systerel
Solutions temps réel sécurisées

AdDoc

(beyond a document generator)

Ada-Europe 2014

Paris, 26 June

Agrément Crédit Impôt Recherche

Systerel Group Organization

SYSTEREL Group
Turnover of € 8 million
Over 100 engineers

systerel
Solutions temps réel sécurisées

- ✓ Safety studies
- ✓ Safety critical software development
- ✓ Formal methods

DELTA SUD-OUEST TECHNOLOGIES
Groupe SYSTEREL

Embedded electronic systems

- ✓ Digital electronics
- ✓ Signal acquisition and conditioning
- ✓ Signal processing

Plan

- Context and objectives
- Method
- Under the hood
- AdDoc / EN 50128:2011
- Demo
- Evolutions
- Results & conclusions

Context and objectives - 1

- For the development of its new generation of Railway Control System, Alstom Transportation has decided to improve the features of tools involved in the development process of safety critical applications.
- The produced software must conform at least with the EN50128:2011 standard (this includes the produced software and its documentation).
- The concerned application is ~450 KLOC Ada_{2005} with a Safety Integrity Level of 4 (the highest).

Context and objectives - 2

- To meet these standard requirements, the documentation has to be **consistent** with the source code.
- Producing this documentation manually is a **costly** and **tedious** activity and its does not guarantee the documentation **exhaustiveness**.

So!

Context and objectives - 3

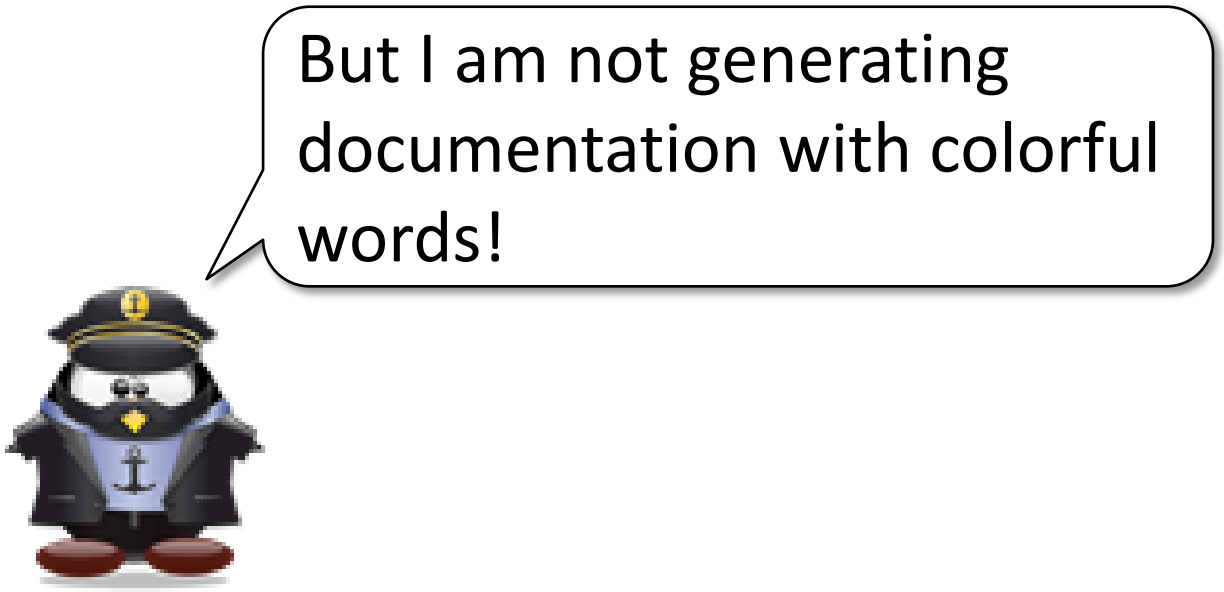
- Alstom Transportation has decided to use a tool able to produce the design documentation in an **automatic** manner from Ada_{2005} source files.

Method - 1

- To avoid wasting energy or resources in "reinventing the wheel", a study of existing tools has been done to find which one could "make the job" with a minimum of effort/adaptations.
- Even if there are already very efficient tools, the effort to adapt them to the specific features wanted by Alstom (to be seen later) was judged too important.

Method - 2

- Thus, Alstom has decided to develop its own tool named AdDoc (Ada Document generator).



Method - 3

- This study also convinced Alstom that the only **reliable** implementation solution should be based on the **ASIS** technology.
- **But why use such a powerful technology as ASIS for documentation purpose only?**
- It was therefore also decided to add some new features to the tool in order to improve the quality of the documentation and of the source code.

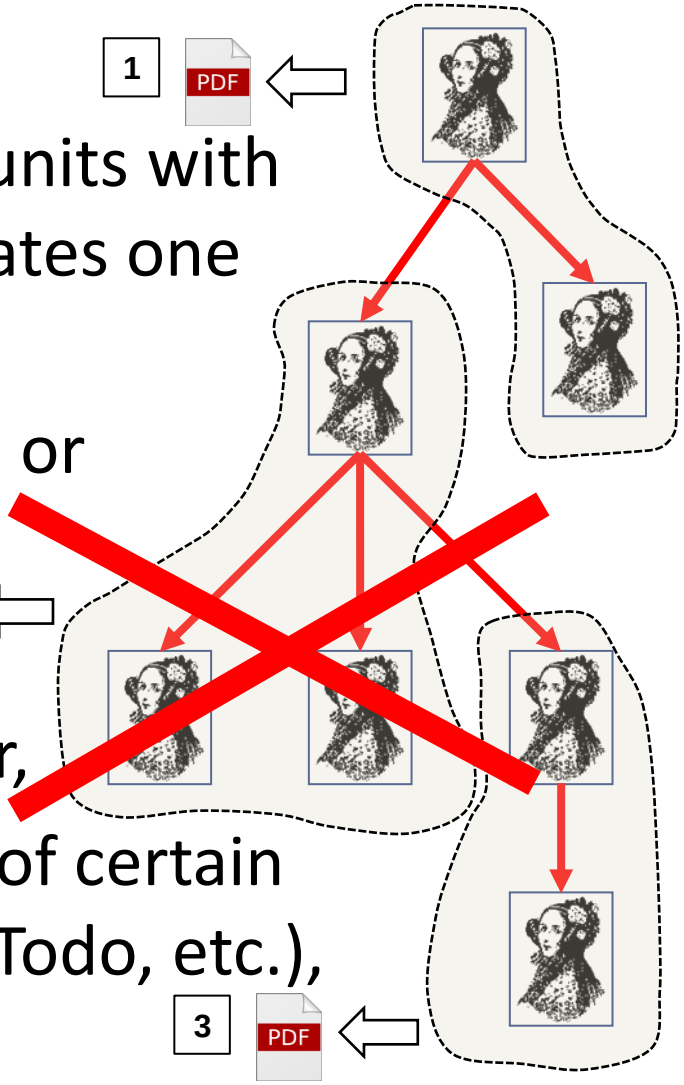
Method - 4

- The current AdDoc tool:
  - Checks the completeness of comments and their consistency with the code → **Force the developer to comment its code**!
  - Checks some coding rules → **Emphasize source readability**!

Method - 5

– Gives the possibility to:

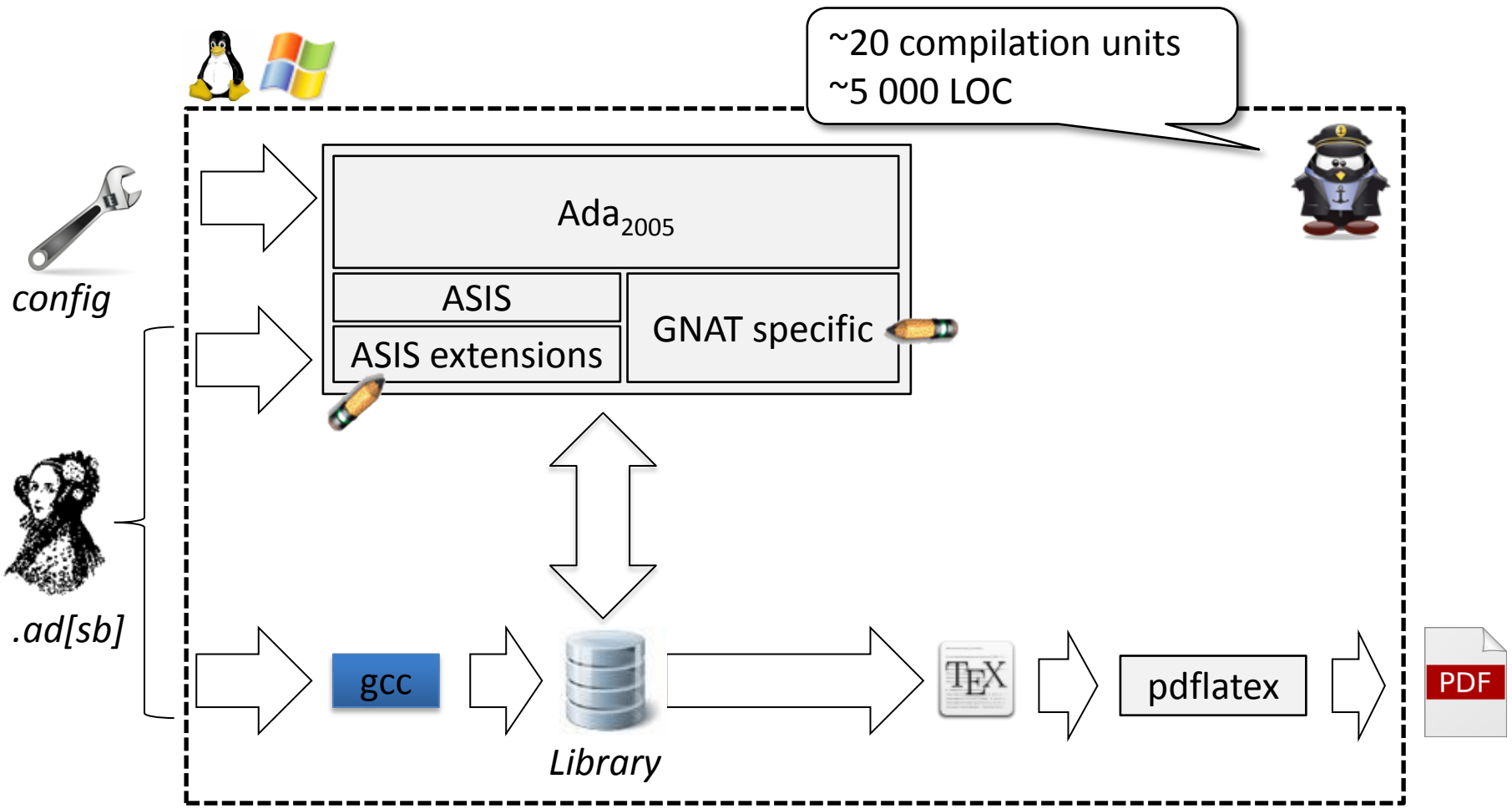
- define rules to map a set of Ada units with modules (~CSCs) : the tool generates one PDF document per CSC,
- ignore a set of compilations units or directories,
- ignore unit bodies,
- use the tool with a cross compiler,
- raise warnings regarding the use of certain predefined Tags (Tbd, Rule, Req, Todo, etc.),
- …



Under the hood - 1

- AdDoc is an Ada_{2005} ASIS application of ~5KLOC that has been specified, developed and validated in approximately **two man-months**!
- Developing this application in such a short time implied to make some "short cuts" by using for example some specific GNAT implementation facilities and also some ASIS extensions.

Under the hood - 2



Under the hood - 3

- One of the difficulties was to access the comment of a syntactic element.
- Therefore, the root package *AdDoc.Comments* as been developed to provide this functionality in a **convenient way**.

```
package AdDoc.Comments is
 type Comments_T is tagged private;
 No_Comments : constant Comments_T;

 function Read(Element : in Asis.Element) return Comments_T;
 ...
private
 ...
end AdDoc.Comments;
```

Under the hood - 4

```
--! xxxxxxxxxxxxxxxxxx
type Private_T

  (--! xxxxxxxxxxxxxxx

   --! xxxxxxxxxxxxxxx

   Discriminant1 : Boolean;

   --! xxxxxxxx

   Discriminant2 : Boolean)

is tagged private;
```

```
--! xxxxxxxxxxxxxxxxx

--! @{

Var1 : Integer := 1;

Var2 : Integer := 2;

--! @}
```

Comment blocks

AdDoc / EN 50128:2011 - 1

- As required by the EN50128:2011 standard, the design documentation shall address:
  1. Identification of all lowest-level software units (e.g. subroutines, methods, procedures) traced back to the upper level,
  2. Their detailed interface with the environment and other components with detailed input and output,
  3. Their safety integrity level without any further apportionment within the component itself,
  4. Detailed algorithms and data structures.

AdDoc / EN 50128:2011 - 2

- The documentation produced by AdDoc has been considered to satisfy points 1), 2) and 3). Regarding the last one, the expected information is available in a document that was already present in Alstom's repository.
- AdDoc is a tool categorized into the class T1: "*generates no outputs which can directly or indirectly contribute to the executable code (including data) of the Software*".

Demo - 1

- With this short demonstration we will show you the ability of AdDoc to:
  - Generate the documentation of a package (here the *Ada_Europe* package),
  - Detect any violation of rules defined by the tool.

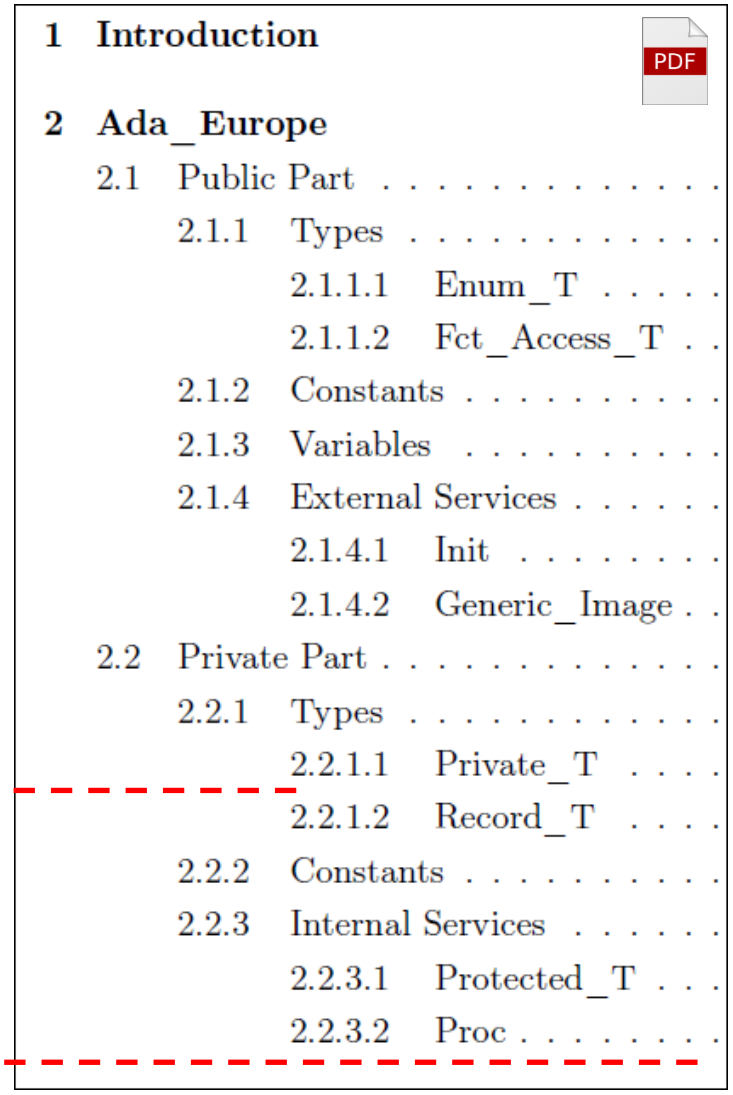
Let's go!

Demo - 2

```
generic
   …
package Ada_Europe is

   type Enum_T is (…);

   type Private_T (…) is tagged private;

   type Fct_Access_T is access function …

   …

   procedure Init (…);

   generic
      …
   function Generic_Image (…) …;
private
    type Private_T (…) is …;
    protected type Protected_T (…) is …
end Ada_Europe;
```

Demo - 3

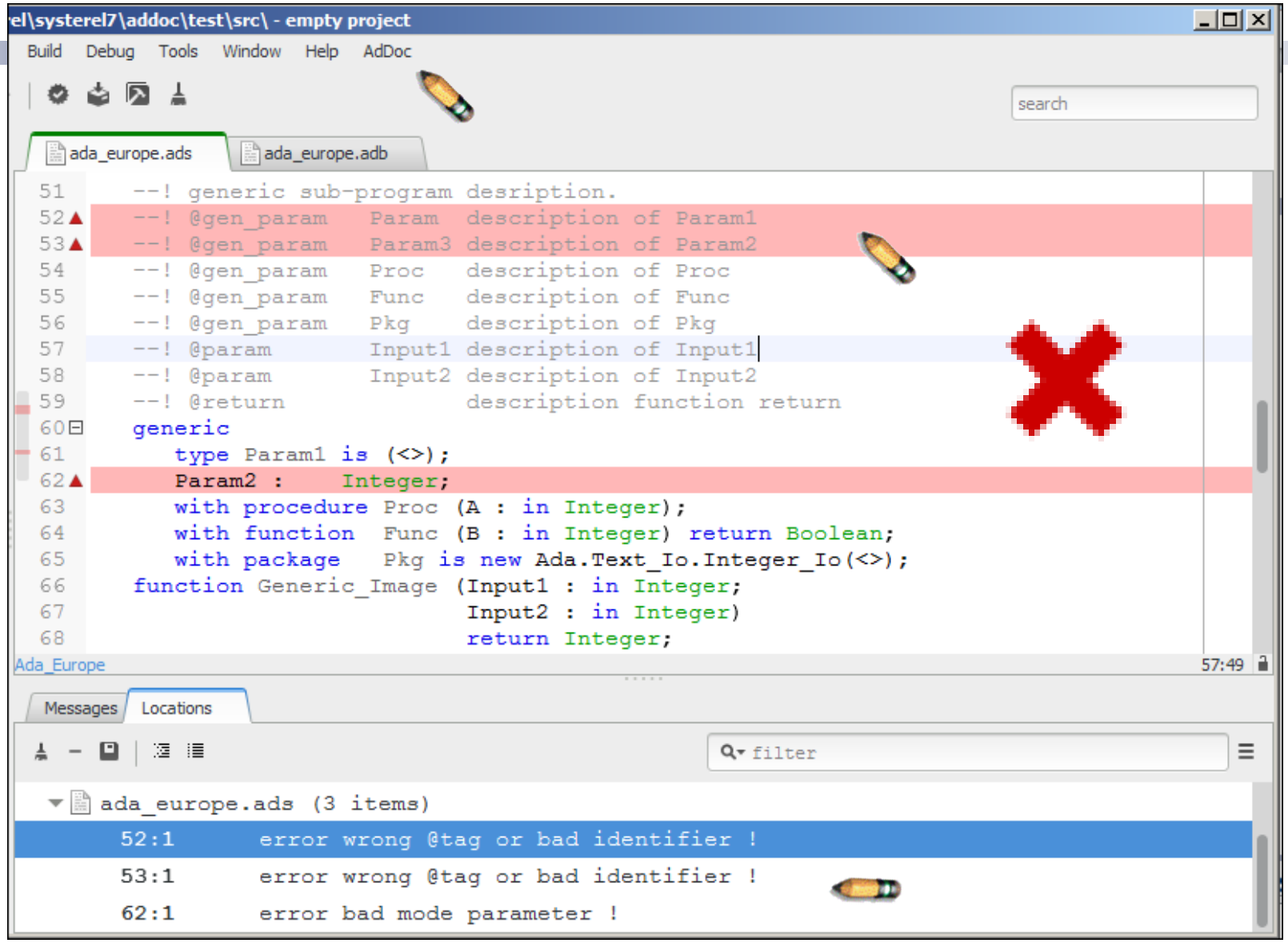
```
package body Ada_Europe is

    type Record_T is record …

    Cst2 : constant Integer := 1;

    procedure Proc (…);

    function Generic_Image (...) …

    procedure Init (…) is …

    procedure Proc (…) is …

    protected body Protected_T is …

end Ada_Europe;
```



Demo - 4

```
--! generic sub-program desription.
--! @gen_param   Param  description of Param1
--! @gen_param   Param3 description of Param2
--! @gen_param   Proc   description of Proc
--! @gen_param   Func   description of Func
--! @gen_param   Pkg    description of Pkg
--! @param       Input1 description of Input1
--! @param       Input2 description of Input2
--! @return             description function return
generic
  type Param1 is (<>);
  Param2 :    Integer;
  with procedure Proc (A : in Integer);
  with function  Func (B : in Integer) return Boolean;
  with package   Pkg is new Ada.Text_Io.Integer_Io(<>);
function Generic_Image (Input1 : in Integer;
                        Input2 : in Integer) return Integer;
```

Demo - 5


el\systerel7\addoc\test\src\ - empty project
Build Debug Tools Window Help AdDoc
search
ada_europe.ads
ada_europe.adb
51 --! generic sub-program desription.
52 --! @gen_param Param description of Param1
53 --! @gen_param Param3 description of Param2
54 --! @gen_param Proc description of Proc
55 --! @gen_param Func description of Func
56 --! @gen_param Pkg description of Pkg
57 --! @param Input1 description of Input1
58 --! @param Input2 description of Input2
59 --! @return description function return
60 generic
61 type Param1 is (<>);
62 Param2 : Integer;
63 with procedure Proc (A : in Integer);
64 with function Func (B : in Integer) return Boolean;
65 with package Pkg is new Ada.Text_Io.Integer_Io(<>);
66 function Generic_Image (Input1 : in Integer;
67 Input2 : in Integer)
68 return Integer;
Ada_Europe
57:49
Messages
Locations
filter
ada_europe.ads (3 items)
52:1 error wrong @tag or bad identifier !
53:1 error wrong @tag or bad identifier !
62:1 error bad mode parameter !

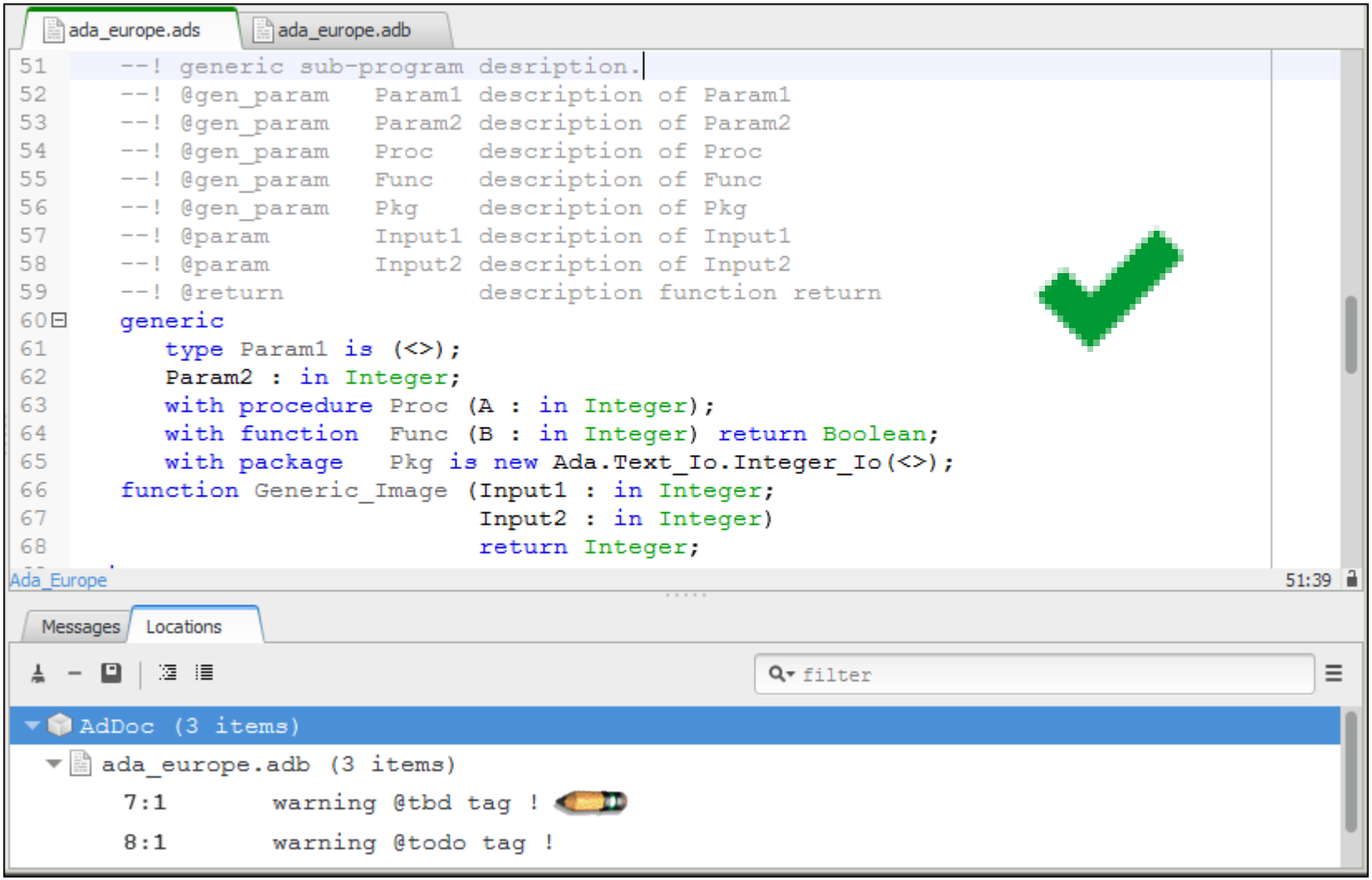
Demo - 6

```
--! generic sub-program desription.
--! @gen_param   Param  description of Param1
--! @gen_param   Param3 description of Param2
--! @gen_param   Proc   description of Proc
--! @gen_param   Func   description of Func
--! @gen_param   Pkg    description of Pkg
--! @param       Input1 description of Input1
--! @param       Input2 description of Input2
--! @return             description function return
generic
  type Param1 is (<>);
  Param2 :    Integer;
  with procedure Proc (A : in Integer);
  with function  Func (B : in Integer) return Boolean;
  with package   Pkg is new Ada.Text_Io.Integer_Io(<>);
function Generic_Image (Input1 : in Integer;
                        Input2 : in Integer) return Integer;
```

Demo - 7

```
--! generic sub-program desription.
--! @gen_param   Param1 description of Param1
--! @gen_param   Param2 description of Param2
--! @gen_param   Proc   description of Proc
--! @gen_param   Func   description of Func
--! @gen_param   Pkg    description of Pkg
--! @param       Input1 description of Input1
--! @param       Input2 description of Input2
--! @return             description function return
generic
  type Param1 is (<>);
  Param2 : in Integer;
  with procedure Proc (A : in Integer);
  with function  Func (B : in Integer) return Boolean;
  with package   Pkg is new Ada.Text_Io.Integer_Io(<>);
function Generic_Image (Input1 ,
                        Input2 : in Integer)return Integer;
```

Demo - 8


ada_europe.ads
ada_europe.adb
51 --! generic sub-program desription.
52 --! @gen_param Param1 description of Param1
53 --! @gen_param Param2 description of Param2
54 --! @gen_param Proc description of Proc
55 --! @gen_param Func description of Func
56 --! @gen_param Pkg description of Pkg
57 --! @param Input1 description of Input1
58 --! @param Input2 description of Input2
59 --! @return description function return
60 generic
61 type Param1 is (<>);
62 Param2 : in Integer;
63 with procedure Proc (A : in Integer);
64 with function Func (B : in Integer) return Boolean;
65 with package Pkg is new Ada.Text_Io.Integer_Io(<>);
66 function Generic_Image (Input1 : in Integer;
67 Input2 : in Integer)
68 return Integer;
Ada_Europe
51:39
Messages
Locations
filter
AdDoc (3 items)
ada_europe.adb (3 items)
7:1 warning @tbd tag !
8:1 warning @todo tag !

Demo - 9

2.1.4.2 Generic_Image

Description : generic sub-program desription.

Design Description : anything that can help to understand the architecture of the procedure

Generic Parameters :

Name	Description
Param1	description of Param1
Param2	description of Param2
Proc	description of Proc
Func	description of Func
Pkg	description of Pkg

Parameters :

Name	In/Out	Type	Description
Input1	in	Integer	description of Input1
Input2	in	Integer	description of Input2

Return : Type Integer : description function return

Demo - 7

```
    --! Description of Private_T
    type Private_T (--! Description of the first discriminant
                    Discriminant1 : Boolean;
                    --! Description of the second discriminant
                    Discriminant2 : Boolean) is tagged private;
    …
private

   --! (described above)
   type Private_T (
      --! (described above)
      Discriminant1 : Boolean;
      --! (described above)
      Discriminant2 : Boolean) is tagged record
      case Discriminant1 is
         --!Description of Field1
         when True  => Field1 : Positive;
         --! Description of Field2
         when False => Field2 : Integer := 1;
      end case;
   end record;
```

Demo - 9

2.2.1.1 Private_T

Description : Description of Private_T

Discriminants :

Field	Type	Description
Discriminant1	Boolean	Descrition of the first discriminant
Discriminant2	Boolean	Descrition of the second discriminant

Fields :

Field	Type	Description
Field1	Positive	Description of Field1
Field2	Integer	Description of Field2

Evolutions - 1

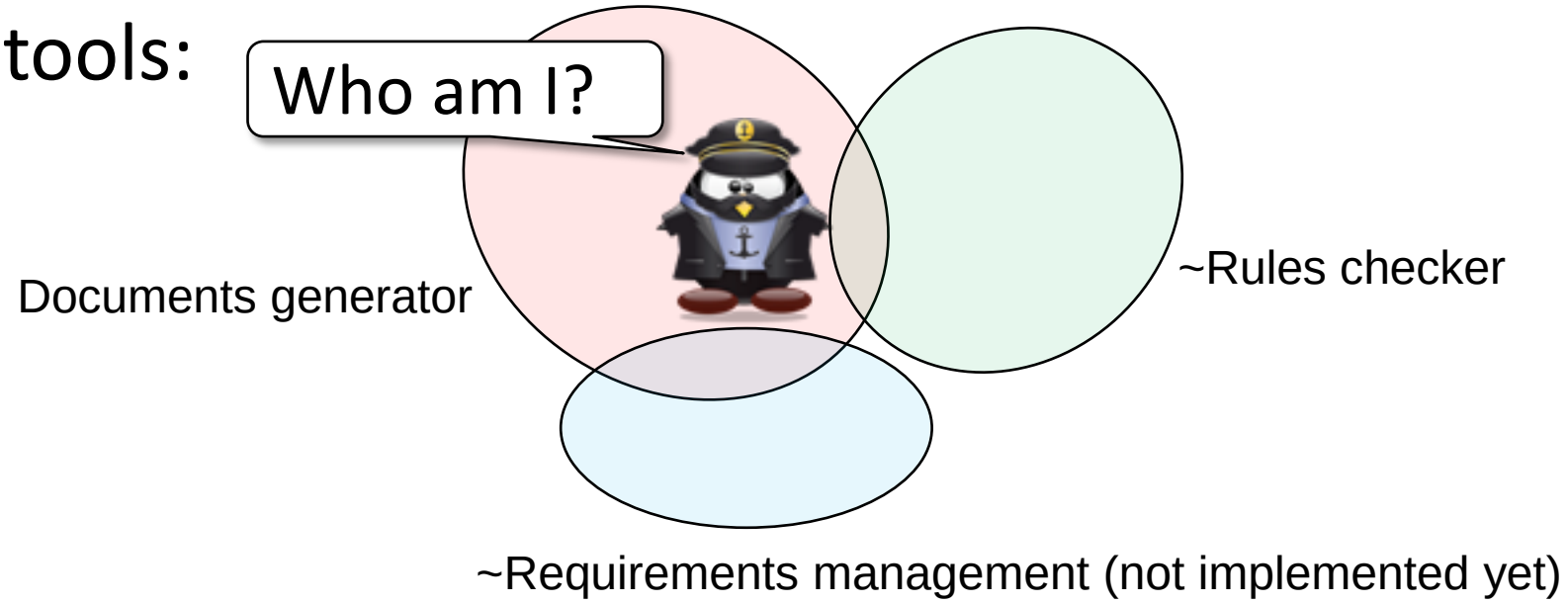
- AdDoc is a **very young** tool working on both Windows and Linux with some improvements already planned such as:
  - Having a full requirements tag extraction in order to capture and to follow them throughout the source code,
  - Providing a convenient way to define document templates (for the moment it uses hard-coded LaTeX files!),

Evolutions - 2

- Having a real integration into GPS (.gpr, GUI, parameters, documentation, etc.),
- Minimizing (or removing?) implementation dependences,
- Having more coding rules and providing a convenient way to add them,
- Adding some specific tags for a better LaTeX output (bold, italic, underline, ...).
- Using Ada_{2012} facilities?,
- …?

Results & conclusions - 1

- AdDoc has now been successfully deployed and integrated in the development process of safety critical application projects (daily builds).
- AdDoc ~is on a boundary between three kind of tools:



Results & conclusions - 2

- To be efficient, it should be used at the beginning of the development phase (otherwise it could be painful!).
- Contrary to what was expected, the difficulty was not **ASIS** (even if it is tricky) but it was the specification of AdDoc:
  - Define the rules to apply to comments,
  - Identify all the possible syntactic constructions and the comments to apply on.

Results & conclusions - 3

- AdDoc is another example that the strength of the Ada language is not only based on its own quality but also on its ability to promote efficient and powerful technology like **ASIS**.


Any questions?