### Gestione Essiccatore

#### Premessa

* Molta logica presente nell’applicativo
* CYB500 prevede funzioni di controllo, visualizzazione e impostazione del lavoro degli essiccatori:
  + switch gestione automatica/manuale
  + calcolo di alcuni parametri (finestre di funzionamento dei modulatori, …)
  + controlli sulla catena di funzionamento degli apparati (start e stop del bruciatore, del tamburo,…)
  + {gestione ricette da produrre (interazione con database)}
  + {minima interazione dosaggio-predosaggio (blocco dei predosatori senza impasti da produrre).}
  + Impostazione dei valori di portata dei predosatori

#### Descrizione

La gestione essiccatore si occupa di visualizzare e gestire il funzionamento del tamburo essiccatore o dei tamburi essiccatori dove è prevista la presenza di due tamburi (ParallelDrum). Si compone di tre parti principali, ovvero:

* Tamburo rotante
* Bruciatore
* Aspirazione gas

Tamburo

#### Parametri

#### Logica

##### RegolazioneAriaAspiratore

Regolazione del ModulatoreFumiTamburo.

### Dettaglio Funzioni

##### ParaTabBruc.bas

* ParaTabBruc\_ReadFile()  
  Lettura parametri su init applicazione, chiamata da ParametriReadFile(), chiamata da Form\_Load().
* ParaTabBruc\_Apply()  
  Aggiornamento interfaccia da lettura parametri , chamata da ParametriApply().

##### BrucAuto.bas

* **GestSetRegBruciatore**(), **GestSetRegBruciatore2**(), **GestioneFunzAutomaticoBruc**(), **GestRegolazioneBruciatore**(tamburo)  
  Funzioni agganciate al ciclo di VideataPrincipale()  
  Controllo di retroazione su elemento di tipo modulatore (up/down).  
  La posizione del modulatore del bruciatore è funzione della temperatura che si vuole ottenere (impostata da parametri) e della portata attuale dell’impianto. Calcolata la portata teorica corretta in dipendenza della temperatura si utilizza una curva parametrica di relazione tra apertura bruciatore (%) portata predosatori (%), si stabilisce la apertura del bruciatore da ottenere. Il controllo applicato è una retroazione con finestra di isteresi del 5% (vedi GestTotRegBruciatore()): nel caso in cui la differenza tra valore richiesto e valore rilevato sia maggiore del 5% viene aperto il modulatore (up), nel caso in cui la differenza **assoluta** tra valore rilevato e valore richiesto sia minore del 5% e superiore dello 0.5% viene attivato un controllo a impulsi di apertura del modulatore di durata minima di 500 ms e massima di 2000ms con ritardo tra impulsi di 3 secondi.
* **AttivaUscitePerRegolazioneEss**(verso As Integer, smSec As Long, tamburo As Integer)  
  Mette in up o down il modulo del bruciatore

##### ControlloBruc.bas

* **RegolazioneAriaAspiratore**(), **ModoRegolazioneDepressioneFiltroConDepressTamburo**()  
  Chiamata dal modulo di gestione filtro. Regola il modulatore di aspirazione del filtro (up, down, none) in base a dei valori di minimo e massimo.
* **FiammaBruciatorePresente\_change**(tamburo As Integer)  
  Gestisce il cambio di stato del bruciatore: se è acceso allora lo passa a gestione manuale, se è spento allora fa partire le procedure di accensione, oltre a arrestare i predosatori in caso la presenza della fiamma non sia bypassata. Se non è presente l’esclusione dell’avviamento a caldo allora viene fermato anche il tamburo. Viene inserito, infine, nel log il dato di consumo di carburante dell’ultima sessione di accensione.
* **VerificaCondizioniAvviamentoBruciatore**(tamburo As Integer, aCaldo As Boolean) As Long  
  Fa un controllo sui parametri per l’avvio del bruciatore, in modo da settare un eventuale codice di errore. Si verifica un errore se:
  + Bruciatore ad olio e
    - pressione insufficiente
    - motore pompa non acceso
  + Temperatura di entrata del filtro superiore alla soglia
  + Non è attivo l’avviamento del bruciatore caldo e il motore del nastro elevatore freddo è fermo
  + Il nastro elevatore caldo è fermo
  + Il motore dell’aspiratore è fermo
  + Il bruciatore non è in posizione di accensione
  + Bruciatore ad olio e temperatura del combustibile è troppo bassa o troppo alta
  + Bruciatore a gas e
    - allarme di pressione del gas alta
    - allarme perdita dalle valvole del bruciatore
  + Allarme della termica del ventilatore
  + Allarme del sensore fumi del tamburo
  + Se avviene una partenza a caldo e le temperature di una dei bitumi è bassa
  + Se avviene una partenza a freddo e il motore di rotazione dell’essiccatore è fermo
* **ConteggioTempoArrestoBruciatore**(tamburo As Integer)  
  Se tutti i predosatori sono spenti e il bruciatore acceso deve partire il tempo di arresto. Il calcolo dei secondi che mancano è dato dal tempo di stop del bruciatore (parametro) meno il tempo passato dallo spegnimento dei predosatori.
* **StopBruciatoreTamburo**(tamburo As Integer)  
  Arresta il bruciatore. Nel caso il bruciatore fosse già in spegnimento, si assicura che Predosatori e tamburo vengano fermati.
* **ControlliFiammaBruciatore**(tamburo As Integer)  
  Esegue i controlli sulla fiamma del bruciatore visualizzando gli specifici allarmi, se presenti.  
  Se i predosatori sono in start e il bruciatore non è acceso allora stoppa i predosatori, sempre che non sia stato selezionato un avvio senza bruciatore. Quindi mette il bruciatore in manuale.  
  Se il tamburo è in preriscaldamento e il bruciatore non è nella posizione di accensione, mette il modulatore in up.
* **ControlloBloccoBruciatore**(tamburo As Integer)  
  Esegue l’asservimento del bruciatore al filtro: se è acceso e l’aspiratore del filtro no allora viene spento anche lui, sempre che non sia selezionato l’esclusione dell’avviamento a caldo.  
  Lo stop al bruciatore viene dato anche se ho una richiesta di blocco della fiamma ma ho ancora il modulatore aperto, o la ventola accesa, o lo start del bruciatore da PLC.
* **AllarmeCombustibile**(tamburo As Integer, inizializza As Boolean), OlioCombInTemperatura\_change(tamburo As Integer)  
  Visualizza a video un eventuale allarme del combustibile
* **ModulatoreBruciatore\_change**(tamburo As Integer)  
  Chiamato nel ciclo di PlcInAnalogici, ricalcola la posizione del modulatore.
* **ModoRegolazioneDepressioneFiltroConDepressFiltroIN**()  
  Se la depressione del filtro rimane all'interno della forchetta prestabilita eseguo il controllo automatico sulla depressione dei due bruciatori e il filtro lo lascio stare, ovvero non tocco il modulatore dell’aspirazione. Se la depressione del filtro esce dalla forchetta prestabilita mi disinteresso del controllo della depressione dei due bruciatori e agisco sulla depressione del filtro per riportarla all'interno della forchetta
* **RegolazioneAriaFredda**(),**RegolazioneImpulsoAriaFredda**(UpDown As Integer, DeltaTemperatura As Integer), **DurataImpulsoAriaFredda**(DeltaTemperatura As Integer) As Integer  
  Gestiscono la regolazione del modulatore dell’aria fredda. La durata dell’impulso mandata al modulatore è ricavata con una specie di PID: Controllo ogni 5 secondi la temperatura di entrata del filtro "TimerAttesaRegolazioneAriaFredda"; più sono vicino al setpoint e minore sarà la durata dell'impulso di apertura o chiusura; l’impulso non può durare più di 2 secondi e non meno di 0,2 secondi.
* **AltaTemperaturaFumiTamburo**(tamburo As Integer)  
  **Mai usata**, spostato tutto in AltaTemperaturaFiltroSw()
* **AltaTemperaturaFiltroSw**(), TempEntrataFiltro\_change()  
  Se viene superata la temperatura massima del filtro, vengono stoppati bruciatore e tamburo
* **GestioneModulatoreBruc**(tamburo As Integer)  
  Se il bruciatore è in fase di spegnimento, il modulatore va in down, se si sta accendendo va in up.
* **LetturaScivoloTamburo**(tamburo As Integer, temperaturaScivolo As Long, plcInAnalogici\_Fatta As Boolean), **ValoreLettoTempScivolo\_change**(tamburo As Integer)  
  Chiamata in PlcInAnalogici().  
  Calcola la temperatura dello scivolo come una media delle ultime 15-30 (a seconda del parametro NumeroLettureScivolo) campionature.
* **TempTorre\_change**(Index As Integer, temperatura As Long), AggiornaTemperaturaTorre()  
  Aggiorna I dati di temperature di un componente (sabbia, inerte, etc) sulla torre.
* **ArrestoBrucTempoX**(tamburo As Integer)  
  Arresta il bruciatore dopo un certo tempo. Se è attivo lo stop della fiamma all’arresto del nastro collettore, e questo è effettivamente fermo, e il bruciatore ha finito la fase di spegnimento, allora arresta il bruciatore. Se non è attivo ho lo stop all’arresto del nastro allora ferma il bruciatore allo stop dei predosatori. Questo significa che il bruciatore ha un suo tempo di spegnimento, ma il suo spegnimento automatico è asservito o al nastro o ai predosatori.
* **BloccoFiammaBruciatore\_change**(tamburo As Integer)  
  Richiamata in PlcInDigitali(). Se il bruciatore è in automatico ed è stato dato il comando di blocco della fiamma, allora passa al manuale. Se è attiva l’esclusione di avviamento a caldo arresta sia il bruciatore che il tamburo, altrimenti solo il bruciatore.
* **StopBruciatore**(tamburo As Integer)  
  Arresta il bruciatore. Si assicura che i predosatori vengano fermati, a meno che non sia selezionata la modalita`di avvio predosatori senza bruciatore.
* **TempUscitaFiltro\_change**()  
  Chiamata in PlcInAnalogici(). Controlla se il filtro è arrivato in temperatura di lavoro.
* **TempFumiTamburo\_change**(tamburo As Integer)  
  Chiamata in PlcInAnalogici().Controlla che la temperatura di uscita fumi non superi la soglia di allarme.
* **LetturaDepressioneBruciatore**(tamburo As Integer, depressioneBruciatore As Long, plcInAnalogici\_Fatta As Boolean), **ValoreLettoDepressioneBruc\_change**(tamburo)  
  Chiamata in PlcInAnalogici().Calcola la depressione del bruciatore come una media delle ultime 15-30 (a seconda del parametro NumeroLettureScivolo) campionature.
* **ArrestoBrucITT**()  
  ITT è il sensore di temperatura fumi all’ingresso del filtro. Ferma il bruciatore se la temperatura è troppo alta.
* **AllarmePerditaValvoleBrucOC\_change**(tamburo As Integer), **AllarmePressioneBrucAlta\_change**(tamburo As Integer), **AllarmePerditaValvoleBruc\_change**(tamburo As Integer), **SicurezzaTempOlioComb\_change**(tamburo As Integer)  
  Chiamate in PlcInDigitali(). Vari allarmi che bloccano il bruciatore.
* **BrucInAccensione**(tamburo As Integer) As Boolean, StartBruciatore(tamburo As Integer)  
  Setta il bruciatore in accensione e fa update a video
* **ModulatoreFumiTamburo\_change**(tamburo As Integer), **DeflettoreAntincendioTamburo\_change**(), **CompressoreBruciatorePressioneInsuff\_change**(), **ModulatoreAspFiltro\_change**(), **BruciatoreInManuale**(tamburo As Integer), **TempSottoMesc\_change**(), **TempSondaAggiuntivaUscitaTamburo\_change**(), **FiltroModulatore\_change**(), **AriaFreddaFiltroModulatore\_change**(), **AriaTamburoModulatore\_change**(tamburo As Integer, apre As Boolean, chiude As Boolean), **TempIngressoTamburo\_change**(), **BruciatorePosizioneAccensione\_change** (tamburo As Integer), TempScambComb\_change(), PressioneInsufficienteOlioCombustibile\_change(tamburo As Integer), StartBruciatoreDaPLC\_change(tamburo As Integer), **PosizioneModulatoreAriaFredda\_change**()  
  Update a video
* **PortaModASetAvvioCaldo**(tamburo As Integer)  
  Modula il bruciatore per riportarlo al set di avvio a caldo
* **PortaModulatoreASet**(target As Long, letturaposizione As Long, Tolleranza As Integer, ByRef fatto As Boolean) As ModulatoreStatusEnum  
  Interviene sul modulatore del bruciatore, lasciando una zona di non intervento per evitare il pendolamento apri/chiudi.
* **ImpulsiContalitriCombustibile\_change**(tamburo As Integer)  
  ricalcola i litri di combustibile utilizzati, come numero di impulsi/impulsi per litro. Update a video
* **ControlloCadutaTamburoFiamma**(tamburo As Integer)  
  Ferma il bruciatore se non c’è l’avviamento a caldo e il tamburo smette di ruotare
* **ValoreForchetta**(valore As Double, min As Double, max As Double, SoloPositivo As Boolean) As Double  
  Normalizza il dato a un valore all’interno della forchetta (quindi non superiore a max o minore a min). **Mai usata**
* **AbilitaAvvCaldo**()  
  Abilita l’avviamento a caldo sull’interfaccia, a condizione che il motore dell’elevatore a caldo sia in moto.

##### Moduli interessati alla gestione essiccatore

BrucAuto.bas

ControlloBruc.bas

ParaTabBruc.bas

#### Metodi utilizzati nella gestione essiccatore:

BrucAuto.bas

306 - Private Sub GestSetRegBruciatore()

291 - Public Sub GestSetRegBruciatore2()

116 - Private Sub GestTotRegBruciatore(tamburo As Integer)

58 - Public Sub GestRegolazioneBruciatore(tamburo As Integer)

12 - Public Sub AttivaUscitePerRegolazioneEss(verso As Integer, smSec As Long, tamburo As Integer)

8 - Private Sub TableShift(ByRef lista() As Double, listaDim As Integer, verso As Integer)

ControlloBruc.bas

205 - Public Sub RegolazioneAriaAspiratore()

148 - Public Sub FiammaBruciatorePresente\_change(tamburo As Integer)

123 - Public Function VerificaCondizioniAvviamentoBruciatore(tamburo As Integer, aCaldo As Boolean) As Long

106 - Public Sub ConteggioTempoArrestoBruciatore(tamburo As Integer)

102 - Public Sub StopBruciatoreTamburo(tamburo As Integer)

78 - Public Sub ControlliFiammaBruciatore(tamburo As Integer)

70 - Public Sub ControlloBloccoBruciatore(tamburo As Integer)

69 - Public Sub AllarmeCombustibile(tamburo As Integer, inizializza As Boolean)

59 - Public Sub ModulatoreBruciatore\_change(tamburo As Integer)

59 - Public Sub GestioneFunzAutomaticoBruc(tamburo As Integer)

56 - Public Sub ModoRegolazioneDepressioneFiltroConDepressFiltroIN()

40 - Public Sub RegolazioneAriaFredda()

40 - Public Sub AltaTemperaturaFumiTamburo(tamburo As Integer)

40 - Public Sub AltaTemperaturaFiltroSw()

39 - Public Sub GestioneModulatoreBruc(tamburo As Integer)

38 - Public Sub ValoreLettoTempScivolo\_change(tamburo As Integer)

37 - Public Sub RegolazioneImpulsoAriaFredda(UpDown As Integer, DeltaTemperatura As Integer)

36 - Public Sub TempTorre\_change(Index As Integer, temperatura As Long)

35 - Public Sub ArrestoBrucTempoX(tamburo As Integer)

33 - Public Sub LetturaScivoloTamburo(tamburo As Integer, temperaturaScivolo As Long, plcInAnalogici\_Fatta As Boolean)

30 - Public Sub BloccoFiammaBruciatore\_change(tamburo As Integer)

29 - Public Sub StopBruciatore(tamburo As Integer)

29 - Public Sub AggiornaGraficaValvolaCombustibile\_Change()

26 - Public Sub TempUscitaFiltro\_change()

25 - Public Sub TempFumiTamburo\_change(tamburo As Integer)

24 - Public Sub LetturaDepressioneBruciatore(tamburo As Integer, depressioneBruciatore As Long, plcInAnalogici\_Fatta As Boolean)

24 - Public Sub ArrestoBrucITT()

23 - Public Sub AllarmePerditaValvoleBrucOC\_change(tamburo As Integer)

22 - Public Sub SicurezzaTempOlioComb\_change(tamburo As Integer)

22 - Public Sub ModulatoreAspFiltro\_change()

22 - Public Sub AllarmePressioneBrucAlta\_change(tamburo As Integer)

22 - Public Sub AllarmePerditaValvoleBruc\_change(tamburo As Integer)

22 - Public Sub AggiornaTemperaturaTorre()

21 - Public Function BrucInAccensione(tamburo As Integer) As Boolean

20 - Public Sub StartBruciatore(tamburo As Integer)

20 - Public Sub ModulatoreFumiTamburo\_change(tamburo As Integer)

20 - Public Sub CompressoreBruciatorePressioneInsuff\_change()

19 - Public Sub TempEntrataFiltro\_change()

19 - Public Sub DeflettoreAntincendioTamburo\_change(tamburo As Integer)

19 - Public Sub BruciatoreModulatore\_change(tamburo As Integer)

18 - Public Sub BruciatoreInManuale(tamburo As Integer)

18 - Public Function DurataImpulsoAriaFredda(DeltaTemperatura As Integer) As Integer

16 - Public Sub TempSottoMesc\_change()

16 - Public Sub PortaModASetAvvioCaldo(tamburo As Integer)

16 - Public Sub ImpulsiContalitriCombustibile\_change(tamburo As Integer)

15 - Public Sub ModoRegolazioneDepressioneFiltroConDepressTamburo()

15 - Public Function PortaModulatoreASet(target As Long, letturaposizione As Long, Tolleranza As Integer, ByRef fatto As Boolean) As ModulatoreStatusEnum

14 - Public Sub ValoreLettoDepressioneBruc\_change(tamburo As Integer)

14 - Public Sub TempSondaAggiuntivaUscitaTamburo\_change()

13 - Public Sub ControlloCadutaTamburoFiamma(tamburo As Integer)

13 - Public Function ValoreForchetta(valore As Double, min As Double, max As Double, SoloPositivo As Boolean) As Double

12 - Public Sub FiltroModulatore\_change()

12 - Public Sub AriaTamburoModulatore\_change(tamburo As Integer, apre As Boolean, chiude As Boolean)

12 - Public Sub AriaFreddaFiltroModulatore\_change()

9 - Public Sub TempIngressoTamburo\_change()

9 - Public Sub OlioCombInTemperatura\_change(tamburo As Integer)

9 - Public Sub BruciatorePosizioneAccensione\_change(tamburo As Integer)

----------------🡪

9 - Public Sub AbilitaAvvCaldo()

8 - Public Sub TempScambComb\_change()

7 - Public Sub PressioneInsufficienteOlioCombustibile\_change(tamburo As Integer)

5 - Public Sub StartBruciatoreDaPLC\_change(tamburo As Integer)

4 - Public Sub PosizioneModulatoreAriaFredda\_change()

ParaTabBruc.bas

213 - Public Function ParaTabBruc\_ReadFile() As Boolean

88 - Public Sub ParaTabBruc\_Apply()

17 - Private Function String2Fuel(fuel As String) As FuelType

#### Oggetti utilizzati nella gestione essiccatore:

ControlloBruc.bas

Public Enum ModulatoreStatusEnum

Definisce gli stati possibili di un modulatore (up/down/none)

Public Type ModulatoreType

Il modulatore è una gestione logica riconducibile ad una valvola, il controllo avviene sulla apertura/chiusura (up/down), il dato di verifica è la posizione reale della valvola, queste grandezze sono espresse in valore percentuale.

Public Type TypeGestioneFumiTamburo  
Rappresenta la gestione del modulatore della valvola di parzializzazione dei fumi all’interno del tamburo, tale gestione è manuale vedi CmdUpDownBruc\_MouseDown() CmdUpDownBruc\_MouseUp().

Public Type TypeGestioneVelocitaTamburo

Rappresenta la gestione della modulazione della velocità del tamburo, tale gestione è manuale vedi CmdUpDownBruc\_MouseDown() CmdUpDownBruc\_MouseUp().