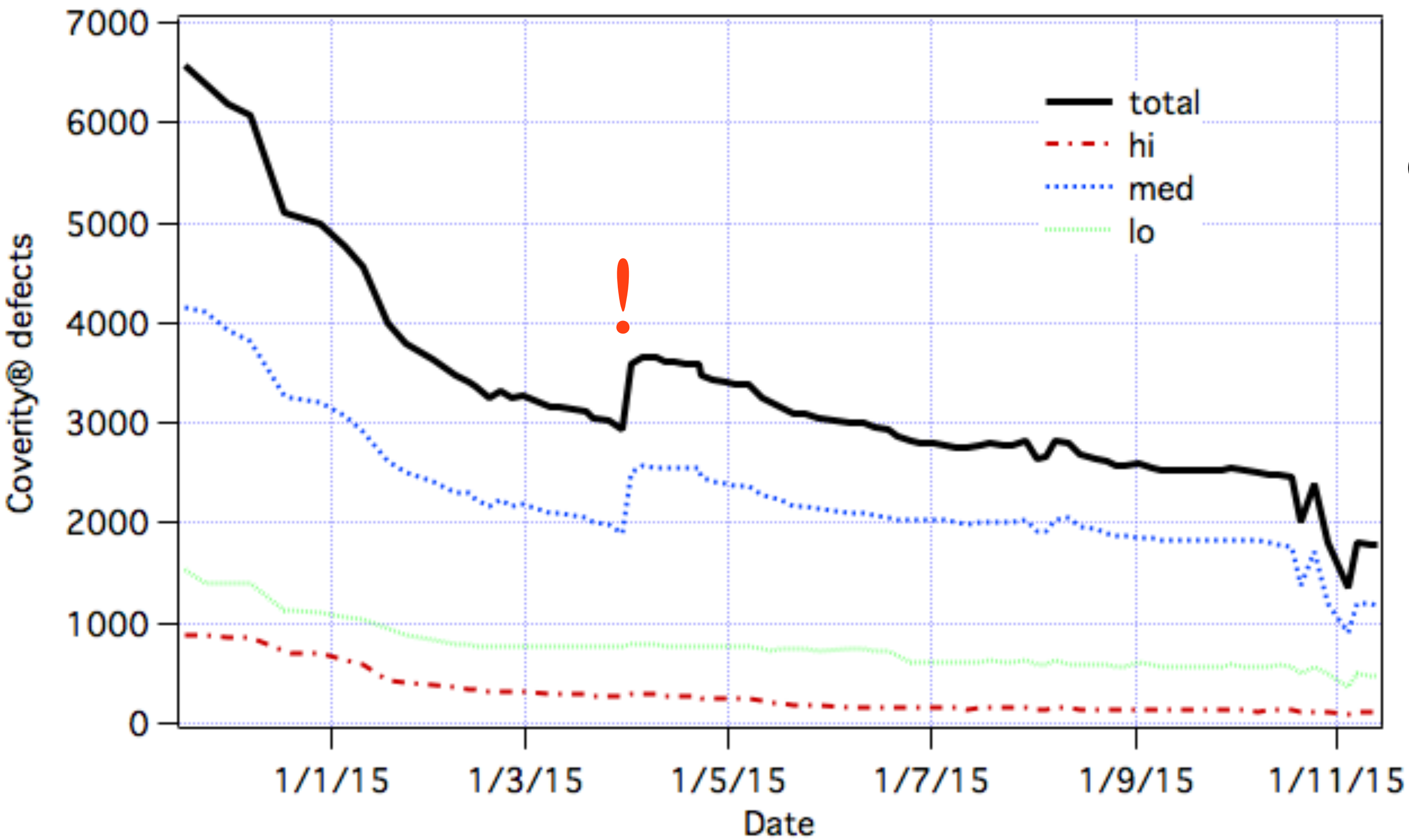


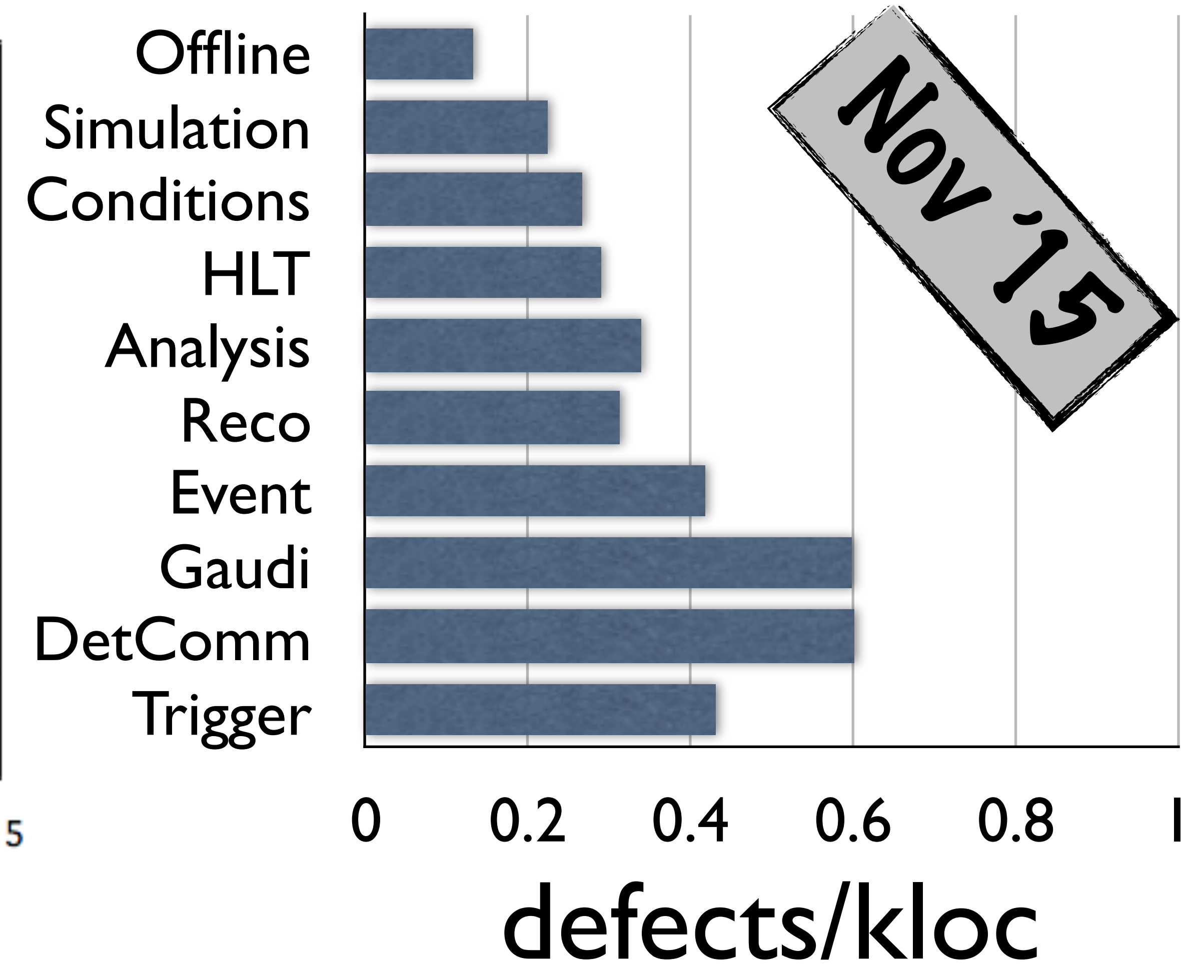


A command line tool, ‘`issues`’ will query the web page and list issues for a specified file

Pitfalls: Coverity’s classification may not correspond to your own; e.g. Uninitialised members are classified ‘high’, as are obvious resource leaks; however a faulty assignment operator which can also leak resources will be ‘low’. Parse errors can occur and are classified ‘low’ but may mask many other defects.

Coverity® progress

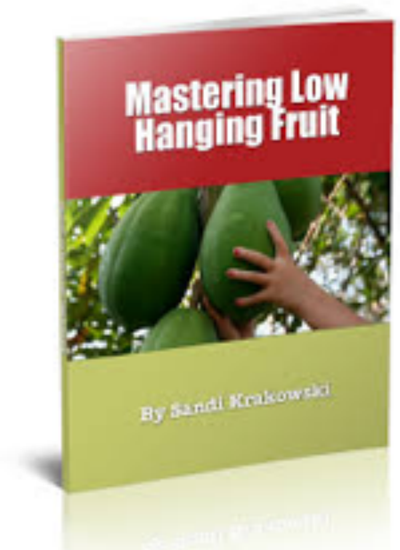




Industry report 2012

"Coverity's analysis found an average defect density of .69 for open source software projects that leverage the Coverity Scan service, and an average defect density of .68 for proprietary code developed by Coverity enterprise customers. Both have better quality as compared to the **accepted industry standard defect density for good quality software of 1.0** [defects/kloc]."

Coverity® pleasures and pitfalls



"spending hours with a memory profiler could save you 5 minutes looking at the coverity report" ☺

False positives (*very few*): ☹
e.g. "restoring ostream format"
"parse errors" (particularly complicated templates)

Near misses:

```
    //set the foreign key
6. return_constant: Function call variableType() returns 4.
CID 11595 (#2-1 of 2): Out-of-bounds read (OVERRUN)
7. overrun-local: Overrunning array of 4 bytes at byte offset 4 by dereferencing pointer &"_FK"[variableType()].
    pixel_columns.createForeignKey(variableType<T>()+"_FK","FK",m_pixeltable,"FK");
    // create indices
```

Misses: `if (m_Analogue[1][strip]<0.0000001 || m_Analogue[1][strip]>-0.0000001){ //`

Other Static Analysers

Installed, to be integrated into reporting system:

cppcheck (running routinely since Nov '15)

- Open source, easily configurable, quick (~1 hr)
- somewhat noisy, more false positives, but *does find additional defects*. Output to XML/static web page. Options for performance (e.g. "use pre-increment"), style.

Include-what-you-use

- Easy to set up; scope is limited to tidying up #includes; web interface available for reports (R. Seuster).

OCLint

- Open source, based on Clang static analysis tools; ~12 hr scan with integrated result-to-web generation.

Investigated:

- **PRQA**: Nice system, flexible, industry standards, can enforce naming conventions; $$, difficult to integrate in our build system

cppcheck

Top level web display

Cppcheck report - AtlasReconstruction:

Defect summary;

440 total

140 memleak
62 literalWithCharPtrCompare
51 uninitMemberVar
42 nullPointer
33 uninitStructMember
25 invalidscanf_libc
17 deallocuse
11 oppositeInnerCondition
10 derefInvalidIterator
9 eraseDereference
8 zerodiv
7 mismatchAllocDealloc
4 doubleFree
4 invalidPrintfArgType_sint
3 bufferAccessOutOfBounds
2 StlMissingComparison
2 compareBoolExpressionWithInt
2 uninitvar
1 arrayIndexOutOfBoundsCond

Line	Id	Severity	Message
	missingInclude	information	Cppcheck cannot find all the include files (use --check-config for details)
AtlasReconstruction/rel_2/Calorimeter/CaloCalibHitRec/src/CaloDmNeighbours.cxx			
399	coutCerrMisusage	error	Invalid usage of output stream: '<< std::cout'.
AtlasReconstruction/rel_2/InnerDetector/InDetAlignment/InDetAlignNtupleTools/src/DetailedIDNtupleTool.cxx			
258	nullPointer	warning	Possible null pointer dereference: vot - otherwise it is redundant to check it against null.
AtlasReconstruction/rel_2/InnerDetector/InDetCalibAlgs/PixelCalibAlgs/src/PixelBarrelSurvey.cxx			
161	mismatchAllocDealloc	error	Mismatching allocation and deallocation: disto
AtlasReconstruction/rel_2/InnerDetector/InDetCalibAlgs/PixelCalibAlgs/src/PixelChargeInterpolationValidation.cxx			
586	mismatchAllocDealloc	error	Mismatching allocation and deallocation: values
AtlasReconstruction/rel_2/InnerDetector/InDetCalibAlgs/TRT_CalibAlgs/share/CalibrateTRT.cpp			
75	uninitMemberVar	warning	Member variable 'Calibrator::resHist' is not initialized in the constructor.
493	memleak	error	Memory leak: hist
650	invalidscanf_libc	portability	scanf without field width limits can crash with huge input data on some versions of libc.
706	invalidscanf_libc	portability	scanf without field width limits can crash with huge input data on some versions of libc.
707	invalidscanf_libc	portability	scanf without field width limits can crash with huge input data on some versions of libc.
708	invalidscanf_libc	portability	scanf without field width limits can crash with huge input data on some versions of libc.
898	uninitStructMember	error	Uninitialized struct member: startdata.det

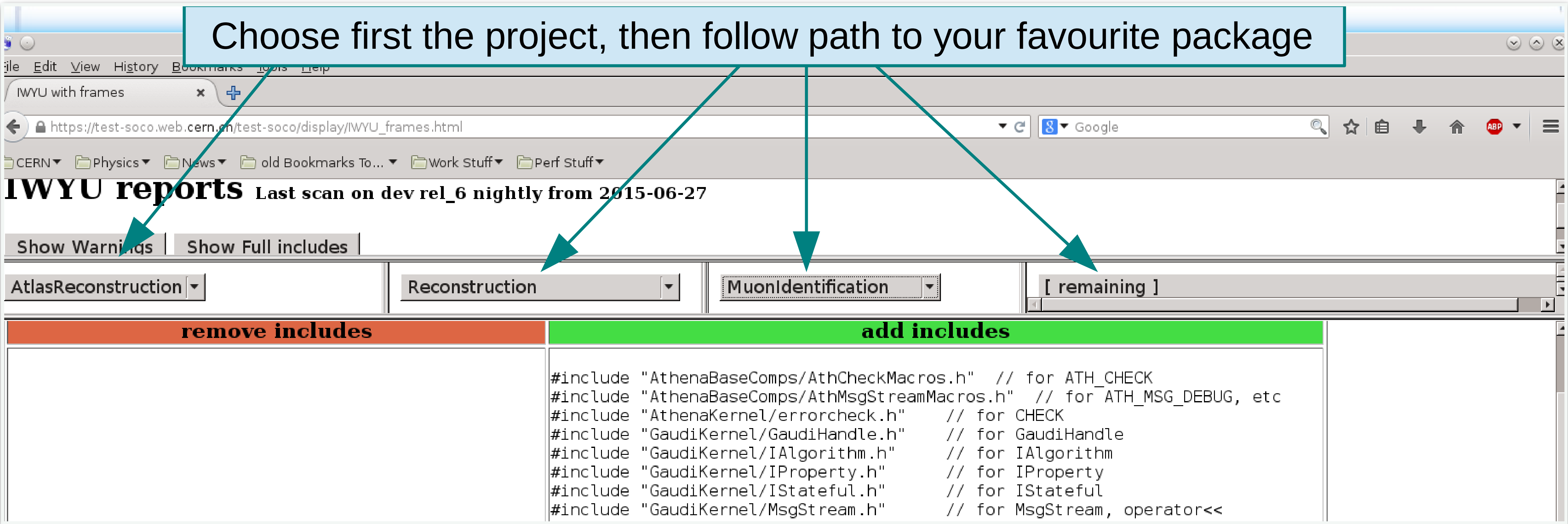
..linked to annotated code

Defects: makeplots.cpp

eralWithCharPtrCompare
271
eralWithCharPtrCompare
306

```
this->cd(1);
Moduletuple->Draw("y:x>>reshist10(40,-1200,1200,40,-1200,1200)",selectionA,"colz");
TH2F *reshist10 = (TH2F*)gPad->GetPrimitive("reshist10");
if (!reshist10) throw string("Variable not found!");
if (variable == "res"){ <--- String literal compared with variable 'variable'. Did you intend to use strcmp() instead?
  reshist10->GetZaxis()->SetRangeUser(0.135, 0.150);
```

Include-what-you-use



…advises on which ‘includes’ to add, remove or can be replaced by a forward class declaration.

Run-time Sanitizers

Implemented in nightly tests of DBG builds:

- **UBSan** (undefined behaviour)

Clear warnings; easy to implement :

```
HepMcParticleLink.h:72:51: runtime error: left shift of 200001 by 16 places cannot be represented in type 'int'
```

Investigated:

- **ASan** (address)

Similar to the debug tool Valgrind but much faster and catches more; output maybe a bit intimidating...

ABI incompatible.

```
==7552==ERROR: AddressSanitizer: heap-buffer-overflow on address 0x61400000ffd4 at pc 0x400825 bp 0x7fff248f7d80 sp 0x7fff248f7d78...
```

- **TSan** (thread)

Currently not useful 'out of the box' for ATLAS

"If all you have is a hammer, everything looks like a nail"
- Psychology of Science (Maslow, 1966)

Coding to solve the line-by-line problems revealed by these tools *might* produce better code, but it won't forcibly produce **good** code.

"Code smell" detectors (tentative results so far):

- **TCToolkit**

~1 hr to scan release, produces >50Mb html.

- code duplication detector (looks useful)
- token tag cloud ("how noisy" the code is)
- class co-occurence matrix (interdependencies)

Continuous testing

```xml
<?xml version="1.0"?>
<atn>
   <TEST name="ISF_EventTests" type="makecheck">
      <package>Simulation/ISF/ISF_Core/ISF_Event</package>
      <author> Elmar Ritsch </author>
      <mailto> elmar.ritsch@cern.ch  </mailto>
      <expectations>
         <errorMessage>Athena exited abnormally</errorMessage>
         <errorMessage>differ</errorMessage>
         <warningMessage> # WARNING_MESSAGE : post.sh> ERROR</warningMessage>
         <successMessage>check ok</successMessage>
         <returnValue>0</returnValue>
      </expectations>
   </TEST>
</atn>
```

'ATN' tests run nightly in all releases;

- Unit tests configured in a dedicated SVN test directory for the package
- XML file describes expected output; maybe simple return code, or comparison with a reference file.

'RTT' tests

- Produce plots of physics quantities (momenta, track parameters etc) and compared with reference plots
- Reviewed by shifters, who report each week.

Summary

- All of these tools are useful as developer aids;
  - Coverity in particular reveals many programming errors and maintainability issues

- Public league tables help motivate groups.
  - None of these will replace peer review or make up for a lack of education (another important topic) or experience.
  - ATLAS will continue to use these tools, integrating the reports from different analyses, and keep an eye on new tools as they become