Isabelle/Solidity for Smart Contracts

Jane Open Access
Dummy University Computing Laboratory, [optional: Address], Country
My second affiliation, Country

Joan R. Public[1]
Department of Informatics, Dummy College, [optional: Address], Country

Abstract

2012 ACM Subject Classification Replace ccsdesc macro with valid one

Keywords and phrases Program Verification, Smart Contracts, Isabelle, Solidity

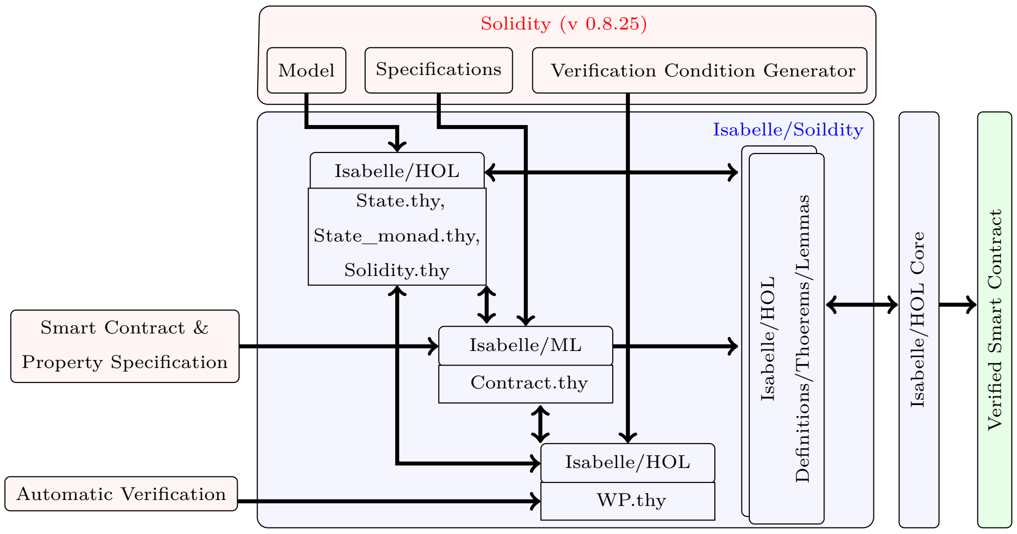
Digital Object Identifier 10.4230/OASIcs.CVIT.2016.23

Funding *Jane Open Access*: (Optional) author-specific funding acknowledgements
Joan R. Public: [funding]

Acknowledgements I want to thank . . .

1 Introduction

2 Overview



[1] Optional footnote, e.g. to mark corresponding author

3 Case Study

Listing 1: Solidity source code for the Casino

```
 1  contract Casino {
 2    enum Coin { HEADS, TAILS } ;
 3    enum State { IDLE, GAME_AVAILABLE, BET_PLACED }
 4    State private state;
 5    address public operator, player;
 6    uint public pot;
 7    bytes32 public hashedNumber;
 8    uint public bet;
 9    Coin guess;
10
11    function createGame(bytes32 hashNum)
12    public byOperator, inState(IDLE) {
13    hashedNumber = hashNum;
14    state = GAME_AVAILABLE;
15    }
16
17    function placeBet(Coin _guess) public payable inState(GAME_AVAILABLE) {
18    require (msg.sender != operator);
19    require (msg.value <= pot);
20    state = BET_PLACED;
21    player = msg.sender;
22    bet = msg.value;
23    guess = _guess;
24  }
25
26    function decideBet(uint secretNumber)
27    public byOperator, inState(BET_PLACED) {
28      require (hashedNumber == keccak256(secretNumber));
29      Coin secret = (secretNumber % 2 == 0)? HEADS : TAILS;
30      if (secret == guess) { pot = pot - bet; player.transfer(bet*2); bet =
            0;
31    } else {
32      pot = pot + bet; bet = 0;
33    }
34    state = IDLE;}
35    function addToPot() public payable byOperator { pot = pot + msg.value;}
36
37    function removeFromPot(uint amount) public byOperator, noActiveBet {
          operator.transfer(amount); pot = pot - amount;}
38    }
```

Listing 1 is a Solidity source code for Casino smart contract from verifyThis competition. The contract has three explicit states: `IDLE`, `GAME_AVAILABLE`, `BET_PLACED` (Line 3). The `creatGame` function allows only operator, enusred by `byOperator` modifier, to creat a game given that the contract is in `IDLE` state, which is eforced by inState(s) modifier. The `creatGame` function assigns a value to `hasNumber` and changes the state of the contract to `GMAE_AVAILABLE`. The `hasNumber` value provided by the operator when creating game is later used to ensure the bet was placed by the operator in the begining of game.

A player may place a bet by calling `placeBet` function in `GMAE_AVAILABLE` state. The `placeBet` function accepts a guess, `_guess`, and changes the sate to `BET_PLACED`. Moreovoer, `require` is used to exclude operator from the bet and also place maximum limit on the bet money.

4 Specification

5 Related Work

6 Conclusion