Vérification par preuve formelle de logiciel de vol spatial

« Preuve de modèle, preuve de programme »

-

CYCLE DE CONFÉRENCES TECHNIQUES SUR LES MÉTHODES FORMELLES DE DÉVELOPPEMENT

Aurore Dupuis (CNES) aurore.dupuis@cnes.fr
Stéphane Duprat (AtoS) stephane.duprat@atos.net

04.02.2014

Motivations

04/02/2014
Stéphane Duprat
Aurore Dupuis

- Main reasons to use verification by proof
  - Quality of verification
    - Exhaustivity
    - Non ambiguous representation
  - Costs
    - Reduce cost of verification phase
    - Reduce cost during total lifecycle of software
    - Reduce maintenance costs

Objectives

04/02/2014
Stéphane Duprat
Aurore Dupuis

- Main objectives:

1. Formal proof integration into the V-development cycle for embedded project
2. Formal proof advantages compared to validation by test
3. Frama-C Technical maturity Evaluation
4. Cost impact evaluation compared to validation by test

Context

04/02/2014
Stéphane Duprat
Aurore Dupuis

- Two space embedded software have been used for this study
  - **Software 1**: Embedded software already validated by test
    - Known validation by test costs
    - Bugs undiscovered by test
  - **Software 2:** Embedded software currently in development
    - Specification and conception undefined
    - Architecture based on components

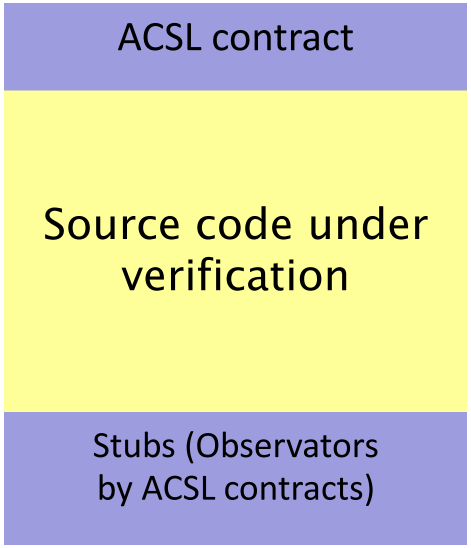
Tooling

04/02/2014
Stéphane Duprat
Aurore Dupuis

- Frama-C platform
  - Deductive proof (Hoare, Dijkstra)
  - Function contracts with ACSL
    - '**requires**' = preconditions
    - '**ensures**' = postcondition
    - '**behavior**' and 'assumes' : fonctionnal cases
    - '**assigns**' : defines side effects

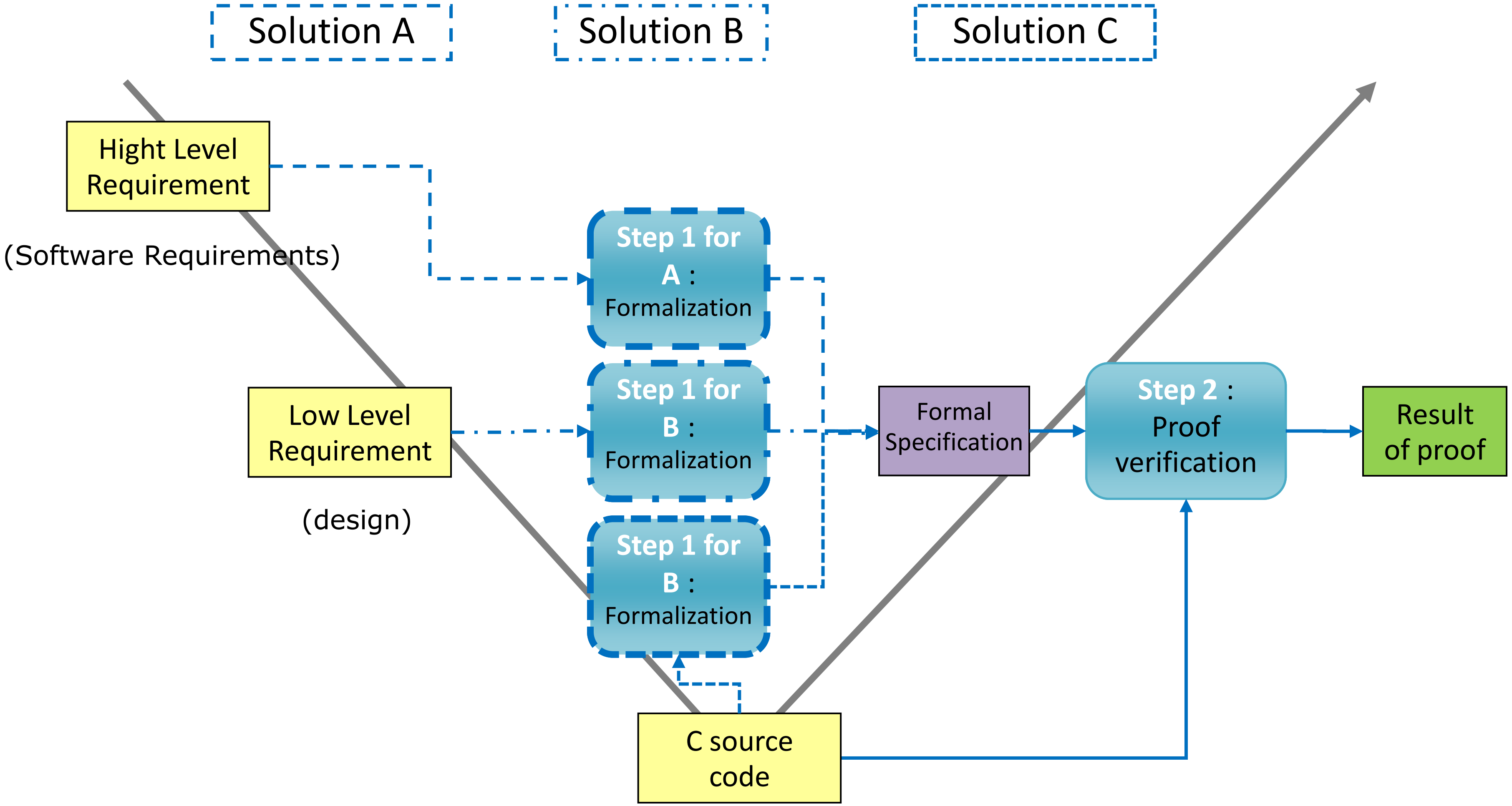
```
/*@
  @ behavior b_neg:
  @   assumes p<0;
  @   ensures P1: \result == -1;
  @ behavior b_pos:
  @   assumes p>=0;
  @   ensures P1: \result == 0;
  @*/
int f1_bis(int p)
{
  ...
```

- Topology of a proof project



Methodology

04/02/2014
Stéphane Duprat
Aurore Dupuis



Study

04/02/2014
Stéphane Duprat
Aurore Dupuis

- Proof on Software 1
  - First apply Solution **B** (formalization at the design level) : considered not relevant for this use case
  - Secondly, Solution **A** (formalization at le Software Requirement level)

  - Results:2 bugs detected

    - One about a comparison between two pointers of a circular buffer.
      - Formalization with the mathematic modulo
      - Problem at the end of a range

    - Second one on the arguments passed to a System Call
      - Formalization of the interface of the `mktime()` system call
      - Missing initialization of an input field
      - Non functional property (not defined in Software Requirement)

04/02/2014
Stéphane Duprat
Aurore Dupuis

Example

```
/*@
  axiomatic math_mod
  {
    logic integer math_mod(integer a, integer b);
    axiom math_mod1 : \forall integer a,b; 0<=a<b && b>0 ==> math_mod(a,b)==a;
    axiom math_mod2 : \forall integer a,b; -b<=a<0 && b>0 ==> 
math_mod(a,b)==a+b;
  }
*/

/*@
  axiomatic detection
  {
    predicate range_ko(integer index1, integer index2, integer size, integer 
delta) = 0<math_mod((index2-index1),size)<delta;

}
*/
```

```
  behavior b2all_range_ok:
    assumes ! range_ko(INDEX_W, INDEX_READ, NB_ELEMT, DELTA_NOM);
    ensures b2all_range_ok:  FLAG_ERROR == \old(FLAG_ERROR);
```

Study

04/02/2014
Stéphane Duprat
Aurore Dupuis

- Proof on Software 2
  - Software with only source code
  - Solution **C** considered as not relevant
  - Solution **B** ReEngineering a design from source code + formalization of the design

  - Results
    - Simple functions well verified without bugs
    - Technical difficulties encountered for other functions
    - Methodological result : function contract for design description

Study

04/02/2014
Stéphane Duprat
Aurore Dupuis

- Formal proof integration into the V-development cycle for embedded project
  - Formalization of high level requirement if better, although HLR are not entirely formalized

- Formal proof advantages compared to validation by test
  - Exhaustive, non ambiguous, no need of hardware to execute tests programs

- Frama-C Technical maturity Evaluation
  - Proof feature was in development, some difficulties with data aliasing (multiple access to same location of memory)

- Cost impact evaluation compared to validation by test
  - Quality of verification already demonstrated
  - Waiting for improvements of the tool to use it in a more general way

Conclusion

04/02/2014
Stéphane Duprat
Aurore Dupuis

- Verification HLR
  - Close to informal specification, good traceability
  - High quality level

- Formal Verification for hard point verification
  - Mix of skills : integrated team (functional specialist + formal proof specialist)

- Current limitation
  - Tool definition : requires program well typed, no low level semantic
  - Tool maturity : need improvements for alias cases, floating points

- For a more extensive usage
  - Context of design or low level requirement:
    Methodologically ok, maturity of tool expected soon
  - For low level:
    Good use case in proof of integration driver + applicative

Thank you

04.02.2014