Formal Verification of Medical Device User Interfaces Using PVS

Software Verification & Validation

BMEVIMMD052

Student: Gergely Ferenc Rácz

Date: 6th December 2017

Outline

- Problem statement
- Case study: Infusion pump
- Solution, Proposed method
  - PVS specification
  - Behavioral model
  - Test input sequences
- Example
- Conclusion

Problem statement

- *Safety* of medical devices
  - Internal operation
  - **Human-device interface (UI)**
- Interaction design is not standardized
- Problems
  - Human factors specialists are not involved
  - User interface prototype is not available
  - SW engineers cannot identify human factors related issues

MŰEGYETEM 1782

https://www.intersil.com/content/dam/Intersil/featured/applications/infusion-pump.jpg

Case study: Infusion pump

- Range of accepted values: (0; 1200]
- Fractional part allowed for: (0; 99]

MŰEGYETEM 1782

Case study: Infusion pump

1. Valid input key sequences are incorrectly registered without the user's awareness.
   - Input key sequence: **`100.1`**
   - Registered as: **`1001`**

   Reason

   Numbers above or equal to 100 cannot have a fractional part.

Case study: Infusion pump

2. Inappropriate feedback is given to the user for error conditions.
   - Input key sequence: **`200.1`**
   - Error message: **`HIGH`**

Reason

The pump erroneously ignores the decimal point in the key sequence and registers the number as 2001.

Case study: Infusion pump

3. Ill-formed input key sequences are silently accepted without the user's awareness.
   - Input key sequence: **`9.9.1`**
   - Registered as: **`9.91`**

Reason

The second decimal point is silently discarded.

Case study: Infusion pump

4. Digits after decimal point are silently discarded without the user's awareness.
   - Input key sequence: **`10.09`**
   - Registered as: **`10`**

Reason

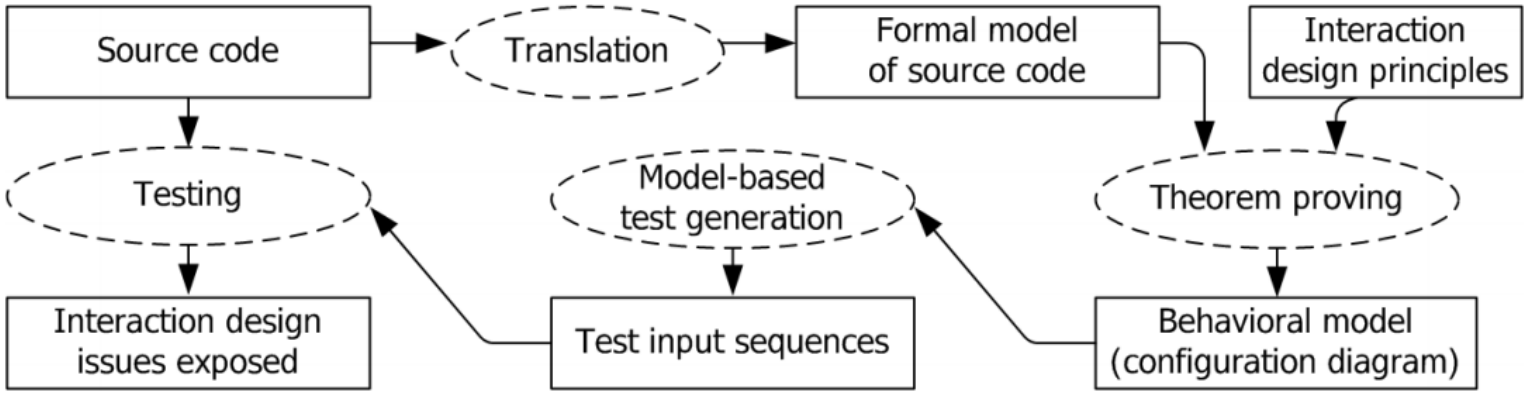
The pump software automatically limits the accuracy of numbers to one decimal digit for values between [10, 100).

Solution

- Model based design
  - Exclude issues during design
  - Must be integrated into development
- **Verification of UI source-code**

1. UI source-code → formal specification
2. Formal specification → behavioral model
   - prove correctness or
   - detect potential interaction design issues
3. Behavioral model → test input sequences

Proposed method



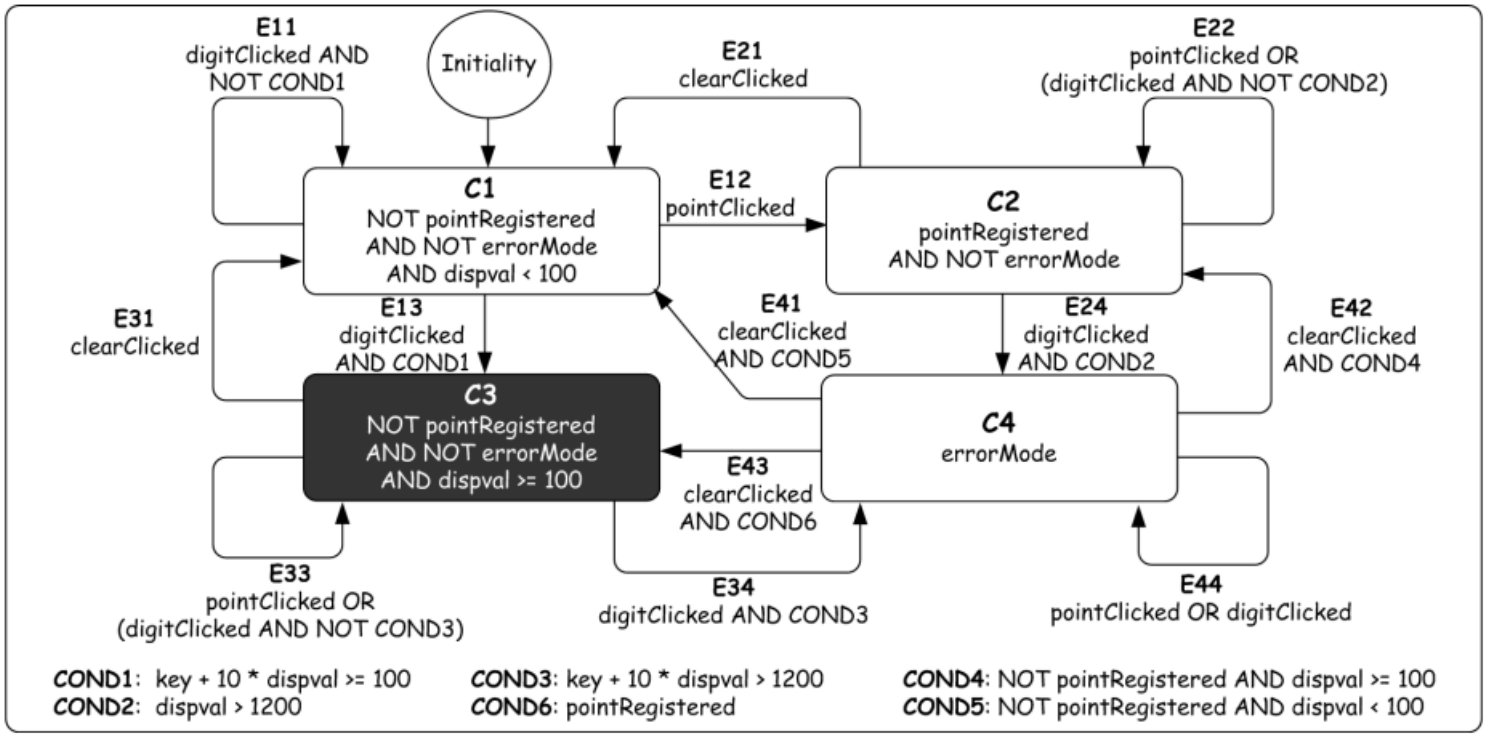
MŰEGYETEM 1782

1. PVS specification

- **PVS**: Prototype Verification System
  - SRI International, California
  - Industrial-level theorem prover
  - Classical typed higher-order logic
- C++ code can be translated into PVS specifications
  - Conditional and iterative statements
  - Computations → states and transitions
  - Functions → higher-order functions

2. Behavioral model

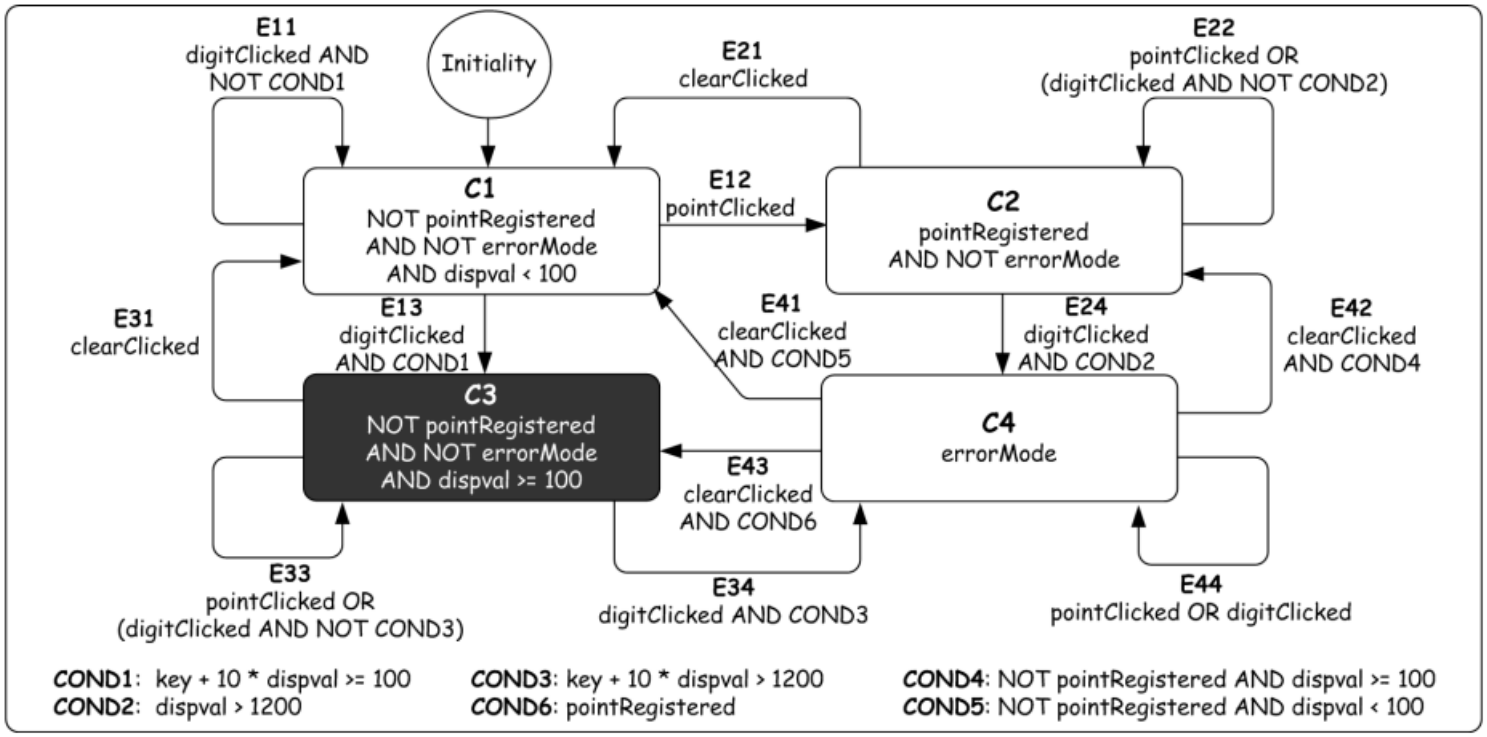
- Build the configuration diagram using PVS and extract the behavioral model.



3. Test input sequences

- Generate test input key sequences
  - Traverse configuration diagram
  - Identify user actions (key presses) associated with its transitions
  - List consecutive actions → test case
- Test input key sequences can be used as test cases
  - E.g. **`9.9.1`**

Example



- $C_1 \xrightarrow{E_{13}} C_3 \xrightarrow{E_{33}} C_3 \xrightarrow{E_{33}} C_3$
- $10 \xrightarrow{0 \text{ pressed}} 100 \xrightarrow{\cdot \text{ pressed}} 100 \xrightarrow{1 \text{ pressed}} 1001$

Conclusion

- Safety of medical devices
  - UI is potential source of issues
- Verification of UI source-code
  - Theorem proving (PVS)
  - Configuration diagrams
- Benefit of PVS over model checking
  - No need for abstraction
  - Behavioral models can also be verified by model checkers

Thank you for your attention!

Reference:

[1] Paolo Masci, Yi Zhang, Paul Jones, Paul Curzon, Harold Thimbleby. Formal Verification of Medical Device User Interfaces Using PVS. *Fundamental Approaches to Software Engineering*, pages 200-214, 2014

MŰEGYETEM 1782