

Experience with C++ Code Quality in ATLAS

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

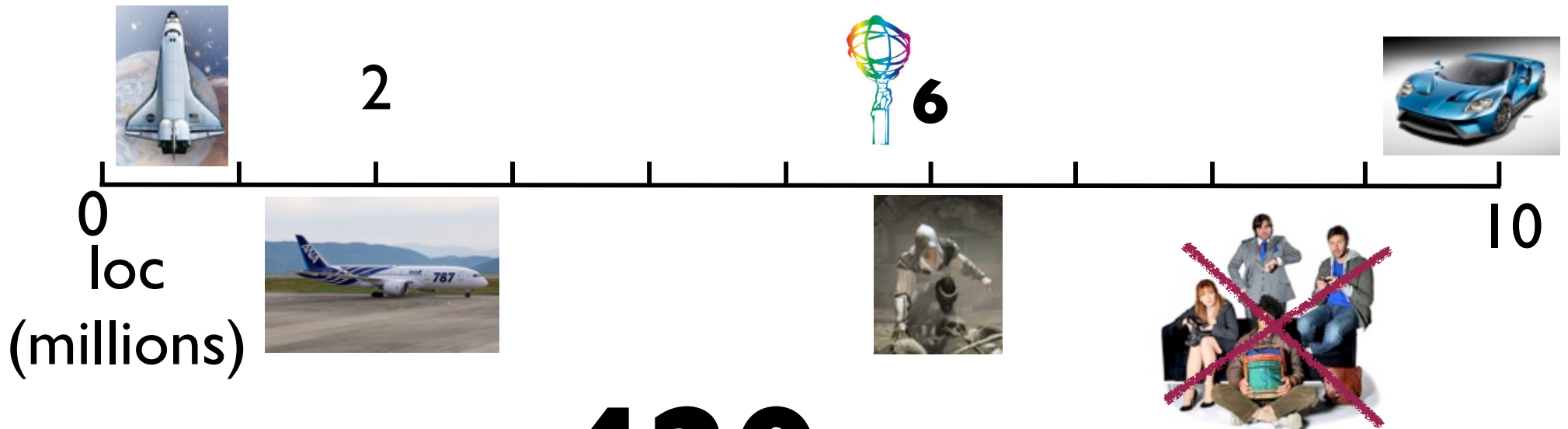
Tools in use
Experience
Progress, Pleasures and Pitfalls,

TOOLS

Gcc-plugin tctoolkit
lwyu PRQA
Coverity
Cppcheck
Clang Ubsan

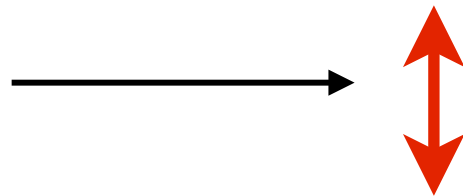
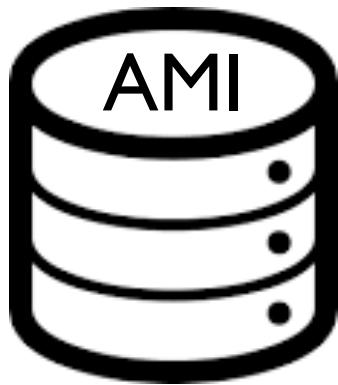
Compilation * Static Analysis * run-time behaviour

Atlas code



~420

developers



~140 'domains'
e.g. Tracking-Fitting



(gcc 4.9)

Compiler checks

Code should always, where possible, compile without warnings.

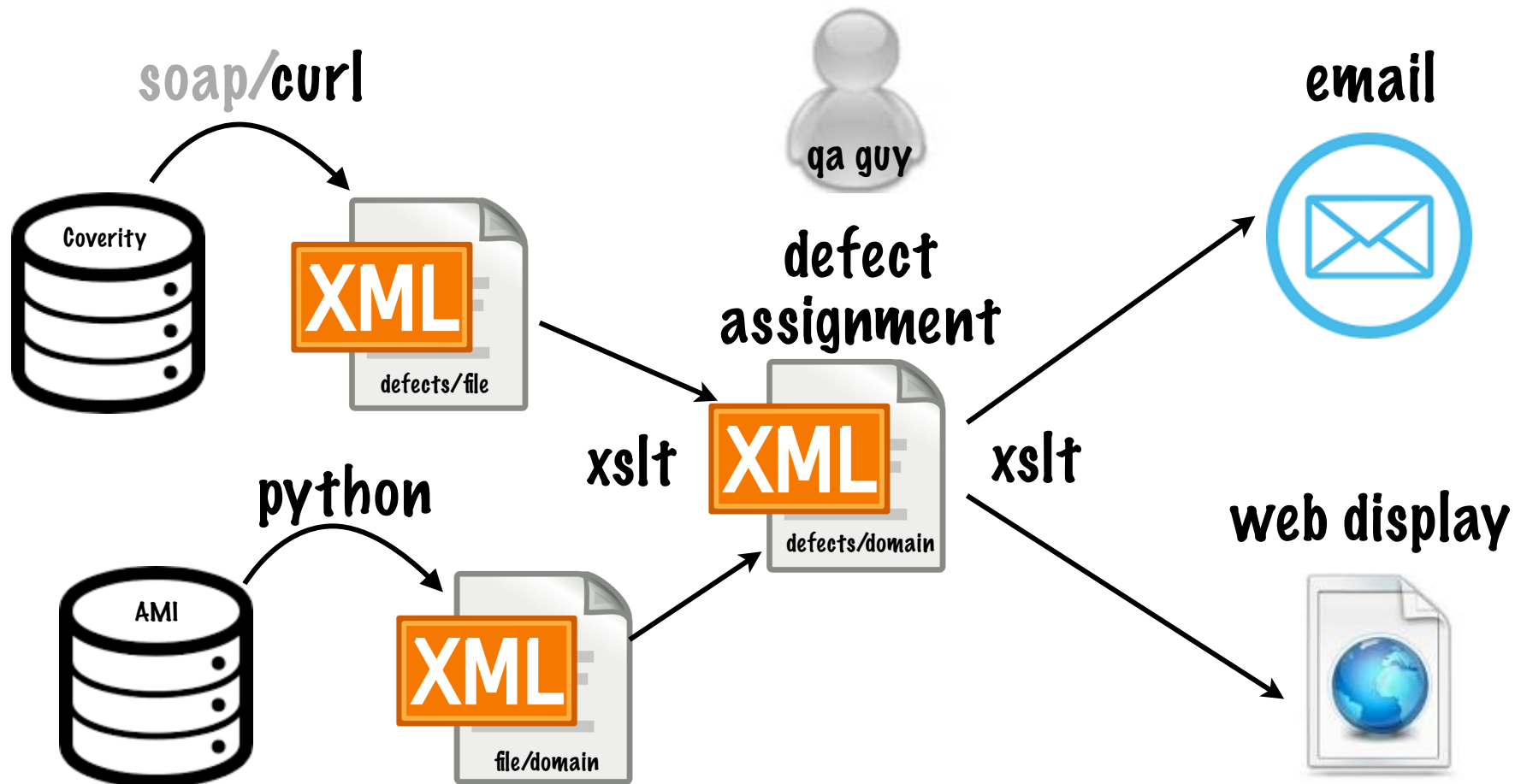
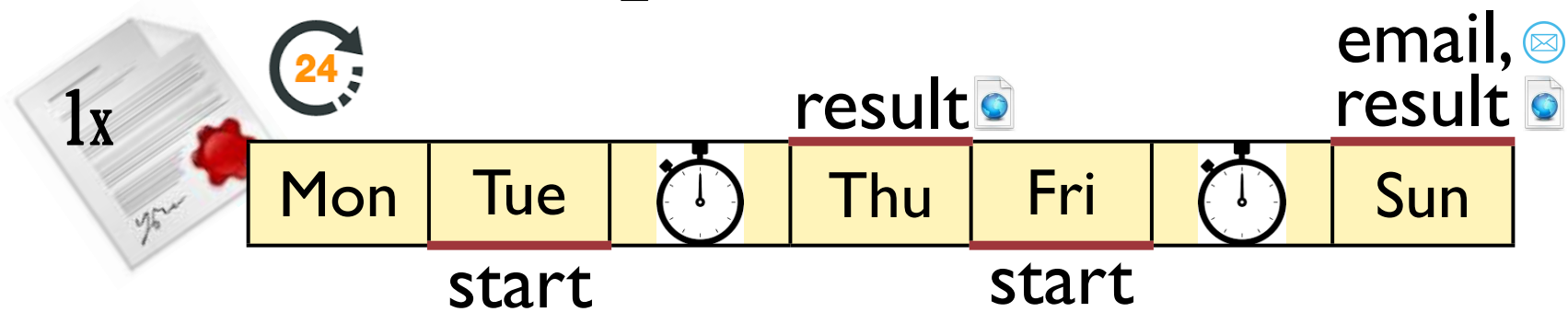
It is beneficial to expose the code to different compilers; e.g. **Clang builds** give additional info:

- 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
- Considering: naming convention checks

Coverity®





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



coverity AtlasEvent Configuration Help Shaun Roe Enter CID(s)

DASHBOARDS
Quality Advisor
Security Advisor
Test Advisor

ISSUES: BY SNAPSHOT
High Impact Outstanding
My Outstanding
Outstanding Defects
Outstanding Security ...
Outstanding Test Rule...
Outstanding Untriaged

ISSUES: PROJECT SCOPE
All In Project
Recently Viewed

FILES
In Latest Snapshot
Uncovered By Tests

FUNCTIONS
High CCM (>15)
In Latest Snapshot
Uncovered By Tests

COMPONENTS
All In Project
High Issue Density (>1)
With Outstanding Issues
With Untriaged Issues

CHECKERS
All In Project

OWNERS
All In Project

SNAPSHOTS

12598 Resource leak High New 1 09/07/2014 Unassigned Un

All 1 issue selected Page 1 of 1

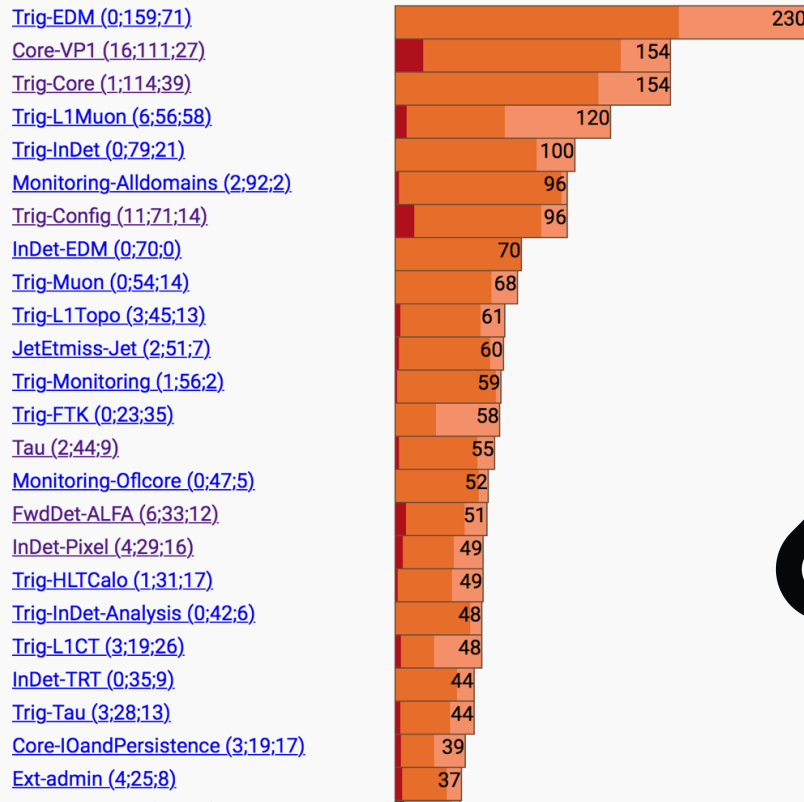
/ForwardDetectors/ALFA/ALFA_CLinkAlg/src/ALFA_CLinkAlg.cxx

```
51 }
52
53 StatusCode ALFA_CLinkAlg::execute()
54 {
55     1. Condition !!this->msgLvl(MSG::DEBUG) , taking false branch
56     2. Condition !!this->msgLvl(MSG::DEBUG) , taking false branch
57     ATH_MSG_DEBUG ("ALFA_CLinkAlg::execute()");
58     3. alloc_fn: Storage is returned from allocation function operator new .
59     4. var_assign: Assigning: pDataEvent = storage returned from new ALFA_CLinkEvent .
60     ALFA_CLinkEvent* pDataEvent=new ALFA_CLinkEvent ();
61     5. noescape: Resource pDataEvent is not freed or pointed-to in LoadAllEventData . [show details]
62     6. Condition !sc_.isSuccess() , taking true branch
63     CID 12598 (#1 of 1): Resource leak (RESOURCE_LEAK)
64     7. leaked_storage: Variable pDataEvent going out of scope leaks the storage it points to.
65     CHECK (LoadAllEventData (pDataEvent));
66     if (m_nDataType==1) pDataEvent->SetDCSFolderIDs (&m_CurrentDCSId);
67     CHECK (evtStore()->record (pDataEvent, "ALFA_CLinkEvent"));
68     CHECK (GenerateXAOD());
69     return StatusCode::SUCCESS;
70 }
71
72 StatusCode ALFA_CLinkAlg::finalize()
73 {
```

12598 Resource leak
The system resource will not be reclaimed and reused, reducing the future availability of the resource.
In ALFA_CLinkAlg::execute(): Leak of memory or pointers to system resources (CWE-404)

Triage
Classification:
Severity:
Action:
Ext. Reference:
Owner:
Enter comments (See the History section below for previous comments)
Apply + Next Apply

Projects and Streams
Detection History
Triage History
Occurrences
1: AtlasEvent
Events contributing to defect:
3 alloc_fn ALFA_CLinkAlg.cxx:57
4 var_assign ALFA_CLinkAlg.cxx:57
5 noescape ALFA_CLinkAlg.cxx:58

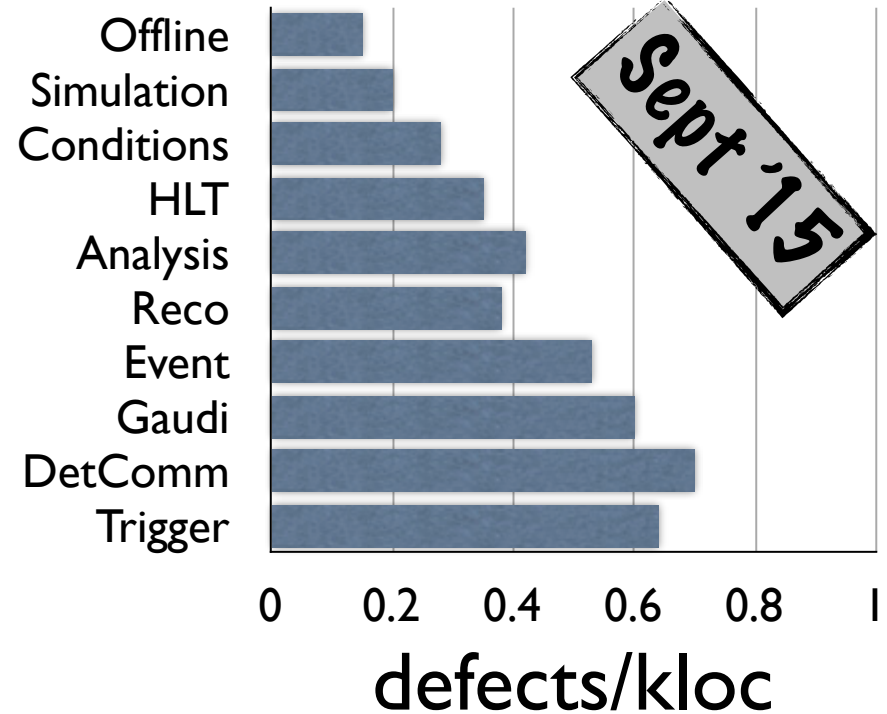
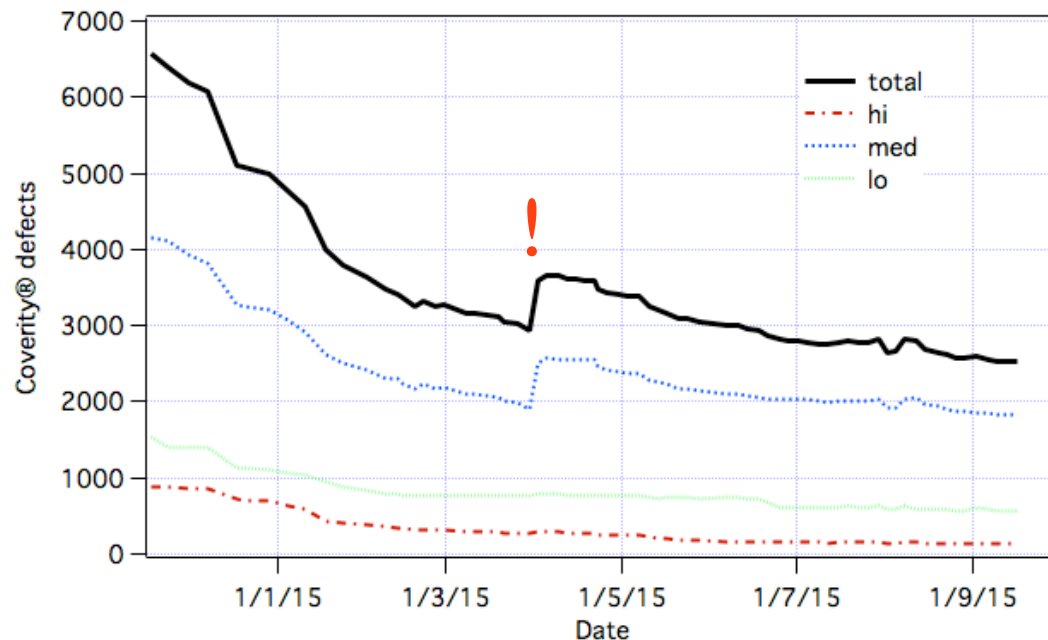


A command line tool, 'issues' will query the web page and list issues for a specified file



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:

```
792 //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()].
793 pixel_columns.createForeignKey(variableType<T>()+ "_FK", "FK", m_pixeltable, "FK");
794 // create indices
```

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

Other Static Analysers

Available, not yet integrated into reporting system:

- **cppcheck**

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).

Investigated:

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

cppcheck

Top level web display

Cppcheck - HTML report - DEV from 2015-05-09 - Iceweasel

File Edit View History Bookmarks Tools Help

Cppcheck - HTML report ... x

seuster.web.cern.ch/seuster/cppcheck/

CERN Physics News old Bookmarks To... Work Stuff Perf Stuff

Cppcheck report - DEV from 2015-05-09:

[Defect summary](#)

16052 total

4692 postfixOperator
4564 variableScope
2787 redundantAssignment
1228 unreadVariable
491 unusedVariable
265 noExplicitConstructor
210 memleak
205 invalidscanf_libc
156 uninitializedMemberVar
139 invalidPrintfArgType_sint
114 catchExceptionByValue
100 nullPointer
79 atSize
63
literalWithCharPtrCompare
56 duplicateBreak
49 passedByValue
41 clarityCondition
41 eraseDereference
41 invalidPrintfArgType_uint
38 assignmentInAssert
38 cstyleCast
37 assertWithSideEffect
33 uninitializedMember
32 duplicateExpression
31 invalidscanf
29 mismatchAllocDealloc
29 unpreciseMathCall
28 selfAssignment
25
duplicateExpressionTernary
24 oppositeInnerCondition
20 clarityCalculation
20 incorrectLogicOperator
16 ignoredReturnValue
16 sprintfOverlappingData
16 uninitvar
16 unreachableCode

| Line | Id | Severity | Message |
|------|---------------------------|-------------|---|
| 159 | invalidscanf_libc | portability | scanf without field width limits can crash with huge input data on some versions of libc. |
| 237 | invalidPrintfArgType_sint | warning | %d in format string (no. 1) requires 'int' but the argument type is 'unsigned int'. |
| 441 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 469 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 2612 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 2698 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 2719 | multiCondition | style | Expression is always false because 'else if' condition matches previous condition at line 2718. |
| 2730 | multiCondition | style | Expression is always false because 'else if' condition matches previous condition at line 2729. |
| 2767 | variableScope | style | The scope of the variable 'beamBackgroundData' can be reduced. |
| 2788 | variableScope | style | The scope of the variable 'isPndmEvent' can be reduced. |
| 2790 | variableScope | style | The scope of the variable 'isCaloEvent' can be reduced. |
| 2791 | variableScope | style | The scope of the variable 'isMinBiasEvent' can be reduced. |
| 2792 | variableScope | style | The scope of the variable 'isMiscEvent' can be reduced. |
| 2793 | variableScope | style | The scope of the variable 'isMetEvent' can be reduced. |
| 2993 | variableScope | style | The scope of the variable 'known' can be reduced. |
| 3671 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 3780 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 4354 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 4491 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 4582 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 4708 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 5397 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 5399 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 5411 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 5619 | sprintfOverlappingData | error | Undefined behavior: Variable 'uniqueName' is used as parameter and destination in s(n)printf(). |
| 5743 | sprintfOverlappingData | error | Undefined behavior: Variable 'uniqueName' is used as parameter and destination in s(n)printf(). |
| 160 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 161 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 175 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 179 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 781 | multiCondition | style | Expression is always false because 'else if' condition matches previous condition at line 780. |
| 791 | multiCondition | style | Expression is always false because 'else if' condition matches previous condition at line 790. |
| 134 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 135 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 648 | redundantAssignment | performance | Variable 'sc' is reassigned a value before the old one has been used. |
| 32 | postfixOperator | performance | Prefer prefix ++/- operators for non-primitive types. |
| 44 | postfixOperator | performance | Prefer prefix ++/- operators for non-primitive types. |
| 188 | unreadVariable | style | Variable 'TheTrigger' is assigned a value that is never used. |

Annotated code

Physics News old Bookmarks To... Work Stuff Perf Stuff

```
2711 if (m_eventsCounter){
2712   h_EvtRejSum->Fill(1);
2713 }
2714
2715 m_failReadyFilterTool= false;
2716 if(m_useReadyFilterTool){
2717   if(!m_ReadyFilterTool->accept()) ifPass = 0;
2718   else if (!m_ReadyFilterTool->accept()) { m_failReadyFilterTool= true; } <--- Expression is always false because 'else if' condition matches previous condition at line 2718.
2719 }
2720 if (m_eventsCounter){
2721   // if !m_useReadyFilterTool h_EvtRejSum->Fill(2);
2722   if (m_failReadyFilterTool) h_EvtRejSum->Fill(2);
2723 }
2724
2725 m_failBadLTTool=false;
2726 if(m_useBadLTTool){
2727   if(!m_BadLTTool->accept()) ifPass = 0;
2728   else if (!m_BadLTTool->accept()) { m_failBadLTTool=true; } <--- Expression is always false because 'else if' condition matches previous condition at line 2729.
2729 }
2730
2731 if (m_eventsCounter){
2732   // if !m_useBadLTTool h_EvtRejSum->Fill(3);
2733   if (m_failBadLTTool) h_EvtRejSum->Fill(3);
2734 }
2735
```

Include-what-you-use

Choose first the project, then follow path to your favourite package

https://test-soco.web.cern.ch/test-soco/display/IWYU_frames.html

IWYU reports Last scan on dev rel_6 nightly from 2015-06-27

Show Warnings Show Full includes

AtlasReconstruction Reconstruction MuonIdentification [remaining]

| remove includes | add includes |
|----------------------------------|--|
| | <pre>#include "AthenaBaseComps/AthCheckMacros.h" // for ATH_CHECK #include "AthenaBaseComps/AthMsgStreamMacros.h" // for ATH_MSG_DEBUG, etc #include "AthenaKernel/errorcheck.h" // for CHECK #include "GaudiKernel/GaudiHandle.h" // for GaudiHandle #include "GaudiKernel/IAgorithm.h" // for IAgorithm #include "GaudiKernel/IProperty.h" // for IProperty #include "GaudiKernel/IStateful.h" // for IStateful #include "GaudiKernel/MsgStream.h" // for MsgStream, operator<< #include "GaudiKernel/ServiceHandle.h" // for ServiceHandle #include "SGTools/BaseInfo.icc" // for BaseInfo::baseinfo #include "SGTools/CLASS_DEF.h" // for ClassID_traits #include "SGTools/DataBucket.icc" // for DataBucket::DataBucket<T> #include "SGTools/DataBucketBase.icc" // for DataBucketBase::cast #include "SGTools/DataProxy.h" // for DataProxy #include "SGTools/DataProxy.icc" // for DataProxy_cast #include "StoreGate/DataHandle.icc" // for DataHandle::cptr, etc #include "StoreGate/tools/SGImplSvc.h" #include "StoreGate/tools/SGImplSvc.icc" // for StoreGateSvc::overwrite, etc class ISvcLocator;</pre> |
| Code: ../src/EventInfoUnlocker.h | |
| | <pre>#include <string> // for string #include "GaudiKernel/StatusCode.h" // for StatusCode, etc class ISvcLocator;</pre> |
| Code: ../src/JobInfo.cxx | |
| | <pre>#include "GaudiKernel/IAgorithm.h" // for IAgorithm #include "GaudiKernel/IProperty.h" // for IProperty</pre> |

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 Valgrind but much faster and catches more; output maybe a bit intimidating...

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

ABI incompatible.

- **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-occurrence matrix (interdependencies)

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 available.