*Documentazione tecnica dell’applicazione CyberTronic 500 - modulo Visual Basic*

Documentazione CyberTronic 500

*Versione: 9.6.46*

Sommario

[Documentazione CyberTronic 500 - Visual Basic 5](#_Toc488940001)

[Breve descrizione di un impianto Marini CyberTronic 500 5](#_Toc488940002)

[Componenti 5](#_Toc488940003)

[Modulo Visual Basic 6](#_Toc488940004)

[Struttura dell’ambiente di sviluppo della applicazione VB6 6](#_Toc488940005)

[Componenti dell'applicazione VB6 6](#_Toc488940006)

[Moduli sorgente componenti il progetto e loro sommaria descrizione 6](#_Toc488940007)

[Descrizione della implementazione per singolo componente 9](#_Toc488940008)

[Colloquio con il PLC 9](#_Toc488940009)

[Scrittura verso PLC 9](#_Toc488940010)

[Lettura da PLC 9](#_Toc488940011)

[Gestione Cisterne 10](#_Toc488940012)

[Panoramica 10](#_Toc488940013)

[Modello dei dati 10](#_Toc488940014)

[Parametrizzazione 11](#_Toc488940015)

[Parco leganti standard 11](#_Toc488940016)

[Logica Gestione completa 11](#_Toc488940017)

[Interfaccia e operazioni 11](#_Toc488940018)

[Emergenze 14](#_Toc488940019)

[Riscaldamenti 15](#_Toc488940020)

[PMB (Polymer Modified Bitumen) 15](#_Toc488940021)

[Logica Gestione semplificata 16](#_Toc488940022)

[Moduli interessati alla gestione cisterne: 20](#_Toc488940023)

[Metodi utilizzati nella gestione cisterne: 20](#_Toc488940024)

[Oggetti utilizzati nella gestione cisterne: 20](#_Toc488940025)

[Gestione Essiccatore 21](#_Toc488940026)

[Premessa 21](#_Toc488940027)

[Descrizione 21](#_Toc488940028)

[Parametri 21](#_Toc488940029)

[Logica 21](#_Toc488940030)

[Dettaglio Funzioni 21](#_Toc488940031)

[Moduli interessati alla gestione essiccatore 26](#_Toc488940032)

[Metodi utilizzati nella gestione essiccatore: 26](#_Toc488940033)

[Oggetti utilizzati nella gestione essiccatore: 26](#_Toc488940034)

[Gestione Predosaggio 27](#_Toc488940035)

[Premessa 27](#_Toc488940036)

[Descrizione 27](#_Toc488940037)

[Parametri e inizializzazione 27](#_Toc488940038)

[Logica 28](#_Toc488940039)

[Calcolo Uscita Predosatori 28](#_Toc488940040)

[Avviamento Predosatori 29](#_Toc488940041)

[Timer 29](#_Toc488940042)

[Retroazione Dosatori-Predosatori 29](#_Toc488940043)

[Nastro del riciclato 30](#_Toc488940044)

[Dettaglio Funzioni 30](#_Toc488940045)

[Moduli interessati alla gestione predosaggio 35](#_Toc488940046)

[Metodi utilizzati nella gestione predosaggio: 35](#_Toc488940047)

[Oggetti utilizzati nella gestione predosaggio: 35](#_Toc488940048)

[Gestione Dosaggio 37](#_Toc488940049)

[Descrizione 37](#_Toc488940050)

[Logica 37](#_Toc488940051)

[Dettaglio Funzioni 37](#_Toc488940052)

[Moduli interessati alla gestione dosaggio 39](#_Toc488940053)

[Metodi utilizzati nella gestione dosaggio: 39](#_Toc488940054)

[Oggetti utilizzati nella gestione dosaggio: 40](#_Toc488940055)

[Gestione Motori 41](#_Toc488940056)

[Descrizione 41](#_Toc488940057)

[Dettaglio Funzioni 41](#_Toc488940058)

[Moduli interessati alla gestione motori 42](#_Toc488940059)

[Metodi utilizzati nella gestione motori: 42](#_Toc488940060)

[Oggetti utilizzati nella gestione motori: 42](#_Toc488940061)

[Gestione Sili/Navette 43](#_Toc488940062)

[Premessa 43](#_Toc488940063)

[Descrizione 43](#_Toc488940064)

[Logica 43](#_Toc488940065)

[Dettaglio Funzioni 43](#_Toc488940066)

[Moduli interessati alla gestione sili: 44](#_Toc488940067)

[Metodi utilizzati nella gestione sili: 44](#_Toc488940068)

[Oggetti utilizzati nella gestione sili: 44](#_Toc488940069)

[Allegati : 45](#_Toc488940070)

# Documentazione CyberTronic 500 - Visual Basic

Questo documento ha lo scopo di descrivere il funzionamento dell’applicazione utilizzata per la gestione degli impianti Marini CyberTronic 500 con particolare riferimento alla parte scritta in Microsoft® Visual Basic 6®.

## Obiettivo del presente documento

L’ obiettivo principale di questo documento è il supporto alla riscrittura o porting delle funzionalità di tale applicazione verso altri ambienti operativi e di sviluppo.

## Breve descrizione di un impianto Marini CyberTronic 500

Il CyberTronic500 è un impianto di produzione asfalto, la modalità di produzione è batch.

## Componenti

Un impianto CyberTronic 500 è composto logicamente dai seguenti moduli:

* Predosaggio (aggregati e riciclato freddo)

Stoccaggio e distribuzione del materiale utilizzato nella produzione di asfalto (aggregati e riciclato freddo)

Il predosaggio è inoltre composto da :

* Tamburo Essiccatore

Mescolazione e riscaldamento del materiale proveniente dalla zona di predosaggio.

* Filtro

Filtraggio e recupero delle polveri generate dal processo di mescolazione aggregati (filler) per loro successivo riuso nel modulo di dosaggio.

* Cisterne (bitume e riciclato caldo)

Stoccaggio e distribuzione del materiale utilizzato nella produzione di asfalto (bitume e riciclato caldo)

* Dosaggio

Cernita e successiva mescolazione del materiale proveniente dal Tamburo essiccatore con aggiunta di bitume e riciclato caldo proveniente dalla zona cisterne e di eventuali additivi.

* Silos

Stoccaggio del materiale prodotto dalla zona di dosaggio (asfalto)

* Gestione Motori

Ogni modulo è a sua volta composto da altri moduli componenti quali nastri, elevatori, bilance, bruciatore, valvole, sensori, attuatori, mescolatore, navetta...

La logica di funzionamento dei componenti dell'impianto è quasi completamente gestita dal PLC attraverso gli I/O digitali e analogici.

L'interazione con l'utente e la presentazione dello stato dei moduli e del funzionamento dell'impianto, nonché una parte di logica, è gestita da software applicativo.

Tale software applicativo è suddiviso, dal punto di vista dello sviluppo, in modulo Visual Basic 6 (VB6) e moduli C#.

## Modulo Visual Basic

Il modulo VB6 in particolare si occupa della presentazione visuale dello stato dell'impianto e della gestione di gran parte dell'interazione dell'utente con l'impianto medesimo (HMI / UX).

### Struttura dell’ambiente di sviluppo della applicazione VB6

### Componenti dell'applicazione VB6

L'applicazione VB6 è suddivisa in moduli sorgente (estensione .bas) e moduli di gestione forms (estensione .frm)

### Moduli sorgente componenti il progetto e loro sommaria descrizione

I moduli sorgente componenti il progetto (.bas) possono essere suddivisi logicamente in :

#### Gestione Comunicazione con il PLC : modulo di comunicazione con il PLC

NetWork.bas

#### Gestione Componenti: moduli di implementazione componenti

GestioneCisterneCommon.bas  
GestioneCisterneTipo5.bas  
GestioneCisterne.bas  
GestioneLivelliSiliFillerVaglio.bas  
GestioneAmperometri.bas  
Viatop.bas  
GestioneFiltroESonde.bas  
GestionePredNastriC.bas  
GestioneRiciclato.bas  
GestioneSilo.bas  
GestioneSiwarex.bas  
GestionePredosatori.bas  
GestioneAquablack.bas  
GestioneSiloGenerale.bas  
GestioneMotori.bas  
GestioneDosaggio.bas  
ControlloBruc.bas  
BrucAuto.bas  
GestioneCaldaie.bas

#### Gestione Funzioni : implementazione delle funzionalità dell’applicazione

GestioneCodaMateriale.bas  
GestioneOreLavoro.bas  
Configurazione.bas  
GestioneAdditivi.bas  
GestioneJob.bas  
MemManuale.bas  
GestioneGrafica  
Trend.bas  
ModPID.bas  
GestioneCmdDeoSoffioAntiadVib.bas  
GestioneAllarmiIconeAllarmi.bas

#### Gestione Parametri: lettura dei parametri di funzionamento del sistema

ParaTabTrend.bas  
ParaTabGeneral.bas  
ParaTabLeg.bas  
ParaTabVarie.bas  
ParaTabAmp.bas  
ParaTabCist.bas  
ParaTabMotor.bas  
ParaTabPred.bas  
ParaTabSilo.bas  
ParaTabComp.bas  
ParaTab.bas  
ParaTabAquablack.bas  
ParaTabAdd.bas  
ParaTabBruc.bas  
ParaTabDebug.bas  
ParaTabSchiumato.bas  
ParaTabComandi.bas

#### Moduli generici : Implementazione di funzionalità generiche

Help.bas  
CodaCircolare.bas  
GestioneLCPC.bas  
Printer.bas  
ControlliVari.bas  
ControlloDatiInseriti.bas  
GestioneBit.bas  
GestioneCaratteri.bas  
Registro.bas  
BusSytem.bas  
DataBase.bas  
GestioneFile.bas  
GestioneLog.bas  
Security.bas  
ModuleShowForm.bas  
GestioneMessaggistica.bas  
GestioneLingue.bas  
Stampe.bas

#### Forms – User Interface e implementazione

frmNewValue.frm

FrmSchiumatura.frm

MessageBox.frm

FrmCalibBilCamion.frm

FormAvvRidotto.frm

FrmSiloGenerale.frm

FrmSiwarexPara.frm

AvviamentoComandi.frm

FormPIDBruc.frm

FormProgRiscaldamentiCaldaie.frm

FrmCalcolaImpasti.frm

AvviamentoMotori.frm

FrmComandiCisterne.frm

FrmNetti.frm

FrmStatoPredosatore.frm

FrmInversionePCL.frm

FrmGestioneTimer.frm

FrmMessaggioAvviso1.frm

CP240.frm

FormAquablack.frm

FrmTaraBilancePN.frm

## Descrizione della implementazione per singolo componente

### Colloquio con il PLC

Il colloquio con il PLC viene implementato in Network.bas e CP240.frm

### Scrittura verso PLC

In CP240.frm è presente la procedura **tmrRicTrasNET\_Timer** , essa viene eseguita allo scadere di alcuni timers, in particolare :

Timer 1: ogni 250ms

Scrittura delle variabili interne sui tags dei PLC (plc4/plc2…) suddivisa per PLC e uscite digitali e analogiche, vedi NetWork.bas PlcOutDigitali / PlcOutAnalogici

Timer 0: ogni 10s

Controllo periodico dello stato della comunicazione

### Lettura da PLC

In CP240.frm sono presenti le procedure :

opcData\_OnDataChanged  
OPCDataCisterne\_OnDataChanged  
OPCDataAquablack\_OnDataChanged  
OPCDataSchiumato\_OnDataChanged

Le quali vengono eseguite sul cambiamento dei tags plc relativi ad ogni plc gestito con un timer di aggiornamento di 300ms.

All’interno di queste procedure vengono letti i tags del PLC e copiati nelle strutture dati che implementano la presentazione e la logica di funzionamento dei componenti gestiti nell’applicazione, vedi Network.bas : PLCInDigitali/PLCInAnalogic

## Gestione Cisterne

### Panoramica

* Logica di funzionamento completamente integrata nel PLC
* CYB500 prevede solo funzioni di controllo e visualizzazione
* Max 6 cisterne bitume
* Max 2 cisterne emulsione
* Max 2 cisterne combustibile

Esistono tre gestioni, dovute alla presenza o meno di un plc dedicato:

* Gestione completa:
  + PLC dedicato (tab plc2 dell’OPCTAGS)
  + Gestione completa delle cisterne (livello, temperatura, valvole) e delle valvola del parco bitume con PLC dedicato.

Il parco viene gestito lanciando la funzionalità (Travasi, Ricircolo, Alimentazione Torre) con la pompa che si vuole usare (scegliendo tra pompa di carico e di circolazione) ; i singoli comandi alle valvole/pompe vengono dati in maniera trasparente per l’operatore

* Gestione reworked:
  + integrata nel PLC generale (tab plc2\_reworked dell’OPCTAGS)
  + nuova gestione completa del parco legante con PLC dedicato, sia come grafica che come gestione delle funzionalità.

Al momento la gestione è specifica per la commessa MP17006 e non è ancora chiaro se andrà a sostituire la gestione completa.

* Gestione ridotta:
  + integrata nel PLC generale (plc4cist)
  + non è presente un PLC dedicato e le funzionalità sono piuttosto ridotte.

Vengono visualizzati livelli, temperature mentre l’unico comando è la gestione della cisterna selezionata per alimentare la torre.

### Modello dei dati

Gestione Cisterne

CYB500

OPCTags.xml

Cisterne.ini

(solo per memorizzazione materiali)

()

ParameterValues.xml

Materiali

Log Allarmi

Plc Main con gestione semplificata

(tab plc4)

Plc Cisterne dedicato con piena gestione

(tab plc2)

### Parametrizzazione

I parametri a livello di gestione generale si trovano nella sezione Bitume del file (ParameterValues.xml), mentre nei vari paragrafi CisternaBitume1..N, CisternaCombustibile1..N, CisternaEmulsione1..N si trovano i parametri specifici per cisterna.

Nel file Cisterne.ini vengono memorizzati i materiali nel caso di Gestione Semplificata; in questo caso infatti è possibile associare i materiali della categoria Bitume direttamente alle cisterne visualizzate nel sinottico. L’associazione viene utilizzata per un controllo di coerenza materiale allo start dosaggio; il programma infatti controlla che il materiale associato al bitume in ricetta sia lo stesso materiale associato alla cisterna selezionata dalla gestione semplificata e in caso di discordanza non permette lo start della produzione.

Nel caso di gestione completa i parametri vengono trasferiti (al primo avvio e ad ogni variazione degli stessi) mediante l’OPCTAGS al PLC dedicato, mentre in caso di gestione completa al PLC principale (sempre mediante l’OPCTAGS).

### Parco leganti standard

#### 

VLINPCAR: valvola sulla linea mandata pompa di carico (NA)

VANTIRITORNO: valvola sulla linea aspirazione pompa di carico (NC)

VENTPCAR: valvola manuale sull’aspirazione pompa di carico (NC)

3VIEBYPASS: valvola tre vie per bypassare la pompa di circolazione (NC)

3VIELINEA: valvola tre vie per alimentare la torre in produzione; questa valvola non è visibile al parco legante e riceve il comando dal PLC principale

### Logica Gestione completa

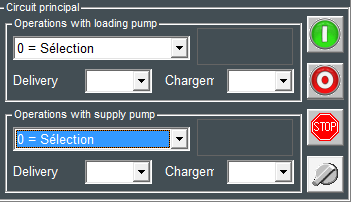
La gestione completa permette la piena gestione di valvole e pompa di carico del parco bitume, la visualizzazione di livelli, temperatura, agitatori e riscaldamenti di ogni cisterna.

Il programma, che risiede in un PLC dedicato, viene parametrizzato da un configuratore in VB6 per il parco specifico coprendo parchi fino ad 8 cisterne (con una valvola di ingresso e una di uscita), 12 valvole sul campo ed una sola pompa di circolazione nell’impianto.

### Interfaccia e operazioni

L’attivazione della gestione completa permette la scelta del pannello gestione cisterne cliccando sull’icona delle cisterne che appare nel Top Bar del Cybertronic; oltre al pannello comandi appariranno le cisterne configurate nella parte bassa del sinottico (dettagliate con livelli, temperature e stato dei riscaldamenti)

Come sopra accennato l’operatore non ha il controllo sul singolo comando alla valvola/agitatore/pompa ma avvia delle funzioni del parco bitume che coinvolgono le cisterne. Tali funzionalità sono svolte dalla pompa di carico e di circolazione e vengono suddivise in due pannelli con tre combo box ciascuno, collocate nel pannello comandi cisterne come di seguito descritto:



**Interfaccia comandi**

**START:** lancia la/le operazioni selezionate

**ABORT:** abortisce la/le operazioni selezionate e torna in alimentazione torre

**EMERGENZA:** si porta allo step di emergenza dove chiude tutto

**AUX E AGITATORI:** comandi ausiliari del parco (riscaldamenti..) che possono essere soltanto accesi o spenti mediante selettori. Oltre ai comandi ausiliari si possono comandare gli agitatori (se configurati nella cisterna) che si accenderanno solo in caso non sia presente un livello minimo o una minima temperatura.

**COMBO OPERAZIONE:** selezione operazione per pompa

**COMBO CISTERNA CARICO:** selezione cisterna da caricare per l’operazione per pompa (visibile in caso di carico, alimentazione esterna con pompa di circolazione e travaso)

**COMBO CISTERNA MANDATA:** selezione cisterna che alimenta per l’operazione per pompa (visibile in caso di ricircolo ed alimentazione torre con pompa di circolazione)

**LISTA OPERAZIONE SELEZIONABILI NELLE COMBO OPERAZIONE:**

* **Nessuna Op con PCar**
* **Carico con PCar**:

si apre la valvola d’entrata della cisterna selezionata in carico dalla combo cisterna di carico, e si apre la valvola di linea pompa di carico; l’operazione parte solo quando l’operatore apre la valvola manuale sull’aspirazione pompa di carico e viene dato un fronte di salita sul consenso esterno pompa di carico

* **Travaso con PCar**

si apre la valvola d’entrata della cisterna selezionata in carico dalla combo, e la valvola d’uscita della cisterna selezionata in mandata dalla combo cisterna di mandata; vengono inoltre aperte la valvola di linea pompa di carico e la valvola di antiritorno.

L’operazione parte solo quando l’operatore chiude la valvola manuale sull’aspirazione pompa di carico e viene dato un fronte di salita sul consenso esterno pompa di carico

* **Ricircolo con PCar**

si aprono la valvola d’entrata e di uscita della cisterna selezionata in travaso dalla combo; vengono inoltre aperte la valvola di linea pompa di carico e la valvola di antiritorno.

L’operazione parte solo quando l’operatore chiude la valvola manuale sull’aspirazione pompa di carico e viene dato un fronte di salita sul consenso esterno pompa di carico

* **Nessuna Op con PCirc**
* **Alimentazione Torre con PCirc**

si aprono la valvola d’entrata e di uscita della cisterna selezionata in alimentazione dalla combo; la 3vie di bypass pompa di circolazione viene mantenuta in posizione a cisterne

* **Carico con PCirc**

si apre la valvola d’entrata della cisterna selezionata in carico dalla combo cisterna di carico e la valvola di antiritorno; la 3vie di bypass pompa di circolazione viene mantenuta in posizione a cisterne

* **Travaso con PCirc**

si aprono la valvola d’entrata della cisterna selezionata in travaso dalla combo e la valvola d’uscita della cisterna selezionata in mandata dalla combo; la 3vie di bypass pompa di circolazione viene mantenuta in posizione a cisterne

* **Ricircolo con PCirc**

si aprono la valvola d’entrata e d’uscita della cisterna selezionata in ricircolo dalla combo; la 3vie di bypass pompa di circolazione viene mantenuta in posizione a cisterne

* **Alimentazione esterna**

si apre solo la valvola di antiritorno e tutte le valvole cisterne rimangono chiuse; la 3vie di bypass pompa di circolazione viene mantenuta in posizione a cisterne e si attende l’apertura della valvola manuale d’entrata pompa di carico

* **Op doppia: carico con pompa di carico e alimentazione torre con pompa di circolazione**

Uno start con la selezione di carico con pompa di carico ed alimentazione torre con pompa di circolazione avvia le operazioni in simultanea ed è molto utilizzato in cantiere per permettere ad un camion di caricare il bitume in una cisterna continuando la produzione.

Praticamente le valvole vengono commutate come nelle singole operazioni ad eccezione della 3vie bypass che viene mantenuta in posizione di bypass per separare la linea di carico delle cisterne dalla pompa di circolazione.

In tal modo è possibile lanciare una singola funzione con la pompa di carico, una funziona con la pompa di circolazione, oppure ancora una funzione con la pompa di carico in contemporanea ad una funzione con la pompa di circolazione.

Nel caso in cui la/le funzionalità non possano essere eseguite nel parco la funzione non viene lanciata e si visualizza un allarme temporaneo di rifiuto operazione nel pannello comandi cisterne.

Di seguito è riportato uno schema a blocchi dello stato del parco bitume (implementato nella gestione PLC):

Start Alimentazione Torre obbligato

Operazione Generica

Start Operazione

Abort Operazione

Start Alimentazione Torre obbligato

Stato di Blocco

(uscite a =0 e valvole NA aperte)

Avvio PLC

Pressione insufficiente

Emergenza

Alimentazione Torre

Emergenza

Come si può notare all’avvio del PLC o in caso di pressione aria insufficiente il parco si porta in uno stato di blocco in cui le uscite vengono azzerate e si aprono le valvole NA (di norma la Cisterna1 e la valvola di linea pompa di carico) per problemi di espansione bitume.

Per partire dallo stato di blocco si è obbligati a selezionare un alimentazione torre in cui si predispone una cisterna ad alimentare l’impianto non appena il PLC principale commuta la 3vie di linea in pesata (durante il dosaggio); questa operazione è la principale in quanto con uno start è possibile lanciare una qualsiasi altra operazione e con un abort si torna sempre ad alimentare la torre.

### Emergenze

In caso di stop di emergenza o di errore ad operazione avviata il parco si porta in Emergenza dove tutte le valvola vengono chiuse; si può ripartire da questo stato soltanto alimentando la torre.

Essendo il parco configurabile, il programma testa la correttezza di ogni finecorsa (lasciando chiuse le valvole non interessate all’operazione).

Pertanto una volta a regime l’operazione, qualsiasi finecorsa errato per più di 2 secondi, porta in emergenza il parco.

Il programma inoltre svolge i controlli basilari di sicurezza quali:

* verificare che una cisterna coinvolta in un operazione non sia fredda (rischiando di bloccare gli attuatori delle valvole)
* verificare che una cisterna in fase di carico non vada in alto livello
* Carico (con pompa di carico o circolazione): si controlla che la cisterna caricata non vada in alto livello
* Travaso (con pompa di carico o circolazione): si controlla che la cisterna caricata non vada in alto livello
* Alimentazione esterna con pompa di carico: si controlla che la cisterna caricata non vada in alto livello

### Riscaldamenti

I riscaldamenti, se abilitati nel PLC delle cisterne, forniscono due uscite digitali per:

* Base: riscaldamento nella parte bassa della cisterna
* Booster: riscaldamento posizionato più in alto nella cisterna

Entrambi lavorano prendendo un set ideale di temperatura, scrivibile anche da pc e posizionato in una label accanto alla temperatura corrente di ogni cisterna in basso nel sinottico.

La base è comandata mediante un PID digitale dove il setpoint è proprio questo set ideale.

Il booster si attiva invece quando la temperatura scende al di sotto di un deltaT (scrivibile nei parametri generali delle cisterne) rispetto al set ideale; per ragioni di sicurezza il booster parte solo se la cisterna non si trova in minimo livello, in maniera che il bitume copra le serpentine del booster stesso.

L’intervento dei riscaldamenti è visibile sulle cisterne del sinottico nel PC.

### PMB (Polymer Modified Bitumen)

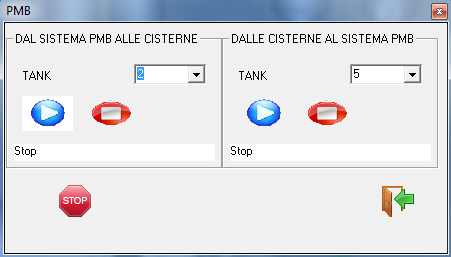
Nella gestione semplificata è prevista anche l’opzione di interfacciamento della gestione del parco bitume Marini con un parco bitume modificato esterno.

Tale parco esterno è attivabile solamente mentre il parco Marini si trova in condizioni di emergenza o in alimentazione torre (non mentre si sta eseguendo un operazione con pompa di carico o di circolazione) e non permette la produzione.

A livello circuitale Il PMB si interfaccia alla linea di carico delle cisterne marini mediante una valvola pneumatica connessa alla mandata di una pompa interna al PMB.

Si interfaccia inoltre alla linea di mandata delle cisterne Marini mediante un’altra valvola pneumatica connessa all’aspirazione di una pompa interna al PMB.

L’interfaccia si presenta così:



Le operazioni possibili sono:

* **Carico di una cisterna Marini dal sistema PMB**

L’operazione viene lanciata e abortita premendo i pulsanti di start e abort dal riquadro di sinistra dopo aver selezionato la cisterna Marini di carico mediante la combobox.

Lo start dell’operazione comanda l’apertura della valvola pneumatica del PMB che si affaccia sulla linea di carico delle cisterne Marini.

Il programma, in fase di accettazione dell’operazione e mentre l’operazione stessa è in corso, verifica che la cisterna Marini da caricare non sia in alto livello o fredda; in tal caso viene tolto il comando di apertura alla valvola pneumatica del PMB e viene alzata un uscita di operazione rifiutata per massimo livello verso il PMB

* **Carico del sistema PMB da una cisterna marini**

L’operazione viene lanciata e abortita premendo i pulsanti di start e abort dal riquadro di destra dopo aver selezionato la cisterna Marini di alimentazione mediante la combobox.

Lo start dell’operazione comanda l’apertura della valvola pneumatizzata del PMB che si affaccia sulla linea di alimentazione delle cisterne marini.

Il programma, in fase di accettazione dell’operazione e mentre l’operazione stessa è in corso, verifica che la cisterna marini in alimentazione non sia in basso livello o fredda, e che il PMB non segnali un alto livello o una sicurezza interna; in tal caso viene tolto il comando di apertura alla valvola pneumatica del PMB e viene alzata un uscita di operazione rifiutata per minimo livello verso il PMB

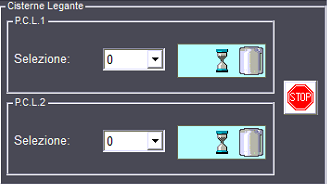
Mediante il pulsante di STOP in basso è possibile interrompere immediatamente una qualsiasi delle due operazioni del PMB.

Ad ogni stop o abort di un’operazione in corso con il PMB, il sistema riporta in alimentazione torre la gestione generale delle cisterne.

### Logica Gestione semplificata

L’abilitazione della gestione semplificata permette solo la visualizzazione di livelli, temperatura, agitatori visualizzati nella parte bassa del sinottico sulle cisterne; abilitando i comandi è inoltre possibile selezionare una cisterna in alimentazione torre senza lanciare nessun’altra operazione.

Il pannello per la selezione della cisterna che alimenta la torre è selezionabile dalla Top bar del Cybertronic e si presenta come segue:



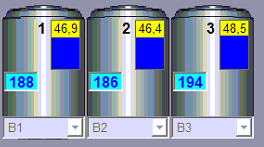
Mediante il parametro numero cisterne su pompa di circolazione si popolano le combo box di “Selezione” associando le giuste cisterne alla pompa di circolazione 1 e 2.

La selezione avviene scegliendo la cisterna dalla combo (0= nessuna cisterna); il PLC alzerà l’uscita che elettricamente andrà ad aprire la valvola di uscita della cisterna selezionata.

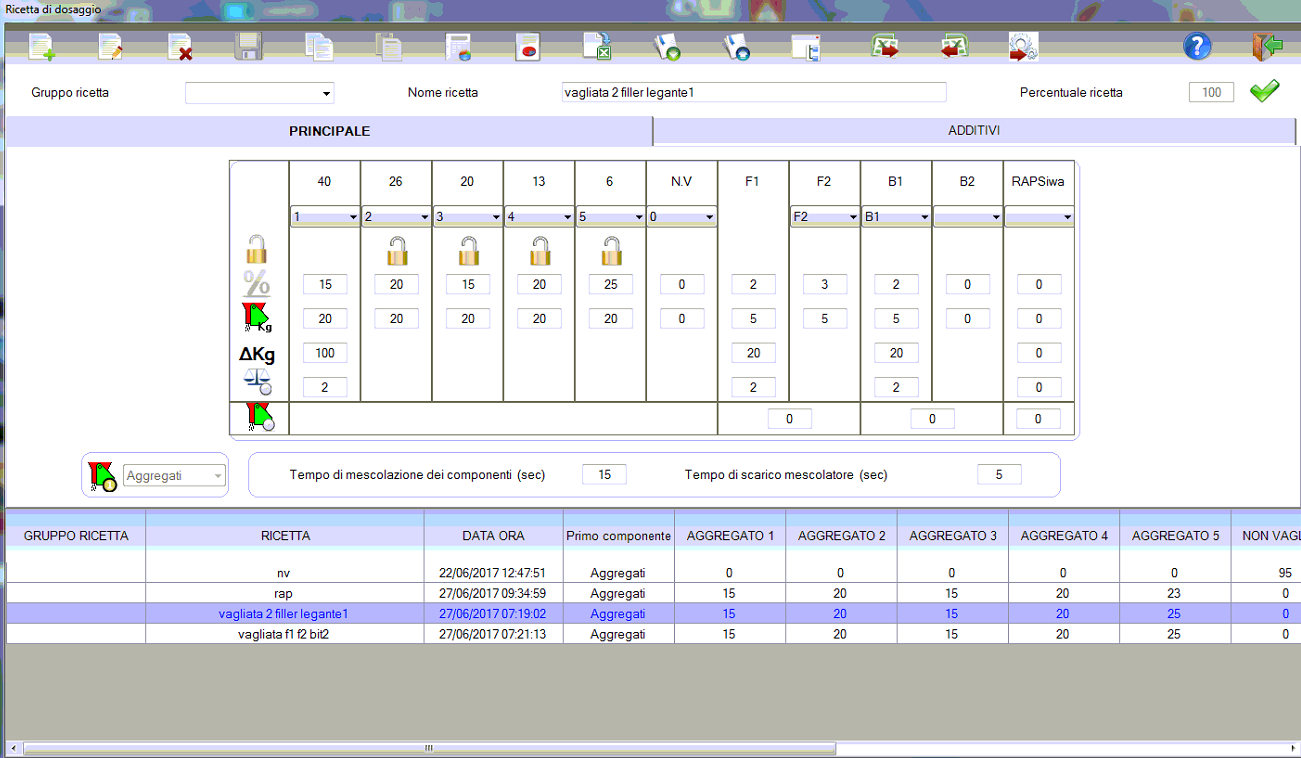
Sul finecorsa di aperto della valvola di uscita della cisterna di selezione sparisce l’icona della clessidra di attesa.

Col pulsante di stop vengono tolti i comandi di selezione.

Una funzionalità presente in questa gestione è il controllo di coerenza tipo di Bitume. Nelle cisterne sul sinottico infatti è possibile assegnare un materiale di categoria Bitume ad ogni cisterna (il quale viene memorizzato nel file Cisterne.ini), come si evidenzia nella seguente immagine:



In fase di produzione, è possibile lanciare soltanto una ricetta di dosaggio dove il bitume selezionato tra i materiale di categoria Bitume (e creato in Gestione Materiali) è lo stesso della cisterna selezionata in alimentazione torre.



La cisterna in alimentazione torre, se coerente con la ricetta si colorerà di verde mentre in caso di incoerenza (può avvenire se ad esempio viene cambiata la cisterna che alimenta la torre a dosaggio avviato) diventerà rossa e, ultimato il ciclo corrente, verrà stoppato il dosaggio.

#### Dettaglio Funzioni

Non esistono funzioni critiche per la gestione delle cisterne. Citiamo le poche funzioni di settaggio.

#### ParaTabCist.bas

Gestione parametri

#### GestioneCisterne.bas

Nessuna funzione con compiti di gestione

#### GestioneCisterneCommon.bas

Principali funzioni:

* CistAccettaErrore(accetta As Boolean)  
  Invia al PLC l’acknowledge dell’errore
* CistSetMateriale(Index As Integer)  
  Legge i dati del materiale usato nelle singole cisterne dalla tabella del database Materiali che ha i seguenti campi:
  + IdMateriale
  + Nome
  + Data
  + Categoria
  + PercUmidita
  + PercBitume
  + Note

Per sapere l’ID associato si guarda a un file Cisterne.ini, dove si trovano sezioni del tipo seguente:

[CisternaBitume1]

Materiale=B1

[CisternaBitume2]

Materiale=B2

…

* CistSetTemperatura(Index As Integer)  
  Setta la temperatura delle cisterne, che avviene mediante controllo PID.
* InviaComandiAgitatori(indice As Integer, stato As Boolean)  
  Accende o spegne gli agitatori di una cisterna
* ScriviDatiComandiAuxCisterneOnOff(ByVal indice As Integer, ByVal avvio As Boolean)  
  Avvia o ferma i comandi ausiliari
* ScriviDatiPidComandiCisterne(indice As Integer)  
  Manda al PLC i parametri del PID
* ScriviDatiRegolazioneTempCisterne()  
  Manda al PLC le nuove impostazioni di temperatura

#### GestioneCisterneTipo5.bas

* SingolaCisternaInviaParametri(ByRef cisterna As OggettoCisterna, cisternaOffset As Integer), SingolaCisternaRidInviaParametri(ByRef cisterna As OggettoCisterna, cisternaOffset As Integer)  
  Manda i dati di una singola cisterna al PLC:
  + Parametri di struttura (cisterna orizzontale, diametro, lunghezza, numero valvole)
  + Parametri di livello (min, max, tipo, soglie allarme)
  + Parametri di temperatura (min, max, soglie allarme, zona morta)
  + Comandi di valvole (inversioni)
* CaricaParametriCisterne()  
  Carica i parametri di configurazione. Si tratta dei dati di scambio col PLC. Parte di essi vengono letti dal file di configurazione xml, altri vengono letti dal PLC in quanto legati ad uno stato attuale dell’impianto (ad esempio le abilitazioni delle valvole di mandata, ritorno, carico e ausiliaria)
* ControllaCisterneAllarmi(ByRef IdDescrizione As Integer, ByRef CodiceAllarme As String), Public Sub ControllaCisterneAllarmiRidotto(ByRef IdDescrizione As Integer, ByRef CodiceAllarme As String)  
  Controlla gli allarmi delle cisterne. Al verificarsi di ogni allarme viene creato un record sul db e viene eseguito l’aggiornamento a video. Nel momento in cui l’allarme viene eliminato dalla lista allarmi viene eseguito l’aggiornamento a database della data di fine allarme e viene eliminata la segnalazione a video del suddetto allarme.
* GestioneComboCisterne(indice As Integer)  
  Gestisce la visualizzazione a video delle cisterne attive. La logica implementata nell’interfaccia prevede che , in dipendenza di alcuni parametri, alcune combinazioni non siano selezionabili dall’operatore:
  + Selezione pompa di carico:
    - In fase di carico non è possibile selezionare come pompa di carico la cisterna di mandata
  + Selezione pompa di alimentazione:
    - In fase di carico, non è possibile selezionare la cisterna di mandata
    - In fase di alimentazione esterna non è possibile selezionare né la cisterna di mandata, né quella di carico
    - In fase di alimentazione torre non è possibile selezionare la cisterna di carico.

A seconda della selezione effettuata vengono settate le variabili del PLC per svolgere l’operazione richiesta (cisterne in gioco e codice dell’operazione da svolgere)

* LeggiDatiPLCCisterneBitume(), LeggiDatiPLCCisterneBitumeRid(), LeggiDatiPLCCisterneCombustibile(), LeggiDatiPLCCisterneEmulsione()  
  Funzioni per raccogliere i valori delle variabili PLC.
* ScriviDatiPLCCisterne(), ScriviDatiPLCCisterneRid()  
  Scrive tutte le variabili di output dell’applicazione sul PLC (per il quale sono input):
* Watchdog
* Acknowledge allarme
* Numero cisterne
* Valore di ritorno dei motori delle pompe
* Valore degli agitatori

### Moduli interessati alla gestione cisterne:

GestioneCisterne.bas

GestioneCisterneTipo5.bas

GestioneCisterneCommon.bas

FrmComandiCisterne.frm

### Metodi utilizzati nella gestione cisterne:

FrmComandiCisterne.frm

Private Sub APButtonAgitatore\_Click(Index As Integer)

Private Sub Form\_Activate()

Private Sub Form\_Load()

Private Sub PosizionaFrameSupp()

Private Sub CmdRefresh(indice As Integer)

Private Sub APButtonCmd\_Click(indice As Integer)

Private Sub imgPulsanteForm\_Click(Index As Integer)

Private Sub TmrCmdOn\_Timer()

Private Sub TxtTempMesc\_GotFocus()

Private Sub TxtTempMesc\_LostFocus()

Private Sub imgPulsanteForm\_MouseMove(Index As Integer, Button As Integer, Shift As Integer, x As Single, Y As Single)

Private Sub imgPulsanteForm\_MouseDown(Index As Integer, Button As Integer, Shift As Integer, x As Single, Y As Single)

Private Sub imgPulsanteForm\_MouseUp(Index As Integer, Button As Integer, Shift As Integer, x As Single, Y As Single)

Private Sub Form\_MouseMove(Button As Integer, Shift As Integer, x As Single, Y As Single)

Private Sub LoadImmaginiPulsantePlus(Index As Integer, stato As StatoPulsantePlus)

Public Sub UpdatePulsantiForm()

GestioneCisterne.bas

Public Sub CistVisualizzaAttesa(cisterna As Integer)

Public Sub CistInizializza()

Public Sub CistVisualizzaCarico(cisterna As Integer)

Public Sub CistVisualizzaErrore(cisterna As Integer)

Public Sub CistVisualizzaIdle(cisterna As Integer)

Public Sub CistVisualizzaLivello(cisterna As Integer, ByVal percento As Integer, ByVal tons As Double)

Public Sub CistVisualizzaRicircolo(cisterna As Integer, aperta As Boolean, chiusa As Boolean, allarme As Boolean)

Public Sub CistVisualizzaSelezione(cisterna As Integer)

Public Sub CistVisualizzaTemperatura(cisterna As Integer, ByVal Value As Double)

Public Sub CistVisualizzaTravaso(cisternaSrc As Integer, cisternaDst As Integer)

Public Sub CistVisualizzaUscita(cisterna As Integer)

Public Sub CistVisualizzaValvolaEntrata1(cisterna As Integer, aperta As Boolean, chiusa As Boolean, allarme As Boolean)

Public Sub CistVisualizzaValvolaUscita1(cisterna As Integer, aperta As Boolean, chiusa As Boolean, allarme As Boolean)

Public Sub CistVisualizzaValvolaUscita2(cisterna As Integer, aperta As Boolean, chiusa As Boolean, allarme As Boolean)

Public Sub CisterneLeggiDatiPLC()

Public Sub CisterneScriviDatiPLC()

Public Sub VisualizzaAgitatoreCisterne(cisterna As Integer, acceso As Boolean)

GestioneCisterneCommon.bas

Public Sub AggiornaGrafPIDCisterne()

Public Sub CistAccettaErrore(accetta As Boolean)

Public Sub CistAzzeramentoTara(cisterna As Integer)

Public Sub CistConnessionePLC()

Public Sub CistGestioneLoop()

Public Sub CistSetMateriale(Index As Integer)

Public Sub CistSetTemperatura(Index As Integer)

Public Sub CistShowMenu(Index As Integer)

Public Sub CisterneCaricaImmagini()

Public Sub CompilaListaCistDosaggio()

Public Sub CreaTagCisterneS7\_Ver9()

Public Sub EnableComboMatCP240(visibility As Boolean)

Public Sub GestioneMaterialeCisterneRidotto()

Public Sub GestioneStatoCisterneRidotto()

Public Sub InviaComandiAgitatori(indice As Integer, Stato As Boolean)

Public Sub LeggiDatiComandiAuxCisterneOnOff(ByVal indice As Integer, ByRef uscita As Boolean, ByRef termica As Boolean)

Public Sub LeggiDatiPidComandiCisterne(indice As Integer)

Public Sub LeggiDatiRegolazioneTempCisterne()

Public Sub ScriviDatiComandiAuxCisterneOnOff(ByVal indice As Integer, ByVal avvio As Boolean)

Public Sub ScriviDatiPidComandiCisterne(indice As Integer)

Public Sub ScriviDatiRegolazioneTempCisterne()

GestioneCisterneTipo5.bas

Private Sub SingolaCisternaInviaParametri(ByRef cisterna As OggettoCisterna, cisternaOffset As Integer)

Private Sub SingolaCisternaRidInviaParametri(ByRef cisterna As OggettoCisterna, cisternaOffset As Integer)

Public Sub AggiornaGraficaStatoCisterna(cisterna As Integer)

Public Sub AggiornaGraficaStatoCisternaCombust(cisterna As Integer)

Public Sub AggiornamentoGraficaOperazioniCisterne()

Public Sub CaricaParametriCisterne()

Public Sub CisterneInviaParametri()

Public Sub ControllaCisterneAllarmi(ByRef IdDescrizione As Integer, ByRef CodiceAllarme As String)

Public Sub ControllaCisterneAllarmiRidotto(ByRef IdDescrizione As Integer, ByRef CodiceAllarme As String)

Public Sub GestioneComboCisterne(indice As Integer)

Public Sub GestioneMDownComandiCisterne(indice As Integer)

Public Sub GestioneMUPComandiCisterne(indice As Integer)

Public Sub GraficaValvolaStandard\_Change(valvola As Integer, ByRef immagine As Object, TipoValvola As TipoValvolaEnum)

Public Sub LeggiDatiPLCCisterneBitume()

Public Sub LeggiDatiPLCCisterneBitumeRid()

Public Sub LeggiDatiPLCCisterneCombustibile()

Public Sub LeggiDatiPLCCisterneEmulsione()

Public Sub PompaCircuitoLegante\_Change(ritorno As Boolean, CodiceAllarme As Integer, ByRef immagine As Object)

Public Sub ScriviDatiPLCCisterne()

Public Sub ScriviDatiPLCCisterneRid()

### Oggetti utilizzati nella gestione cisterne:

GestioneCisterneCommon.bas

Public Type MotoreS7

Public Type PIDCisterne

Public Type GestioneCisterne

Public Type ContalitriType

Public Type OggettoValvolaPLC

GestioneCisterneTipo5.bas

Public Type OggettoCisterna

Public Type OggettoDBScambioDatiCisterneVecchiaStruttura

Public Type OggettoDBScambioDatiCisterne

GestioneCisterneCommon.bas

Public Enum StatoValvola

Public Enum TipiGestioneCiterneEnum

Public Enum ListaOperazioniCircuito

Public Enum PlcTagCisterneEnum

Public Enum PlcTagCisterneRidottoEnum

Public Enum ValvoleBitumeEnum

Public Enum ValvoleEmulsioneEnum

Public Enum ValvoleCombustibileEnum

GestioneCisterneTipo5.bas

Public Enum TipoValvolaEnum

## 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

Dosaggio

Tamburo

Predosaggio

### 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 , chiamata 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 2000 ms 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  
  Esegue 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 modalità 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().

## Gestione Predosaggio

### Premessa

* Molta logica presente nell’applicativo
* CYB500 prevede funzioni di controllo, visualizzazione e impostazione del lavoro dei predosatori:
  + switch gestione automatica/manuale
  + calcolo di alcuni parametri (umidità, …)
  + controlli sulla catena di funzionamento degli apparati (start e stop dei predosatori e dei nastri)
  + 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

Il predosaggio si occupa di calibrare i materiali che alimentano l’essiccatore:

* Inerti: materiale vergine
* Riciclato: materiale di recupero

Il dosaggio viene eseguito attraverso l’impiego di tramogge associate ad un nastro estrattore, eventualmente dotato di bilancia. Il tutto viene convogliato in un nastro collettore, poi in un nastro lanciatore e infine nell’essiccatore.

n

n

Tamburo

Nastro Lanciatore

Nastro Collettore

Nastro estrattore

Tramoggia

Tramoggia

In ogni predosatore viene immagazzinato un materiale con certe caratteristiche (tipo, diametro, umidità, …)

### Parametri e inizializzazione

Sul caricamento della Form e sulla applicazione di eventuali cambiamenti vengono utilizzati i parametri presenti nel file xml (Parameters.xml).

I dati di umidità sono ricavati dalla tabella di database Materiali.

### Logica

##### Gestione ricette da produrre

L’alimentazione dell’essiccatore con i differenti materiali è guidata da una ricetta, che viene impostata dall’operatore, selezionandola tra quelle presenti nella tabella Predosaggio del database. La ricetta di predosaggio può essere impostata mediante il job in esecuzione.

Predosaggio

Calcolo Uscita Predosatori

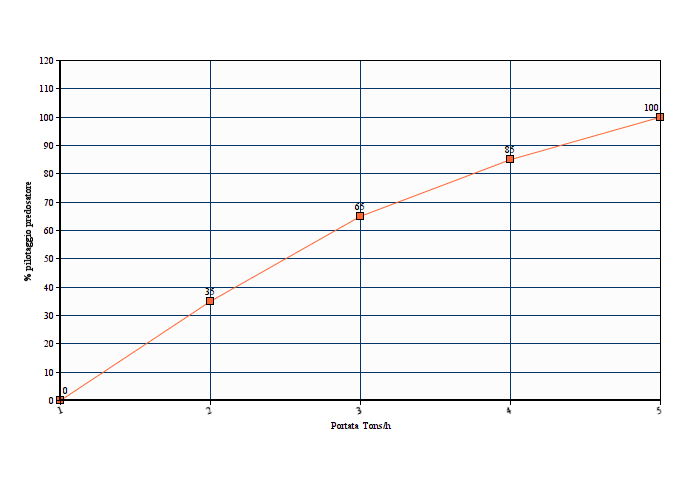
Scelta Ricetta

PLC

### Calcolo Uscita Predosatori

La proporzione con la quale i predosatori devono rilasciare una certa quantità di materiale è espressa in percentuali.

Il materiale in uscita dal predosatore non ha un andamento proporzionale con il settaggio impostato, ma segue un grafico, approssimato nel programma con una interpolazione lineare a 5 punti. In base alla portata richiesta dalla ricetta si calcola a che percentuale di pilotaggio far funzionare il predosatore.



La regolazione del motore del nastro estrattore viene attuata mediante le bilance presenti sullo stesso e un controllo PID sull’errore di portata.

Quindi, riassumendo:

% di materiale da ricetta

Calcolo portata teorica (Grafico)

Set Nastro Estrattore

PLC

### Avviamento Predosatori

In vari punti del codice viene valutata la possibilità di far funzionare i predosatori in automatico o manuale.

L’avvio in automatico dei predosatori è soggetta ad alcune condizioni. L’avvio **NON** può avvenire:  
- Se non è selezionata una ricetta  
- Se i sili del filler sono troppo pieni  
- Se viene utilizzato del riciclato caldo ma il nastro elevatore freddo e il nastro lanciatore non sono accesi  
- Se viene utilizzato il riciclato caldo ma l’essiccatore non è in funzione  
- Se non è accesa la fiamma del bruciatore (a meno di forzatura) e i predosatori sono impostati per il prossimo set  
- Se viene utilizzato il riciclato caldo ed è escluso l’allarme di temperatura bassa del filtro  
- Se viene utilizzato il riciclato caldo e la temperatura dei filtri è minore di quella di lavoro  
- Se c’è il blocco per alta temperatura dei predosatori  
- Se il nastro collettore è fermo (gestione semiautomatica)

Qualsiasi condizione di allarme provoca lo stop dei predosatori e il passaggio in modalità manuale.

### Timer

Molti task inerenti la gestione dei predosatori sono inseriti in uno specifico timer del VB6:

* Stato e impostazioni di ogni predosatori
* Accensione del bruciatore: se i predosatori sono tutti spenti allora parte il timer di spegnimento del bruciatore
* Ricalcolo del riciclato freddo totale caricato nella ricetta
* Lancio della regolazione ponderale del riciclato
* Richiamo della gestione della fillerizzazione (chiamata qui per non fare un altro timer)
* Aggiornamento del log nella tabella StoricoPredosaggio. Oltre ai dati del predosaggio vengono inseriti anche quelli della torre/dosaggio/tramogge nella tabella StoricoPredosaggioTorre.

### Retroazione Dosatori-Predosatori

Quando i cicli di dosaggio da eseguire meno i cicli di dosaggio eseguiti raggiungono un valore impostato nei parametri o dall’operatore (CicliStopPred), vengono bloccati i predosatori. In pratica, se in dosaggio non c’è necessità di altro materiale per finire i cicli di impasto, non c’è bisogno che i predosatori alimentino la torre di dosaggio.

### Nastro del riciclato

Il deflettore del riciclato può essere impostato su 2 valori:

* Anello: il riciclato è caldo e finisce nel tamburo
* Elevatore: il riciclato è freddo e va direttamente in torre

Se abilitato il deflettore a Elevatore/Anello (AbilitaDeflAnElev su parameters.xml)

Nel caso di riciclato caldo, si calcola una retroazione PID (proporzionale-integrativo-derivativo) sul nastro. Prima calcolo il peso teorico del riciclato:

La bilancia sul nastro restituisce il peso reale. A quel punto viene applicato il controllo PID sul delta tra peso reale e peso teorico e viene trovato il valore da applicare al nastro come retroazione.

### Dettaglio Funzioni

#### ParaTabPred.bas

Funzioni per lettura e caricamento parametri

#### GestionePredNastriC.bas

Principali funzioni:

* DatiSetPredosaggi()  
  Calcola l’umidità totale dei predosatori, sul comando di start dei predosatori e sul cambio stato degli stessi
* AvviamentoGestionePredosatori()  
  Gestisce l’avviamento dei predosatori. Se i predosatori sono in automatico e sono vergini e sono accesi => se l’essiccatore non è acceso e i tamburi non sono caldi => manda allarme e passa a gestione manuale dei predosatori lanciando PassaInManualePredosatori().
* ControlloPredAutomatico(nastro As Byte) As Boolean  
  Controlla che i predosatori non partano se il nastro collettore è fermo e la gestione è semiautomatica
* PredosatoriVerginiAccesi() As Boolean  
  Verifica se almeno un predosatore di materiale vergine è acceso.
* PredosatoriRiciclatiAccesi() As Boolean  
  Verifica se almeno un predosatore del riciclato è acceso.
* GestioneStopPredosatori()  
  Gestisce lo stop dei predosatori. Viene lanciata quando i cicli di dosaggio da eseguire meno i cicli di dosaggio eseguiti raggiungono un valore impostato nei parametri o dall’operatore (CicliStopPred).
* ControllaRicettaPredVuota()  
  Controlla se la ricetta è vuota. Se è vuota, non trasferisce i set (percentuali di pilotaggio) CP240.OPCData.items(PLCTAG\_NM\_PRED\_RICFUT\_Inerte\_Set1).Value = 0
* AvvioPredAutomatico()  
  Fa partire i predosatori in automatico.
* PredosatoreSetCalcolato(ByRef Pred As PredosatoreType, portataTeorica As Double) As Integer  
  Calcola il valore da attribuire al Set dei predosatori n base alle ton/h (portata rilevata in cinque punti) tenendo presente l’andamento del grafico.   
  Si usa una interpolazione lineare in 5 punti. Grafico con portata sulle x e % di potenza al dosatore sulle Y. Data la portata teorica si imposta la % di lavoro, calcolandola con la linea tra due punti.
* ValoreUscitaAnalogicaPred(riciclato As Boolean, ValoreSet As Integer, ValoreSetReale As Integer) As Double  
  Usata da PredosatoreUscitaAnalogica per calcolare il valore della uscita analogica dei predosatori.  
  Premessa: ogni uscita o ingresso analogico del plc va da 0 a 27648, sia che si tratti di 0-20 mA, 4-20 mA o 0-10V. Il valore massimo di scala (quello che consideriamo 1 in una scala da 0 a 1) si trova in Parameter.xml. Ad esempio il valore del voltaggio dei predosatori è dato dalla variabile Vpred.  
  <Vpred Type="2" TagOPC="TagDaSistemare" Default="8" MinValue="-10" MaxValue="+10" DescrID="504" TtipID="0" MeasUnitID="994" ReadAccess="1" WriteAccess="3" Printable="1" EditMode="4" />  
  Quindi nel nostro caso dei predosatori sappiamo che al massimo si accetta un voltaggio di 8V pari nel plc a 22118 (unità del predosatore).  
  Se ora moltiplico il valore di uscita analogico calcolato (che è una percentuale di potenza voglio andare) con questo valore ottengo il dato da spedire al PLC.
* PredosatoreRegolazionePonderale(ByRef Pred As PredosatoreType)  
  Controllo PID sul motore del singolo nastro estrattore
* PulsanteStopPred()  
  Mette in manuale i predosatori.
* RichiamoRicettaPredos()  
  Setta sui predosatori i dati della ricetta. Lavora sulla tabella di database Predosaggio, PredosaggioLog. Sono tabelle dove il lavoro dei predosatori è espresso con percentuali.
* AggiornaProssimaRicettaPredos()  
  Aggiorna a video la prossima ricetta.
* PassaInManualePredosatori()  
  Fa passare la gestione dei predosatori in Manuale. Il tag PLC interessato è: PLCTAG\_NM\_PRED\_Auto\_Man. Se non è selezionata l’attesa di fine ricetta allora blocco i predosatori (PredosatoriArrestoImmediato) altrimenti la finisco.
* PredosatoriRicConBilancia() As Boolean  
  Dice se i predosatori del riciclato hanno bilancia
* *VisualizzaPortateNastri()*Visualizza o meno i controlli (in CP240) dedicati alla visualizzazione e memorizzazione della portata (effettiva o teorica) che transita sui nastri elevatori
* StartPredosatori\_change()  
  Aggiorna a video
* LeggeFileUmiditaPredosatoreRic(),LeggeFileUmiditaPredosatore ()  
  Leggono da un file specifico i dati dell’umidità dei predosatori. Sono i file Umidita-Predosatore.ini, Umidita-PredosatoreRic.ini.
* RinfrescaOrigineDatiPredosaggio(NomeRicPredSel As String, Optional AggiornaCP240 As Boolean), RinfrescaNomeRicPreDosaggio()  
  Aggiornamenti a video
* NastroRicRegolazionePonderale()  
  Abilita la regolazione ponderale del riciclato. Se ho le bilance su ogni riciclato posso calcolare il peso che teoricamente dovrebbe avere il nastro riciclato e col PID gestire la quantità da far uscire istantaneamente. Se ho un solo nastro controllo quello e basta.
* PesoNastroInerti(),PesoNastroRiciclato(),PesoNastroRiciclatoParDrum()  
  Chiamata su PlcInAnalogici, calcola il peso dei nastri di inerti, riciclato, riciclato al tamburo
* GestioneConsumi()  
  Invocata direttamente da VideataPrincipale; al max una chiamata al secondo. Aggiorna totalizzazione peso dei nastri elevatori; se la bilancia non è inclusa utilizza un valore teorico
* ResetTotalizzatoriNastri(nastro As Integer)  
  Manda al PLC il comando di reset del totalizzatore di un nastro e aggiorna a video il dato dei consumi.

#### GestionePredosatori.bas

* VisualizzaPredosatoriImpostati(), VisualizzaPredosatoriRicImpostati(), DisposizionePredNC1, DisposizionePredNC2(), DisposizionePredNC3(), DisposizionePredNRicJolly(), DisposizionePredNCRic2()  
  Funzioni di visualizzazione sul sinottico di vari componenti opzionali (per esempio i nastri collettori possono essere 1, 2 o 3
* PredosatoreInizializza()  
  Inizializza i predosatori allo start dell’applicazione (legge anche i file di umidità)
* PredosatoreUscita\_change(ByVal predosatore As Integer), PredosatoreRicUscita\_change(ByVal predosatore As Integer)  
  Gestisce l’evento di cambio dell’uscita di pilotaggio di un predosatore.
* SetPredosatoreUscita(ByVal predosatore As Integer, valore As Boolean), SetPredosatoreRicUscita(ByVal predosatore As Integer, valore As Boolean)  
  Esegue il cambio dell’uscita di pilotaggio di un predosatore.
* PredosatoreRitorno\_change(ByVal predosatore As Integer), PredosatoreRiciclatoRitorno\_change(ByVal predosatore As Integer)  
  Gestisce l’evento di cambio del ritorno di un predosatore.
* PredosatoreMinimoVuoto\_change(ByVal predosatore As Integer), PredosatoreRiciclatoMinimoVuoto\_change(ByVal predosatore As Integer)  
  Gestisce l’evento di minimo di un predosatore.
* PredosatoriCalcoloUmiditaTotaleTmp(Umidita() As Double, tutti As Boolean) As Double, PredosatoriCalcoloUmiditaTotale(tutti As Boolean) As Double, PredosatoriRiciclatoCalcoloUmiditaTotaleTmp(Umidita() As Double, Index As Integer) As Double, PredosatoriRiciclatoCalcoloUmiditaTotale(Index As Integer) As Double  
  Viene calcolata l’umidità totale in base alla portata istantanea dei predosatori
* RitardoVibratorePredVuoto(),RitardoSoffioPredVuoto()  
  Ritardi di vibrazione e soffio su predosatore vuoto.
* TermicaPredosatori\_change()  
  Se arriva il segnale di allarme termico dei predosatori, li blocco tutti. Fino a che la termica non si abbassa disabilito a video il pulsante di start dei predosatori.
* PredosatoreRitardato(ByRef Pred As PredosatoreType) As Boolean  
  Verifica se un predosatore ha una gestione con ritardo o immediata.
* PredosatoreOttieniSet(riciclato As Boolean, ByVal predosatore As Integer) As Integer  
  Ottiene il pilotaggio attuale di un predosatore
* PredosatoreCambiaSet(riciclato As Boolean,ByVal predosatore As Integer, valore As Integer,cambioManuale As Boolean), PredosatoreInserisciSet(ByRef Pred As PredosatoreType, valore As Integer, cambioManuale As Boolean)  
  Setta il pilotaggio di un predosatore. Il nuovo set viene messo in coda
* PIDcontroller  
  controllo PID
* PredosatoreInVolumetrico(riciclato As Boolean, predosatore As Integer, attivo As Boolean), PredosatoreVolumetricoPonderale(ByRef Pred As PredosatoreType, attivo As Boolean), PredosatoreRegolazionePonderale(ByRef Pred As PredosatoreType)  
  Funzioni per la regolazione ponderale volumetrica dei predosatori. Viene usato il valore della bilancia per pilotare l’uscita del predosatore dopo un calcolo PID dell’errore tra il teorico e il peso effettivo.
* PredosatoreVerificaSet(ByRef Pred As PredosatoreType, forza As Boolean)  
  Funzione di verifica dei valori del set di un predosatore. Aggiorna i dati di portata del prodosatore in base alle pesate della bilancia Siwarex. In base ai parametri impostati e alle condizioni di funzionamento attuali cambia lo stato del predosatore e ne imposta il prossimo set.
* PredosatoriSet\_timer()  
  Timer per le routine dei predosatori. A intervalli regolari (250 ms) controlla:   
  - stato e impostazioni di ogni predosatore  
  - accensione del bruciatore. Se i predosatori sono tutti spenti allora parte il timer di spegnimento del bruciatore  
  - ricalcola il riciclato freddo  
  - lancia la regolazione ponderale del riciclato  
  - richiama la gestione della fillerizzazione  
  - aggiorna o crea un record di log nella tabella StoricoPredosaggio
* GestioneStoricoPredosaggio(), GestioneStoricoPredosaggioAggiungi(), GestioneStoricoPredosaggioAggiungiPred(rs), GestioneStoricoPredosaggioAggiungiTorre(rs)  
  Funzioni di registrazione su database delle misurazioni istantanee di curva e portata dei predosatori. Non so perché poi si sia aggiunta la torre e non sia stata messa in un timer a parte.
* PredosatoriInManuale(), PredosatoreManuale(riciclato As Boolean, predosatore As Integer, start As Boolean, forza As Boolean)  
  Funzioni per mettere i predosatori in gestione manuale
* PredosatoriInStartAutomatico(start As Boolean)  
  Setta i predosatori in automatico ed in start (o stop)
* PredosatoriRicalcolaSet()  
  Ricalcola il set di uscita dei predosatori
* PredosatoriAccesi(riciclato As Boolean, nastro As Integer) As Boolean  
  Verifica che almeno un predosatore sia acceso
* PredosatoriArrestoImmediato(riciclato As Boolean, nastro As Integer)  
  La routine è invocata sul passaggio in manuale, su allarmi dei predosatori e su stop del bruciatore/tamburo. Arresta immediatamente una linea di predosaggio
* SetRiduzioneProduzione(nuovaRiduzione As Integer)  
  Percentuale di riduzione attuale della produzione predosatori. Non accetta valori minori del 30% o superiori al 100%.
* PredosatoreOttieniMaterialeLogId(ByRef Pred As PredosatoreType) As Long, PredosatoreOttieniNome(ByRef Pred As PredosatoreType) As String, PredosatoreOttieniPortata(ByRef Pred As PredosatoreType) As Integer  
  Ricava id materiale, nome e portata di un predosatore
* ControlloCondizioniStartPreDosaggio() As Boolean, StartPreDosaggio()  
  Condizioni per lo start del predosaggio. Non può partire :  
  - Se non è selezionata una ricetta  
  - Se i sili del filler sono troppo pieni  
  - Se viene usato del riciclato caldo ma il nastro elevatore freddo e il nastro lanciatore non sono accesi  
  - Se viene usato il riciclato caldo ma l’essiccatore non ruota  
  - Se non è accesa la fiamma del bruciatore (a meno di forzatura) e ci sono dei predosatori impostati per il prossimo set  
  - Se viene usato il riciclato caldo ed è escluso l’allarme di temperatura bassa del filtro  
  - Se viene usato il ric. caldo e la temperatura dei filtri è minore di quella di lavoro  
  - Se c’è il blocco per alta temperatura dei predosatori  
  - Se il nastro collettore è fermo (gestione semiautomatica)
* PredosatoriArrestoImmediato(riciclato As Boolean, nastro As Integer)  
  La funzione è invocata sul passaggio in manuale, su allarmi dei predosatori e su stop del bruciatore/tamburo. Arresta immediatamente una linea di predosaggio.

#### ParaTabPred.bas

* CaricaDatiGrafPredosatore(ByRef Pred As PredosatoreType)  
  carica dai parametri i valori delle curve di portata/set dei predosatori
* PlcGriglieVibranti()  
  Passa dei parametri al PLC relativi alla presenza e ai ritardi di start e stop dei vibratori dei predosatori del riciclato
* LeggiUmiditaPredSQL()  
  legge da database (tabella MaterialiLog) l’umidità, associandola al predosatore che ha in uso un certo materiale.

### Moduli interessati alla gestione predosaggio

GestionePredNastriC.bas

GestionePredosatori.bas

ParaTabPred.bas

### Metodi utilizzati nella gestione predosaggio:

GestionePredNastriC.bas

243 - Public Sub NastroRicRegolazionePonderale()

135 - Public Sub GestioneConsumi()

107 - Public Sub PesoNastroRiciclato()

99 - Public Sub PesoNastroInerti()

82 - Public Sub RichiamoRicettaPredos()

71 - Public Sub PesoNastroRiciclatoParDrum()

59 - Public Sub AggiornaProssimaRicettaPredos()

58 - Public Sub ResetTotalizzatoriNastri(nastro As Integer)

45 - Public Sub PassaInManualePredosatori()

37 - Public Function PredosatoreSetCalcolato(ByRef Pred As PredosatoreType, portataTeorica As Double) As Integer

37 - Public Function ControlloPredAutomatico(nastro As Byte) As Boolean

35 - Public Sub RinfrescaOrigineDatiPredosaggio(NomeRicPredSel As String, Optional AggiornaCP240 As Boolean)

35 - Public Sub RinfrescaNomeRicPreDosaggio()

32 - Public Function ControllaRicettaPredVuota()

27 - Public Sub StartPredosatori\_change()

27 - Public Sub PulsanteStopPred()

27 - Public Function PrimoPredosatoreDelNastro(nastro As NastriPredosatori) As Integer

26 - Public Sub LeggeFileUmiditaPredosatoreRic()

25 - Public Sub PreparazioneAvvPred()

24 - Public Function ValoreUscitaAnalogicaPred(riciclato As Boolean, ValoreSet As Integer, ValoreSetReale As Integer) As Double

23 - Public Sub PesoNastroRiciclato\_change()

22 - Public Sub PesoNastroInerti\_change()

22 - Public Sub LeggeFileUmiditaPredosatore()

15 - Public Sub VisualizzaPortateNastri()

14 - Public Sub AvviamentoGestionePredosatori()

13 - Public Sub PesoNastroRiciclatoParDrum\_change()

13 - Public Sub GestioneStopPredosatori()

13 - Public Function PredosatoriRicConBilancia() As Boolean

9 - Public Sub DatiSetPredosaggi()

9 - Public Function PredosatoriRiciclatiAccesi() As Boolean

8 - Public Function PredosatoriVerginiAccesi() As Boolean

7 - Public Function SelezioneRicettaPredosaggioCambiata() As Boolean

6 - Public Sub AvvioPredAutomatico()

GestionePredosatori.bas

200 - Public Sub PredosatoriSet\_timer()

166 - Public Function ControlloCondizioniStartPreDosaggio() As Boolean

143 - Public Sub PredosatoriArrestoImmediato(riciclato As Boolean, nastro As Integer)

143 - Public Sub PredosatoreVerificaSet(ByRef Pred As PredosatoreType, forza As Boolean)

131 - Public Sub GestioneStoricoPredosaggioAggiungi()

114 - Public Sub PredosatoreMinimoVuoto\_change(ByVal predosatore As Integer)

111 - Public Sub StartPreDosaggio()

90 - Private Sub PredosatoreInserisciSet(ByRef Pred As PredosatoreType, valore As Integer, cambioManuale As Boolean)

88 - Public Sub PredosatoreRiciclatoMinimoVuoto\_change(ByVal predosatore As Integer)

88 - Public Function PredosatoriAccesi(riciclato As Boolean, nastro As Integer) As Boolean

83 - Public Function PIDcontroller( \_

68 - Private Sub PredosatoreCambiaStato(ByRef Pred As PredosatoreType, nuovoStato As StatoPredosatoreType)

66 - Private Sub PredosatoreProssimoSet(ByRef Pred As PredosatoreType, forzaTempi As Boolean)

61 - Private Sub GestioneStoricoPredosaggioAggiungiPred(ByRef rs As adodb.Recordset)

59 - Public Sub PredosatoreRicUscita\_change(ByVal predosatore As Integer)

50 - Public Function PredosatoriRiciclatoCalcoloUmiditaTotaleTmp(Umidita() As Double, Index As Integer) As Double

48 - Public Sub VisualizzaPredosatoriRicImpostati()

48 - Public Function ChkCoherenceMaterial(IdPredosaggio)

47 - Public Sub VisualizzaPredosatoriImpostati()

46 - Private Sub PredosatoreRegolazionePonderale(ByRef Pred As PredosatoreType)

43 - Private Sub DisposizionePredNC1(i As Integer)

40 - Public Sub SelectFeederRecipeByCS(IdRecipe As Long)

40 - Public Function CalcolaSetNastri(Index As Integer) As Double

37 - Public Sub RitardoVibratorePredVuoto()

36 - Private Sub DisposizionePredNC2(i As Integer)

32 - Public Sub SetPredosatoreUscita(ByVal predosatore As Integer, valore As Boolean)

32 - Private Sub PredosatoriVerificaAllarme()

31 - Public Sub PredosatoreUscita\_change(ByVal predosatore As Integer)

27 - Public Sub PredosatoreCambiaSet(riciclato As Boolean, ByVal predosatore As Integer, valore As Integer, cambioManuale As Boolean)

26 - Public Sub SetRiduzioneProduzione(nuovaRiduzione As Integer)

25 - Public Sub PredosatoriRicalcolaSet()

25 - Private Sub PredosatoreVolumetricoPonderale(ByRef Pred As PredosatoreType, attivo As Boolean)

24 - Public Sub PredosatoriInStartAutomatico(start As Boolean)

24 - Public Sub PredosatoreManuale(riciclato As Boolean, predosatore As Integer, start As Boolean, forza As Boolean)

24 - Public Function PredosatoriCalcoloUmiditaTotaleTmp(Umidita() As Double, tutti As Boolean) As Double

22 - Private Sub PredosatoreUscitaAnalogica(ByRef Pred As PredosatoreType, ValoreSet As Integer)

22 - Private Sub DisposizionePredNC3(i As Integer)

20 - Public Sub RitardoSoffioPredVuoto()

20 - Public Function PredosatoreOttieniSet(riciclato As Boolean, ByVal predosatore As Integer) As Integer

20 - Private Sub VisualizzaMinMaxPonderale(ByRef Pred As PredosatoreType)

20 - Private Sub AvvisoPredosatoreVuoto(ByRef Pred As PredosatoreType)

18 - Public Function PredosatoreOttieniMaterialeLogId(ByRef Pred As PredosatoreType) As Long

17 - Private Sub DisposizionePredNCRic2(i As Integer)

16 - Public Sub PredosatoriInManuale()

16 - Public Sub PredosatoreInizializza()

16 - Public Sub GestioneStoricoPredosaggio()

15 - Public Sub CalcolaNumeroPredosatoreSiwarex()

14 - Public Sub TermicaPredosatori\_change()

14 - Private Sub PredosatoreAutomatico(Pred As PredosatoreType, start As Boolean)

13 - Public Sub PredosatoreRitorno\_change(ByVal predosatore As Integer)

13 - Public Sub PredosatoreRiciclatoRitorno\_change(ByVal predosatore As Integer)

13 - Private Sub PredosatoreRiciclatoAcceso(ByVal predosatore As Integer, ByVal acceso As Boolean)

12 - Public Sub ChkResetColorMateriali()

12 - Public Function PredosatoriRiciclatoCalcoloUmiditaTotale(Index As Integer) As Double

12 - Private Sub GestioneStoricoPredosaggioAggiungiTorre(ByRef rs As adodb.Recordset)

11 - Public Function PredosatoriCalcoloUmiditaTotale(tutti As Boolean) As Double

10 - Private Sub PredosatoreAcceso(ByVal predosatore As Integer, ByVal acceso As Boolean)

8 - Public Sub PredosatoreInVolumetrico(riciclato As Boolean, predosatore As Integer, attivo As Boolean)

8 - Public Function PredosatoreOttieniPortata(ByRef Pred As PredosatoreType) As Integer

8 - Private Function PredosatoreRitardato(ByRef Pred As PredosatoreType) As Boolean

7 - Public Sub SetPredosatoreRicUscita(ByVal predosatore As Integer, valore As Boolean)

6 - Public Sub VisualizzaRiduzioneProduzione()

6 - Public Function PredosatoreOttieniNome(ByRef Pred As PredosatoreType) As String

5 - Private Sub DisposizionePredNRicJolly(i As Integer, position As Integer)

5 - Private Sub DisposizionePredNCRic1(i As Integer)

ParaTabPred.bas

108 - Public Function ParaTabPred\_ReadFile() As Boolean

83 - Public Sub LeggiUmiditaPredSQL()

82 - Public Sub ParaTabPred\_Apply()

72 - Public Sub CaricaDatiGrafPredosatore(ByRef Pred As PredosatoreType)

42 - Private Sub LeggiConfigurazionePredosatore(Pred As PredosatoreType)

31 - Private Sub ParaTabPred\_ToDB()

26 - Public Sub PlcGriglieVibranti()

### Oggetti utilizzati nella gestione predosaggio:

GestionePredNastriC.bas

Public Type TolleranzaNastro

GestionePredosatori.bas

Public Type SetPredosatoreType

Public Type PidType

Public Type CurvaPredosatoreType

Public Type GraficoPredosatoreType

Public Type MotorePredosatoreType

Public Type PredosatoreType

Public Type FeederVibratorWorkingCycle\_TYPE

GestionePredNastriC.bas

Public Enum NastriPredosatori

GestionePredosatori.bas

Public Enum CambioSetPredosatoreType

Public Enum StatoPredosatoreType

Public Enum BilanciaNastro

## Gestione Dosaggio

### Descrizione

Le gestione dosaggio si occupa di leggere i dati della ricetta di dosaggio, verificarne la correttezza, inviarli al PLC, gestire condizioni e vincoli per la esecuzione del ciclo di dosaggio e monitorare l’andamento del ciclo medesimo.

### Logica

La logica della gestione dosaggio è essenzialmente di controllo sulle condizioni e i vincoli di esecuzione del ciclo di dosaggio. I dati della ricetta vengono prelevati dal database e inviati al PLC opportunamente formattati. Il dosaggio viene gestito nella funzione CicloJob() richiamata da VideataPrincipale(), le altre funzioni interessate sono la CambioDosJob(), la StartJob() e la StartDosaggio(). Si tratta di un pattern di tipo distinta , lista di tasks da eseguire in sequenza.

### Dettaglio Funzioni

#### GestioneDosaggio.bas

##### InvioFormulaDosaggio()

Invia al PLC i dati relativi alla ricetta di dosaggio da eseguire, i dati sono reperiti nel database , tabella Dosaggio.

##### SommaComponenti

Somma componenti dosaggio e scrittura record di storico dosaggio.

##### GestioneStatoDosaggio

Chiamata in network.bas - opcData\_OnDataChanged() , gestisce il dosaggio, fra le altre cose chiama la funzione LeggiNettiResiduiCP240()

##### LeggiNettiResiduiCP240

Aggiorna i dati relativi ai pesi dei componenti del dosaggio in corso che vengono letti dal PLC.

##### StartDosaggio

Esegue le operazioni di inizio dosaggio e chiama la funzione ControlloCondizioniStartDosaggio().

##### ControlloCondizioniStartDosaggio

Controlla le condizioni per poter iniziare il dosaggio.

##### SelezionaRicettaDosaggio

Seleziona il recordset nella tabella Dosaggio se ok chiama la GestioneCambioRicettaDosaggio()

##### GestioneCambioRicettaDosaggio

Verifica che sia possibile il cambio ricetta dosaggio e chiama la RichiamoRicettaDos()

##### RichiamoRicettaDos

Esegue alcune verifiche e chiama la InvioFormulaDosaggio()

##### StartDosaggio

Chiamata da StartJob() – GestioneJob.bas, inizia il ciclo di dosaggio.

##### TrasformaRicettaS7

Invia la ricetta di dosaggio al PLC rispettando i seguenti vincoli:

Il PLC attualmente vuole che la ricetta sia composta così:

Aggregato 1 : set in percentuale + ordine pesata

Aggregato 2 : set in percentuale + ordine pesata

...

Aggregato 6 : set in percentuale + ordine pesata

ordine pesata deve essere sempre diverso da zero se il set% è maggiore di zero

ordine pesata deve avere valori da 1 a 6

Se ho portine con apertura accoppiata

il primo aggregato della serie deve contenere in % il totale

gli aggregati successivi devono avere il set% a zero e l'ordine di pesata uguale all'aggregato che contiene la somma:

ESEMPIO

Aggregato1 set% = 60% ordine pesata = 1

Aggregato2 set% = 0% ordine pesata = 1

Aggregato3 set% = 0% ordine pesata = 1

Aggregato4 set% = 20% ordine pesata = 2

Aggregato5 set% = 20% ordine pesata = 3

In questo modo si apriranno contemporaneamente le portine aggregati 1,2,3 e la bilancia raggiungerà il 60% del peso totale dell'impasto, poi verranno effettuate le due pesate singole da 20% ciascuna dell'aggregato 4 e poi del 5.

OrdinePesata contiene la sequenza di pesata (1= prima pesata, 2=seconda pesata, ecc..)

1. Scorro i set da Aggregato1 a Aggregato6

2. Se l'aggregato ha set <> 0 guardo il suo OrdinePesata

3. Se l'ordine pesata = posizione Aggregato (Aggregato1 + Ordine=1) verifico se ci sono portine accoppiate

4. Se l'ordine pesata <> posizione Aggregato (Aggregato3 + Ordine=1)

### Moduli interessati alla gestione dosaggio

GestioneDosaggio.bas

### Metodi utilizzati nella gestione dosaggio:

GestioneDosaggio.bas

874 - Public Sub InvioFormulaDosaggio()

625 - Public Sub SommaComponenti()

398 - Public Sub LeggiNettiResiduiCP240()

250 - Public Function ControlloCondizioniStartDosaggio() As Boolean

171 - Public Sub RefreshDatiFormNetti()

138 - Public Sub InizializzaComponenti()

134 - Public Sub GestioneStatoDosaggio()

109 - Public Sub SelezionaRicettaDosaggio()

100 - Public Sub TrasformaRicettaS7()

97 - Public Sub ControlloLivelliAltiTramogge()

92 - Public Sub AggiornaSetKgCP240()

89 - Public Sub RichiamoRicettaDos()

87 - Public Sub ComponenteInit( \_

85 - Public Sub TimeOutDosaggio()

80 - Public Sub StartDosaggio()

80 - Public Sub CfgPortineScaricoAgg()

78 - Public Sub RiempieBufferPortine(rs As adodb.Recordset, ByRef arrayPortine() As RicettaTramogge)

71 - Public Sub ForzaSetTempi(Optional soloagg\_ric As Boolean)

64 - Private Sub CalcolaVelocitaInvertPesateFiller(KgF1 As Double, KgF2 As Double, KgF3 As Double)

63 - Public Function VerificaSetSicurezzeBilance() As Boolean

61 - Public Sub AggiornaVelocitaInvertPesateFiller(KgF1 As Double, KgF2 As Double, KgF3 As Double)

58 - Public Sub ArrestoEmergenzaDosaggio()

58 - Private Sub RicettaParametriBitumeGravita()

55 - Public Sub PosizionaBitume123()

54 - Public Sub AggiornaOrdinePesateForzato()

53 - Public Sub CalcoloProgressImpasto()

52 - Public Sub ComponenteLivello(ByRef componente As ComponenteType, Value As Integer)

51 - Public Sub SetDosaggioScarico()

51 - Public Sub RiempiBufferKgVoli()

50 - Public Sub ComponenteInPesata(ByRef componente As ComponenteType, Value As Boolean)

48 - Public Function CalcoloTotaleImpasto() As Double

47 - Public Sub RiempiBufferViatopScarMixer()

43 - Public Sub ArrestoPCLInDosaggio()

42 - Public Sub RinfrescaNomeRicDosaggio()

42 - Public Sub PosizionaAdditivi()

42 - Public Sub GestioneDosaggioBitumeEsterno()

40 - Public Sub ForzaSetAggregati(Index As Integer)

39 - Public Sub RiempiBufferAggregatiFiller()

38 - Public Sub CalcolaVelocitaInvertPesateFillerManuale()

38 - Public Function VerificaRiduzioneImpastoCasoSacchi(ByRef riduzione As Integer, inIncremento As Boolean) As Boolean

37 - Public Sub ForzaSetFiller(Index As Integer)

33 - Public Sub RinfrescaOrigineDatiDosaggio(NomeRicDosSel As String)

33 - Public Function RiduzioneImpastoCasoSacchi(ByRef min As Integer, ByRef passo As Integer) As Boolean

32 - Public Sub DosaggioAutoMan(auto As Boolean)

31 - Public Sub StatusCalibrazioneBilPN()

31 - Public Sub CalcolaTempoCiclo()

29 - Public Sub DatiResiduiNetti()

29 - Public Sub AbilitazioneCambioRicetta(abilita As Boolean)

26 - Public Sub RefillTargheCamion()

26 - Public Sub LeggiNettiParziali()

25 - Public Sub PosizionaDeflettoreVaglio()

22 - Public Sub GestioneDeflettoreByPassTamburoParallelo()

22 - Public Sub ComponentePesoOut(ByRef componente As ComponenteType, Value As Double)

21 - Public Sub SelectDosingRecipeByCS(IdRecipe As Long)

21 - Public Sub PosizionaDeflettoreByPassTamburoParallelo()

20 - Public Sub VerificaRitornoPesataFiller(componente As Integer, comando As Boolean, ritorno As Boolean)

18 - Public Sub SetRiduzioneImpasto(nuovaRiduzione As Integer)

18 - Public Sub SetCicliDosaggioDaEseguire(ByVal cicli As Long)

18 - Public Sub SelectClienteByCS(IdCliente As Long)

18 - Public Sub LivelloTeoricoOut(ByRef componente As ComponenteType, Value As Double)

18 - Public Sub GraficaTempoMescolazione()

17 - Public Sub PulsanteControlloPortineManuale(Optional automatico As Boolean)

17 - Public Sub AggiornaGraficaScaricoMescolatore()

16 - Public Sub SetSelezioneCircuitoBitume2(bitume2 As Boolean)

16 - Public Sub Grafica\_DeflTramScivScarBilRic()

15 - Public Sub SegnalaCambioRicettaAlPlc()

15 - Public Sub ScambiaPompaCircLegante()

15 - Public Sub RinfrescaOrigineDatiClientiCamion(NomeCliente As String)

15 - Public Sub GestioneBitumeGravita()

15 - Public Function GrandezzaImpastoPLC() As Integer

14 - Public Sub LivelloTeoricoIn(ByRef componente As ComponenteType, Value As Double)

14 - Public Sub ComponenteTemperatura(ByRef componente As ComponenteType, Value As Double)

14 - Public Sub ComponenteSetCalcolato(ByRef componente As ComponenteType, Value As Double)

13 - Public Sub RinfrescaOrigineDatiClienti(NomeCliente As String)

13 - Public Sub GestioneCambioRicettaDosaggio()

12 - Public Sub ComponenteSetDisplay(ByRef componente As ComponenteType, Optional decimalDigit As Integer)

12 - Public Sub ComponenteSet(ByRef componente As ComponenteType, Value As Double)

12 - Private Sub RefreshRicettaAquablack()

11 - Public Sub StatoAccoppiataDisplay(ByRef componente As ComponenteType, Stato As Boolean)

11 - Public Sub PulsanteStopCicliDosaggio()

11 - Public Function RicalcolaSetRicetta(SetOLD As Double, KgImpasto As Long, PercBitume As Double, KgNew As Long) As Double

10 - Public Sub LetturaPesoMescolatore()

10 - Public Function ImpastoPeso() As Long

8 - Public Sub SetDimensioneImpastoKg(nuovaDimensione As Long)

8 - Public Sub ScritturaPesoMescolatore()

5 - Public Sub AzzeraBufferViatopScarMixer()

5 - Public Function AdditivoFlomacInserito() As Boolean

2 - Public Sub AzzeramentoDatiTrasRic()

104 - Public Sub ScaricoAggregati\_change()

71 - Public Sub RAPSiwaInPesata\_change()

66 - Public Sub TempoMescolazione\_change()

61 - Public Sub MescolatoreChiuso\_change()

56 - Public Sub DosaggioInCorso\_change()

47 - Public Sub MescolazioneInCorso\_change()

47 - Public Sub MescolatoreAperto\_change()

46 - Public Sub ValorePortinaBitume\_change(portinaBitume As Integer)

40 - Public Sub Valv3VieBitume2Emulsione\_Change()

37 - Public Sub BilRAP\_Change()

37 - Public Sub BilFiller\_change()

33 - Public Sub ScaricoBitume\_change(gravita As Boolean)

33 - Public Sub BilBit\_change()

33 - Public Sub BilAgg\_change()

32 - Public Sub TemperaturaLegante\_change(indiceEtichetta As Integer, temperatura As Long)

30 - Public Sub BilRAPSiwa\_change()

27 - Public Sub ScambioTuboTroppoPieno\_Change()

24 - Public Sub BilTamponeRiciclato\_change()

23 - Public Sub ScaricoFiller\_change()

23 - Public Sub BitumeInSpruzzatura\_change()

22 - Public Sub VagliatoNonVagliato\_change()

21 - Public Sub ConsensoScaricoBilance\_change()

18 - Public Sub PressioneAriaInsufficente\_change()

18 - Public Sub PesataRAP\_Change()

17 - Public Sub Valv3VieSpruzzatriceVersoTorre\_Change()

17 - Public Sub PortinaNV\_change()

17 - Public Sub DeflTramScivScarBilRic\_change()

16 - Public Sub PortinaAgg\_change(portina As Integer)

15 - Public Sub DeflettoreNonPassa\_change()

15 - Public Sub CicliDosaggioEseguiti\_change()

15 - Public Sub ScambioFillerRecuperoInApporto\_Change()

13 - Public Sub CambioRicettaPrenotato\_change()

12 - Public Sub FuoriTollBitume\_change()

11 - Public Sub FuoriTollViatop\_change()

11 - Public Sub FuoriTollRiciclato\_change()

11 - Public Sub FuoriTollFiller\_change()

11 - Public Sub FuoriTollContalitri\_change()

11 - Public Sub FuoriTollAggregati\_change()

11 - Public Sub ContalitriErroreTimeOutAvvio\_change()

11 - Public Sub ContalitriErroreTimeOutArresto\_change()

9 - Public Sub BilAdditivoBacCNT\_change()

8 - Public Sub AggiornaGraficaFlomac\_Change()

7 - Public Sub ComandoScaricoMixer\_change()

7 - Public Sub BitumeDaCircuitoMarini\_change()

7 - Public Sub BitumeDaCircuitoEsterno\_change()

7 - Public Sub BitSpruzzato\_change()

7 - Public Sub BilanciaInertiPortinaChiusa\_change()

7 - Public Sub BilanciaInertiPortinaAperta\_change()

7 - Public Sub BilanciaFillerPortinaChiusa\_change()

### Oggetti utilizzati nella gestione dosaggio:

GestioneDosaggio.bas

Public Type ComponenteType

Public Type ComponenteContalitriType

Public Type TypeRicettaS7

Public Type RicettaTramogge

Public Type BilanciaPnComboType

## Gestione Motori

### Descrizione

Logica di funzionamento quasi completamente integrata nel PLC

### Dettaglio Funzioni

#### ParaTabMotor.bas

##### ParaTabMotor\_ReadFile()

Lettura dei parametri e invio dati al PLC

#### GestioneMotori.bas

##### ControlloAsservimenti()

Esegue alcuni controlli sul funzionamento dei motori

##### SetMotorManagement()

Gestisce il cambio della modalità di gestione dei motori corrente , chiama da varie funzioni di UI e dalla VerificaMotoriAccesi()

### Moduli interessati alla gestione motori

GestioneMotori.bas

GestioneAmperometri.bas

ParaTabMotor.bas

AvviamentoMotori.frm

GestioneOreLavoro.bas

### Metodi utilizzati nella gestione motori:

GestioneMotori.bas

208 - Public Sub MotoreAggiornaGrafica(motore As Integer)

171 - Public Sub MotoreUscita\_change(motore As Integer, Optional CambioStato\_per\_PausaLavoro As Boolean)

104 - Public Sub MotoreRitornoTermica(motore As Integer)

92 - Public Sub VerificaMotoriAccesi()

80 - Public Sub SetMotorManagement(newMotorManagement As MotorManagementEnum)

63 - Public Sub SetMotoreRitorno(motore As Integer, valore As Boolean)

60 - Public Sub SetMotoreAllarme(motore As Integer, valore As Integer)

58 - Public Sub ControlloAsservimenti()

52 - Public Sub PulsanteManualePremuto()

46 - Public Sub SetMotoreUscita(ByVal motore As Integer, valore As Boolean, Optional CambioStato\_per\_PausaLavoro As Boolean)

45 - Public Sub MotorManagementPlcCountDown\_change(motoreChanged As Boolean)

39 - Public Function MotoreCocleaRecuperoErrore() As Boolean

39 - Public Function MotoreCocleaPreseparatriceErrore() As Boolean

38 - Public Sub MotorManagementPlc\_change()

38 - Public Sub MotoriInAutomatico\_change()

31 - Public Function VerificaMotorePred(ByRef motore As MotorePredosatoreType, criterioAllarme As String) As Boolean

25 - Public Sub SetMotoreBlocco(motore As Integer, valore As Boolean)

25 - Public Sub GraficaDustFix()

22 - Public Sub MotorManagementPlcSirena\_change()

21 - Public Function VerificaMotore(ByRef motore As MotoreType, criterioAllarme As String) As Boolean

21 - Public Function MotoreAcceso(motore As Integer) As Boolean

20 - Public Sub SetMotoreRitornoReale(motore As Integer, valore As Boolean)

20 - Public Sub SetMotoreForzatoSpento(motore As Integer, valore As Boolean)

19 - Public Sub SetMotoreForzatoAcceso(motore As Integer, valore As Boolean)

17 - Public Sub NMSetMotoreUscita(ByVal motore As Integer, valore As Boolean)

17 - Public Sub ChkAvvioMotoriFillerizzazione(motore As Integer, comando As Boolean)

16 - Public Sub GestionePulsantiTipoFunzMot(Stato As Boolean)

15 - Public Sub DimensionaFiamma(tamburo As Integer)

15 - Public Sub ControlloMixerDosaggioStart()

14 - Public Sub NMSetMotoreForzato(ByVal motore As Integer)

14 - Public Function MotoreCocleaRecuperoAcceso() As Boolean

14 - Public Function MotoreCocleaPreseparatriceAcceso() As Boolean

13 - Public Sub SetMotoreRitornoIndietro(motore As Integer, valore As Boolean)

12 - Public Sub SetMotoreForzatoDarwin(motore As Integer, valore As Boolean)

12 - Public Sub AvvioStopAutomaticoMotori()

12 - Public Function CambioRicettaNVtoV() As Boolean

11 - Public Sub UpdateManagement(newMotorManagement As MotorManagementEnum)

11 - Public Sub FineAttesaStopAutomaticoMotori()

10 - Public Sub GestioneVaglio()

10 - Public Sub AggiornaGraficaValvolaTSF\_Change()

10 - Public Sub AbilitaPulsFormInversione()

9 - Public Sub MotoreRitorno\_change(motore As Integer)

8 - Public Sub NMSetMotoreUscitaInv(ByVal motore As Integer, valore As Boolean)

8 - Public Sub MotoreRitornoReale\_change(motore As Integer)

8 - Public Sub InizioAttesaStopAutomaticoMotori()

6 - Public Sub RimettiAutomaticoMotori()

6 - Public Sub MotorManagementPlcTroppoPienoNV\_change()

5 - Public Sub ControlloPuliziaFiltro()

4 - Public Sub MotoreSicurezza\_change(motore As Integer)

4 - Public Function MotoriInAutomatico() As Boolean

2 - Public Sub AccendiVaglioCambioRicetta()

GestioneAmperometri.bas

61 - Public Sub ValoreAmperometri\_change(amp As Integer)

32 - Public Sub ValoreAmperometriBar\_change(amp As Integer)

32 - Public Function FiltroAmperometro(amp As Integer, Value As Integer) As Integer

ParaTabMotor.bas

150 - Public Sub ParaTabMotor\_Apply()

110 - Public Function ParaTabMotor\_ReadFile() As Boolean

84 - Public Sub CaricaTestiMotori()

AvviamentoMotori.frm

93 - Private Sub MergeAvvMotori()

63 - Private Sub Form\_Load()

60 - Public Sub VisualizzaMotoreAcceso(motore As Integer)

59 - Private Sub APButtonStartStopMotore\_Click(Index As Integer)

45 - Private Sub imgPulsanteForm\_Click(Index As Integer)

38 - Public Sub DebugMotore(motore As Integer)

35 - Private Sub LoadImmaginiPulsantePlus(Index As Integer, stato As StatoPulsantePlus)

25 - Public Sub CambioMod()

21 - Private Sub LblOreParz\_DblClick(Index As Integer)

21 - Private Sub Form\_MouseMove(Button As Integer, Shift As Integer, x As Single, Y As Single)

19 - Public Sub AggiornaOreMotori()

18 - Private Sub LblOreTot\_DblClick(Index As Integer)

18 - Private Sub imgPulsanteForm\_MouseMove(Index As Integer, Button As Integer, Shift As Integer, x As Single, Y As Single)

16 - Public Sub ForzPLC(motore As Integer, valore As Boolean)

12 - Public Sub UpdatePulsantiForm()

9 - Public Sub UpdateManagement(managemode As MotorManagementEnum)

9 - Private Sub PasswordLevel()

6 - Public Sub ShowMe(ByRef parente As Form)

4 - Private Sub imgPulsanteForm\_MouseUp(Index As Integer, Button As Integer, Shift As Integer, x As Single, Y As Single)

4 - Private Sub imgPulsanteForm\_MouseDown(Index As Integer, Button As Integer, Shift As Integer, x As Single, Y As Single)

4 - Private Sub Form\_Activate()

3 - Private Sub txtVelMot\_DblClick(Index As Integer)

2 - Private Sub StatusBar1\_PanelClick(ByVal Panel As MSComctlLib.Panel)

2 - Private Sub CmdPassword\_Click()

2 - Private Sub ChkAvvForzatoPCL\_Click()

GestioneOreLavoro.bas

56 - Public Sub CalcolaOreLavoroMotori()

26 - Public Sub LeggeFileOreLavoro()

18 - Private Sub ScriveFileOreLavoro(motore As Integer)

16 - Public Sub AzzeraOreLavoroMotori(motore As Integer, totale As Boolean)

### Oggetti utilizzati nella gestione motori:

GestioneMotori.bas

137 - Public Type TamburoType

120 - Public Enum MotoriEnum

93 - Public Type MotoreType

15 - Public Type MotorePausaLavoro

8 - Public Enum MotorManagementEnum

7 - Public Enum MotorSequenceEnum

7 - Public Enum AvviamentoMotoriRidottoEnum

6 - Public Enum StatusMotori '20161014

4 - Public Enum FuelType

## Gestione Sili/Navette

### Premessa

Nessuna logica presente nel codice, tutta la logica è implementata nel PLC.

### Descrizione

Gestione dei sili e delle navette presenti nell’impianto.

### Logica

L’applicativo VB si occupa di visualizzare i livelli e lo stato dei sili e la posizione della eventuale navetta di trasporto del prodotto finale. Vengono inoltre recepiti i comandi operatore da inviare al PLC che si occupa della effettiva gestione.

### Dettaglio Funzioni

#### GestioneSilo.bas

##### SiloS7InviaParametri Assegna alle variabili PLC i parametri associati (destinazione navetta)

##### SiloS7ScriviPosizioni

Invia al PLC le posizioni dei sili gestiti

### Moduli interessati alla gestione sili:

GestioneSilo.bas

GestioneSiloGenerale.bas

ParaTabSilo.bas

### Metodi utilizzati nella gestione sili:

GestioneSilo.bas

294 - Public Function DefinisciTAG\_Silo() As Integer

115 - Public Sub SiloS7Leggi(forza As Boolean)

95 - Public Sub SiloS7GestioneAllarmi(ByRef IdDescrizione As Integer, ByRef IndirizzoPLC As String)

78 - Private Sub SiloS7Position\_changed()

64 - Public Sub CalcoloQuotePosGraficaSiloS7()

57 - Public Sub SiloS7InviaParametri()

53 - Public Sub AbilitaOggettiSiloS7(full As Boolean)

45 - Public Sub ScriviDestinazioneSilo(destinazione As Integer)

36 - Private Function SiloImmagineStato(siyncroOn As Boolean, fcAperto As Boolean, piena As Boolean) As String

35 - Public Sub SiloS7ScriviPosizioni()

28 - Public Function Linearizza(valore As Double, K1min As Double, K2max As Double, LoLim As Double, HiLim As Double) As Double

25 - Public Sub SiloS7IconStatusUpdate()

24 - Public Function SiloS7GetPosizioneSilo(asse As Integer, silo As String) As Double

23 - Private Sub SiloS7Stato\_changed()

19 - Public Sub ResetCmdSilo()

18 - Public Sub SetStatoSiloS7AutoMan()

17 - Public Sub SiloS7Scrivi()

14 - Public Sub SetStatoSiloS7Jog()

12 - Public Sub SiloS7WarnigEvent()

12 - Public Sub SetStatoSiloS7Stop()

12 - Private Sub SiloS7Speed\_changed()

6 - Public Sub SetSiloS7Start()

2 - Public Sub SiloS7ShowPosition()

GestioneSiloGenerale.bas

149 - Public Sub PosizionaSiliCP240()

82 - Public Sub LetturaTemperaturaSilo\_change()

78 - Public Sub MessaggioSiloToPlus(DatoSilo As DatoSiloMsgPlusType, Optional NumeroTelescarico As Integer)

77 - Public Sub RegistraScaricoSiloDB()

71 - Public Sub SiloGenerale()

59 - Public Sub CelleSiloLeggiTXT(NumeroModuloSilo As Integer)

54 - Public Sub SegnalazioneScaricoBennaNavetta(InScarico As Boolean)

48 - Public Sub AggiornaPesoSilo(DestinazioneSilo As Integer)

47 - Public Sub ScaricoSiloSenzaCelleCarico(Peso As String)

37 - Public Sub CelleSiloScriviTXT(NumeroModuloSilo As Integer)

33 - Public Sub SegnalazioneBennaSu(BennaSu As Boolean)

32 - Public Sub ValoreTempSilo\_change(silo As Integer, temperatura As Long)

31 - Public Sub VerificaSirenaLivelloAlto(indice As Integer)

31 - Public Sub CelleSiloValoreLetto\_change(silo As Integer)

30 - Public Sub CelleSiloInizializza()

26 - Public Sub AttivazioneSilo(silo As Integer, siloAttivo As Boolean, ritornoSilo As Boolean)

26 - Public Function ScompartiCompattaSalta(ByVal scomparto As Integer) As Integer '20151202 NUOVA GESTIONE SILI DEPOSITO

26 - Public Function GetIdDosaggioLOGScarico(IdDosaggioAct As Integer) As Integer

26 - Public Function GetIdDosaggioLogFromIdDosaggio(IdDosaggioAct As Double) As Integer

26 - Public Function GetDescrFromIdDosaggioLOG(IdDosaggioAct As Double) As String

24 - Public Sub PesoCamion\_change()

22 - Public Function DestinazioneSiloAppartieneModuloSilo(destinazione As Integer, NumeroModulo As Integer) As Boolean

19 - Public Sub VisualizzaSiloAttivo(usaFrmSilo As Boolean)

19 - Public Function ShowConfermaCambioSilo() As Boolean

16 - Public Sub PortinaScaricoSilo\_Change(NumeroSiloScaricato As Integer)

16 - Public Function VerificaEsistenzaSilo(silo As Integer) As Boolean

15 - Public Sub PosizioneDeodorante()

15 - Public Function RicavaDestinazioniDaModuloSilo(NumeroModulo As Integer, StringaConfigurazione As String) As String

13 - Public Function SiloVBToPlc(indice As Integer) As Integer

13 - Public Function PlcToSiloVB(indice As Integer) As Integer

13 - Public Function GetSiloIndex(silo As String) As Integer

11 - Public Sub TelescarichiSilo\_Change(silo As Integer)

11 - Public Function GetSiloString(silo As Integer) As String

11 - Public Function CalcolaNumeroModuliSilo(Stringa) As Integer

10 - Public Sub VisualizzaSiloPieno(silo As Integer)

10 - Public Sub LivelloAltoScomparto\_change(indice As Integer)

10 - Public Function ScompartiSiliPLC\_a\_PC(ByVal scomparto As Integer) As Integer '20151202 NUOVA GESTIONE SILI DEPOSITO

10 - Public Function ScompartiSiliPC\_a\_PLC(ByVal scomparto As Integer) As Integer '20151202 NUOVA GESTIONE SILI DEPOSITO

10 - Public Function ScompartiCompatta(ByVal scomparto As Integer) As Integer '20151202 NUOVA GESTIONE SILI DEPOSITO

10 - Public Function RicavaModuloSiloDaDestinazione(destinazione As Integer) As Integer

9 - Public Sub SiloLeggiTXTSenzaCelle(scomparto As Integer) '20151202 NUOVA GESTIONE SILI DEPOSITO

8 - Public Sub CaricoSilo\_change(silo As Integer)

8 - Public Sub AggiornaImgSemaforo(verde As Boolean, attesa As Boolean, semaforo As Boolean)

7 - Public Sub SiloScriviTXTSenzaCelle(scomparto As Integer) '20151202 NUOVA GESTIONE SILI DEPOSITO

7 - Public Function AzzeraSeNegativo(valore As Variant) As Variant

6 - Public Sub AggiornaAnalogicaPesaCamion\_change()

4 - Public Sub NavettaPosizioneCarico\_change()

4 - Public Function GetSiloFromConfigSilo(Index As Integer) As String

3 - Public Sub CelleSiloScriviTXTCamion()

3 - Public Sub CelleSiloLeggiTXTCamion()

2 - Public Sub TelescaricoSilo\_change(scomparto As Integer)

2 - Public Function SiloConfig2Combo(silo As String) As String

2 - Public Function GetSiloHexCode(silo As String) As String

ParaTabSilo.bas

146 - Public Sub ParaTabSilo\_Apply()

107 - Public Function ParaTabSilo\_ReadFile() As Boolean

74 - Public Sub WritePositioneSiloToXml()

46 - Public Sub ReadPositioneSiloFromXml()

### Oggetti utilizzati nella gestione sili:

GestioneSilo.bas

5 - Public Enum SiloStateType

GestioneSiloGenerale.bas

10 - Public Type SiloType

8 - Public Type DeodoranteSilo

6 - Public Type DatoSiloMsgPlusType

3 - Public Type Semaforo\_Type

## Allegati :

moduli-vb.txt : moduli componenti il progetto

objects-vb.txt : oggetti utilizzati

classes-vb.txt : classi utilizzate

forms-vb.txt : forms

lista-funzioni-bas.txt : procedure contenute nei moduli .bas (sorgenti vb)

lista-funzioni-frm.txt : procedure contenute nei moduli .frm (forms e callbacks)

Summary report.docx : code analysis report

html-doc : code navigation folder