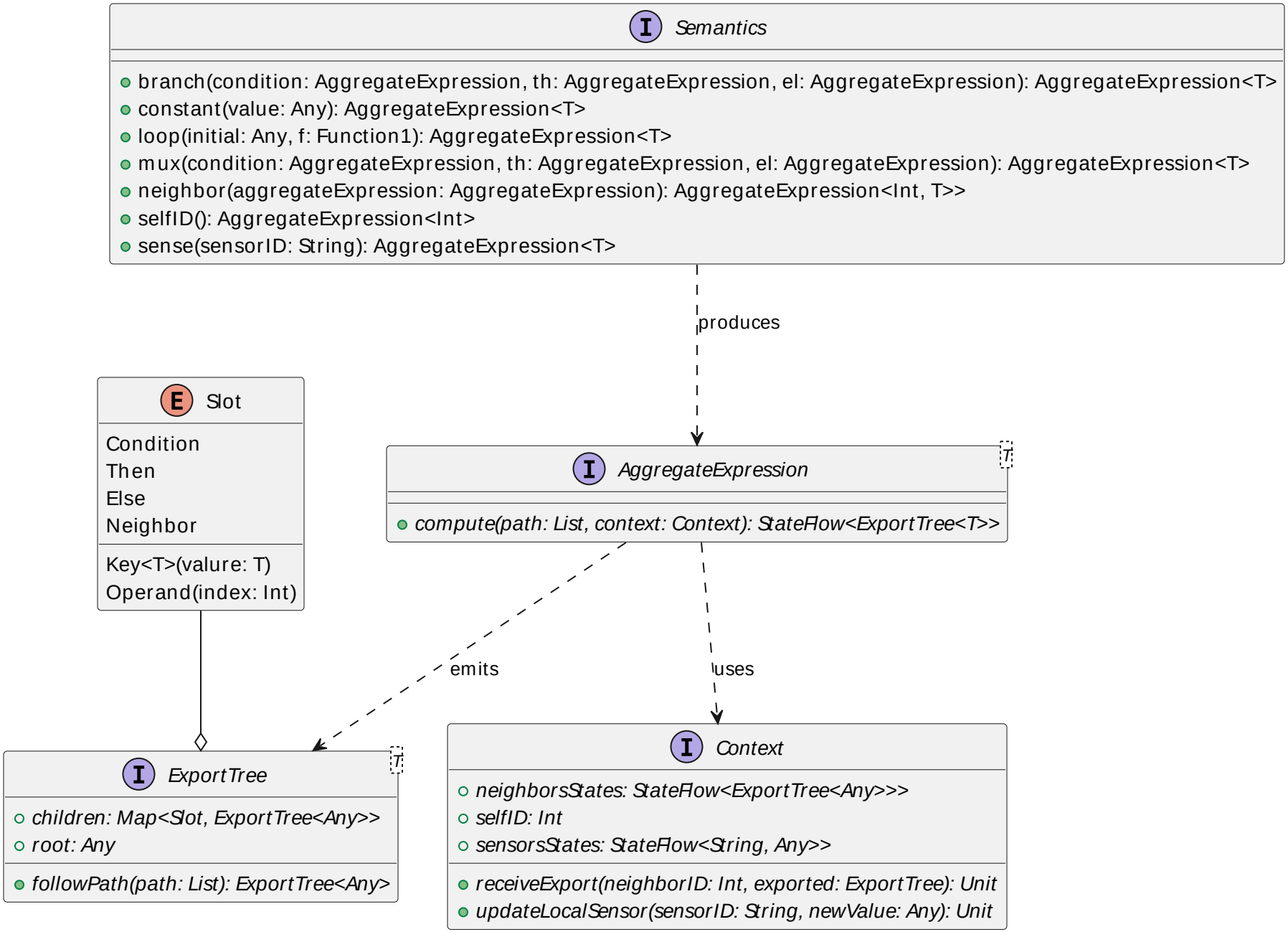
Semantics
branch(condition: AggregateExpression, th: AggregateExpression, el: AggregateExpression): AggregateExpression<T>
constant(value: Any): AggregateExpression<T>
loop(initial: Any, f: Function1): AggregateExpression<T>
mux(condition: AggregateExpression, th: AggregateExpression, el: AggregateExpression): AggregateExpression<T>
neighbor(aggregateExpression: AggregateExpression): AggregateExpression<Int, T>>
selfID(): AggregateExpression<Int>
sense(sensorID: String): AggregateExpression<T>
produces
Slot
Condition
Then
Else
Neighbor
Key<T>(valure: T)
Operand(index: Int)
AggregateExpression
T
compute(path: List, context: Context): StateFlow<ExportTree<T>>
emits
uses
ExportTree
T
children: Map<Slot, ExportTree<Any>>
root: Any
followPath(path: List): ExportTree<Any>
Context
neighborsStates: StateFlow<ExportTree<Any>>>
selfID: Int
sensorsStates: StateFlow<String, Any>>
receiveExport(neighborID: Int, exported: ExportTree): Unit
updateLocalSensor(sensorID: String, newValue: Any): Unit