



STATIC ANALYSIS FOR ADA, C/C++ AND PYTHON: DIFFERENT LANGUAGES, DIFFERENT NEEDS.



Maurizio Martignano
Spazio IT – Soluzioni Informatiche s.a.s
Via Manzoni 40
46051 San Giorgio Bigarello, Mantova
<https://www.spazioit.com>

Agenda



June 2021

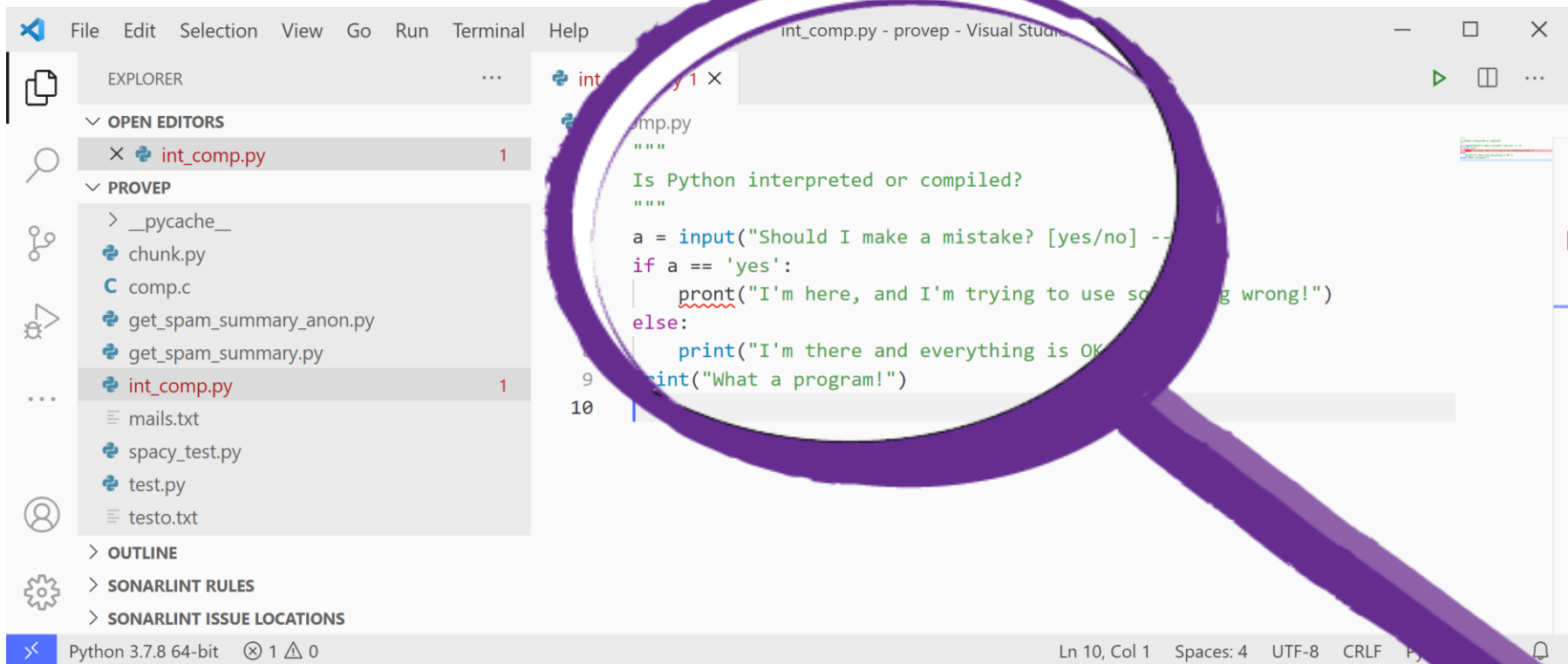
2

Agenda



- Why Static Analysis?
- Ada
- C/C++
- Python
- Future Activities

Why Static Analysis?



Why Static Analysis?



	Metrics Structure Analysis	Guidelines / Standards	Bugs Finding (alternative to testing?)	“Compilation”
Ada	M	M	L	
C/C++	L	M	M	
Python	L	M	M	L

M: more important

L: less important



Metrics, Maintainability Guidelines, Standards

Some Ada Analyzers used @ Spazio IT



	Metrics Structure Analysis	Guidelines / Standards	Bugs Finding (Abstract Interpretation, Symbolic Execution)
AdaControl	M	M	
GNAT tools	M GNATtmetric	M GNATcheck	A CodePeer
PolySpace for Ada	M	M	A
Understand	M	M	

M: mostly used for
A: also used for

Ada Static Analysis



- Lots of the issues that can be found in a system written in C /C++ are prevented from happening by design (of Ada language itself); e.g. many data conversion errors are prevented from happening by Ada strong typing.
- The compilation system is “very strict”.
- In many Ada projects thorough testing is mandatory.
- Testing may include the execution of run-time checks.

Metrics to improve Maintainability



- In 2015 Spazio IT developed for AIRBUS an Ada SonarQube Plugin able to compute the so called “Maintainability Index” as a function(Analysability, Changeability, Stability, Testability).
- In turn these other characteristics were based on metrics like (LOCs, Cyclomatic Complexity, Comment/Code Ratio, etc...).
- The overall computation followed a model implemented by AIRBUS and very close to ISO/IEC 9126, ISO/IEC 25010.

Metrics to improve Maintainability (cont)



- The **maintainability index** proved to correspond well with the «experienced» actual maintainability of real case projects.
- The **issues** found by the tool identified the elements, the points requiring a fix to improve the overall maintainability of the analysed projects.

Metrics to improve Maintainability (cont)



A screenshot of a web application interface for managing code quality issues. The browser address bar shows 'localhost:9000/issues?resolved=false'. The application has a navigation menu with 'Issues' selected. A search bar is present with the text 'Search for projects, sub-projects and file'. On the left, there is a 'Filters' sidebar with sections for 'Display Mode' (Issues selected, Effort), 'Type' (Bug: 0, Vulnerability: 0, Code Smell: 87), 'Severity' (Blocker: 0, Critical: 0, Major: 14, Minor: 37, Info: 36), 'Resolution', 'Status', 'Creation Date', 'Rule', and 'Tag'. The main area displays a list of issues for the project 'XMLAda / sax/sax-htable.adb'. Each issue entry includes a title, a metric value (e.g., 'construct_nesting = 5.0 (in range [5.0, 7.0])'), a severity icon (Code Smell), status (Info, Open, Not assigned), a timestamp ('9 days ago'), a severity level (L60, L135, L165, L213), and action icons (refresh, filter, tags). The issues listed are: 'Set_With_Hash#60', 'Get_Ptr_With_Hash#135', 'Remove#165', 'Remove#165', 'Remove_All#213', and 'Remove_All#213'. At the bottom, the project 'XMLAda / sax/sax-state_machines.adb' is partially visible. The top right of the interface shows '1 / 87 issue' and navigation controls.



Bugs Finding Guidelines, Standards

Some C/C++ Analyzers used @ Spazio IT



	Metrics Structure Analysis	Guidelines / Standards	Bugs Finding (Abstract Interpretation, Symbolic Execution)
PC-Lint-Plus	M	M	A
Cppcheck	A	A	M
Clang-SA Clang-Tidy			M
FB-Infer			M

M: mostly used for

A: also used for

C/C++ Static Analysis



- C/C++ compilers, if properly used, can be as strict as Ada compilers (e.g. <https://www.adacore.com/gems/gem-33>)

```
Command Prompt
C:\SMART\PROG\ProveAda>gnat compile Static_Check.adb
gcc -c static_check.adb
static_check.adb:8:17: non-local pointer cannot point to local object
gnatmake: "static_check.adb" compilation error

C:\SMART\PROG\ProveAda>clang -o dp.exe -Wall -Werror dp.c
dp.c:18:11: error: address of stack memory associated with local variable 'x'
    returned [-Werror,-Wreturn-stack-address]
    return &x;
           ^
1 error generated.

C:\SMART\PROG\ProveAda>
```

C/C++ Static Analysis (cont)



- In some C/C++ projects thorough testing is mandatory.
- In the Clang compilation system the compiler and the static analyzers (Clang-SA and Clang-Tidy) are based on the very same libraries → **the Compiler is the Static Analyzer**
- GCC analysis capabilities are continuously evolving (also thanks to the competition with “Clang”).
- Abstract Interpretation / Symbolic Execution in Clang static analyzers and FB Infer are already rather powerful and improving...

C/C++ Static Analysis (cont)



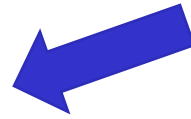
```
93 static Basic b;  
94 static MeasData m;  
95  
96 // Only use on 0-terminated strings!  
97 static int skip_to_next(char ** sp, const char ch) {  
98     int steps;  
99     while (ch != 0 && (**sp) != ch) {  
100         (*sp)++;  
101         steps++;  
102     }
```

3 ← 'steps' declared without an initial value →

4 ← Assuming the condition is true →

5 ← Loop condition is true. Entering loop body →

6 ← The expression is an uninitialized value. The computed value will also be garbage



Clang Bugs Finding displayed in Clang UI



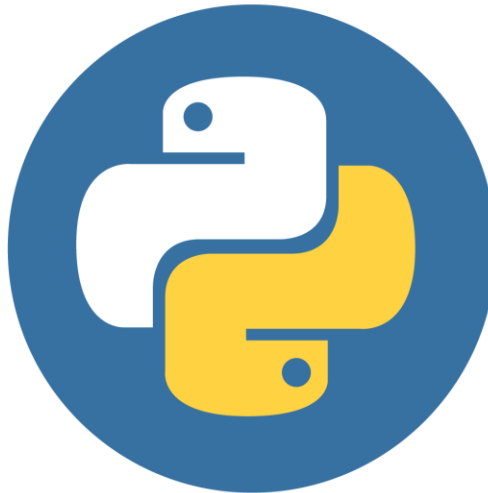
Clang Bugs Finding displayed in SonarQube

The expression is an uninitialized value. The computed value will also be garbage

Bug Major +6

- 1 Calling 'skip_to_next'
- 2 Entered call from 'gpgsaParser'
- 3 'steps' declared without an initial value
- 4 Assuming the condition is true
- 5 Entering loop body
- 6 The expression is an uninitialized value. The computed value will also be garbage

Undefined or garbage value returned to caller



Bugs Finding Guidelines, Standards Compilation

Some Python Analyzers used @ Spazio IT



	Metrics Structure Analysis	Guidelines / Standards	Bugs Finding (Logical Errors, Security Issues, Abstract Interpretation)	Compilation
Bandit	A	A	M	
Flake8	M	M		
Pylint	A	A	M	A
SQ Python Plugin	A	A	M	

M: mostly used for
A: also used for

Is Python interpreted or compiled?



```
int_comp.py - provep - Visual Studio Code
int_comp.py 1 x
int_comp.py
1 """
2 Is Python interpreted or compiled?
3 """
4 a = input("Should I make a mistake? [yes/no] --> ")
5 if a == 'yes':
6     pront("I'm here, and I'm trying to use something wrong!")
7 else:
8     print("I'm there and everything is OK.")
9     print("What a program!")
10
```

```
Anaconda Prompt (Anaconda3)
(base) C:\SMART\PROG\provep>python int_comp.py
Should I make a mistake? [yes/no] --> yes
Traceback (most recent call last):
  File "int_comp.py", line 6, in <module>
    pront("I'm here, and I'm trying to use something wrong!")
NameError: name 'pront' is not defined

(base) C:\SMART\PROG\provep>pylint int_comp.py
*****
Module int_comp
int_comp.py:6:4: E0602: Undefined variable 'pront' (undefined-variable)
```

```
>&2 echo "Migrations complete."

python -u manage.py runserver 0.0.0.0:8080
```

Python Static Analysis



- In case of very large codebases, “linting” can be used also as “compiler”, to guarantee the syntactical correctness of the entire codebase.
- In many Python (not TDD)-projects thorough testing is **not** mandatory.

Python Static Analysis (cont)



Issues

Not secure | sonarsv.spazioit.com/project/issues?id=my%3Adjango-crm&resolved=false&tags=brain-overload

Projects Issues Rules Quality Profiles Quality Gates

Search for projects... Log in

Django-CRM master

April 19, 2021, 10:34 AM Version 2021-04-19-01

Overview Issues Security Hotspots Measures Code Activity

Project Information

Search for tags...

- convention 337
- bad-practice 224
- accessibility 221
- brain-overload 128**
- unused 122
- wcag2-a 107
- html5 86
- obsolete 86
- python3 56
- clumsy 52
- design 47
- pitfall 31
- user-experience 9
- suspicious 7
- confusing 3

15 shown Show More

Ctrl + click to add to selection

Directory

File

↑ ↓ to select issues ← → to navigate 1 / 128 issues 4d 7h effort

accounts/api_views.py

- Refactor this function to reduce its Cognitive Complexity from 22 to the 15 allowed. Why is this an issue?** 3 months ago L60
Code Smell Critical Open Not assigned 12min effort brain-overload
- Refactor this function to reduce its Cognitive Complexity from 52 to the 15 allowed. Why is this an issue?** 6 months ago L157
Code Smell Critical Open Not assigned 42min effort brain-overload
- Function has a complexity of 17 which is greater than 15 authorized. Why is this an issue?** 6 months ago L157
Code Smell Critical Open Not assigned 12min effort brain-overload
- This function has 5 returns or yields, which is more than the 3 allowed. Why is this an issue?** 3 months ago L157
Code Smell Major Open Not assigned 20min effort brain-overload
- Refactor this code to not nest more than 4 "if", "for", "while", "try" and "with" statements. Why is this an issue?** 9 days ago L193
Code Smell Critical Open Not assigned 10min effort brain-overload
- Refactor this code to not nest more than 4 "if", "for", "while", "try" and "with" statements. Why is this an issue?** 9 days ago L207
Code Smell Critical Open Not assigned 10min effort brain-overload
- Refactor this function to reduce its Cognitive Complexity from 59 to the 15 allowed. Why is this an issue?** 6 months ago L256

Python Static Analysis (cont)



The screenshot shows the SonarQube web interface for a project named 'Django-CRM'. The browser address bar shows the URL: `sonarsrv.spazioit.com/project/issues?id=my%3Adjango-crm&open=AXjpRQsKVfFARlqz45cs&resol...`. The interface includes a navigation bar with 'Projects', 'Issues', 'Rules', 'Quality Profiles', and 'Quality Gates'. A search bar and 'Log in' button are also present. The main content area displays two code snippets with associated issues:

- Issue 1:** Located in `common/api_views.py`, the issue is 'status is used before it is defined. Move the definition before.' It is categorized as a 'Bug' with a severity of '+4'. A tooltip below the issue provides instructions: 'alt + ↑ ↓ to navigate issue locations'.
- Issue 2:** Located in `events/models.py`, the issue is 'Rename field "event_type" to prevent any misunderstanding/clash with field "EVENT_TYPE" defined on line 14'. It is categorized as a 'Code Smell' with a severity of '+1'. A tooltip below the issue shows '2 of 2 shown'.

The code snippets are as follows:

```
830 ashwi...         return Response(
831                 {
832                     "error": True,
833                     "errors": "You do not have permission to perform this action",
834                 },
835                 status=status.HTTP_403_FORBIDDEN,
836             )
837 ashwi...         params = request.query_params if len(request.data) == 0 else request.data
838 ashwi...         user = User.objects.get(id=pk)
839 66010...
840
841         if params.get("status"):
842             1 status = params.get("status")
843             if 2 status == "Active":
844                 user.is_active = True
845             elif 3 status == "Inactive":
846                 user.is_active = False
847         else:
848             return Response(
849                 {"error": True, "errors": "Please enter Valid Status for user"},
850                 status= 4 status.HTTP_400_BAD_REQUEST,
851             )
852         user.save()
853
854         context = {}
```

Future/Current Activities



sonar
qube



Future/Current Activities (cont)



- Keeping up to date with the evolution of Clang-SA, Clang-Tidy and FB Infer.
- Improving their integration with SonarQube.
- Applying these technologies and tools to the SAFe Toolset (https://spazioit.com/pages_en/sol_inf_en/code_quality_en/safe-toolset-en/) and ISVV projects where Spazio IT is going to be involved this year and the next ones. (e.g. adding Python support to the SAFe Toolset).

Current (04/2021) Clang SA – SonarQube Integration



The screenshot displays the SonarQube interface for a project named 'Node_js' on the 'master' branch. The left pane shows the source code for 'deps/icu-small/source/common/locale.cpp'. The code includes a function 'Locale::getLocale(int locid)' that initializes a 'Locale *localeCache' and returns it. Annotations highlight several issues: 'localeCache' is initialized here, assuming 'localeCache' is equal to NULL, assuming pointer value is null, taking true branch, and returning null reference. The middle pane shows a list of 8/138 issues, with the selected issue 'Returning null reference' (Bug +5) detailed below. The right pane shows the code with red markers corresponding to the annotations and the issue details.

```
2193 }
2194
2195 const Locale &
2196 Locale::getLocale(int locid)
2197 {
2198     Locale *localeCache = getLocaleCache()
2199     U_ASSERT((locid < eMAX_LOCALES)&&(locid >= 0));
2200     if (localeCache == NULL) {
2201         // Failure allocating the locale cache.
2202         // The best we can do is return a NULL reference.
2203         locid = 0;
2204     }
2205     return localeCache[locid]; /*operating on NULL*/
2206 }
2207
2208 /*
2209 This function is defined this way in order
2210 initialization and static destruction.
2211 */
2212 Locale *
2213 Locale::getLocaleCache(void)
2214 {
```

Returning null reference Bug +5

- 1 Calling 'Locale::getLocale'
- 2 Entered call from 'Locale::getCanadaFrench'
- 3 'localeCache' initialized here
- 4 Assuming 'localeCache' is equal to NULL
- 5 Returning null reference

alt + ↑ ↓ to navigate issue locations

Returning null reference Why is this an issue? 7 months ago L2205 clangsa, core

8 / 138 issues

deps/cares/src/ares_process.c

The left operand of '==' is a garbage value Bug +17

deps/icu-small/source/common/locale.cpp

Node_js master

Last analysis had 3 warnings April 7, 2021, 7:35 AM Version 15.13.0

Projects Issues Rules Quality Profiles Quality Gates

Search for projects... Log in

Current (04/2021) FB Infer – SonarQube Integration



The screenshot displays the SonarQube web interface. On the left, a code editor shows a C file with several lines of code. The main area shows a list of issues, with one issue selected and its details expanded. The issue is titled "pointer 'tagPtr' last assigned on line 207 could be null and is dereferenced at line 208, column 9". The issue is classified as "Critical" and "Open". The code snippet shows the following:

```
202  /* Allocate piggypacked memory chunk containing
203  * - tag structure,
204  * - tag begin string,
205  * - tag end string
206  */
207  tagPtr = ns_malloc(sizeof(Tag) + (size_t)slen + (size_t)elen);
208  tagPtr->type = type;
```

The issue details panel shows the following information:

- Issue title: pointer 'tagPtr' last assigned on line 207 could be null and is dereferenced at line 208, column 9.
- Severity: Critical
- Status: Open
- Not assigned: 5min effort
- Created: 9 days ago
- Location: L208

Thank you for your time!

