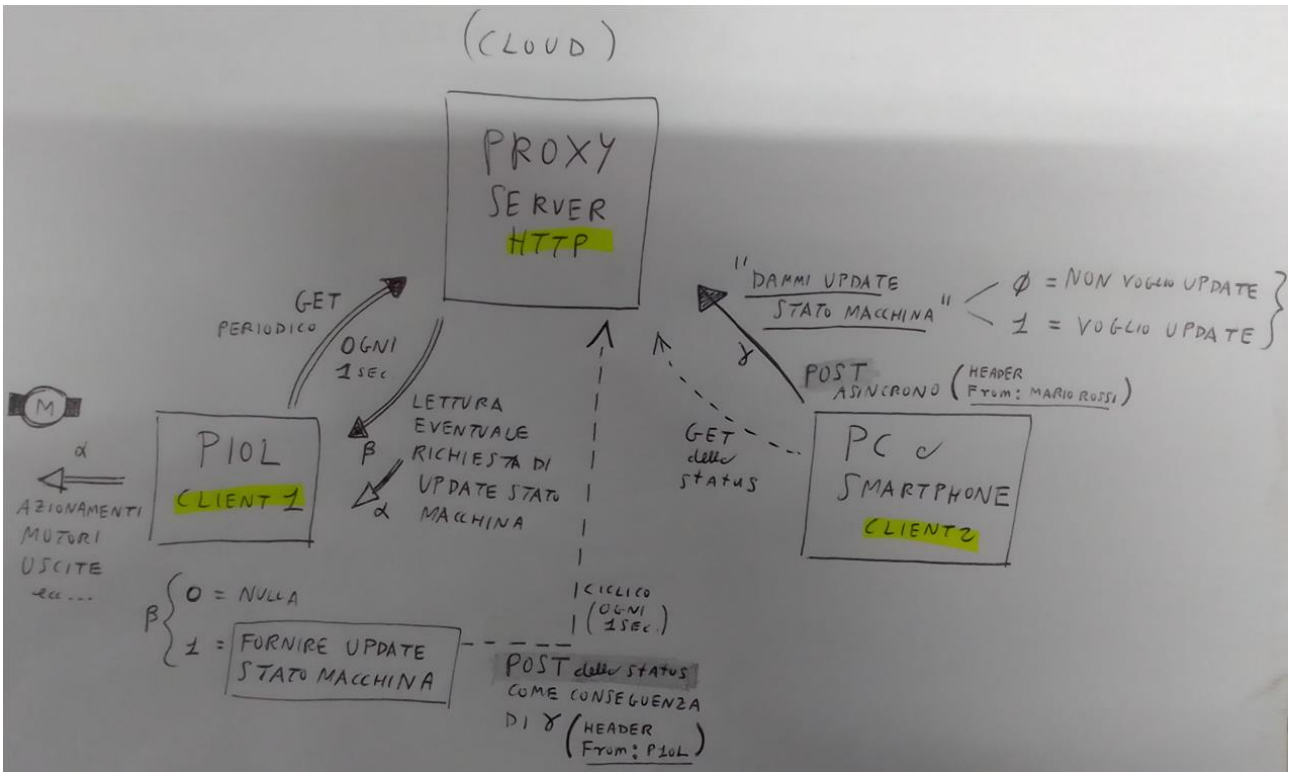
Breve nota generale sulla teleassistenza nell'ecosistema Syel + Codesys: utilizzo dell'end-point nell'URL + header http "From: "

20-12-2022

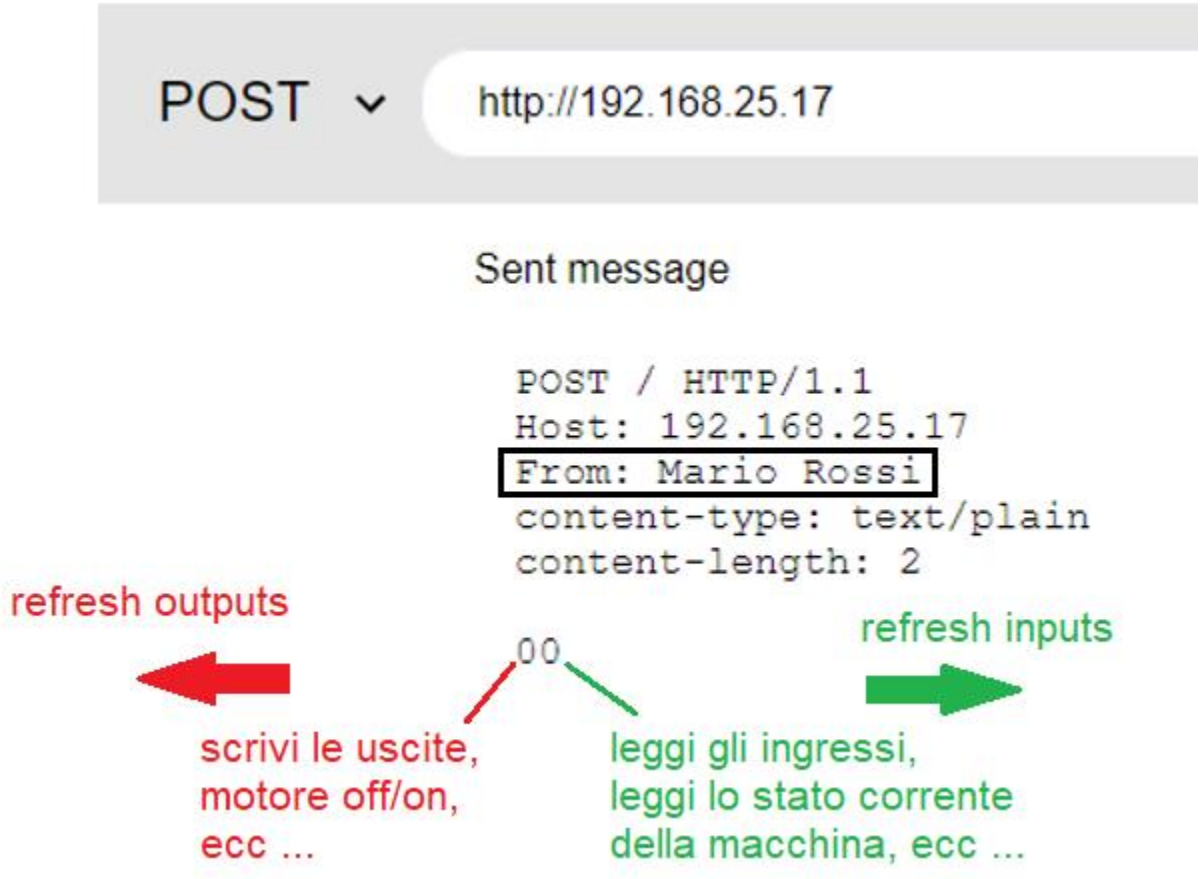
Draft, version 1.0

Abbiamo già discusso la possibilità di implementare una forma di teleassistenza che si sviluppi via protocollo HTTP1.1: in merito ci si riferisca al documento "Breve nota generale sulla teleassistenza nell'ecosistema Syel + Codesys" del "16-12-2022", "Draft, version 1.0" per maggiori dettagli. Abbiamo pertanto già esaminato la situazione relativa all'esempio da noi sviluppato, il quale prevede il seguente schema a blocchi di massima:



Il server è stato implementato da un HMI a 5 pollici by Syel posto nella stessa LAN dei clients, su cui gira un SW nostro, che implementa un server http custom, il quale discerne da quale nodo, da quale client http, arriva sia un metodo POST che un metodo GET, grazie unicamente all'header http "From: ".

Ad esempio, nel caso della POST inviata dal client PC o smartphone verso il server:



Il server distingue questa POST dalla POST inviata periodicamente, es: ogni 1 secondo, dal P10L by Syel (HMI che controlla il bus di campo, quindi la macchina), unicamente grazie all'header "From: ", in quanto abbiamo, nel nostro esempio:

"From: Mario Rossi"

Per la POST inviata dal cliente

"From: P10L"

Per la POST inviata dall'HMI by Syel P10Linux

È importante per tale server distinguere la paternità della POST, in quanto in base a chi l'ha inviata, il contenuto della risposta che il server deve mandare sarà diverso, e soprattutto l'azione di sub-mitting informativa dentro il server sarà diversa. In ciascun caso, abbiamo infatti un array diverso da riempire, con informazioni diverse.

Stesse logiche sono valide per la distinzione della GET arrivata, ovvero GET arrivata dal cliente, oppure GET arrivata da P10L. Il server, in funzione del client che ha inviato la GET, deve rispondere con informazioni diverse.

Abbiamo esteso l'esempio, rendendolo maggiormente aderente alla realtà di una teleassistenza con cloud/web-server http, es: AWS (Amazon Web Services) e simili, in modo da utilizzare anche un end-point all'interno dell'URL fornito dal client di turno.

Prima l'end-point non era stato usato nell'URL, quindi era stato lasciato semplicemente "/" nei messaggi http, ovvero il simbolo del path main-root, senza specificare un percorso, una risorsa, da sub-mittare/postare (scrivere, upload) verso il server oppure una risorsa (es: un file di testo) da ottenere in lettura (in download) dal server.

Esempio:

Adesso le cose sono cambiate … per esempio:

Questa è la POST che PC o smartphone del cliente invia al server, dove i due byte già discussi (evidenziati in verde) costituiscono il file "req.txt" ("request.txt"), ovvero il file di testo (stringa ASCII) che il client vuole sub-mittare verso il server.

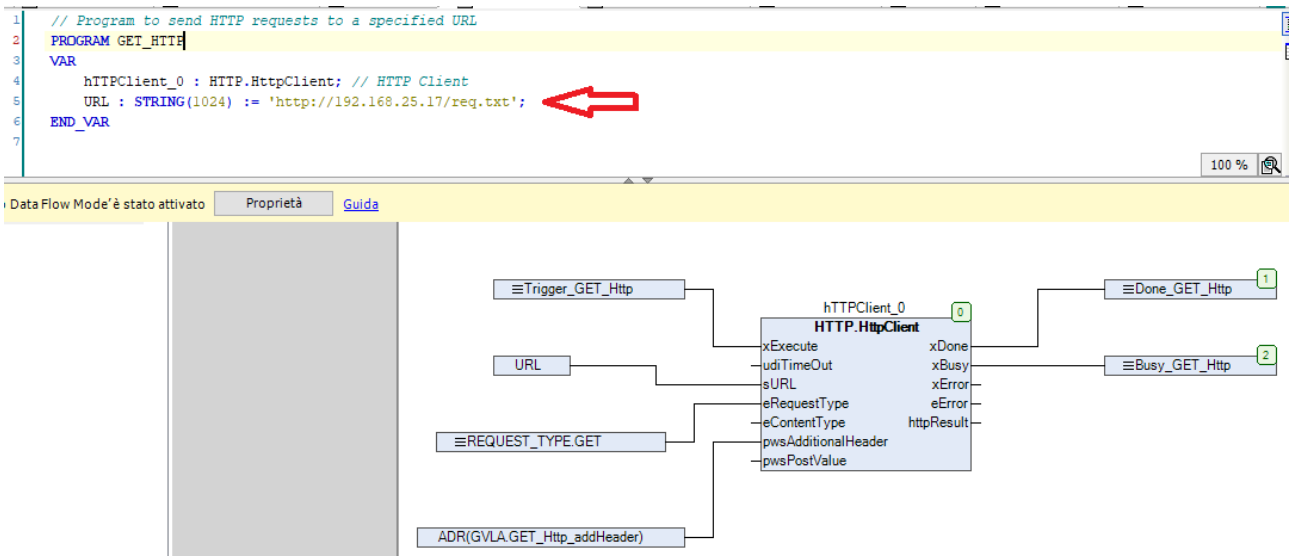
"/req.txt" è l'end-point verso il quale (es: il path nel filesystem del server) il server stesso registrerà la risorsa inviata dal client, ovvero la stringa ASCII suddetta.

Altro esempio:

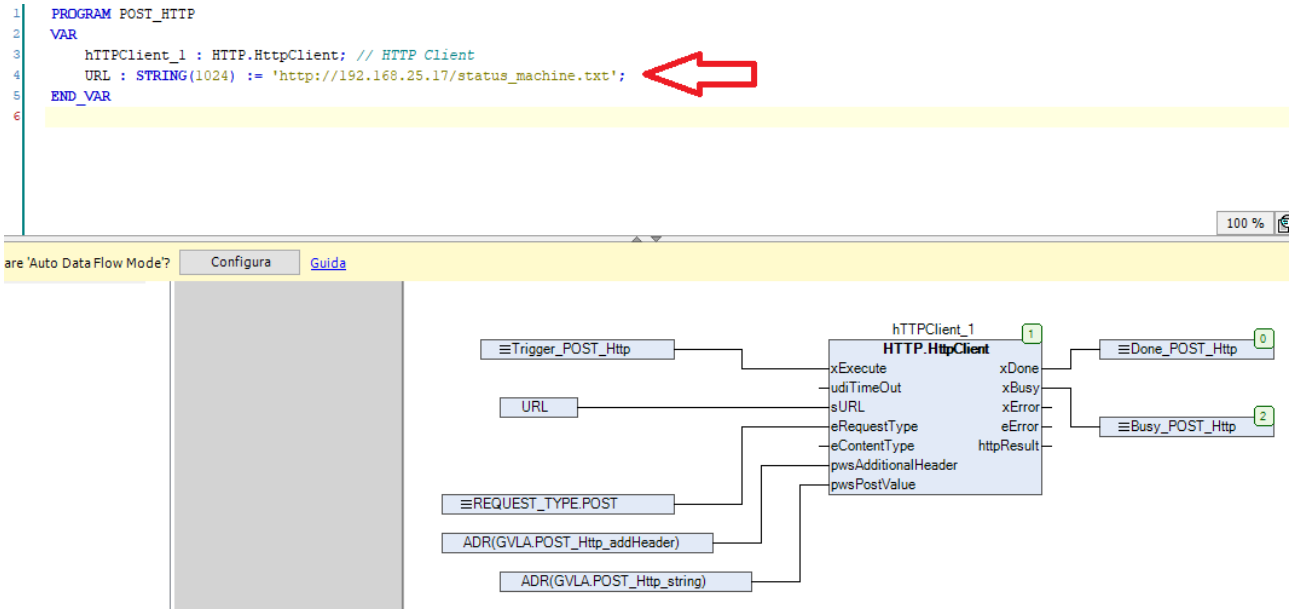
Il PC o smartphone sta richiedendo la risorsa testuale "status_machine.txt" dal server, file che è stato sub-mittato in precedenza dal P10L, e che contiene lo stato corrente della macchina.

Lato Codesys su P10L abbiamo la seguente situazione:

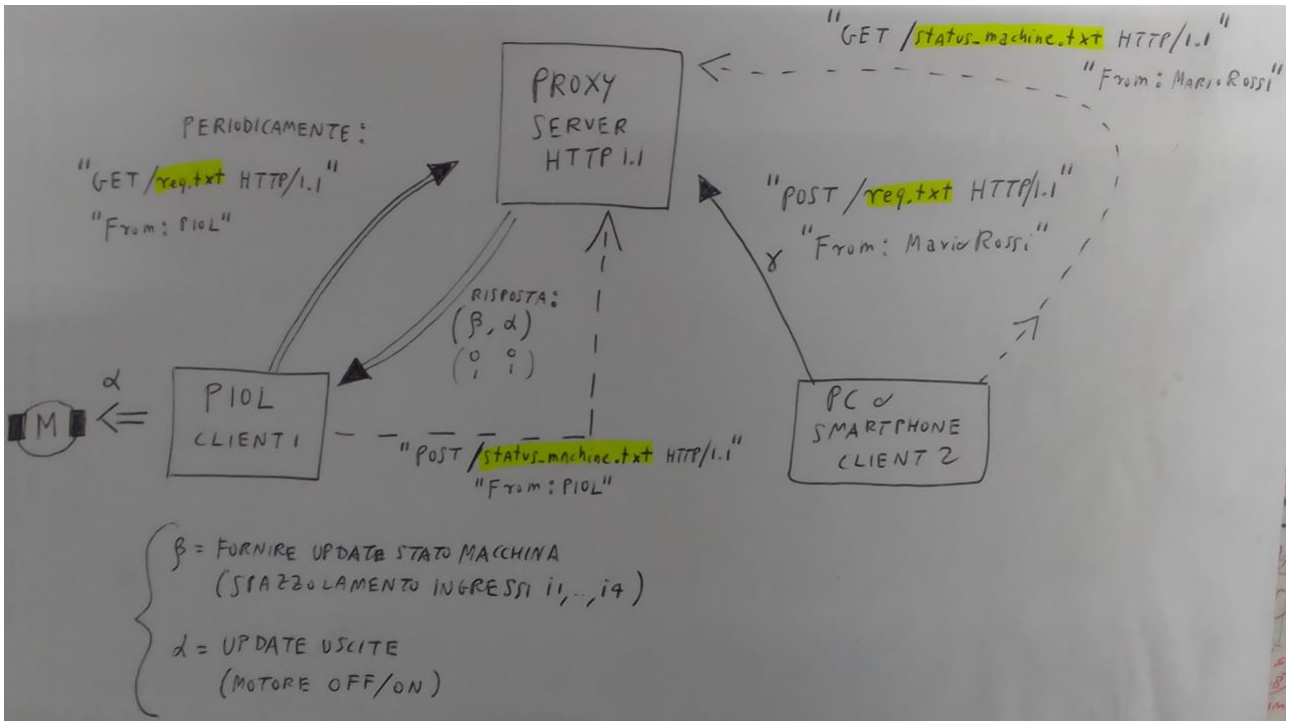
GET:

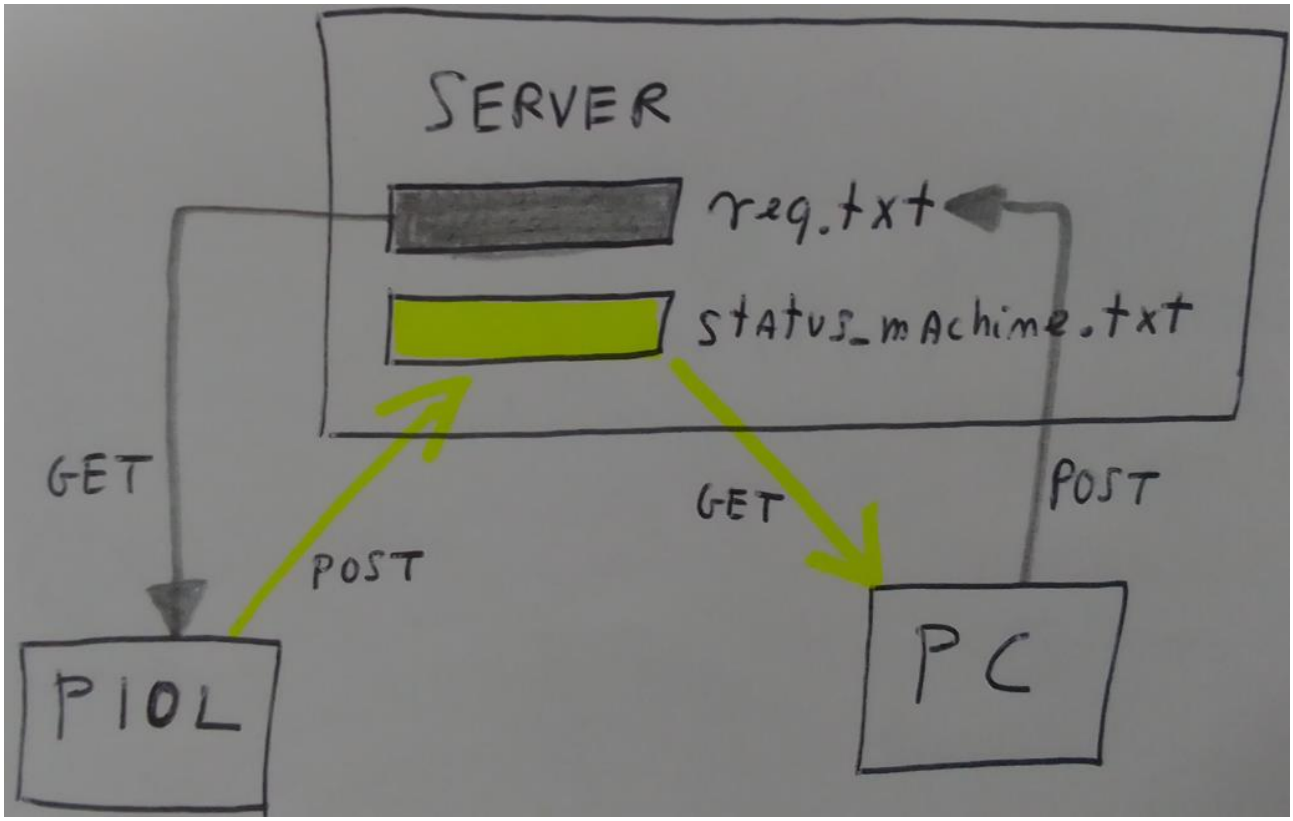


POST:



Questa è la situazione generale del sistema:


PROXY
SERVER
HTTP 1.1
"GET /status_machine.txt HTTP/1.1"
"From: Mario Rossi"
PERIODICAMENTE:
"GET /req.txt HTTP/1.1"
"From: PIOL"
"POST /req.txt HTTP/1.1"
"From: Mario Rossi"
RISPOSTA:
(β, α)
(0 ; 1)
α
M
PIOL
CLIENT 1
"POST /status_machine.txt HTTP/1.1"
"From: PIOL"
PC o
SMARTPHONE
CLIENT 2
β = FORNIRE UPDATE STATO MACCHINA
(STAZZONAMENTO INGRESSI i1,..,i4)
α = UPDATE USCITE
(MOTORE OFF/ON)


SERVER
req.txt
status_machine.txt
GET
POST
GET
POST
PIOL
PC