An Integrated CI/CD Workflow for eXecutable Domain-Specific Modelling Languages

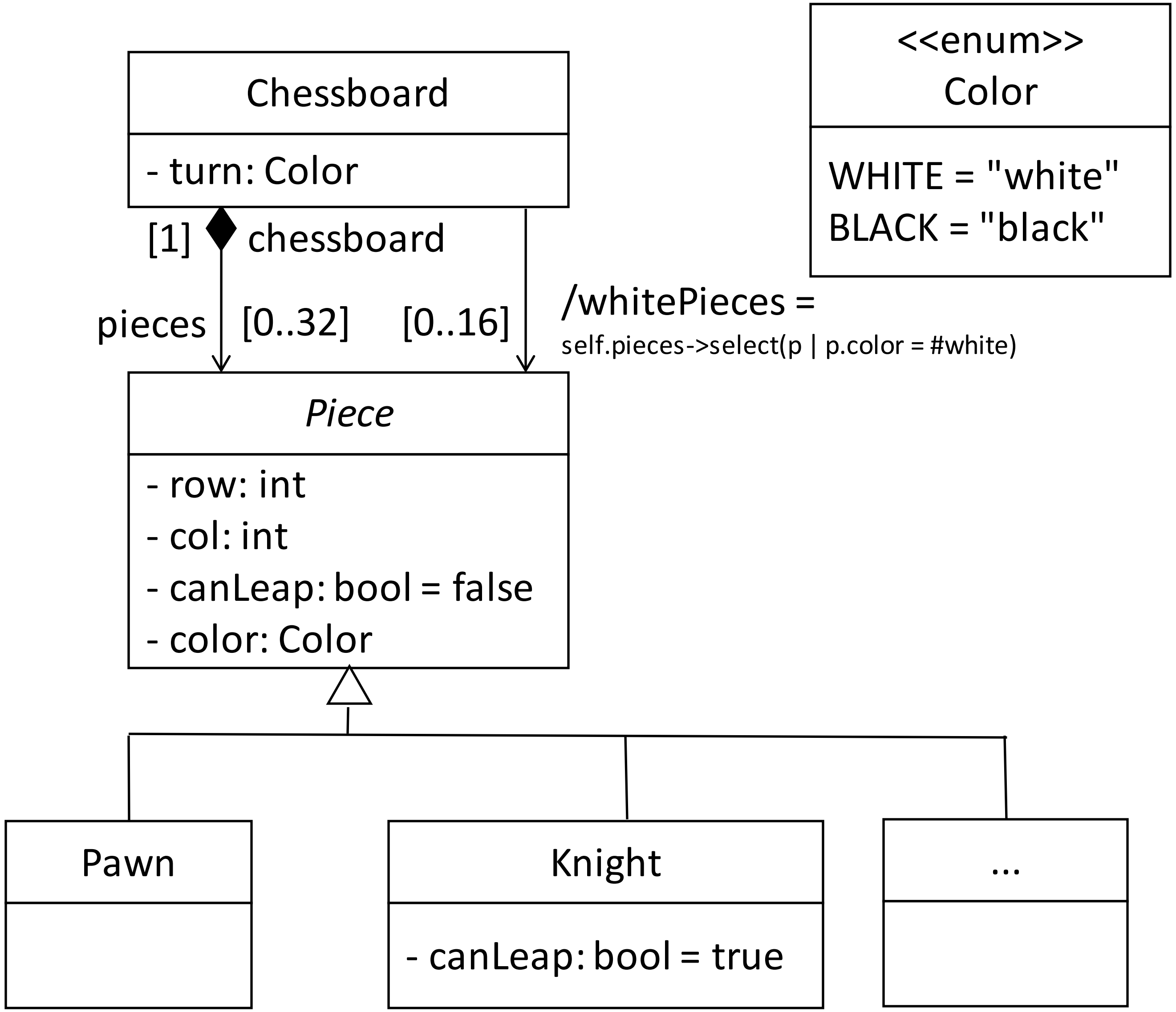
DevOps Oct. 2021

Nicolas Hili <nicolas.hili@univ-grenoble-alpes.fr>

IUT2

Université Grenoble Alpes

Example: Chess



Chessboard

- turn: Color

[1] chessboard

pieces [0..32] [0..16]

/whitePieces =
self.pieces->select(p | p.color = #white)

<<enum>>
Color

WHITE = "white"
BLACK = "black"

Piece

- row: int
- col: int
- canLeap: bool = false
- color: Color

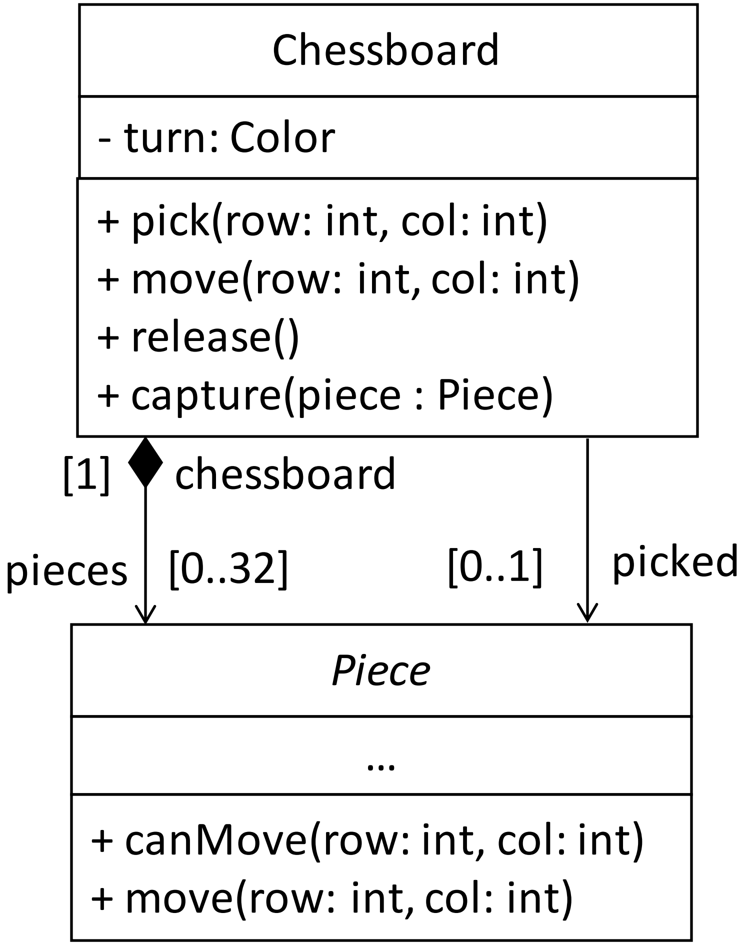
Pawn

Knight

- canLeap: bool = true

...

Example: Chess



Chessboard

- turn: Color

+ pick(row: int, col: int)
+ move(row: int, col: int)
+ release()
+ capture(piece : Piece)

[1] chessboard

pieces [0..32]

[0..1] picked

Piece

…

+ canMove(row: int, col: int)
+ move(row: int, col: int)

Return **TRUE** if:

The piece is moved according to its movement rules

and

The destination is on the board and unoccupied

or

The destination is occupied by an opponent's piece

and

Every cell b/w the current location and the destination is empty

or

The piece can leap

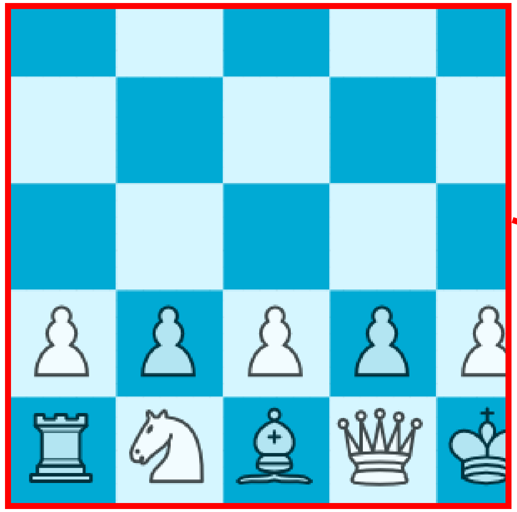
and

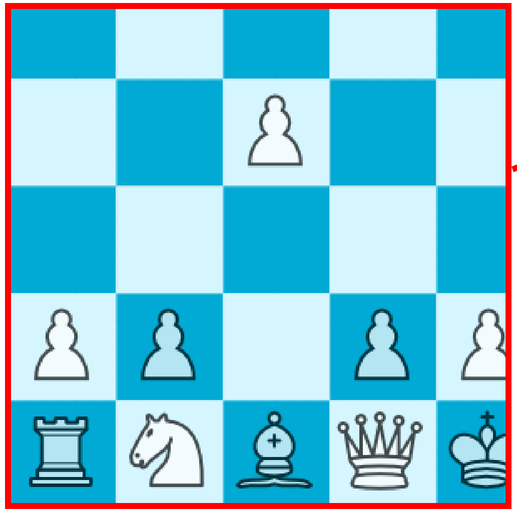
The player's king will not be in check at the end of the turn

Return **FALSE** otherwise

What would be the specification of the function `canMove(row: int, col: int)`?

Test-Driven Development





Test suite #1: pawn moving

Test case #1: pawn can only move forward or diagonally

Test case #2: pawn can move two squares on its first move

Description: on its first move a pawn can advance two squares along the same file, provided both squares are unoccupied.

Initial model state:

Operation name: 'move'

Expected model state:

Operation context: 'White Pawn 3'

Operation parameters: [0, 0]

Expected operation result: true

Test case #3: pawn can move one square one its first move

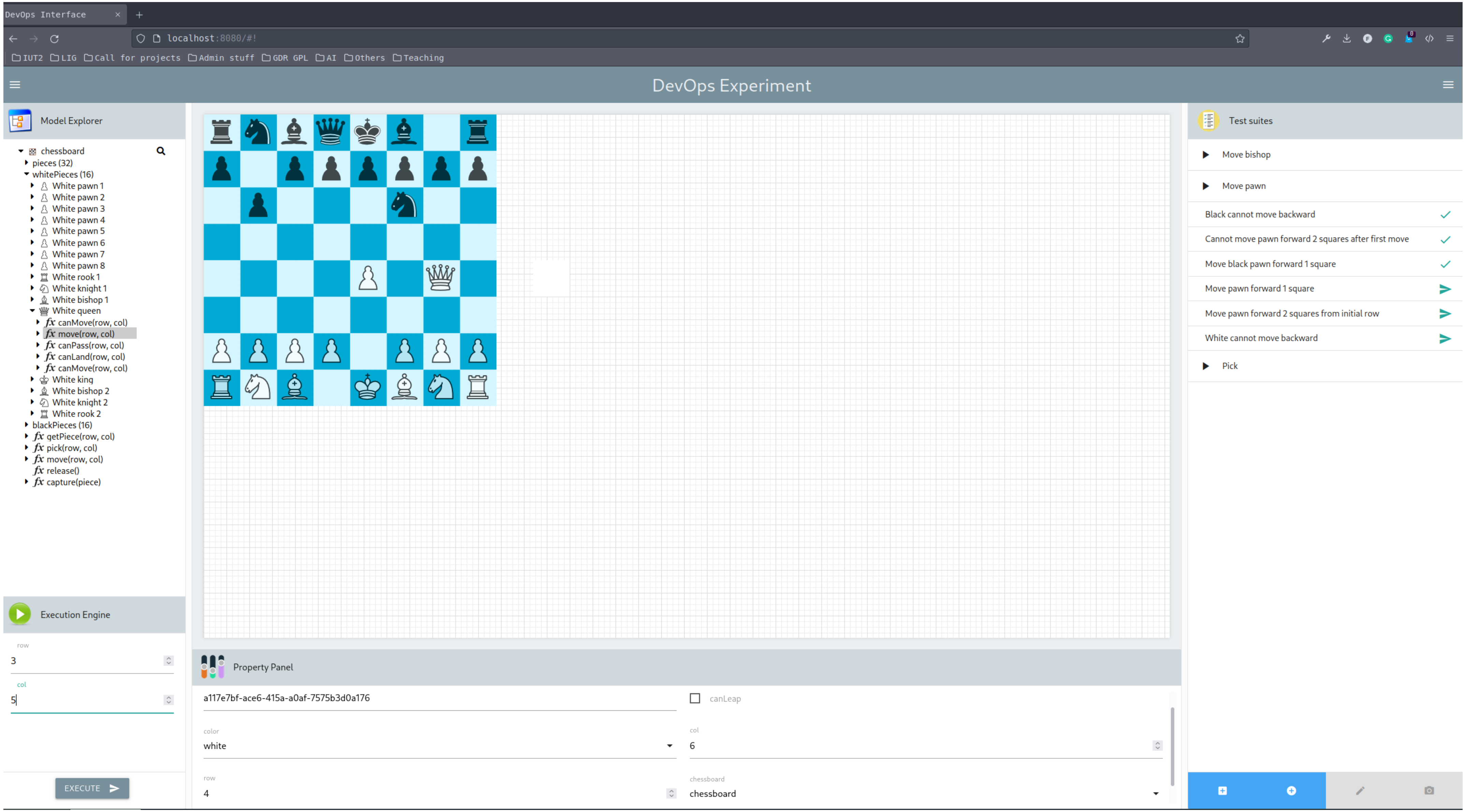
Test case #4: pawn can move one square on subsequent moves

Test case #5: pawn can move diagonally to capture

...

Test suite #2: ...

FlexiMeta



- chessboard
  - pieces (32)
  - whitePieces (16)
    - White pawn 1
    - White pawn 2
    - White pawn 3
    - White pawn 4
    - White pawn 5
    - White pawn 6
    - White pawn 7
    - White pawn 8
    - White rook 1
    - White knight 1
    - White bishop 1
    - White queen
      - fx canMove(row, col)
      - fx move(row, col)
      - fx canPass(row, col)
      - fx canLand(row, col)
      - fx canMove(row, col)
    - White king
    - White bishop 2
    - White knight 2
    - White rook 2
  - blackPieces (16)
  - fx getPiece(row, col)
  - fx pick(row, col)
  - fx move(row, col)
  - fx release()
  - fx capture(piece)

row: 3

col: 5

Move bishop	
Move pawn	
Black cannot move backward	✓
Cannot move pawn forward 2 squares after first move	✓
Move black pawn forward 1 square	✓
Move pawn forward 1 square	➤
Move pawn forward 2 squares from initial row	➤
White cannot move backward	➤
Pick	

a117e7bf-ace6-415a-a0af-7575b3d0a176

□ canLeap

color: white

col: 6

row: 4

chessboard: chessboard

GitLab workflow

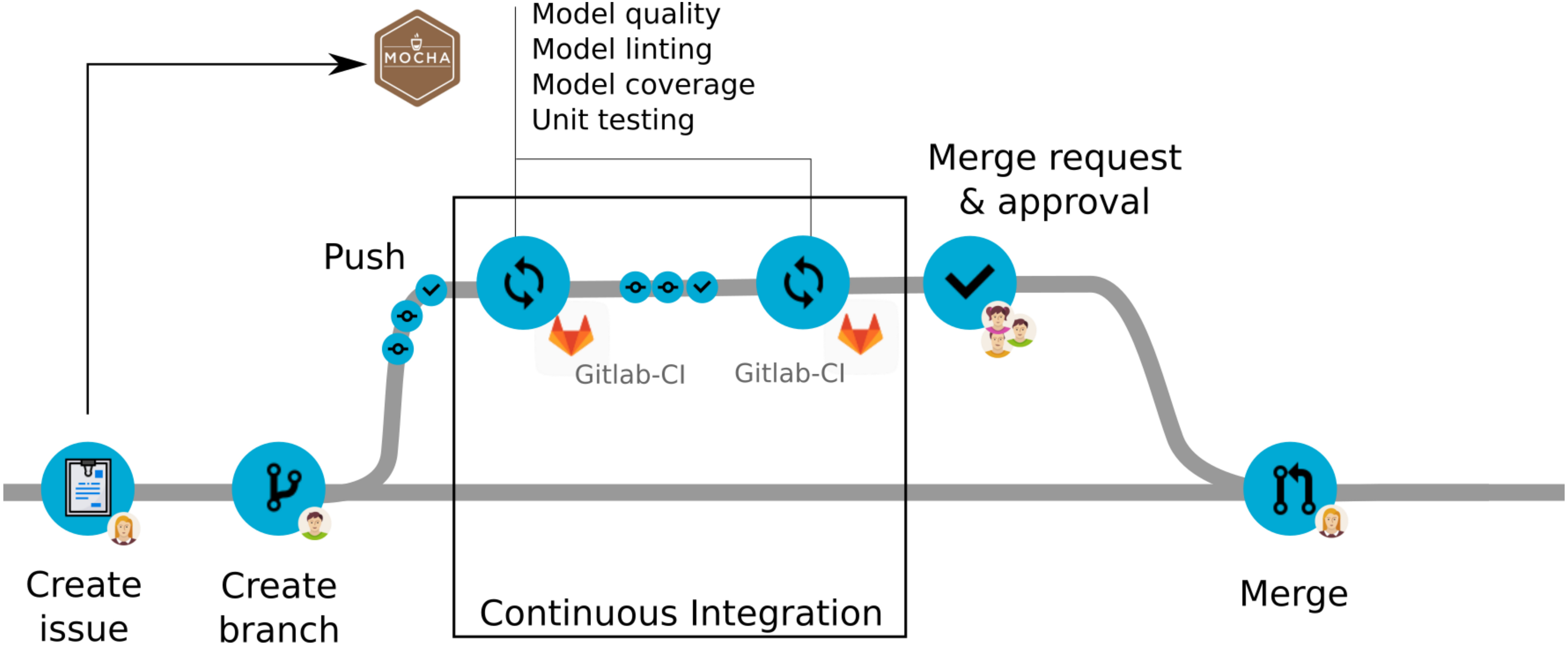


Figure adapted from https://docs.gitlab.com/ee/ci/introduction/