*Documentazione tecnica dell’applicazione CyberTronic 500  
modulo Visual Basic - Versione: 9.6.46*

Documentazione CyberTronic 500

Sommario

[Documentazione CyberTronic 500 - Visual Basic 5](#_Toc488830148)

[Breve descrizione di un impianto Marini CyberTronic500 5](#_Toc488830149)

[Componenti 5](#_Toc488830150)

[Modulo Visual Basic 6](#_Toc488830151)

[Struttura dell’ambiente di sviluppo della applicazione VB6 6](#_Toc488830152)

[Componenti dell'applicazione VB6 6](#_Toc488830153)

[Moduli sorgente componenti il progetto e loro sommaria descrizione 6](#_Toc488830154)

[Descrizione della implementazione per singolo componente 9](#_Toc488830155)

[Colloquio con il PLC 9](#_Toc488830156)

[Scrittura verso PLC 9](#_Toc488830157)

[Lettura da PLC 9](#_Toc488830158)

[Gestione Cisterne 1](#_Toc488830159)

[Gestione Essiccatore 1](#_Toc488830160)

[Dettaglio Funzioni 2](#_Toc488830161)

[Predosaggio 1](#_Toc488830162)

[Dosaggio 1](#_Toc488830163)

[Motori 1](#_Toc488830164)

[Sili 1](#_Toc488830165)

[Allegati : 2](#_Toc488830166)

# Documentazione CyberTronic 500 - Visual Basic

Questo documento ha lo scopo di descrivere il funzionamento dell’applicazione utilizzata per la gestione degli impianti Marini CyberTronic500 con particolare riferimento alla parte scritta in Microsoft® Visual Basic 6®.

L’ obiettivo principale di questa descrizione è il supporto alla riscrittura o porting delle funzionalità di tale applicazione verso diversi ambienti di sviluppo.

## Breve descrizione di un impianto Marini CyberTronic500

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

## Componenti

Un impianto CyberTronic500 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/PLCInAnalogici …

### 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 si ha 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

OPCTags.xml

ParameterValues.xml

Gestione Cisterne

CYB500

Materiali

Log Allarmi

Cisterne.ini (solo per memorizzazione materiali)

()

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 legante “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

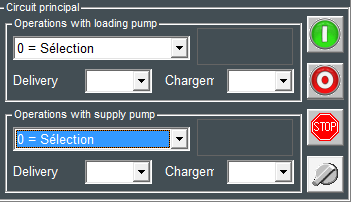
La gestione completa permette la piena gestione di valvole e pompa di carico del parco bitume, la visualizzazione di livelli, temperatura, agitatori e riscaldamenti di ogni cisterna.

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 ed una d’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.

A lato pratico le valvole vengono commutate come nelle singole operazione ad eccezione della 3vie bypass che viene mantenuta in posizione di bypass per separare la linea di carico delle cisterne.

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):

Pressione insufficiente

Avvio PLC

Emergenza

Start Alimentazione Torre obbligato

Start Alimentazione Torre obbligato

Abort OP

Start OP

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 le valvole non interessate all’operazione chiuse).

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.

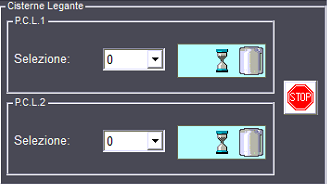
Il booster si attiva invece quando la temperatura scende al di sotto di un deltaT (scrivibile nei parametri generali delle cisterne) rispetto al set ideale; per ragioni di sicurezza il booster parte solo se la cisterna non si trova in minimo livello, in maniera che il bitume copra le serpentine del booster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

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

#### 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:



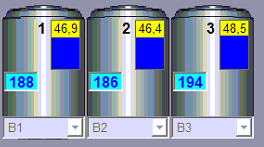
Mediante il parametro numero cisterne su pompa di circolazione si popolano le combo box di “Selezione” associando le giuste cisterne alla pompa di circolazione 1 e 2.

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.

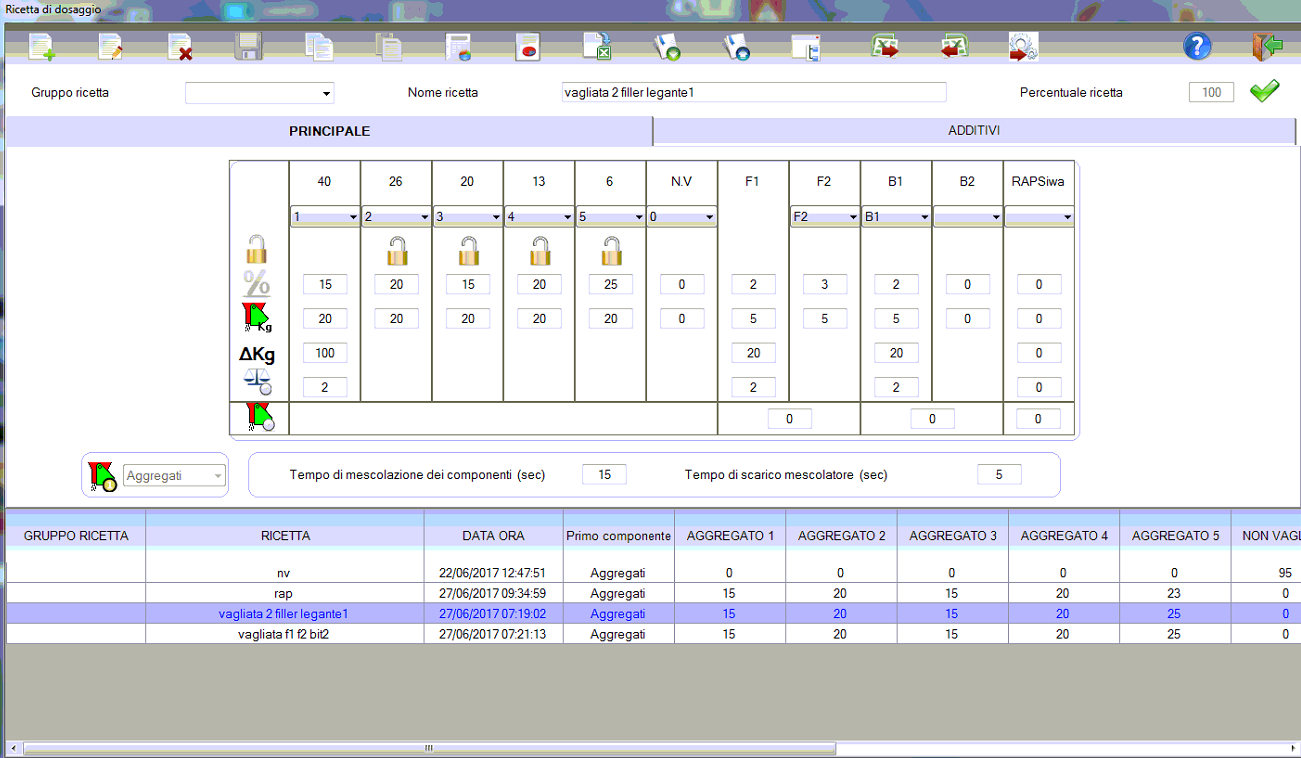
Sul finecorsa di aperto della valvola di uscita della cisterna di selezione sparisce l’icona della clessidra di attesa.

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  
  Fa un controllo sui parametri per l’avvio del bruciatore, in modo da settare un eventuale codice di errore. Si verifica un errore se:
  + Bruciatore ad olio e
    - pressione insufficiente
    - motore pompa non acceso
  + Temperatura di entrata del filtro superiore alla soglia
  + Non è attivo l’avviamento del bruciatore caldo e il motore del nastro elevatore freddo è fermo
  + Il nastro elevatore caldo è fermo
  + Il motore dell’aspiratore è fermo
  + Il bruciatore non è in posizione di accensione
  + Bruciatore ad olio e temperatura del combustibile è troppo bassa o troppo alta
  + Bruciatore a gas e
    - allarme di pressione del gas alta
    - allarme perdita dalle valvole del bruciatore
  + Allarme della termica del ventilatore
  + Allarme del sensore fumi del tamburo
  + Se avviene una partenza a caldo e le temperature di una dei bitumi è bassa
  + Se avviene una partenza a freddo e il motore di rotazione dell’essiccatore è fermo
* **ConteggioTempoArrestoBruciatore**(tamburo As Integer)  
  Se tutti i predosatori sono spenti e il bruciatore acceso deve partire il tempo di arresto. Il calcolo dei secondi che mancano è dato dal tempo di stop del bruciatore (parametro) meno il tempo passato dallo spegnimento dei predosatori.
* **StopBruciatoreTamburo**(tamburo As Integer)  
  Arresta il bruciatore. Nel caso il bruciatore fosse già in spegnimento, si assicura che Predosatori e tamburo vengano fermati.
* **ControlliFiammaBruciatore**(tamburo As Integer)  
  Esegue i controlli sulla fiamma del bruciatore visualizzando gli specifici allarmi, se presenti.  
  Se i predosatori sono in start e il bruciatore non è acceso allora stoppa i predosatori, sempre che non sia stato selezionato un avvio senza bruciatore. Quindi mette il bruciatore in manuale.  
  Se il tamburo è in preriscaldamento e il bruciatore non è nella posizione di accensione, mette il modulatore in up.
* **ControlloBloccoBruciatore**(tamburo As Integer)  
  Esegue l’asservimento del bruciatore al filtro: se è acceso e l’aspiratore del filtro no allora viene spento anche lui, sempre che non sia selezionato l’esclusione dell’avviamento a caldo.  
  Lo stop al bruciatore viene dato anche se ho una richiesta di blocco della fiamma ma ho ancora il modulatore aperto, o la ventola accesa, o lo start del bruciatore da PLC.
* **AllarmeCombustibile**(tamburo As Integer, inizializza As Boolean), OlioCombInTemperatura\_change(tamburo As Integer)  
  Visualizza a video un eventuale allarme del combustibile
* **ModulatoreBruciatore\_change**(tamburo As Integer)  
  Chiamato nel ciclo di PlcInAnalogici, ricalcola la posizione del modulatore.
* **ModoRegolazioneDepressioneFiltroConDepressFiltroIN**()  
  Se la depressione del filtro rimane all'interno della forchetta prestabilita eseguo il controllo automatico sulla depressione dei due bruciatori e il filtro lo lascio stare, ovvero non tocco il modulatore dell’aspirazione. Se la depressione del filtro esce dalla forchetta prestabilita mi disinteresso del controllo della depressione dei due bruciatori e agisco sulla depressione del filtro per riportarla all'interno della forchetta
* **RegolazioneAriaFredda**(),**RegolazioneImpulsoAriaFredda**(UpDown As Integer, DeltaTemperatura As Integer), **DurataImpulsoAriaFredda**(DeltaTemperatura As Integer) As Integer  
  Gestiscono la regolazione del modulatore dell’aria fredda. La durata dell’impulso mandata al modulatore è ricavata con una specie di PID: Controllo ogni 5 secondi la temperatura di entrata del filtro "TimerAttesaRegolazioneAriaFredda"; più sono vicino al setpoint e minore sarà la durata dell'impulso di apertura o chiusura; l’impulso non può durare più di 2 secondi e non meno di 0,2 secondi.
* **AltaTemperaturaFumiTamburo**(tamburo As Integer)  
  **Mai usata**, spostato tutto in AltaTemperaturaFiltroSw()
* **AltaTemperaturaFiltroSw**(), TempEntrataFiltro\_change()  
  Se viene superata la temperatura massima del filtro, vengono stoppati bruciatore e tamburo
* **GestioneModulatoreBruc**(tamburo As Integer)  
  Se il bruciatore è in fase di spegnimento, il modulatore va in down, se si sta accendendo va in up.
* **LetturaScivoloTamburo**(tamburo As Integer, temperaturaScivolo As Long, plcInAnalogici\_Fatta As Boolean), **ValoreLettoTempScivolo\_change**(tamburo As Integer)  
  Chiamata in PlcInAnalogici().  
  Calcola la temperatura dello scivolo come una media delle ultime 15-30 (a seconda del parametro NumeroLettureScivolo) campionature.
* **TempTorre\_change**(Index As Integer, temperatura As Long), AggiornaTemperaturaTorre()  
  Aggiorna I dati di temperature di un componente (sabbia, inerte, etc) sulla torre.
* **ArrestoBrucTempoX**(tamburo As Integer)  
  Arresta il bruciatore dopo un certo tempo. Se è attivo lo stop della fiamma all’arresto del nastro collettore, e questo è effettivamente fermo, e il bruciatore ha finito la fase di spegnimento, allora arresta il bruciatore. Se non è attivo ho lo stop all’arresto del nastro allora ferma il bruciatore allo stop dei predosatori. Questo significa che il bruciatore ha un suo tempo di spegnimento, ma il suo spegnimento automatico è asservito o al nastro o ai predosatori.
* **BloccoFiammaBruciatore\_change**(tamburo As Integer)  
  Richiamata in PlcInDigitali(). Se il bruciatore è in automatico ed è stato dato il comando di blocco della fiamma, allora passa al manuale. Se è attiva l’esclusione di avviamento a caldo arresta sia il bruciatore che il tamburo, altrimenti solo il bruciatore.
* **StopBruciatore**(tamburo As Integer)  
  Arresta il bruciatore. Si assicura che i predosatori vengano fermati, a meno che non sia selezionata la 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().

### Predosaggio

### [Dosaggio](Gestione%20Dosaggio.docx)

### [Motori](file:///C:\Users\uts.MARINI\Documents\projects\new-project\cyber500-doc\Gestione%20Motori.docx)

### [Sili](Gestione%20Sili.docx)

## 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