**Email 1:**

Sinceramente non ricordo bene il significato di tutte le colonne presenti nel file, potresti per favore spiegarle nuovamente? Inoltre, secondo te quali informazioni di quel file potrebbero essere d'aiuto per risolvere il problema del frame rate che abbiamo avuto l'anno scorso?

A noi servirebbe di capire principalmente:

    - se ci sono proprietà che vengono messe in lock per troppo tempo

    - se ci sono moduli che richiedono di essere eseguiti con una frequenza superiore a quella realmente richiesta

    - se ci sono moduli troppo lenti

    - se ci sono proprietà che richiedono molto tempo per essere lette e/o scritte

    - se ci sono dei tempi di attesa in cui nessun modulo è in esecuzione a causa di semafori bloccati in attesa di dati da NaoQi

Questi sono le informazioni utili che a me sono venute in mente, se te ne hai delle altre sono le benvenute :)

**Responce 1**

il significato delle colonne è:  
1) timestamp reale in millisecondi da quando è partito l'RAgent  
2) valore errato/non funzionante :-)  
3) timestamp rispetto al thread  
4) ID del thread  
5) tipo di evento  
6) oggetto dell'evento  
7) altre info  
  
esempio:  
1634601 -1375 1476774 3055285952 LOCK motion:stats/activityCounter  
1634781 -1375 1476774 3055285952 UNLOCK motion:stats/activityCounter  
  
spiegazione:  
il thread numero [3055285952](tel:3055285952), appartenente al modulo motion, ha lockato la  
property stats/activityCounter all'istante

1634601 e l'ha liberata  
all'istante 1634781 di tempo reale; il lock è quindi durato  
1634781-1634601=180ms  
dal suo punto di vista, ha lockato la property all'istante 1476774 e l'ha  
subito rilasciata, mantenendola lockata 0ms di tempo di thread: questo  
significa che negli altri 180ms sono stati altri thread a prendere tempo,  
lui, dal suo punto di vista, ha lockato e slocckato subito la property  
  
quindi:  
- per capire le property che vengono lockate per troppo tempo, controlla  
il tempo di thread (terza colonna, cioè quanto LORO tengono lockate le  
property)  
- per capire la frequenza di scheduling di un modulo, guarda l'intervallo  
di tempo reale (prima colonna) tra due START: dopo una modifica di un mese  
fa, nessun thread dovrebbe essere schedulato più velocemente di 100Hz  
- per capire quanto impiega una iterazione di un modulo (tra lo START e  
l'END), usa, ovviamente, il tempo di thread (terza colonna)  
- property che richiedono molto tempo per essere lette/scritte: è lo  
stesso che vedere il tempo tra LOCK e UNLOCK (quindi il timestamp della  
terza colonna)  
- tempi di attesa... riguardo ai tempi di attesa, i lock esterni (tipo  
l'attesa di eventi NaoQi) dovrebbero stare FUORI dal  
session->start/session->end, in questo modo l'attesa non viene computata  
(erroneamente) nel tempo di esecuzione del modulo e hai anche un'idea di  
quanta attesa c'è per avere dati da NaoQi (o da qualunque altra fonte  
dati, ad esempio socket, Player, ecc.)

**Responce 2**

2) in aggiunta a quelle informazioni, sugli eventi START e END, c'è anche  
l'informazione esplicita (scheduling, duration) già calcolata dal sistema  
(in secondi, in questo caso, in pratica sono le due differenze START-START  
e START-END di cui ti parlavo nell'altra email, già calcolate)

**Email 2**

La domanda è questa: nel file di log c'è un solo START ed una serie di END per un singolo modulo e non riusciamo a capire quando effettivamente un modulo è terminato oppure no. Inoltre vorrebbe sapere cosa rappresentano gli END intermedi.

Inoltre quale è il significato della "duration"? Indica il tempo che ha richiesto motion per essere eseguito?

**Responce 3**

in teoria lo START corrisponde a session->start() (o a SESSION\_START), mentre l'END corrisponde a session->end() o a session->terminat() (o a ESSION\_END\_CATCH\_TERMINATE o affini)... da questo log uno sarebbe portato ad affermare che c'è un end() senza il corrispettivo start()  
  
duration è il calcolo del tempo tra START e END in secondi utilizzando il timer del thread (schedule, che sta sullo start, è il tempo in secondi dall'ultimo start() dello stesso modulo, sempre in secondi, utilizzando il timer reale)