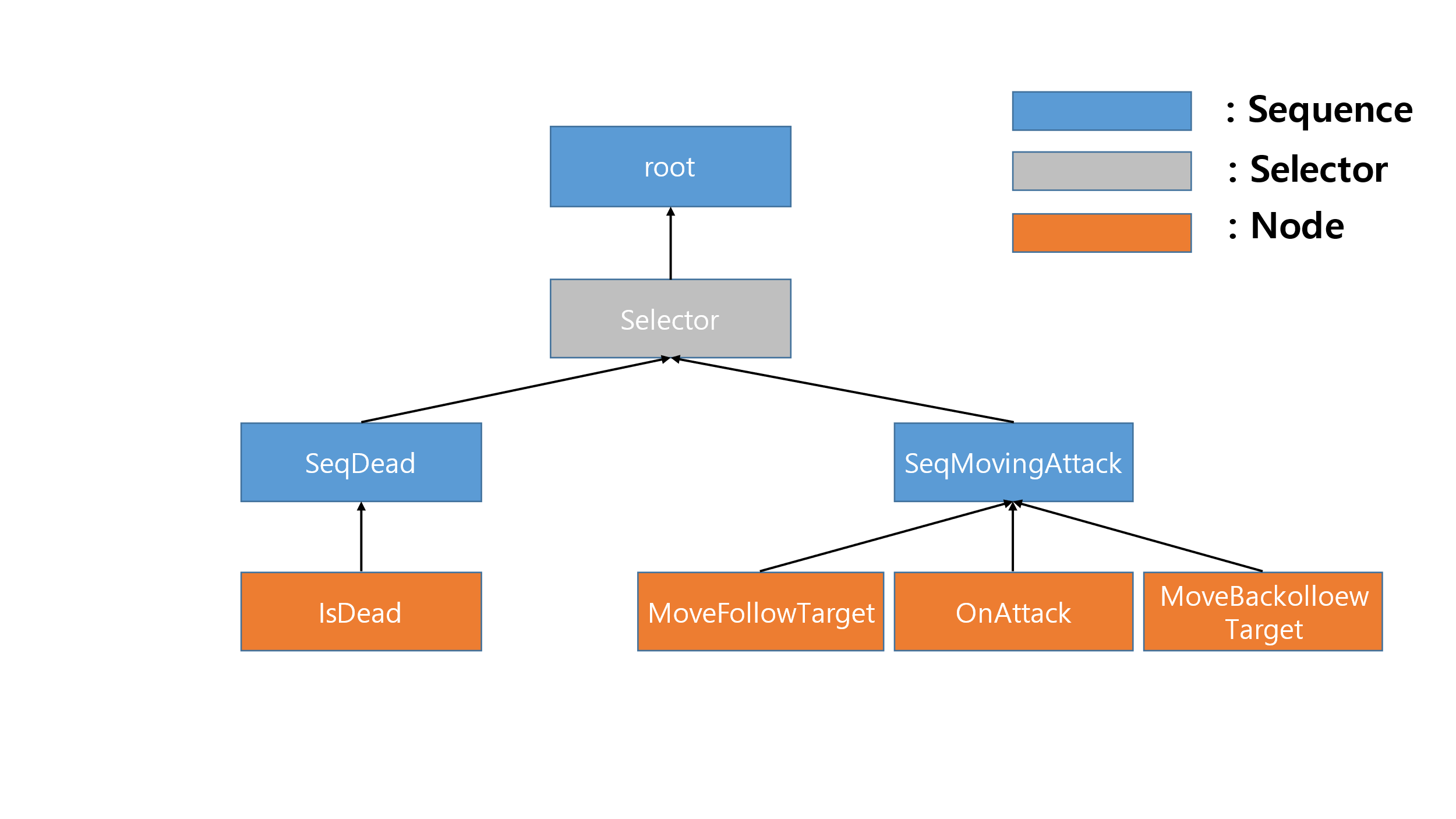
**-BT에 대한 설명**

탱크 AI는 위 그림과 같은 구조로 형성되었다. Root가 Selector에게서 탱크가 수행되어야할 Node들에 대한 정보를 전달받아 그 노드의 함수들을 실행시킨다. 이때, Selector는 탱크의 죽음 조건부터(SeqDead) 체크하고, 죽었다고 판단되면 root에게 전달하여 적을 소멸시킨다. 죽음 조건이 충족되지 않았다면 Selector가 다음 시퀀스로(SeqMovingAttack) 넘어가 MoveFollowTarget, MoveBackollowTarget, AddBullet( OnAttack 노드에서 실행되는 함수 ) 함수들 중 조건에 알맞는 함수를 실행시킨다.

**-노드들의 함수에 대한 설명**

|  |
| --- |
| MoveFollowTarget() |
| INPUT : *Obj1\_Pos, Obj2\_Pos, speed, follow , nTime, Obj2\_Delta\_Esti, Obj2\_Pre\_Pos*  IF *follow* == true do;  Dir <- *Obj2\_Pos* – this.position + *Obj2\_Delta\_Esti*  Dir.Normalize()  Rot <- LookRotation(Dir).Angley  This.Rotation <- Rot  this.position <- this.position + Dir \* *speed*  IF *nTime* % 100 <= 1 do;  IF *Obj2\_Pre\_Pos* != (0,0,0) do;  Obj2\_Delta\_Pos <- *Obj2\_Pos* - *Obj2\_Pre\_Pos*  *Obj2\_Delta\_Esti.x* <- RBE(*Obj2\_Delta\_Pos.x*, *Obj2\_Delta\_Esti.x*)  *Obj2\_Delta\_Esti.z* <- RBE(*Obj2\_Delta\_Pos.z, Obj2\_Delta\_Esti.z*)  End IF  *Obj2\_Pre\_Pos* <- *Obj2\_Pos*  IF Distance(*Obj1\_Pos, Obj2\_Pos*) < 30.0f do;  *follow* <- FALSE  else  *follow* <- TRUE  End IF  End IF  RETURN TRUE  End IF  RETURN FALSE: |

1수도코드는 MoveFollowTarget 함수에 대해 설명한다. MoveFollowTarget 함수는 follow 변수가 true일 때 적의 위치를 예상하여 조준하고 전진하는 처리를 하는 함수이다. 일정주기마다 적의 위치를 예측하는 계산을 하고 적의 이전위치에 대한 변수를 적의 현재위치로 갱신하며 적과의 거리에 따른 follow 변수에 변화를 준다.

follow는 true일때 전진을, false일때 후진을 의미하도록 설정하였다.

*Obj1\_Pos* 는 탱크 자신의 위치를 의미한다.

*Obj2\_Pos* 는 적의 위치를 의미한다.

*speed*는 자신의 속력을 의미한다.

*nTime*은 게임이 시작된 후 일정주기로 일정수치만큼 증가한다.

*Obj2\_Delta\_Esti* 는 Dir에 더해지는 벡터로, 적의 경로를 예측한 값을 나타낸다. .  
*Obj2\_Pre\_Pos*는 적의 이전 위치를 의미한다.

Normalize()는 벡터를 단위벡터화하는 함수이다.

RBE() 함수는 recursive bayesian estimation을 이용하여 적의 위치 벡터에 대한 변화량을 각 성분마다 추정하기 위해서 정의한 함수이다.

|  |
| --- |
| MoveBackollowTarget() |
| INPUT : *Obj1\_Pos, Obj2\_Pos, speed, follow , nTime, Obj2\_Delta\_Esti, Obj2\_Pre\_Pos*  IF *follow* == true do;  Dir <- *Obj2\_Pos* – this.position + *Obj2\_Delta\_Esti*  Dir.Normalize()  Dir.RotationY(30)  Rot <- LookRotation(Dir).Angley  This.Rotation <- Rot  this.position <- this.position - Dir \* *speed*  IF *nTime* % 100 <= 1 do;  IF *Obj2\_Pre\_Pos* != (0,0,0) do;  Obj2\_Delta\_Pos <- *Obj2\_Pos* - *Obj2\_Pre\_Pos*  *Obj2\_Delta\_Esti.x* <- RBE(*Obj2\_Delta\_Pos.x*, *Obj2\_Delta\_Esti.x*)  *Obj2\_Delta\_Esti.z* <- RBE(*Obj2\_Delta\_Pos.z, Obj2\_Delta\_Esti.z*)  End IF  *Obj2\_Pre\_Pos* <- *Obj2\_Pos*  IF Distance(*Obj1\_Pos, Obj2\_Pos*) < 30.0f do;  *follow* <- FALSE  else  *follow* <- TRUE  End IF  End IF  RETURN TRUE  End IF  RETURN FALSE: |

2수도코드는 MoveBackollowTarget 함수에 대해 설명한다. MoveoBackollowTarget 함수는 follow 변수가 false일 때 적의 위치를 예상하여 회전하여 조준하고 이동방향을 틀며 후진하는 처리를 하는 함수이다. 일정주기마다 적의 위치를 예측하는 계산을 하고 적의 이전위치에 대한 변수를 적의 현재위치로 갱신하며 적과의 거리에 따른 follow 변수에 변화를 준다.

follow는 true일때 전진을, false일때 후진을 의미하도록 설정하였다.

*Obj1\_Pos* 는 탱크 자신의 위치를 의미한다.

*Obj2\_Pos* 는 적의 위치를 의미한다.

*speed*는 자신의 속력을 의미한다.

*nTime*은 게임이 시작된 후 일정주기로 일정수치만큼 증가한다.

*Obj2\_Delta\_Esti* 는 Dir에 더해지는 벡터로, 적의 경로를 예측한 값을 나타낸다. .  
*Obj2\_Pre\_Pos*는 적의 이전 위치를 의미한다.

Normalize()는 벡터를 단위벡터화하는 함수이다.

RBE() 함수는 recursive bayesian estimation을 이용하여 적의 위치 벡터에 대한 변화량을 각 성분마다 추정하기 위해서 정의한 함수이다.

|  |
| --- |
| IsDead() |
| INPUT : *Obj1\_Hp*  IF *Obj1\_Hp* <= 0 do;  RETURN FALSE  End IF  RETURN TRUE |

3수도코드는 IsDead 함수에 대해 설명한다. IsDead 함수는 자신의 HP가 0이하가 될 시 false를 반환하며, 이는 IsDead 노드에서 true로 변환되어 selector에게 전달하는 기능을 가진다.

|  |
| --- |
| AddBullet() |
| INPUT : *Obj1\_Hp , Dir, nTime*  IF  *Obj1\_Hp* > 0 do;  IF *nTime* %100 ==0 do;  bullet <- Prefab\_bullet  bullet.Dir<- this.*Dir*  bullet.parent <- null  bullet.layer <- " Obj1\_Bullet "  bullet.position <- this.position  *nTIme++;*  RETURN TRUE  End IF  End IF  RETURN FALSE: |

*Obj1\_Hp* 는 자신의 HP를 의미한다.

4수도코드는 AddBullet 함수에 대해 설명한다. AddBullet 함수는 일정 주기마다 총알을 발사하는 처리를 하기 위한 함수이다.

*Obj1\_Hp* 는 총알 자신의 탱크의 HP를 의미한다.

Dir은 총알 자신의 탱크가 바로보고있는 방향벡터을 의미한다.

nTime은 게임이 시작된 후 일정주기로 일정수치만큼 증가한다.