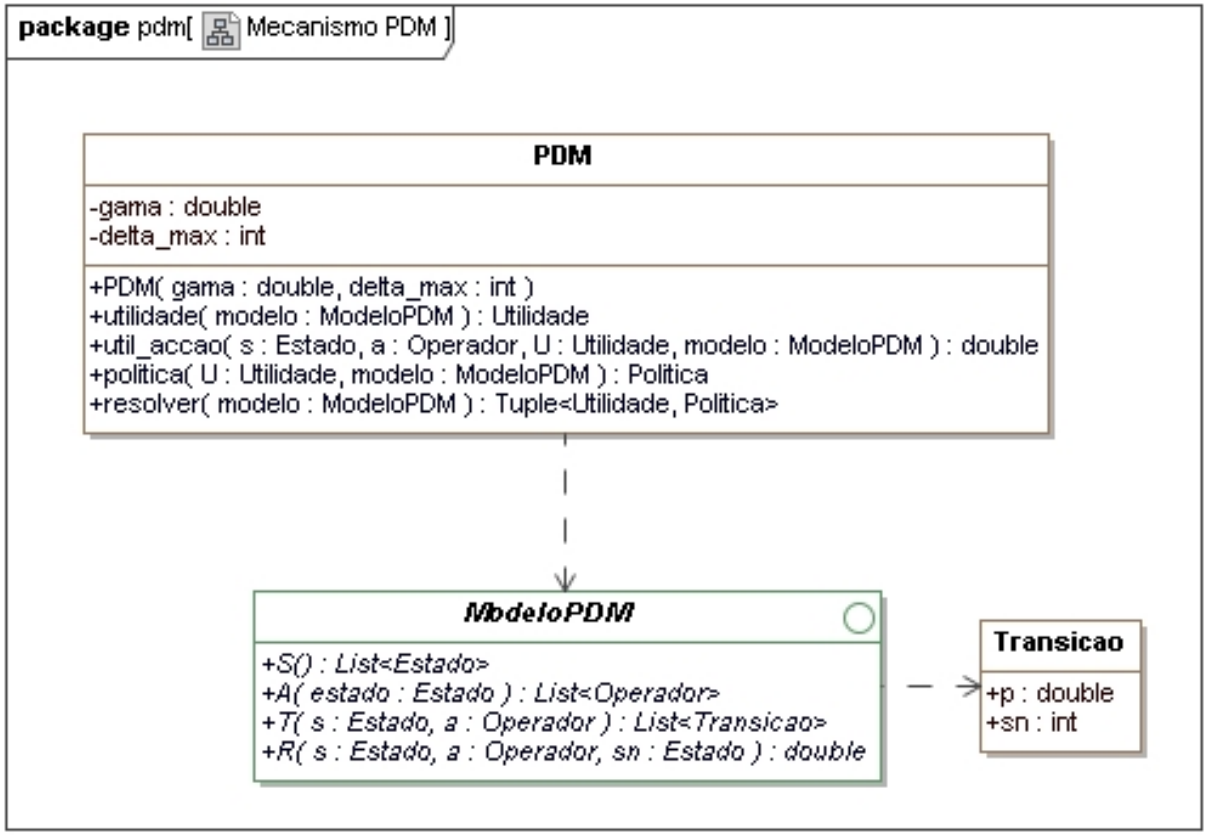
PROCESSOS DE DECISÃO DE MARKOV



CÁLCULO DA UTILIDADE

Iteração da utilidade de estado

Iniciar $U(s)$:

$$U(s) \leftarrow 0, \ \forall s \in S$$

Iterar $U(s)$:

$$U(s) \leftarrow \max_a \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma U(s')\right], \ \forall s \in S$$

No limite:

$$U \rightarrow U^{\pi *}$$

Critério de paragem de iteração?

- Diferença máxima de actualização < Δ_{max} (limiar de convergência)

```
function utilidade:
    U ← 0, ∀s ∈ S
    do:
        U_ant ← U
        δ ← 0
        for s in S:
            U[s] ← max_{a∈A(s)} U_acção(s, a, U_ant)
            δ ← max{δ, |U[s] - U_ant[s]|}
    while δ ≥ Δ_max:
    return U
```

```
function U_acção(s, a, U):
    return Σ_s' T(s, a, s')[R(s, a, s') + γU[s']]
```

MECANISMO PDM

```
def utilidade(self, modelo):
    S, A = modelo.S, modelo.A
    U = {s : 0 for s in S()}
    while True:
        Uant = U.copy()
        delta = 0
        for s in S():
            U[s] = max(self.util_accao(s, a, Uant, modelo) for a in A(s))
            delta = max(delta, abs(U[s] - Uant[s]))
        if delta < self._delta_max:
            break
    return U

def util_accao(self, s, a, U, modelo):
    T, R, gama = modelo.T, modelo.R, self._gama
    return sum(p * (R(s, a, sn) + gama * U[sn]) for p, sn in T(s, a))

def politica(self, U, modelo):
    S, A = modelo.S, modelo.A
    pol = {}
    for s in S():
        pol[s] = max(A(s), key=lambda a: self.util_accao(s, a, U, modelo))
    return pol
```