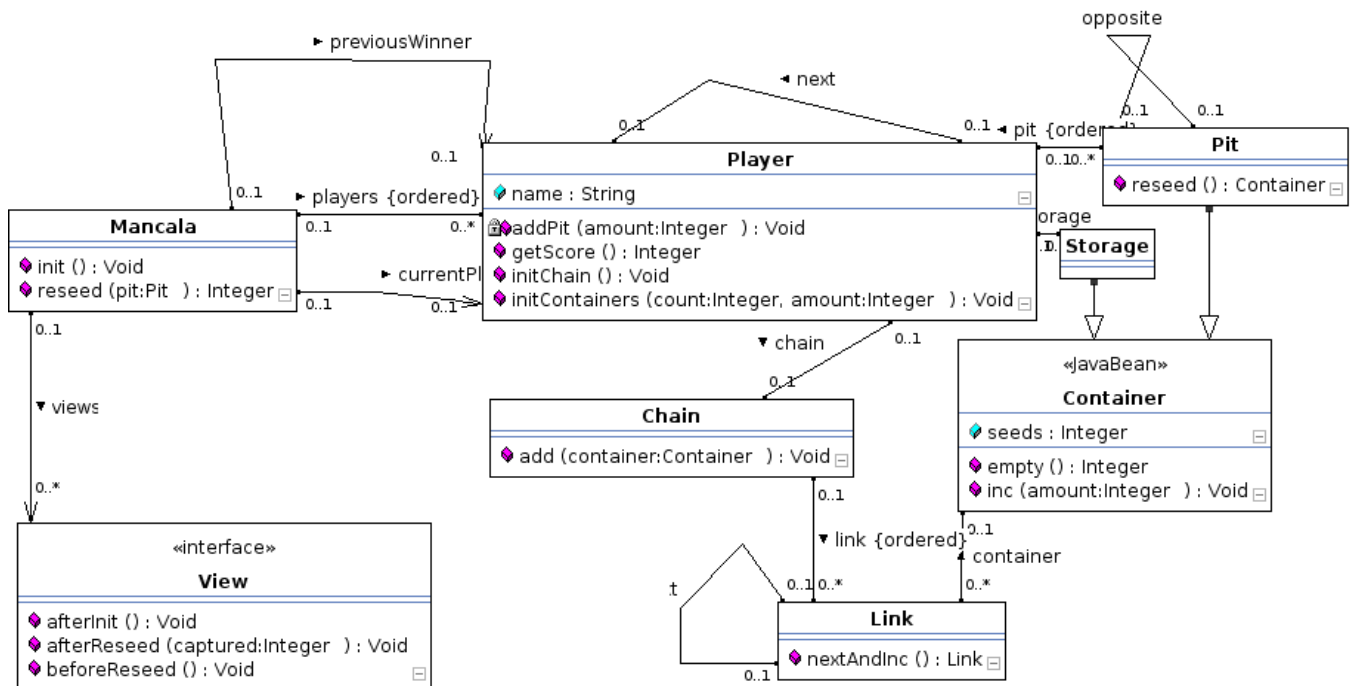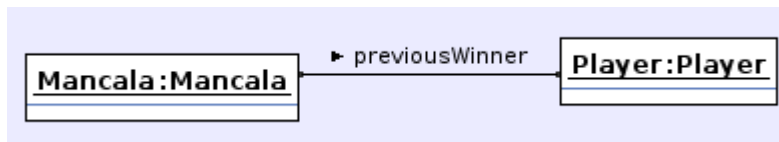# Diagrams

1.Class Diagram:
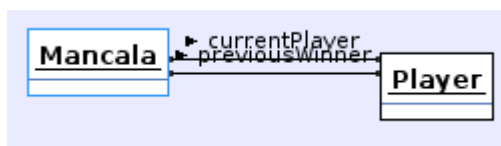

2. User Story 2

Pre-condition:

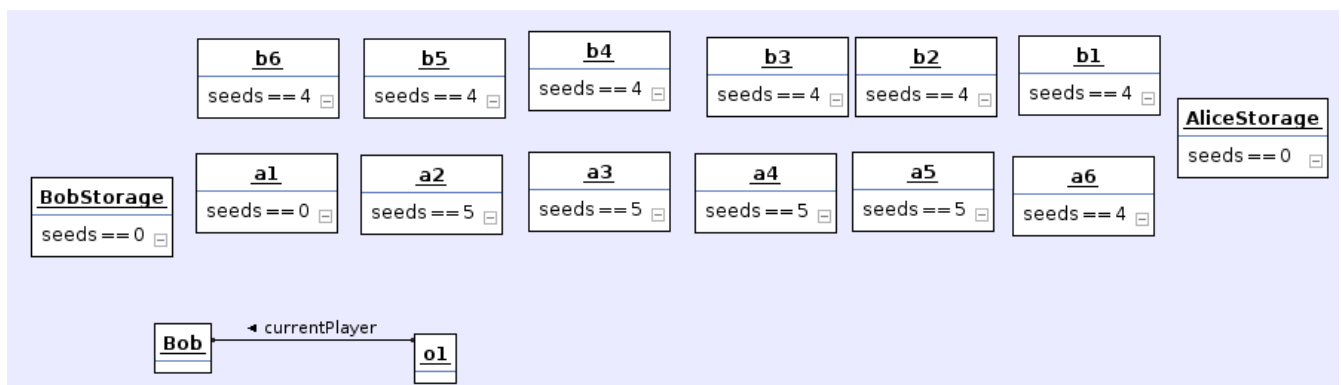
Post-condition:

3. User Story 3

Pre-condition:
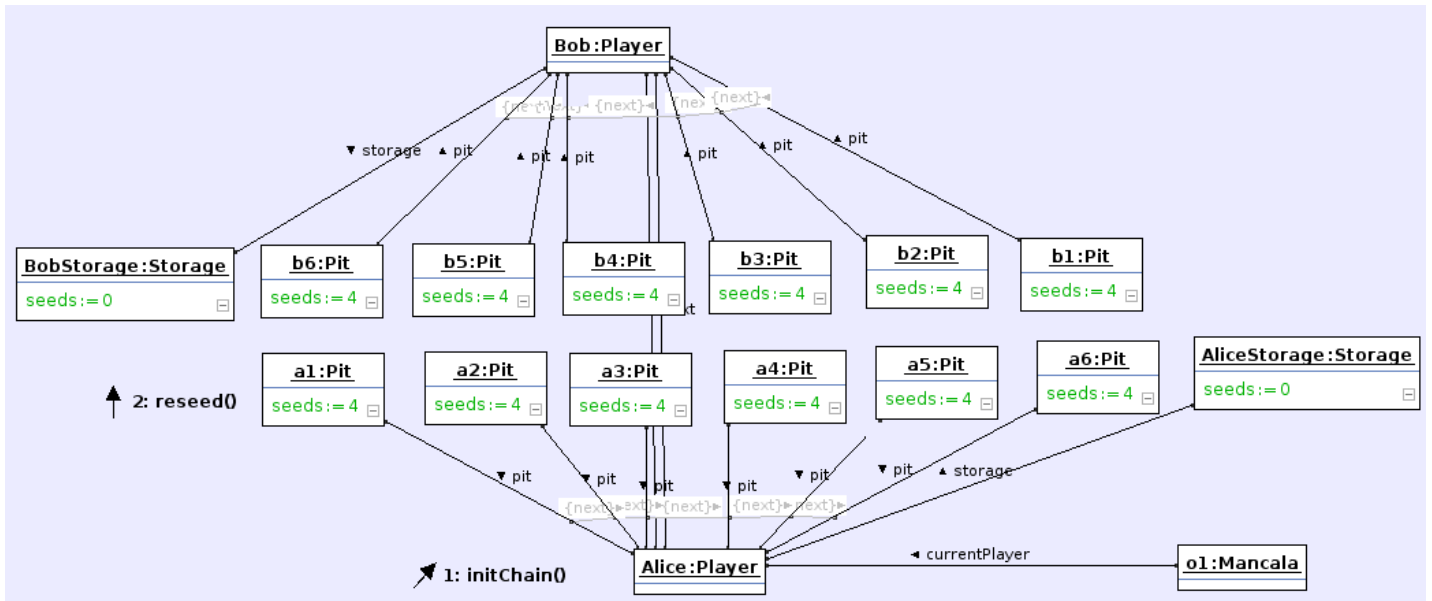
Post-condition:

4. User Story 4:

Pre-condition:

Post-condition:

BobStorage	b6	b5	b4	b3	b2	b1
seeds == 0	seeds == 4	seeds == 4	seeds == 4	seeds == 4	seeds == 4	seeds == 4

a1	a2	a3	a4	a5	a6	AliceStorage
seeds == 4	seeds == 4	seeds == 0	seeds == 5	seeds == 5	seeds == 5	seeds == 1

◄ currentPlayer

Alice ——— Mancala
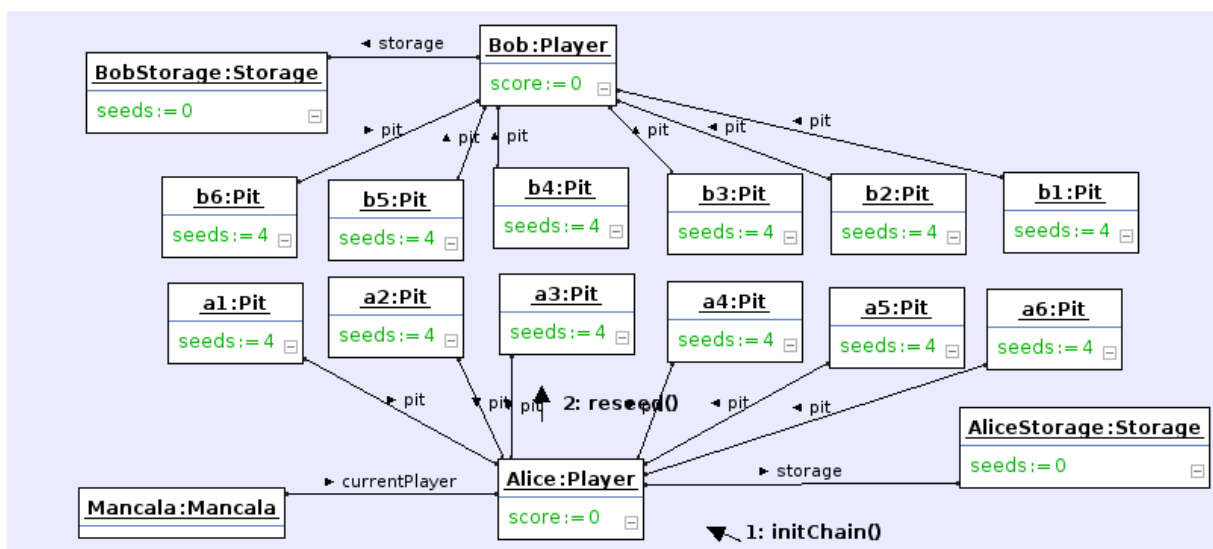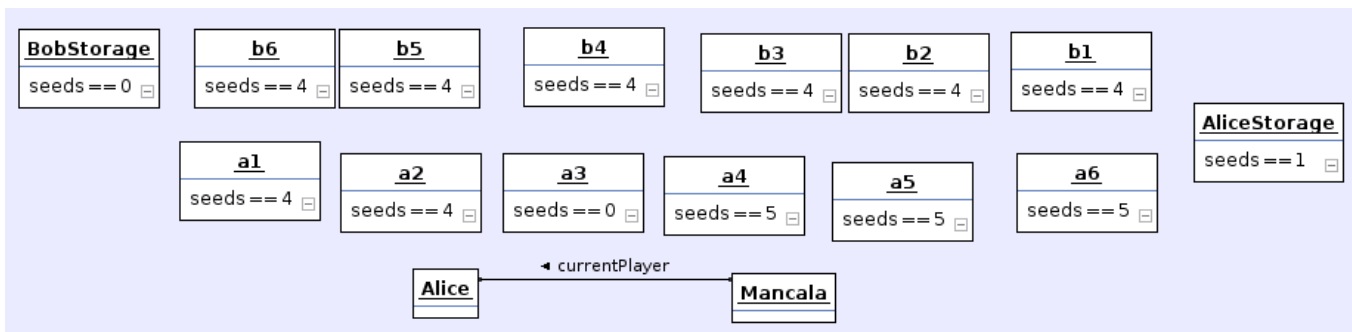
5.User Story 5

Pre-condition:
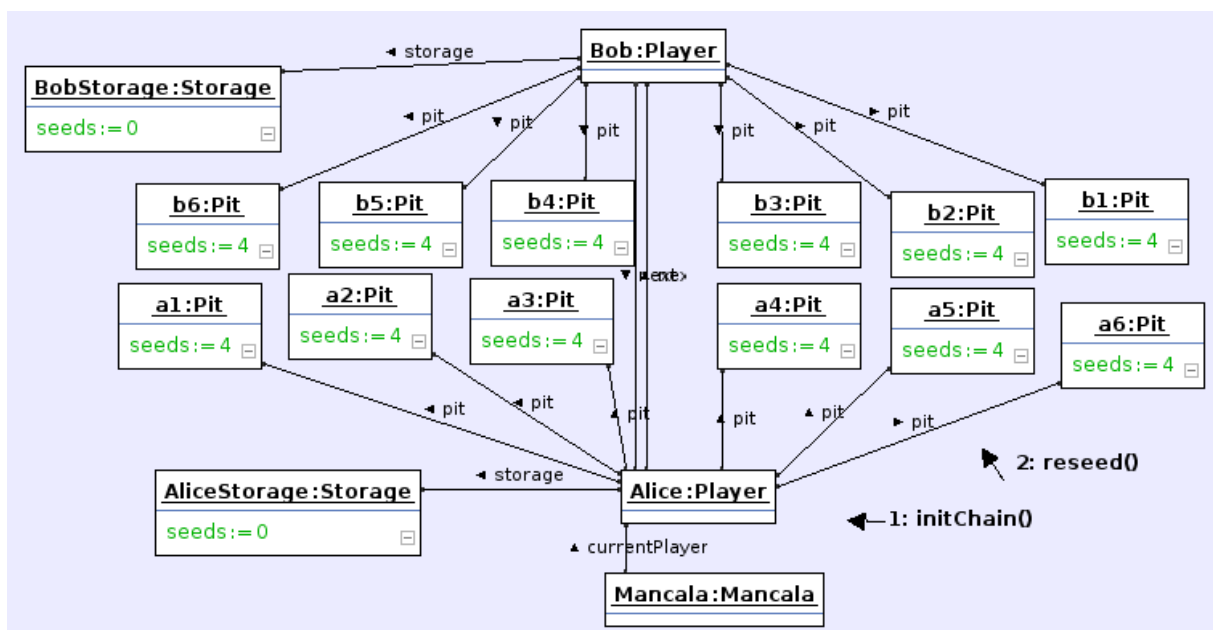
Post-condition:
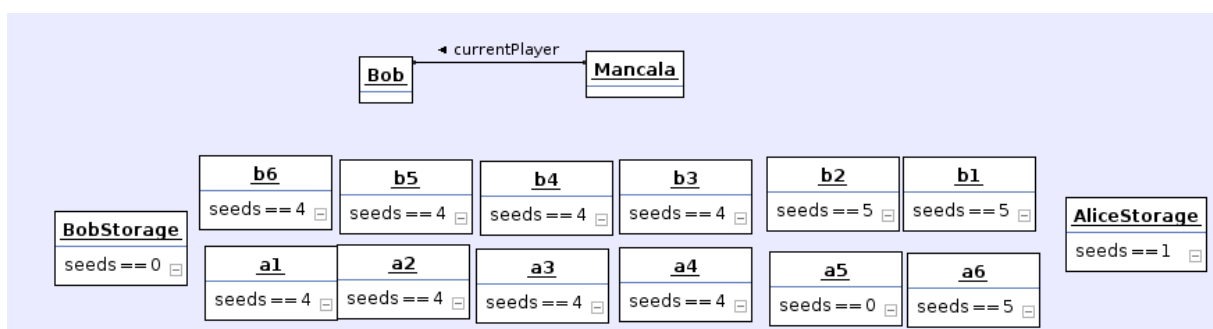
6. User Story 6

Pre-condition:
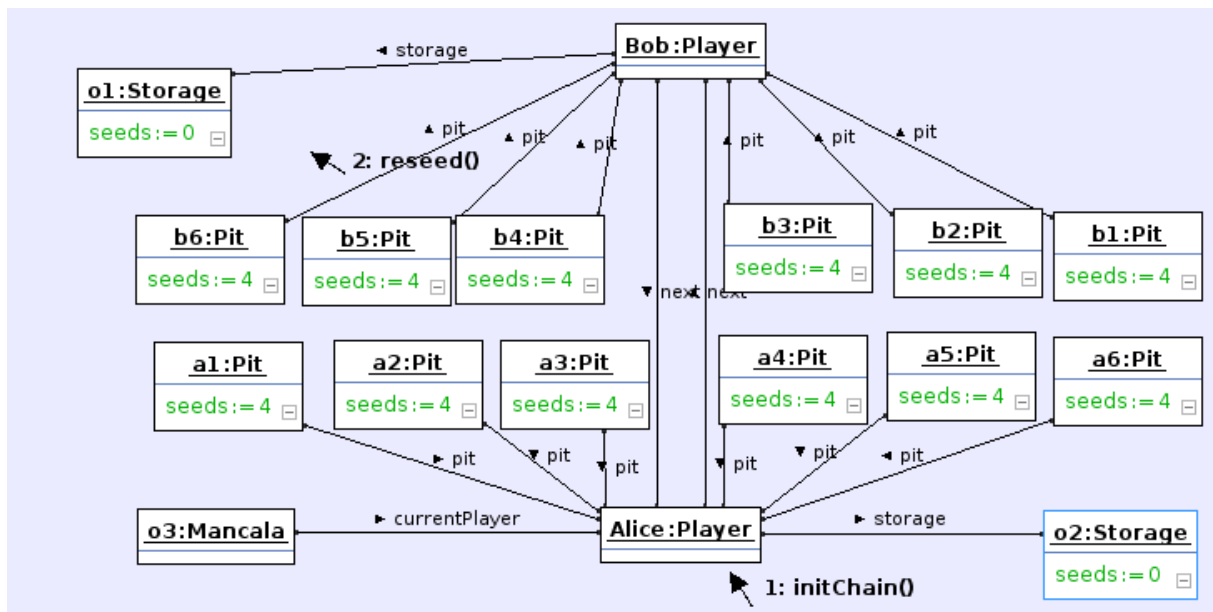
Post-condition:
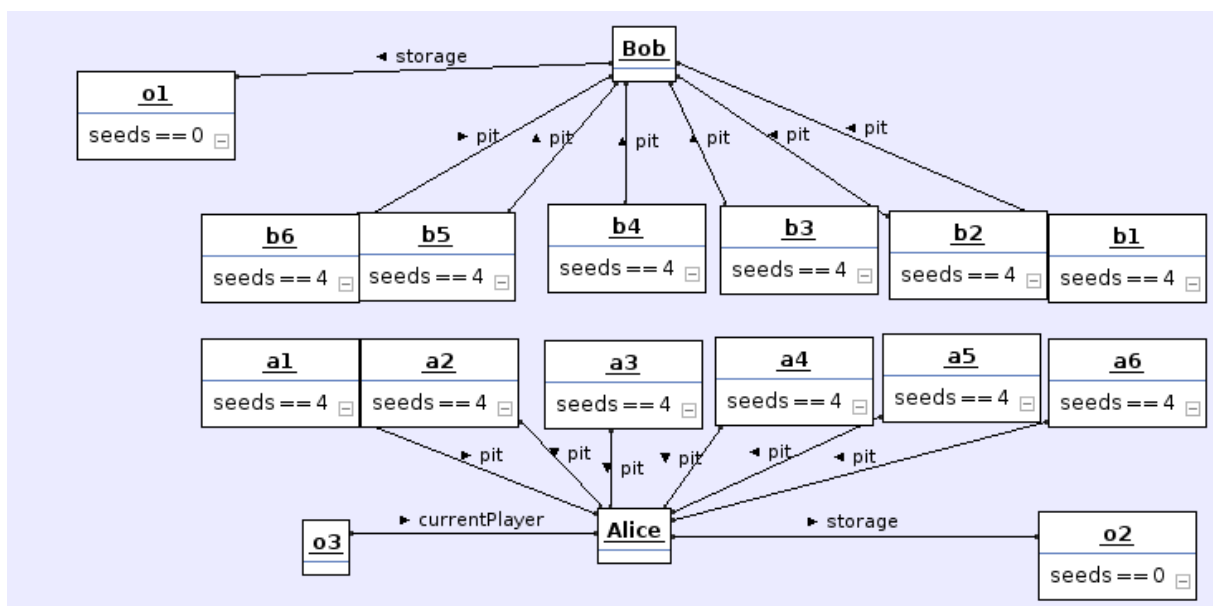
7. User Story 7

Pre-condition:
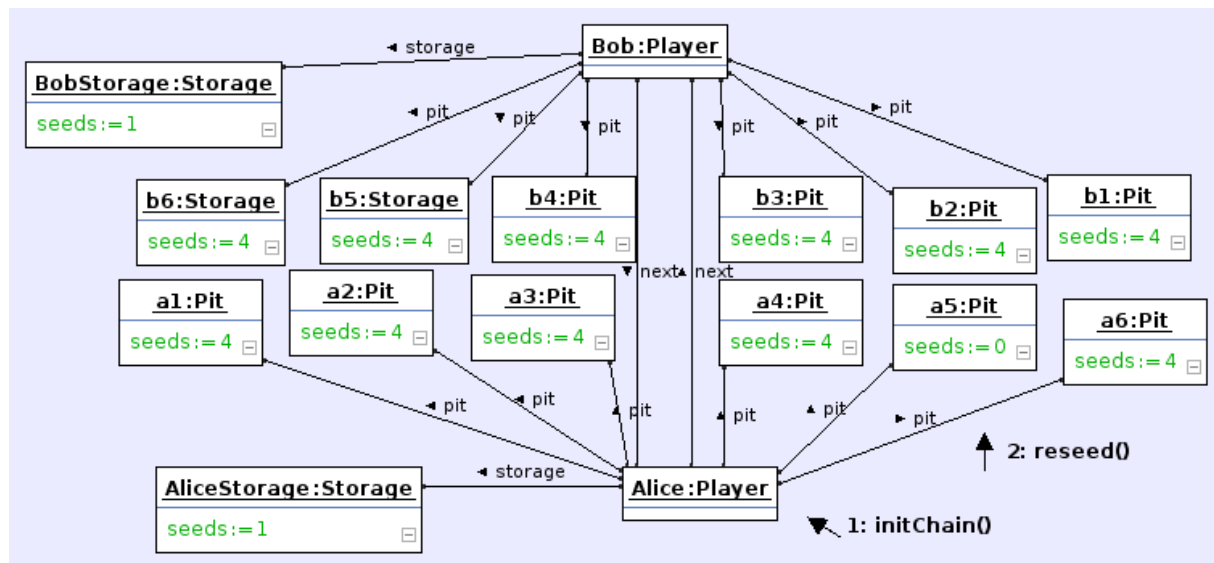
Post-condition:
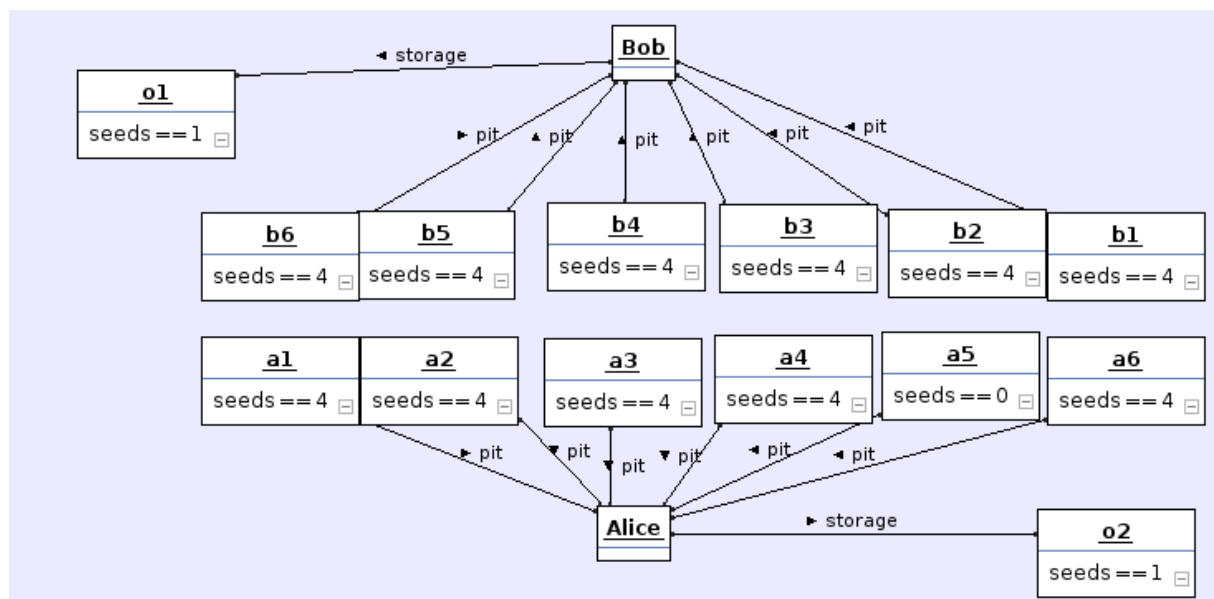
8. User Story 9:

Pre-condition:
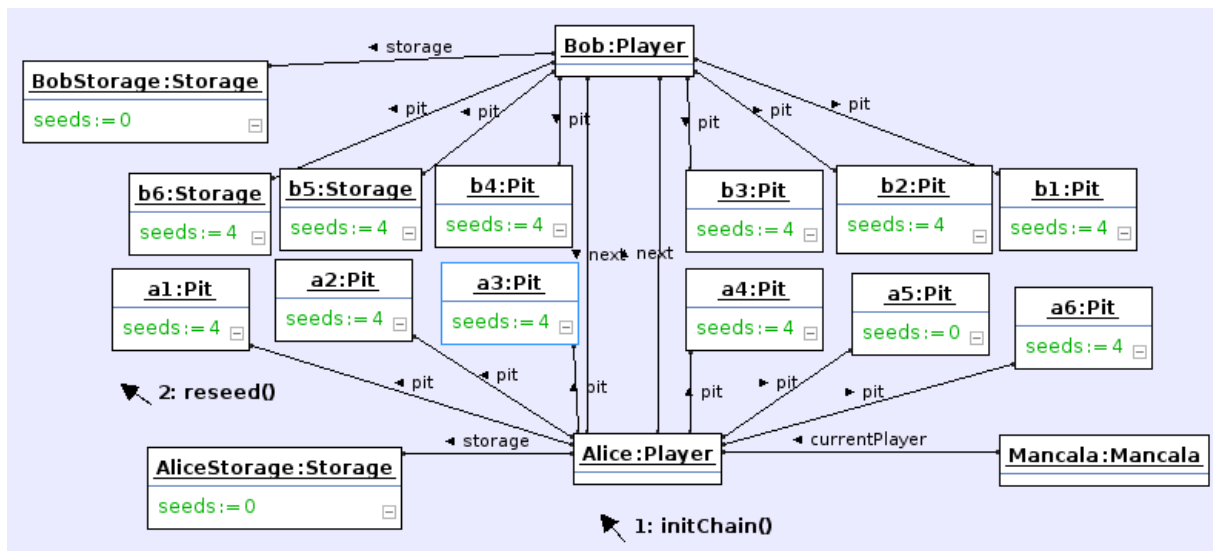
Post-condition:

9. User Story 10

Pre-condition:
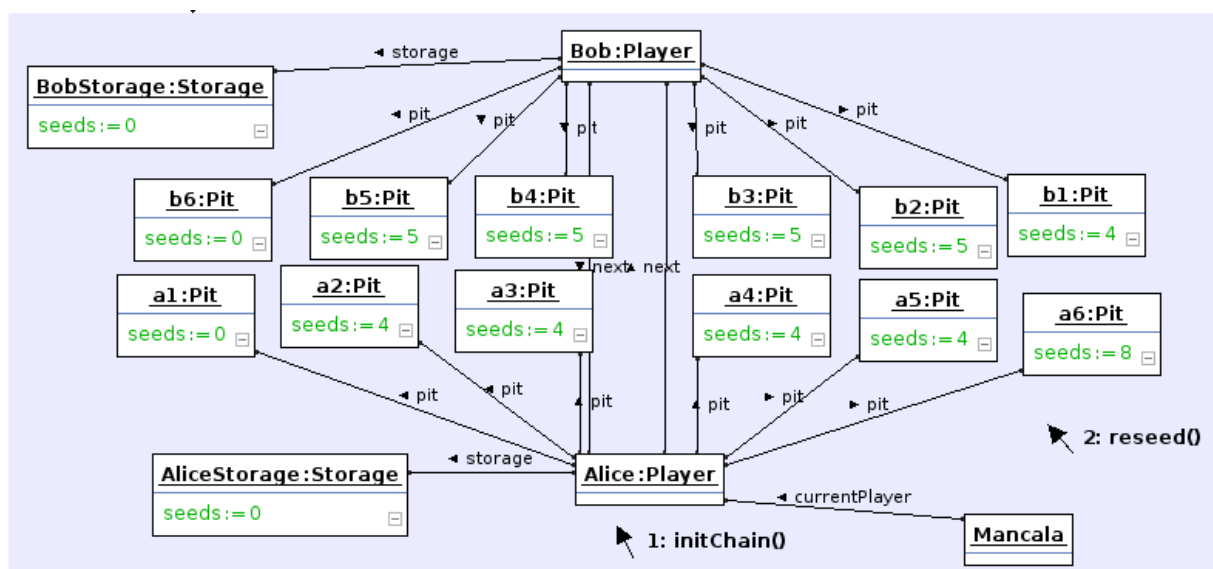
Post-condition:
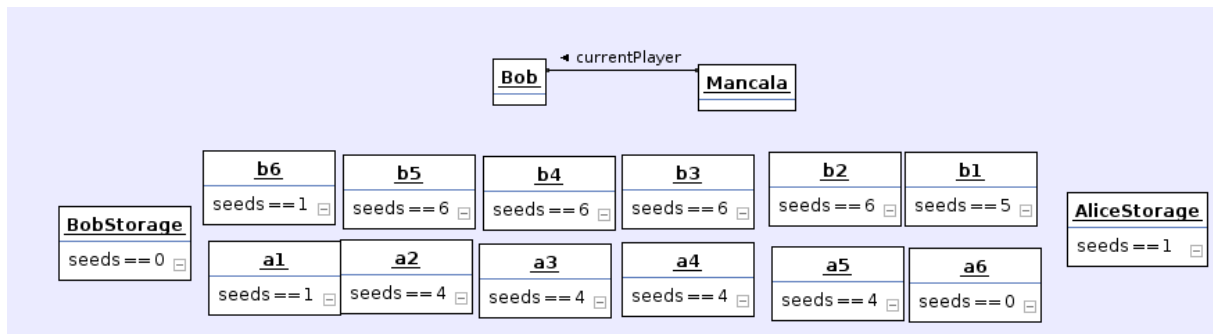
10. User Story 11

Pre-condition:
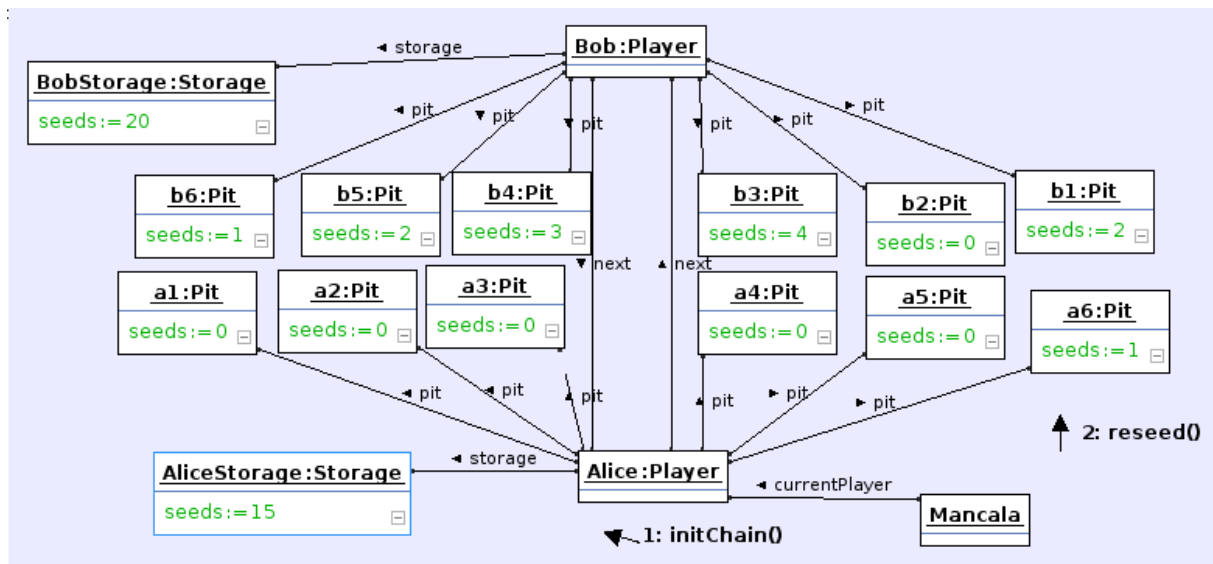
Post-condition:
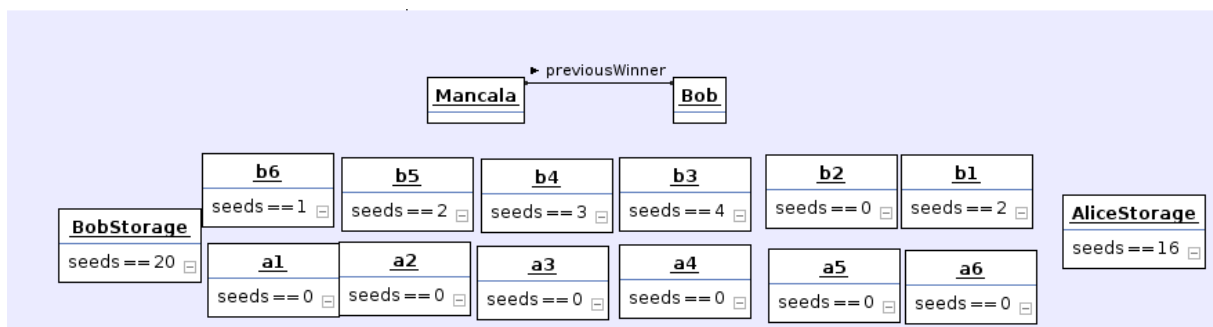
11. User Story 12

Pre-condition:
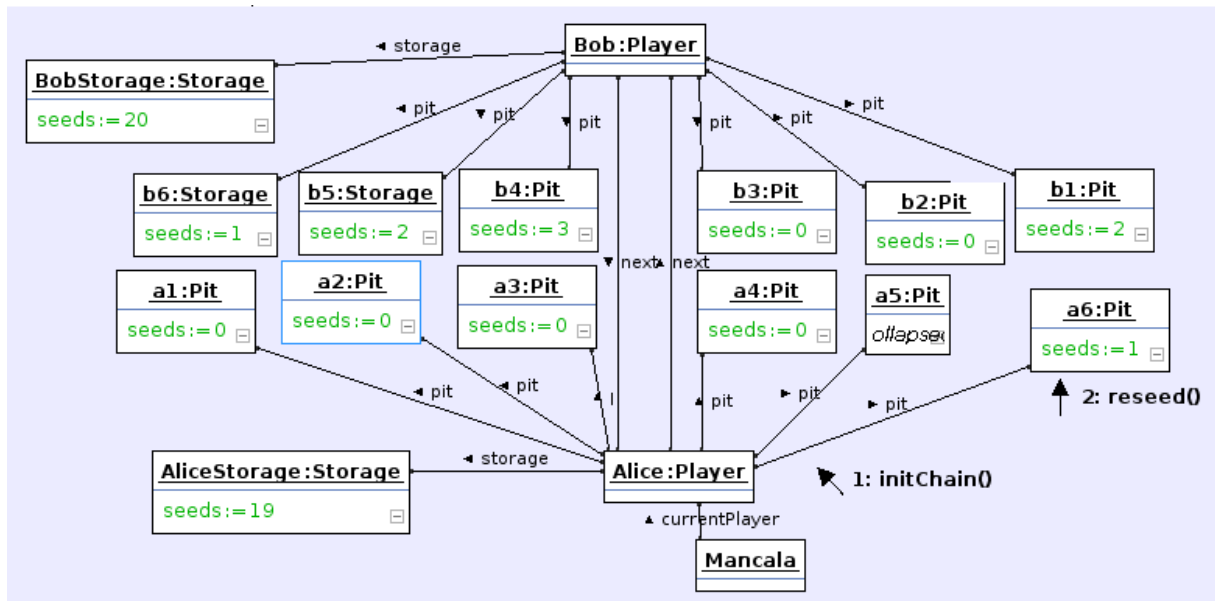
Post-condition:
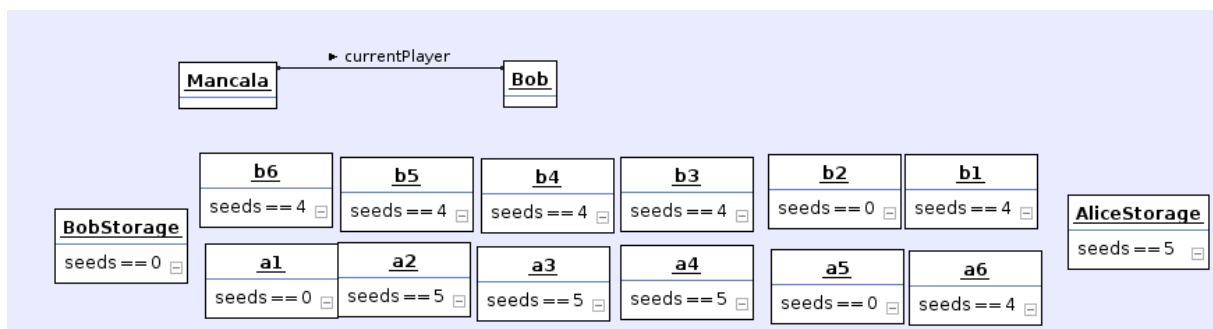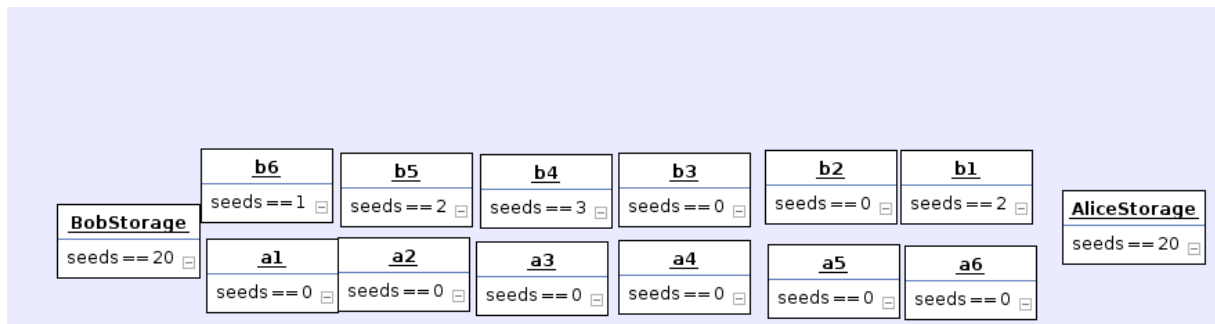
12. Chain
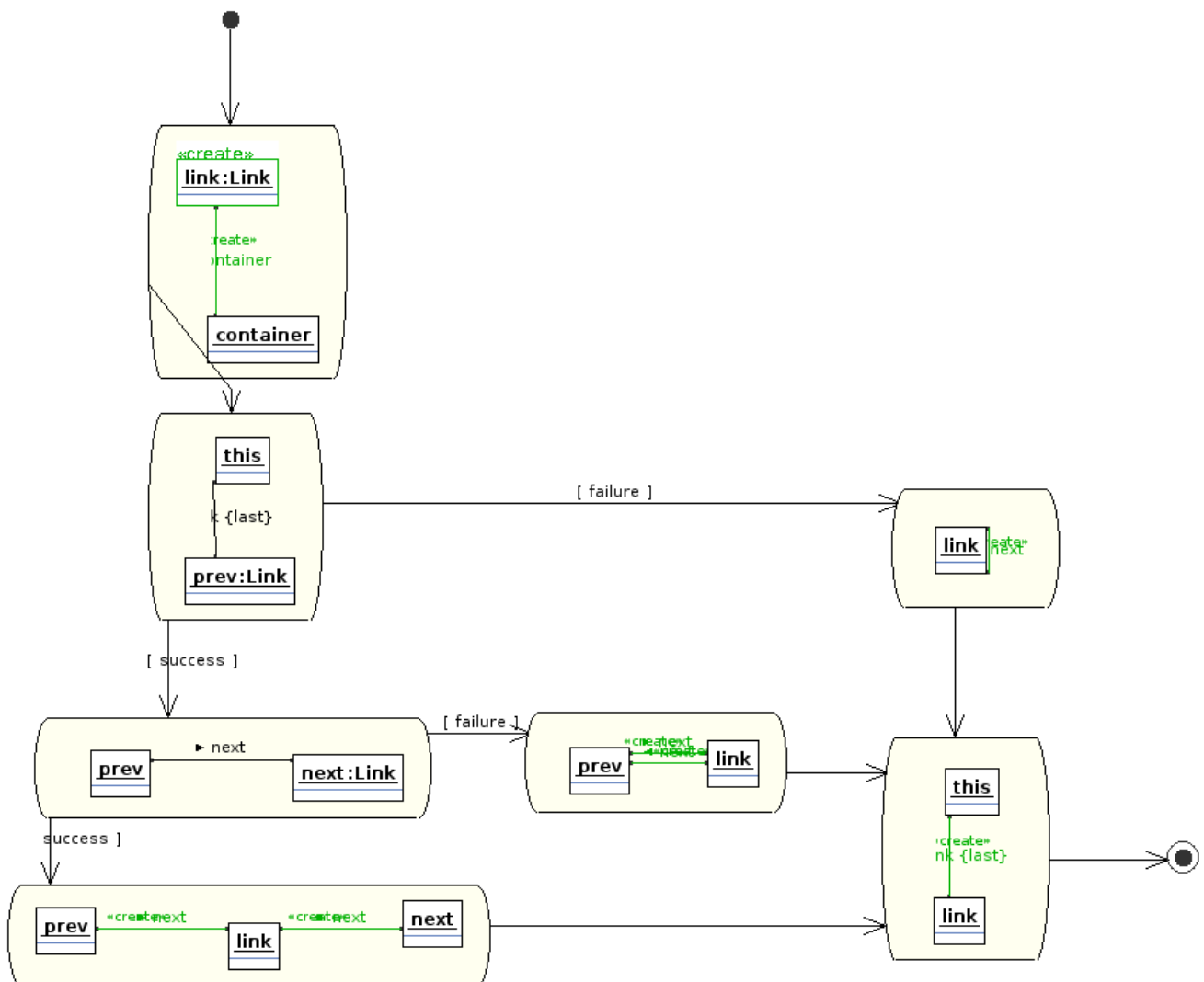
13.Container which is for empty

Container::empty (): Integer
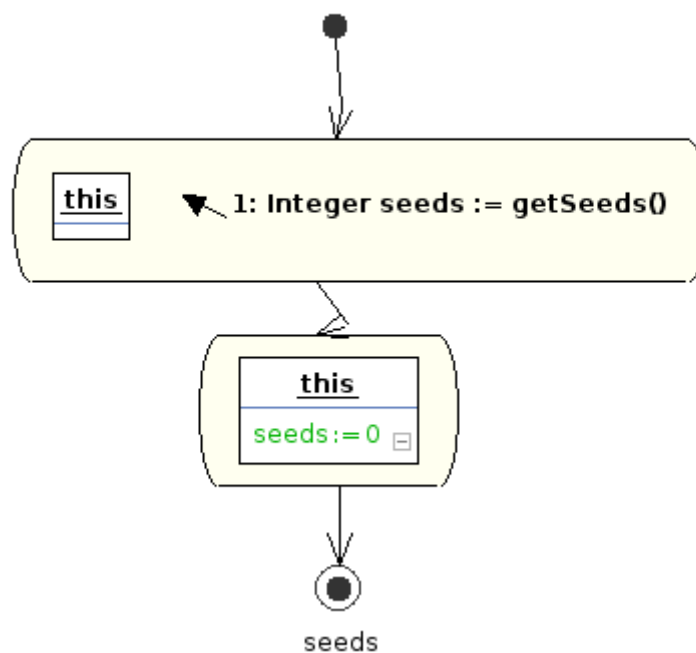
seeds

14. Container which is increased by seeds

Container::inc (amount: Integer): Void

```
┌─────────────────────────────────────────┐
│ ┌──────┐                                 │
│ │ this │  ▼◣ 1: Integer seeds := getSeeds() │
│ └──────┘                                 │
└─────────────────────────────────────────┘
```

```
┌─────────────────────────────────┐
│  ┌───────────────────────────┐  │
│  │           this            │  │
│  ├───────────────────────────┤  │
│  │ seeds := seeds + amount  ⊟ │  │
│  └───────────────────────────┘  │
└─────────────────────────────────┘
```
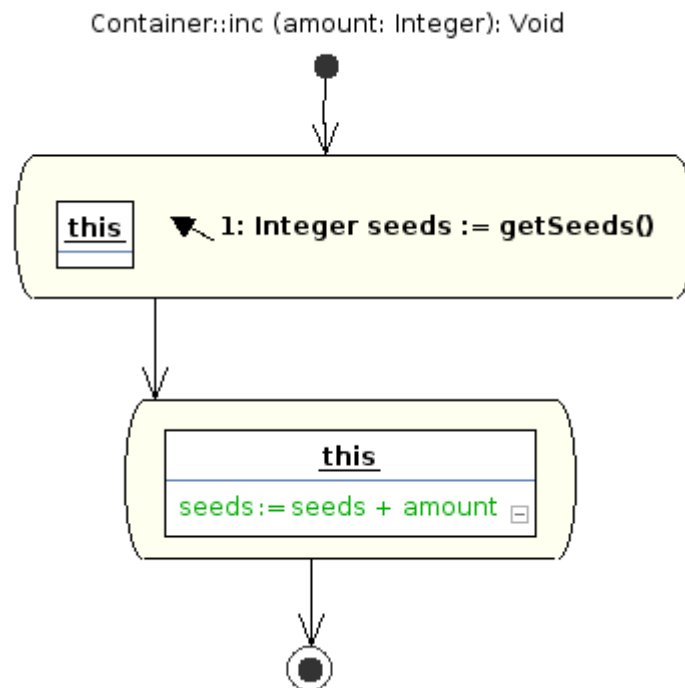
15.Link which is on the next position

Link::nextAndInc (): Link

```
┌───────────────────────────────────────────────────────────────┐
│        ▸ next              ▸ container                         │
│ ┌──────┐   ┌──────────┐        ┌────────────────────┐         │
│ │ this │   │ next:Link │        │ container:Container │         │
│ └──────┘   └──────────┘        └────────────────────┘         │
│                                       ▼◣ 1: inc(1)             │
└───────────────────────────────────────────────────────────────┘
```

next
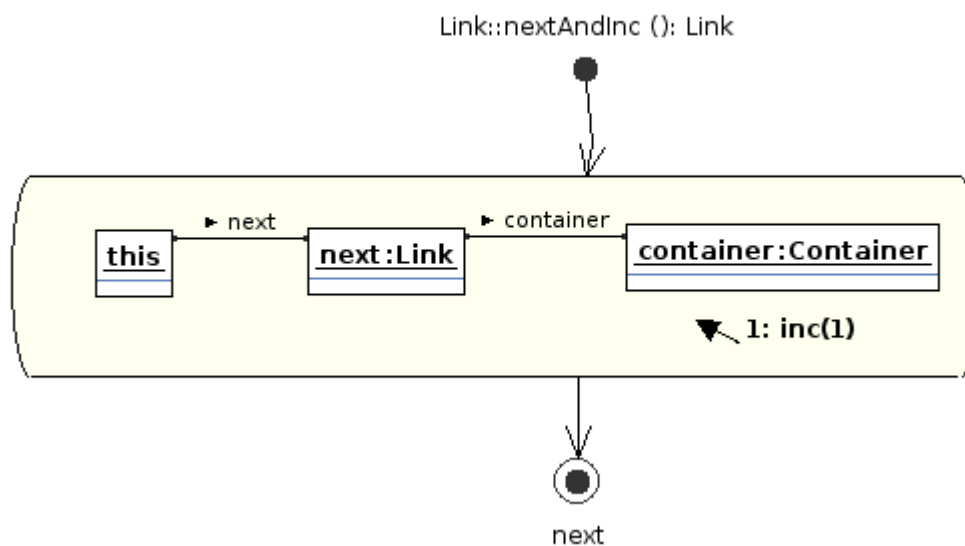
16. Reseeding from Mancala

Mancala::reseed (pit: Pit): Integer

// ensure pit belongs to current player

this ▸ currentPlayer → player:Player ◂ pit ← pit

▾ storage

storage:Storage

[failure] ⊙ -1

this ▸ views → view:View

1: afterReseed(captured)

[success]

// ensure pit is not empty

pit
seeds > 0 ⊟

[failure] ⊙ -1

[success]

this ▸ views → view:View

◥ **1: beforeReseed()**

end]

// perform the actual reseeding

«create»
last:Container := pit.reseed()

[else]   [last == storage]

// turn moves to next player

this «destroy»currentPlayer ——— player:Player

«create»
▸currentPlayer   ▾ next

next:Player

⊙

this ▸ views → view:View

◥ **1: afterReseed(-1)**

[end]

[failure]

// check if last seed ended up in player's empty pit

lastPit := (Pit) last ▸ pit → **player**
seeds == 1 ⊟

[success]

// get number of seeds

lastPit ▸ opposite → **opposite:Pit**   **storage**

◤ 1: Integer lastSeeds := empty()

◤ 2: Integer oppositeSeeds := empty()

◤ 3: inc(lastSeeds)

◤ 4: inc(oppositeSeeds)

5: Integer captured := lastSeeds + oppositeSeeds

captured

17. Initial for Mancala

Mancala::init (): Void

this — ► players {last} — previousPlayer:Player

this — ► players {first} — firstPlayer:Player

[each time]

this — ► players — player:Player

previousPlayer «create» player

1: initContainers(6, 4)

previousPlayer := player

[end]

this — ► players — player:Player

[each time]

player

1: initChain()

[end]

this — ► previouswinner — previousWinner:Player

this — ► players — player:Player — previousWinner

collapsed ⊞

[success]

[failure]

this — ► currentPlayer — player

this — firstPlayer

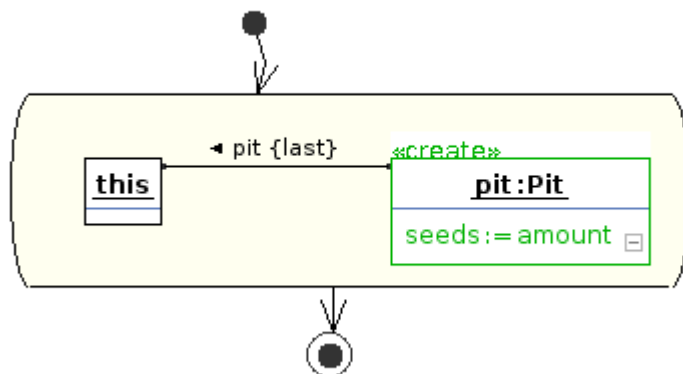// <add comment>

this — ► views — view:View

1: afterInit()

end]

18.Reseeding from Pits

19.Adding pit for the player

Player::addPit (amount: Integer): Void

20. Getting score from the player

Player::getScore (): Integer

this ► storage storage:Storage

storage.getSeeds()
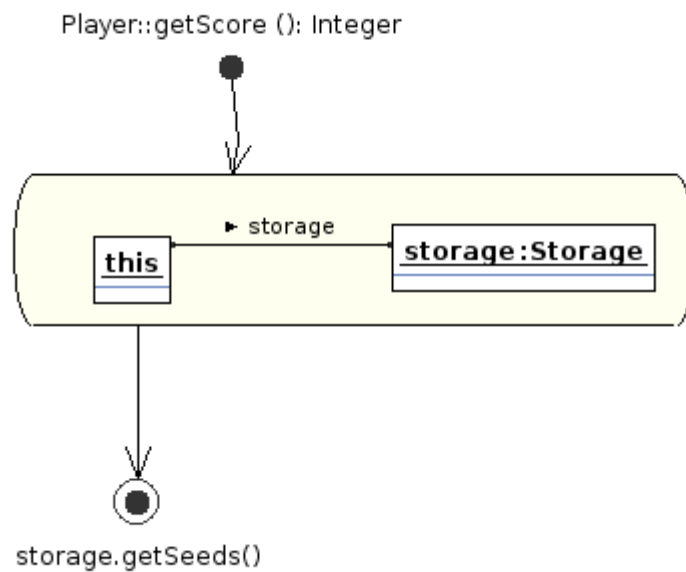
21. Initial chain for the player

Player::initChain (): Void

player:= this ► chai «create» chain:Chain

player ◄ pit pit:Pit

[each time] chain 1: add(pit)

end]

[player == this] chain 1: add(storage)

[else]

player ► next next:Player

[else]

player:= next

[player == this]
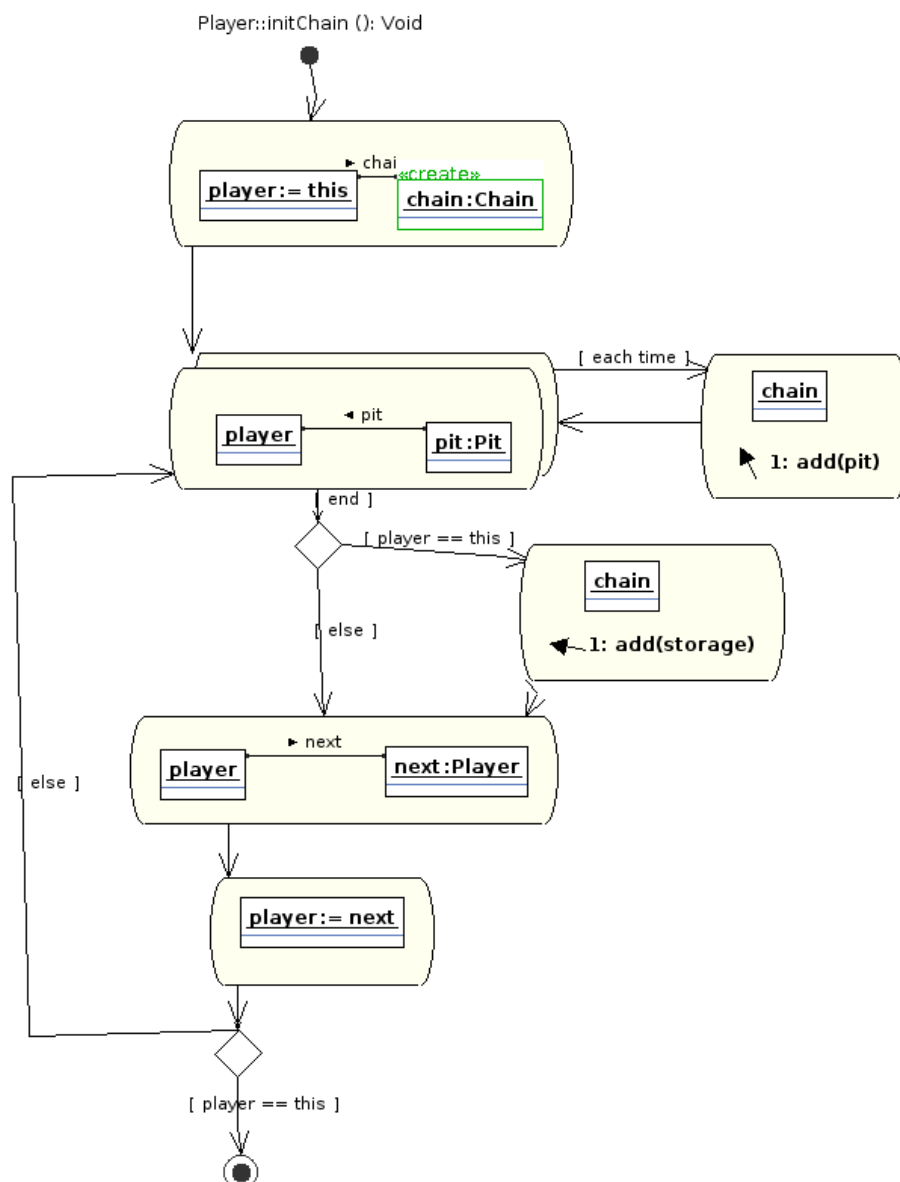
22. Initial container for the player

Player::initContainers (count: Integer, amount: Integer): Void