Problems (December 5)

Interazione con enti non-QActor. Una entità scritta in un qualsiasi linguaggio deve poter (**qactors2018/sez. 5.1, 5.6**) :

1. Inviare un messaggio a uno specifico attore di un QActor system
   - it.unibo.mbot.intro/src/**radargui.qa**
   - it.unibo.mbot.intro/runnable/ **it.unibo.ctxRadarBase.MainCtxRadarBase.java**
2. Generare eventi gestibili da un QActor system (Unity, WEnv)
3. Ricevere informazioni da un (attore di un) QActor system

Basi di conoscenza: un QActor deve poter consultare/aggiornare conoscenze temporanee e/o permanenti [data-structures, files, database (NoSql) , actorKb (**qactors2018/sez. 3**)
it.unibo.qa2018.tests/src/ **atheoryUsage.qa introspection.qa,**]

Long-lasting actions : un QActor deve eseguire azioni di lunga durata senza perdere la possibilità di gestire messaggi o e/o eventi (**qactors2018/sez. 3**)
(*asynchronous actions* (**qactors2018/sez. 4.1**) : **it.unibo.qa2018.tests/src/asynchFibo .qa**, *azioni interrompibili*?)

Risposte: Un QActor vorrebbe **inviare informazione al mittente di un dispatch** (`sendToSender, sendAnswerToServer` in QActor.java)

Sistemi dinamicamente configurabili : un QActor dovrebbe poter essere aggiunto in modo dinamico a un sistema funzionante (**qactors2018/sez. 9, it.unibo.qa2018.pivot**, **it.unibo.mbot.inro.radarUsage.qa**)