**ESTRATTORE**

**Assunzioni:**

**Sintassi Annotazioni:**

<startAnnotation> <miniSLID>: <controlStatement> [<serviceName>(<guard>)]<endAnnotation>

// miniSL: call map(jobs,i)

startAnnotation = //

miniSLID = miniSL

controlStatement = call

serviceName = map

guard = jobs,i

endAnnotation = “”

| **keyword** | **definition** |
| --- | --- |
| startAnnotation | definisce l’identificatore dell’inizio del commento in riga |
| miniSLID | identificativo dell’annotazione, ciò che fa per noi di un commento un’annotazione |
| controlStatements | insieme di keyword utilizzate per identificare un pezzo di codice in modo da poterlo tradurre in miniSL: **for, call, if, else, end, function, invoke** |
| serviceName | nome del servizio esterno invocato tramite call, in caso si usi il control statement “function” o “invoke” questo sarà il nome (fnName) della funzione interna |
| guard | guardia dello statement nel caso di if o for. Nel caso di function invoke o call parliamo di parametri |
| endAnnotatio | definisce l’identificatore della fine del commento di riga, in questo caso è lasciato vuoto (default) |

**Esempio Annotazioni:**

**// miniSL: call main(jobs,m,r)**

**function main(jobs: Job[], m: number, r: number): void {**

**// miniSL: for(i,m)**

**for (let i = 0; i < m; i++) {**

**// miniSL: call map(jobs,i)**

**map(jobs, i);**

**// miniSL: for(j,r)**

**for (let j = 0; j < r; j++) {**

**// miniSL: call reduce(jobs,i,j)**

**reduce(jobs, i, j);**

**}**

**// miniSL: end**

**}**

**// miniSL: end**

**}**

**// miniSL: end**

**corrispondente codice miniSL:**

service map : (void) -> void;

service reduce : (void) -> void;

(jobs, m, r, isVipUser, username, param) => {

for(i in range(0, m)) {

call map(jobs, i)

for(j in range(0, r)) {

call reduce(jobs, i, j)

}

}

}