


DEFINIZIONE DI UNA STRUTTURA AD OGGETTI PER LA RAPPRESENTAZIONE INTERNA DELLE INTERROGAZIONI DEL LINGUAGGIO PERLA

Candidato: Alessandro PERRUCCI
Relatore: Chiar.mo Prof. Fabio A. SCHREIBER



ART DECO

(Adaptive InfRasTructures for DECentralized Organizations)

- Progetto finanziato dal MIUR
- Sviluppo di tecniche e strumenti per la diffusione delle “networked enterprises” tra le piccole e medie imprese
- Un caso di studio: monitoraggio della produzione e trasporto di vini di alta qualità

Caso di studio

- Obiettivo: monitorare i parametri importanti del processo produttivo e del trasporto di vini (temperatura, umidità, accelerazione, ecc.) e garantirne la tracciabilità
- E' necessaria una vasta rete di sensori di tipo eterogeneo, ognuna con una sua interfaccia di controllo e interrogazione (WSN, RDIF, GPS, ecc.)
- Difficile per l'utente recuperare e incrociare i dati
- Soluzione: il linguaggio PERLA



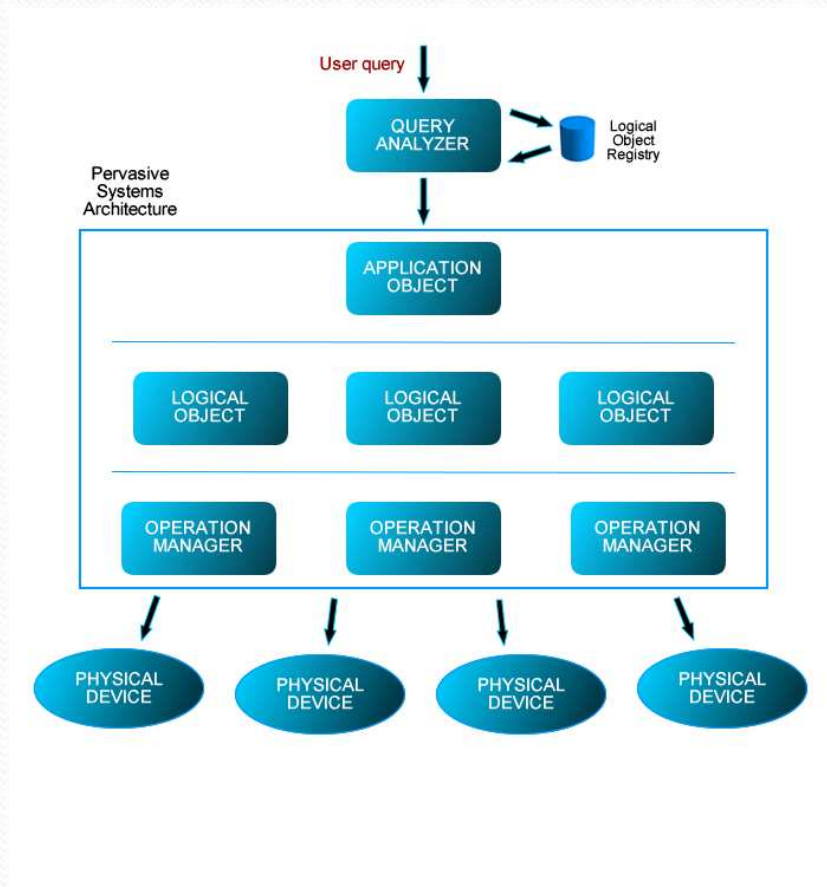
PERLA

(PERvasive LAnguage)

- Linguaggio completamente dichiarativo di alto livello
- Permette di interrogare un sistema pervasivo in modo simile a come si interroga una base di dati
- Nasconde all'utente l'elevata complessità del sistema e della programmazione di basso livello
- E' stato pensato per essere eseguito al di sopra di una architettura a 3 livelli

Architettura

- Architettura a 3 livelli:
 - livello di accesso fisico ai dispositivi
 - livello degli oggetti logici
 - livello applicativo
- L'analizzatore di query è posto al di sopra dell'architettura
- Esso riceve in ingresso la query e ne fa il parsing
- La query viene divisa in sottoquery che vengono inviate agli esecutori



Un esempio di query

```
CREATE OUTPUT STREAM EnvironmentParam (sensorID ID, temp FLOAT,  
    humidity FLOAT, pressure FLOAT DEFAULT -1, locationX FLOAT,  
    locationY FLOAT)
```

```
INSERT INTO STREAM EnvironmentParam (sensorID, temp, humidity,  
    locationX, locationY)
```

LOW:

```
    EVERY 30 m SELECT ID, AVG(temp, 30 m), AVG(humidity, 30 m)  
    DEFAULT -1, locationX, location  
HAVING AVG(temp, 30 m)<10 OR AVG(temp, 30 m)>35  
SAMPLING EVERY 10 m  
EXECUTE IF EXISTS(temp) AND is_in_vineyard(locationX, locationY)  
    REFRESH EVERY 5 m
```

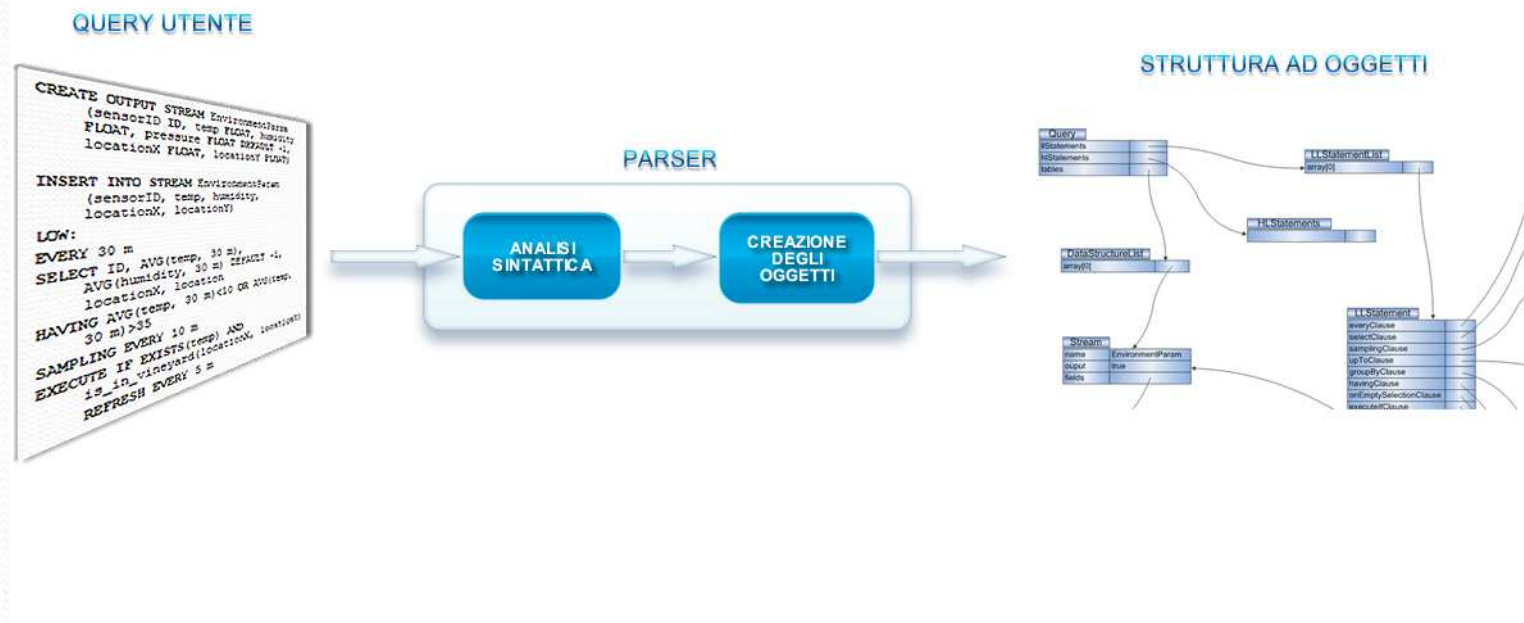



L'obiettivo del progetto

- Il parser deve analizzare le query sottoposte dall'utente e le deve tradurre in un formato utilizzabile dal motore di esecuzione
- È necessaria quindi una struttura ad oggetti che consenta di tradurre ogni query inserita dall'utente

L'obiettivo del progetto è la progettazione e realizzazione di una struttura ad oggetti per la rappresentazione interna delle query

Il parser



Un esempio: la clausola Sampling

```
<Sampling Clause> →
  SAMPLING
  {
    <On Event Clause> |
    <Sampling IfEvery Clause>
    [<On Unsupported SR Clause>] [<Refresh Clause>]
    [<Where Clause>]
  }
```

```
<On Event Clause> →
  ON EVENT <Event List>

<Event List> →
  <Logical Object Event> { ',' <Logical Object Event> }*
```

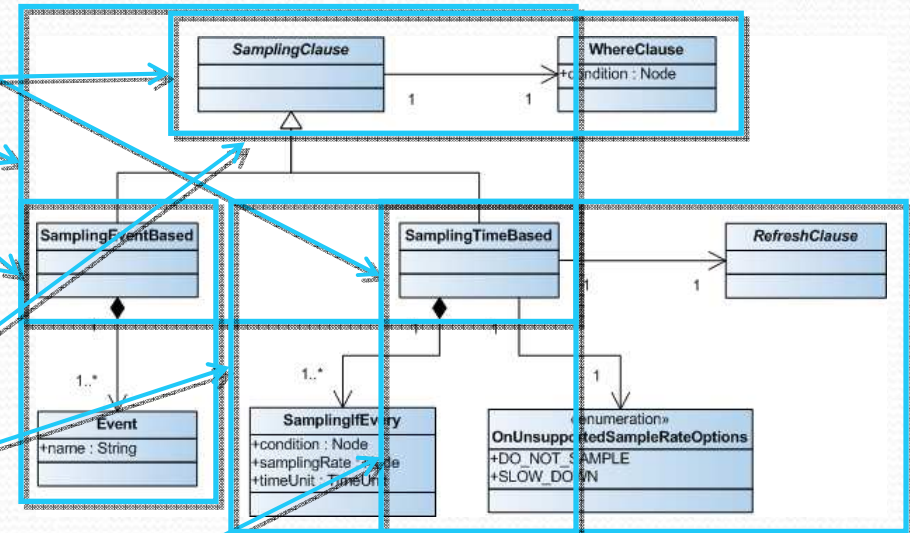
```
<Sampling IfEvery Clause> →
  {
    {<Sampling If Clause> <Sampling Every Clause> }*
    ELSE <Sampling Every Clause> |
    <Sampling Every Clause>
  }
```

```
<Sampling If Clause> →
  IF <Condition>
```

```
<Sampling Every Clause> →
  EVERY <Expression> <Time Unit>
```

```
<On Unsupported SR Clause> →
  ON UNSUPPORTED SAMPLE RATE { DO NOT SAMPLE | SLOW DOWN }
```

```
<Where Clause> →
  WHERE <Condition>
```



Un esempio di struttura ad oggetti

```
CREATE OUTPUT STREAM EnvironmentParam
(sensorID ID, temp FLOAT, humidity
FLOAT, pressure FLOAT DEFAULT -1,
locationX FLOAT, locationY FLOAT)
```

```
INSERT INTO STREAM EnvironmentParam
(sensorID, temp, humidity,
locationX, locationY)
```

LOW:

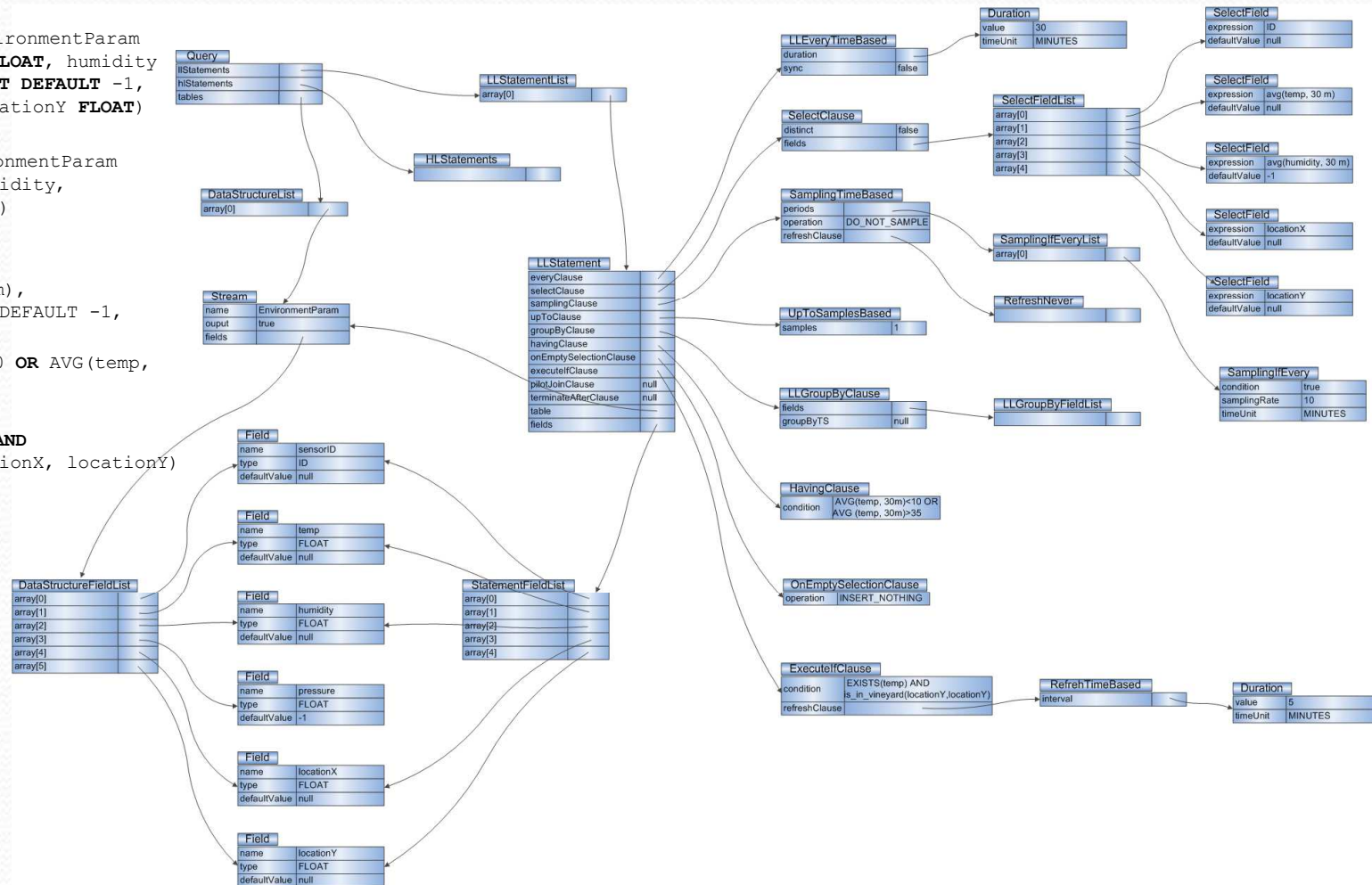
EVERY 30 m

```
SELECT ID, AVG(temp, 30 m),
AVG(humidity, 30 m) DEFAULT -1,
locationX, location
```

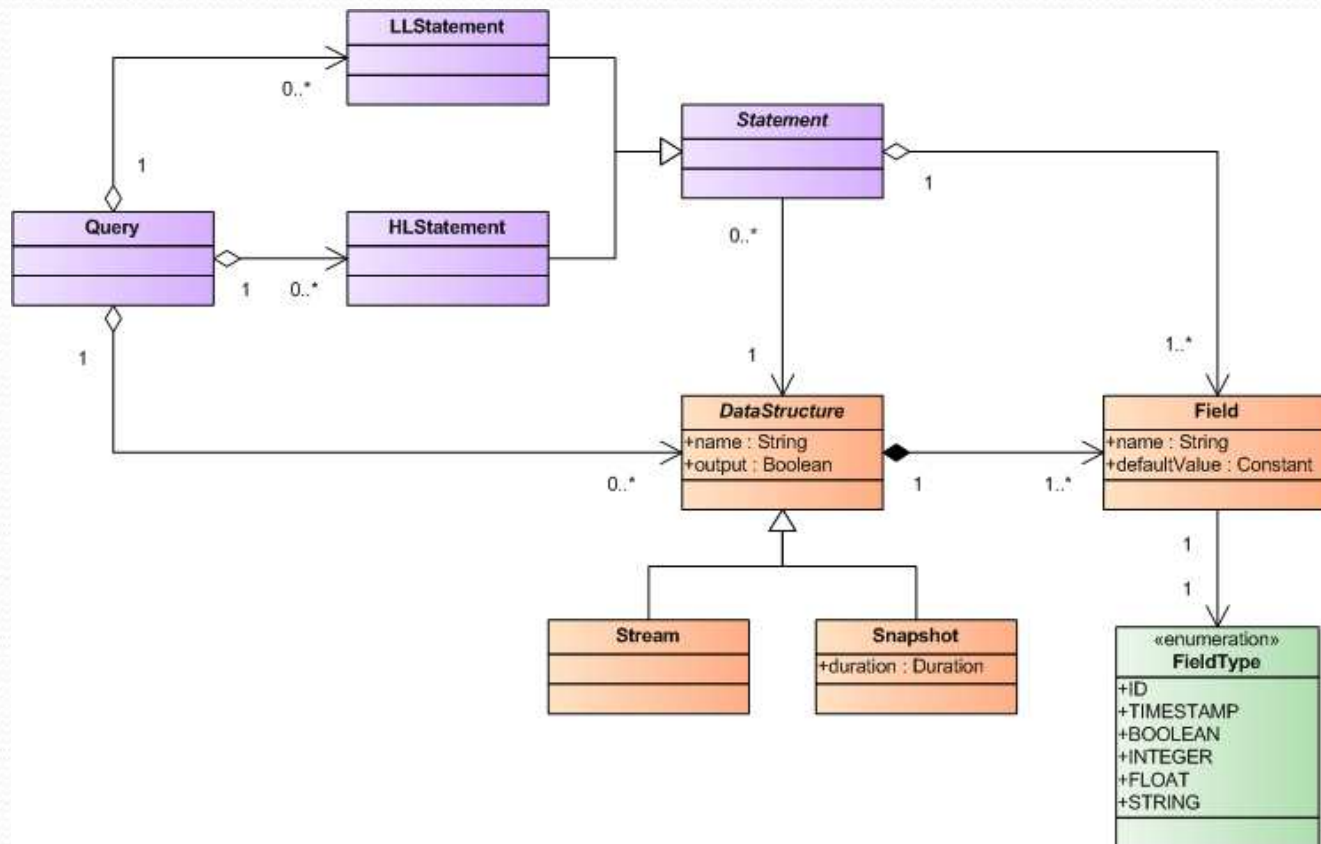
```
HAVING AVG(temp, 30 m)<10 OR AVG(temp,
30 m)>35
```

SAMPLING EVERY 10 m

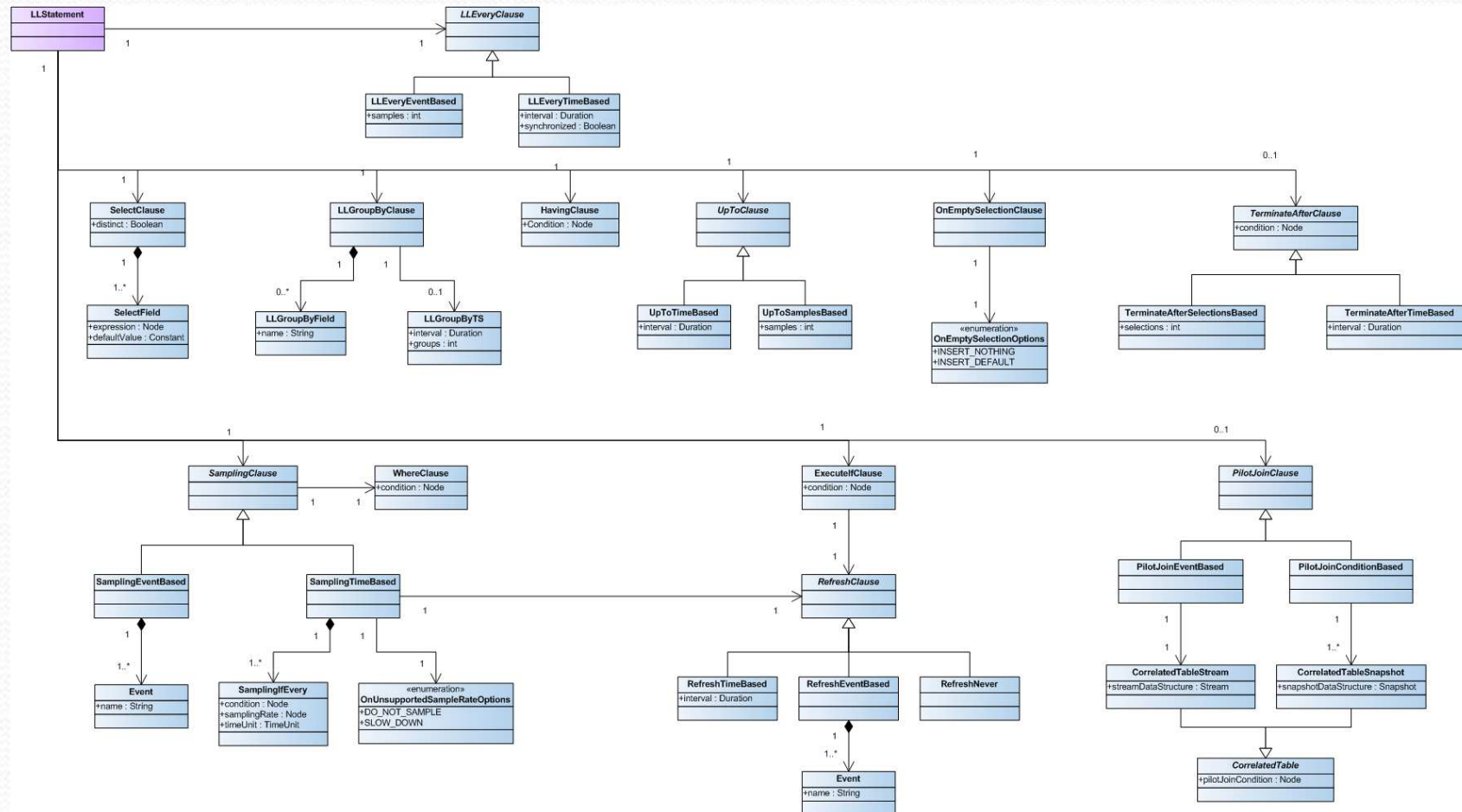
```
EXECUTE IF EXISTS(temp) AND
is_in_vineyard(locationX, locationY)
REFRESH EVERY 5 m
```



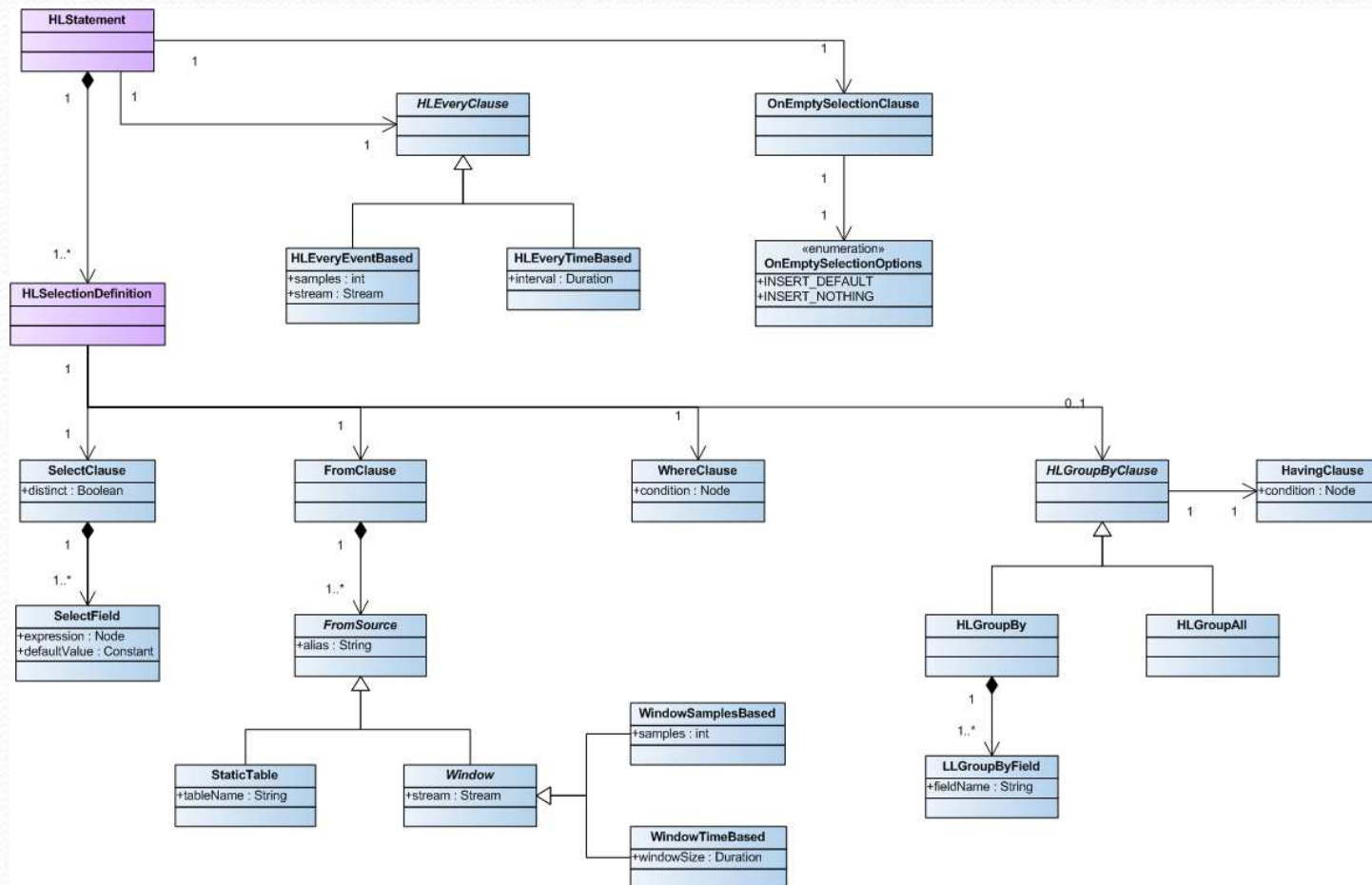
Class diagram (1/3)



Class diagram (2/3)



Class diagram (3/3)



Conclusioni e stato dell'arte

- Le classi progettate sono state implementate in Java
- Un progetto precedente ha realizzato una parte del parser utilizzando JavaCC
- In questo momento sono in corso di svolgimento un progetto per completare il parser e un secondo per definire gli oggetti logici

