

Novembar 2003

Specijalna poglavlja softwareskih sistema

OPIS TIPIČNIH FUNKCIJA I ARHITEKTURE SCADA SOFTWARESKIH PAKETA SA MMI SOFTWAREOM

Uvod u FIX software

FIX, **FIX MMI** i **iFIX** su Intellution proizvodi u FIX familiji industriskog softwarea za upravljanje procesima.

U 1988, lansiranje **FIX DMACS** (Distributed manufacturing Automation and Control Software), bio je prvi SCADA software baziran na PC računaru, sa distribuiranim klijent/server arhitekturom. U kasnijim verzijama on se koristio na raznim platformama uključujući DOS, VMS, OS/2, Windows, Windows NT.

Za upravljanje i nadzor batch procesa (diskontinualnih ili šaržnih), koristi se **FIX BOS** (Batch Operations Supervisor). **FIX BOS** integrira SCADA mogućnosti **FIX** paketa sa višim nivom specifičnih za batch procese management funkcija preko relacije baze podataka.

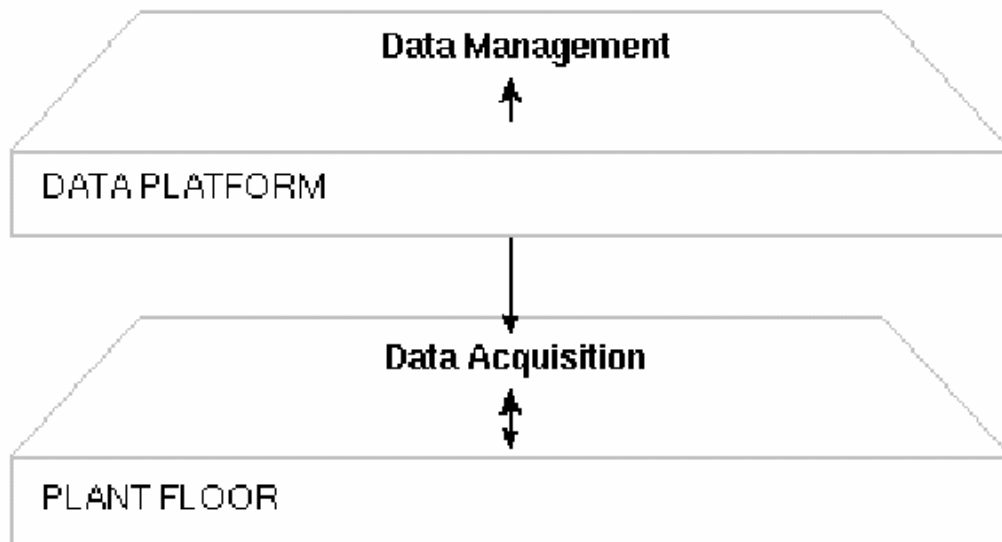
FIX funkcije

Prikazivanje podataka

FIX software kao paket namjenjen upravljanju i nadzoru procesa, pruža operatorima prikaz procesnih podataka u realnom vremenu, kao i druge softwareske aplikacije u postrojenju. Ovo prikazivanje podataka u relnom vremenu je ključno za efikasnije korištenje resursa i personala i za veći nivo automatizacije.

Bazne funkcije

FIX, **iFIX** i **FIX MMI** su softwareski sistemi. Jezgro softwarea izvršava bazne funkcije koje dozvoljavaju specifičnim aplikacijama da izvršavaju svoje doznačene zadatke. Dvije osnovne funkcije su : akvizicija podataka i management podataka. Slijedeća slika ilustrira ove bazne funkcije:



Akvizicija podataka

Akvizicija podataka je sposobnost dobijanja podataka iz procesa i njihovo procesiranje u obliku korisnom za operatore. Podatci mogu biti takodjer izdavani u proces (pisani) , uspostavljajući na taj način dvosmjernu vezu sa procesom.

FIX software ne zahtjeva neki svoj specifični hardware da bi prikupljao podatke sa procesa. On komunicira direktno sa I/O uredjajima koji već postoje putem softwareskog intefejsa koji se naziva - I/O drajver (driver). Intellution ima široku lepezu ovih I/O drajvera za najpoznatije familije I/O modula i uredjaja.

U većini slučajeva FIX I FIX MMI mogu raditi sa I/O hardwareom koji je već instaliran na postrojenju.

Management podataka

Nakon što su podatci prikupljeni oni se obradjuju i usmjeravaju prema zahtjevima softwareskih aplikacija. Ovaj postupak se definiše kao management podataka (data management).

Intellution FIX familija softwarea se izvršava na standardnim PC računarima. Različiti dijelovi postrojenja mogu koristiti različit računarski hardware. Pošto sve FIX platforme imaju sposobnost da komuniciraju sa čvorovima pod različitim platformama, mogu se uvezati u jedinstvenu mrežu. Jezgro FIX softwarea transparentno povezuje razmjene podataka izmedju različitih platformi.

Integritet podataka

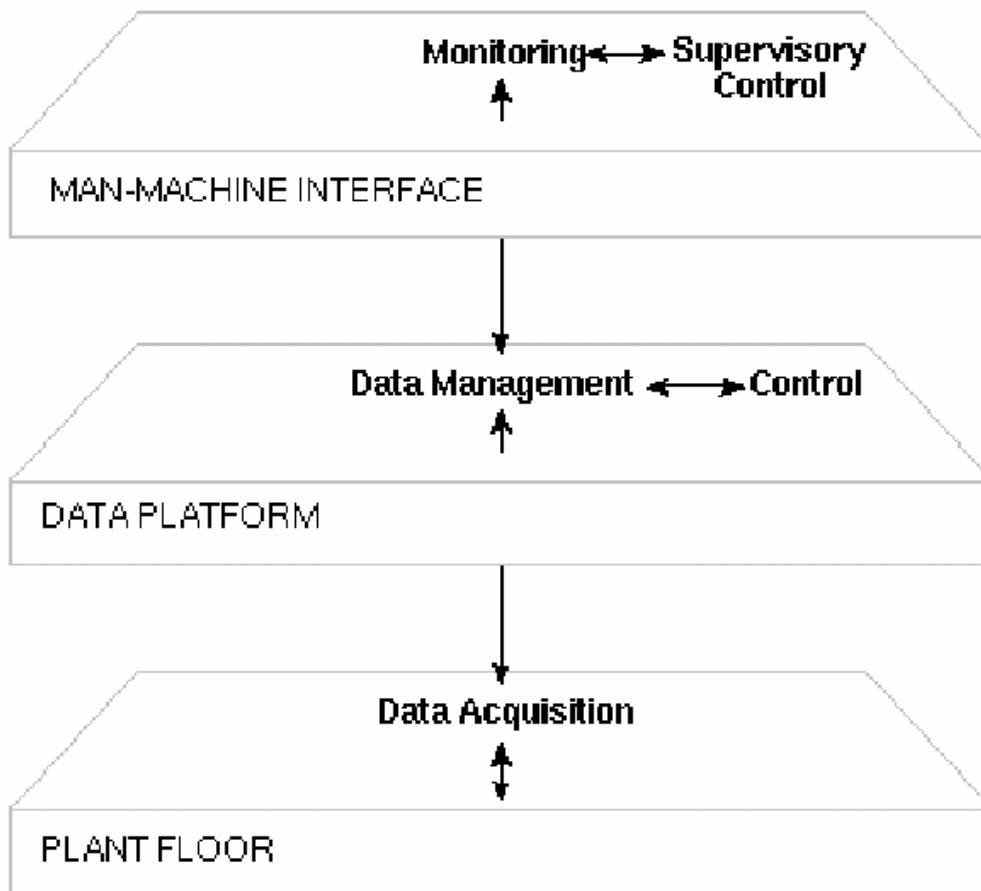
Osnovne funkcije akvizicije podataka i managementa obezbjedjuju osnovu za sve funkcije upravljanja procesima koje FIX i FIX MMI mogu izvršavati. Apsolutni integritet podataka je u srcu Intellution-ovog softwarea i predstavlja inherentni dio njegove razvojne filozofije.

MMI i SCADA funkcije

Prvi i najvažiji step u automatizaciji procesa je da se operatori koriste efikasnije u vodjenju procesa. Tradicionalni paneli kontrolnih sala se mogu zamjeniti sa računarima i grafičkim displejima koji izvršavaju Intellution software. Računar može obezbjediti mnoge od funkcija u kontrolnoj sali, kao :

- monitoring
- supervizorsko upravljanje
- alarmi
- upravljanje i regulacija

Slijedeća slika ilustruje ove SCADA funkcije :



Monitoring

Monitoring je sposobnost prikazivanja podataka iz procesa operatorima u realnom vremenu. Moćni numerički , tekstualni i grafički formati su na raspolaganju da bi se podatci učinili što dostupnijim operatorima.

Supervizijsko upravljanje

Supervizijsko upravljanje je sposobnost nadziranja u realnom vremenu prošireno mogućnošću da operatori mogu mjenjati zadate vrijednosti (set points), izlaze i druge vrijednosti direktno sa ekrana. Pošto FIX software može čitati i upisivati procesne podatke , može se uspostaviti supervizijska stanica upravljanja da bi određivala koje procesne tačke mogu biti i upisivane i čitane a koje se mogu samo čitati (read-only).

Alarmi

Bez obzira da li operatori rade sa nadzorne stanice ili supervizijske stanice upravljanja, oni imaju potrebu za mogućnost trenutnog uočavanja izuzetnih stanja i događaja u procesu. Alarmiranje predstavlja dakle sposobnost prepoznavanja ovih nenormalnih stanja i trenutno izvještavanje o ovim događajima.

Alarmi se generišu na granicama normalnih stanja procesnih veličina koje unose operatori odnosno dizajneri sistema upravljanja. FIX aplikacije mogu koristiti alarme na vrlo različite načine da bi upozorili operatore. Moguće je takodjer selektivno konfigurisati komuniciranje ovih alarmnih poruka između FIX čvorova u mreži.

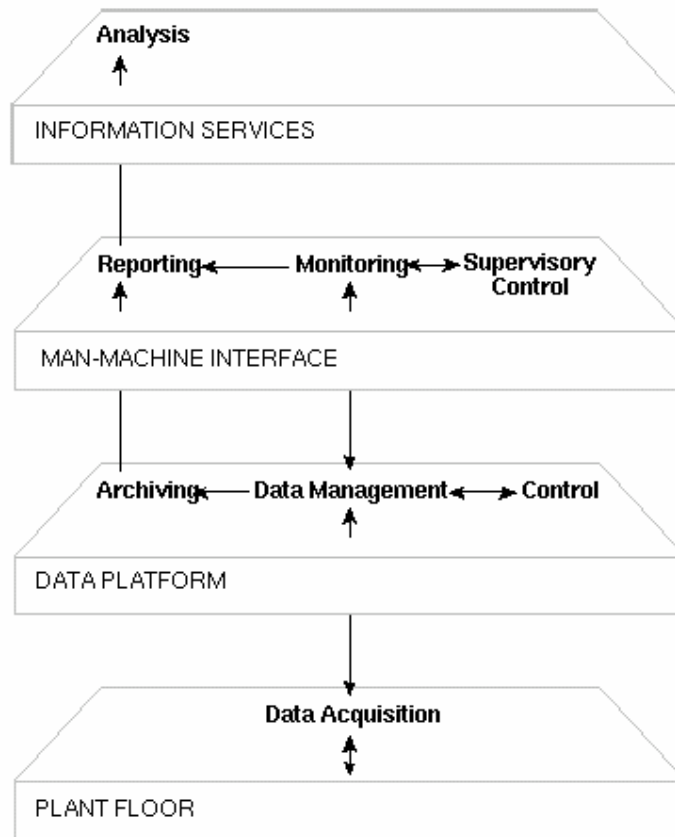
Upravljanje

Upravljanje (control) je sposobnost da se automatski primjene algoritmi koji podešavaju procesne vrijednosti i time održavaju ove vrijednosti unutar postavljenih granica. Upravljanje ide jedan korak dalje od supervizijskog upravljanja time što odklanja potrebu za interakcijom sa operatorom. Za ovo se može koristiti računar da upravlja ili cijelim procesom ili dijelom procesa. Intellution software u okviru FIX paketa omogućava kontinualno upravljanje , batch (šaržno) upravljanje i daje i mogućnost statističkog upravljanja.

Funkcije izvještavanja (reporting)

Podatci u realnom vremenu su samo jedan nivo procesiranja podataka. Mnoga postrojenja zahtjevaju mogućnost izvještavanja ili pohranjivanja podataka u realnom vremenu radi kasnije analize. FIX i FIX MMI imaju mogućnost kreiranja izvještaja o kritičnim informacijama iz sistema i procesa.

Naredna slika ilustrira ove funkcije izvještavanja :



Arhiviranje podataka

Bilo koja procesna vrijednost u sistemu može biti uzorkovana i pohranjena u sistemu u fajl podataka (data file) sa brzinom koju specificira operator. U svakom trenutku vremena, podaci se mogu restaurirati iz fajlova podataka da se od njih kreiraju trend displeji historijskih podataka. Menadžeri i operatori u procesu mogu koristiti ove podatke da ispituju događaje koji su prethodili kritičnom događaju , nakon što riješe urgentnije probleme. Arhivirani podatci su moćno orudje u analizi i optimizaciji vođenja procesa.

Izveštavanje

Intellution obezbjedjuje funkcije koje omogućuju operatorima da pristupaju FIX podacima putem standardnih industrijskih protokola za razmjenu podataka kao što su DDE, ODBC i SQL. Operatori mogu kreirati detaljne izvještaje sa spreadsheetovima (tabličnim izvještajima) , kao što je Microsoftov Excel, koji može da prima prikupljene i izračunate real time i historijske podatke.

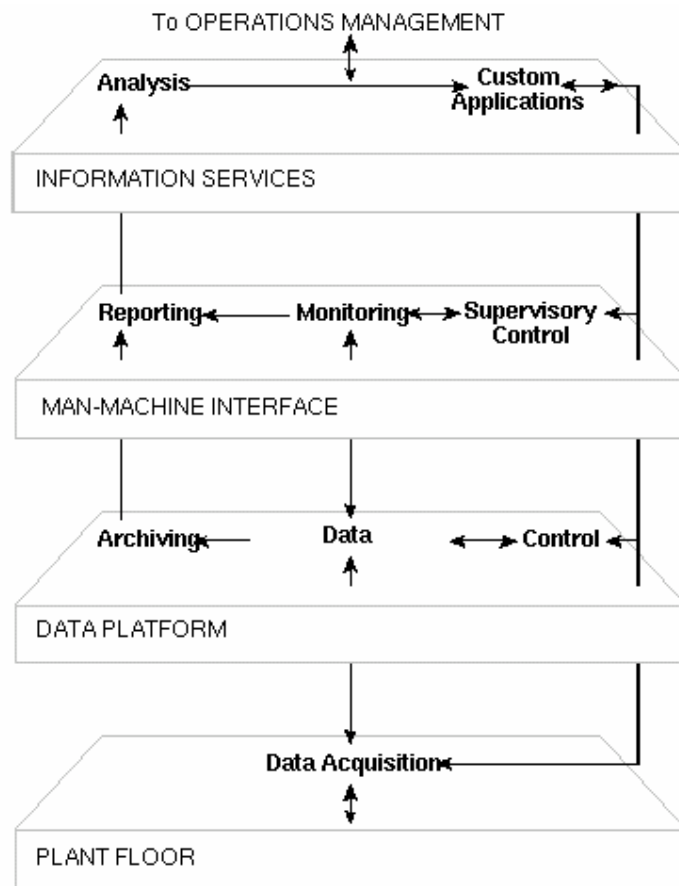
Ovi izvještaji obezbjedjuju važan off-line alat za inženjere procesa koji nadziru performansu procesa.

Funkcije otvorene arhitekture

Mnogi procesi imaju specifične potrebe koje se mogu riješiti time što se ima pristup podacima u realnom vremenu.

Intellution obezbjedjuje set Visual BASIC , C, C++ funkcija ovih jezika koje obezbjedjuju pristup pisanju i čitanju bilo kojeg procesnog podatka u sistemu. Otvorena arhitektura FIX sistema obezbjedjuje IT inženjerima alate da mogu pisati softwareske aplikacije koje rješavaju specifične probleme procesa.

Slijedeća slika ilustrira funkcije otvorene arhitekture:



Funkcije otvorene arhitekture

Otvorena arhitektura takodjer dozvoljava inženjerima da mogu razviti aplikacije koje obezbjedjuju ključne podatke u realnom vremenu za software managementa operacija i drugim aplikacijama u postrojenju.

Podrška za Microsoft OLE 2.0 omogućava IT inženjerima da razviju VisualBASIC aplikacije koje će interaktivirati sa podacima iz procesa.

Mogućnost distribuiranog procesiranja

Isto tako važno kao što je to šta software može učiniti, je kako je to učinjeno. FIX procesne mogućnosti dozvoljavaju veliki broj konfiguracija i procesnih strategija.

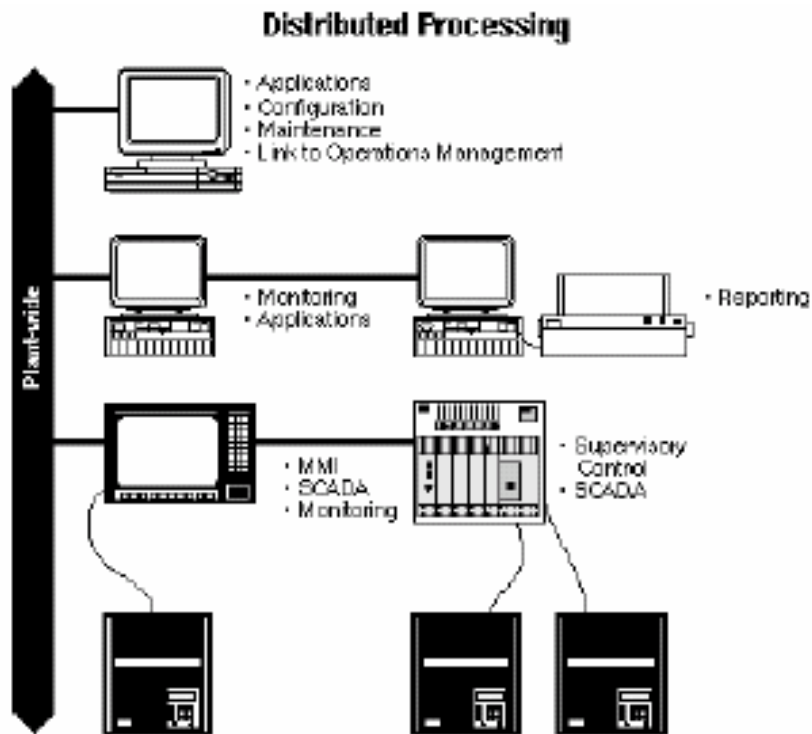
Distribuirano procesiranje

Najintegralniji dio softwareskog dizajna je njegova sposobnost da izvršava istinsko distribuirano procesiranje putem mreže. Mnogi sistemi rade na hijerarhijski način što čini pojedinačne računare ranjivim na otkaze sistema bilo gdje u mreži. Arhitektura FIX i FIX MMI dozvoljava postrojenjima da distribuiraju kritične funkcije izmedju svih kompjutera (čvorova), na mreži. Svaki čvor može komunicirati sa svim ostalim čvorovima na mreži, ali lokalni zadatci nisu nužno zavisni od drugih čvorova.

Centralizirano procesiranje

Neke aplikacije trebaju samo jedan čvor da bi izvršavale svoje funkcije. FIX software radi takodjer vrlo glatko i u okviru samo jednog računara. Vrlo je lako konvertovati distribuirani čvor u samostalni čvor ili pak samostalni čvor u distribuirani čvor.

Slijedeća slika pokazuje distribuirani i samostalni proces:



Centralized or Stand Alone Processing



Vremenski bazirano procesiranje

Većina aplikacija radi tako da prikuplja i obradjuje podatke u regularnim intervalima, definiranim u sekundama, minutama ili satima. FIX software može izvršavati bilo koju kombinaciju vremenski baziranog procesiranja. Ovo dozvoljava operatoru da balansira sistemske resurse izmedju podataka koji trebaju biti prikupljeni brzo i podataka koji se mogu prikupljati u toku dužih intervala. Čak i procesiranje do brzine od 50 ms (0.05 sec) je moguće postići na savremenijim P III i P IV baziranim PC konfiguracijama..

Procesiranje bazirano na izuzeću (exception based)

Često je efikasnije procesirati podatke nakon ključnih događaja , kao što je to promjena zadate vrijednosti ili zatvaranje ili otvaranje kontakta. Procesiranje koje je triggerovano događajima a ne trenutkom vremena je poznato kao procesiranje bazirano na izuzeću (exception based processing).

Procesiranje može biti triggerovano sa:

- promjenom podatka
- neplanirane poruke od procesnog hardwarea
- operatorskim akcijama
- softwareskim aplikacijama

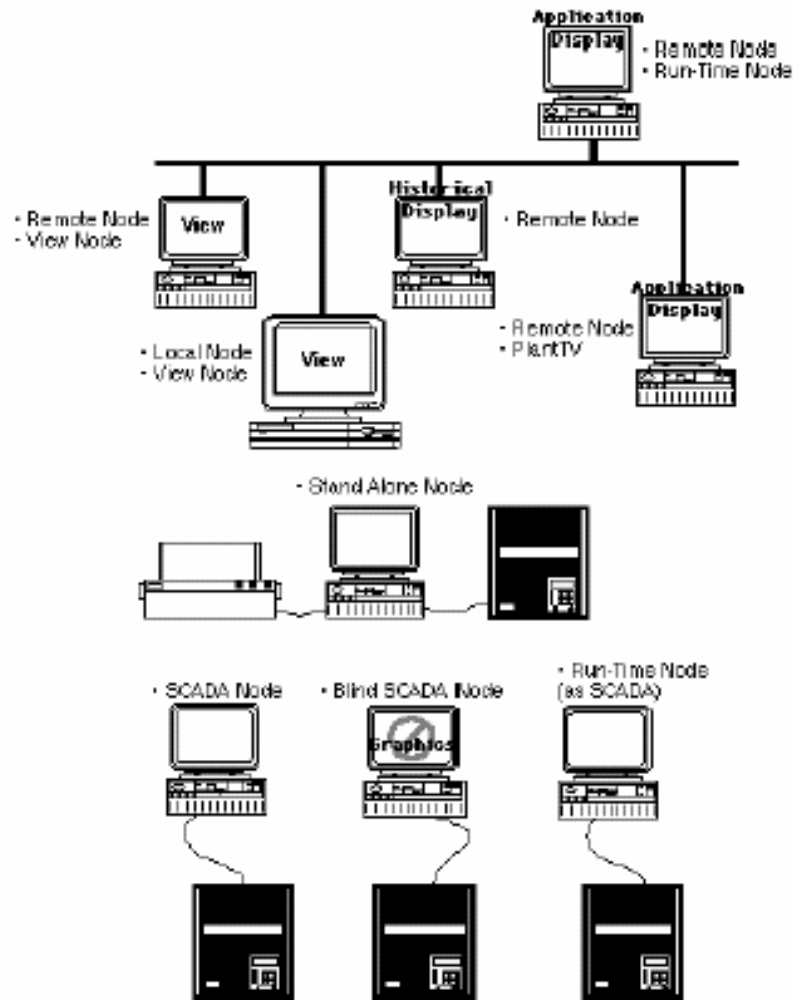
Procesiranje bazirano na izuzeću je bitno za istinski distriburane SCADA aplikacije koje nadziru veliki broj I/O uredjaja. Na primjer, naftno polje može biti nadzirano putem velikog broja RTU jedinica (remote terminal unit – daljinskih terminala), Podatci iz RTU jedinica se ne mjenjaju suviše često, tako da nema potrebe da se podaci prikupljaju u fiksnom intervalu vremena. Medjutim, kada se podatci promjene, operatori trebaju to odmah da saznaju. FIX prati ove promjene i trenutno ih procesira.

FIX čvor može izvršavati simultano i vremenski bazirane i procesiranje bazirano na izuzeću. Ova sposobnost FIX daje korisniku mogućnost korištenja najbolje strategije procesiranja za svaku tačku podatka u sistemu.

ARHITEKTURA SISTEMA

Čvorovi

Čvor je bilo koji kompjuter na kojem se izvršava FIX software. Koncept čvora je ilustriran na slijedećoj slici:



FIX i FIX MMI tipovi čvorova

Lokalni i daljinski čvorovi

Kada radimo sa distribuiranim FIX sistemom, lokalnim čvorom se naziva čvor (računar) na kojem radimo. Daljinski se odnosi na bilo koji čvor sa kojim imamo potrebu da uspostavimo komunikaciju da bi pristupili njihovim podacima.

Samostalni čvor

Kada radimo sa centraliziranim FIX sistemom, samostalni čvor se odnosi na čvor koji izvršava sve funkcije. Samostalni čvorovi ne koriste mrežu.

SCADA server

SCADA server (ili SCADA čvor) izvršava FIX akviziciju podataka i management tih podataka. Obično, SCADA čvor je rezidentan u samom postrojenju i ima direktnu vezu sa hardwareom procesa (transmitterima, sensorima, aktuatorima).

Slijepi SCADA server

Slijepi SCADA server (ili slijepi SCADA čvor) ne koristi nikakav grafički software. Ova konfiguracija dozvoljava kompjuteru da koristi više resursa za akviziciju i management podataka sa procesa.

Bazični čvor

Bazični čvor je bilo koji kompjuter koji ima instaliran FIX bazni čvor software i ne izvršava niti jednu SCADA funkciju.

Izvršni čvor (run-time node)

Izvršni čvor ne dozvoljava nikakvu modifikaciju operatorskih displeja ili procesne baze podataka. Korisnik instalira prethodno konfigurirane fajlove na ove čvorove tako da operatori mogu nadzirati proces , mjenjati procesne parametre (zadate vrijednosti , itd.) , i potvrdjivati alarme.

TV postrojenja (PlantTV)

TV postrojenja je čvor koji dozvoljava managerima, supervisorima i ostalim da mogu pristupiti podacima iz procesa. Medjutim ovi podatci mogu biti samo nadzirani i nikakve izmjene nisu moguće niti potvrdjivanje alarma.

Klijent gledanja (View client)

Najpopularniji tip čvora je onaj na kojem se izvršava View program (gledanja). View je grafički displej u realnom vremenu koji je dio grafičke aplikacije. Pojam View čvor (ili View klijent) znači da čvor izvršava View program, ali na čvoru se mogu izvršavati i druge aplikacije.

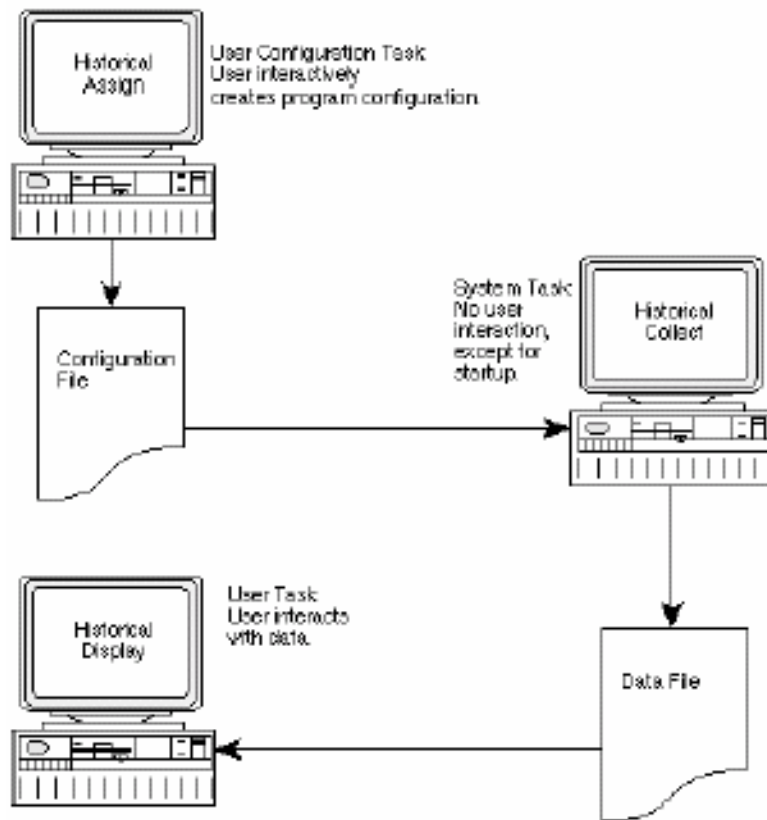
Programi

FIX i FIX MMI su višeprogramski sistemi (multitasking). Svaki FIX čvor može izvršavati istovremeno mnogo programa. Interno, kritični programi imaju prioritetni pristup sistemskim resursima i programi se mogu prazniti (preempted) , da bi se moglo odgovoriti na zahtjev još kritičnijom zahtjevima za resurse sistema.

Svi FIX i FIX MMI programi mogu biti klasificirani u tri tipa:

- sistemski zadatci (programi)
- zadatci konfiguracije korisnika programa
- korisnički zadatci (programi)

Slijedeća slika ilustrira odnose izmedju ovih programa:



Konfiguracioni programi korisnika

U korisničkim programima su uključene instrukcije i logika koja nadzire i kontrolira proces. Korisnički programi kreiraju konfiguracione fajlove. Sistemske aplikacije čitaju konfiguracione fajlove kada ih startamo i koriste podatke koje nalaze u njima da bi izvršavali doznačene zadatke.

Sistemske zadatke

Sistemske zadatke djeluju zajedno sa procesom u realnom vremenu. Ovi taskovi dobijaju njihove instrukcije iz konfiguracionih fajlova i zahtjevaju vrlo malo ili nikako interakcije sa operatorom. U opštem slučaju sistemske zadatke imaju prioritet u pristupu sistemskim resursima.

Korisnički zadatke

Korisnički zadatke su programi sa kojima operator interaktira da bi radio sa procesom i procesnim podacima. Korisnički zadatke takodje kreiraju i koriste konfiguracione fajlove.

Aplikacija historijskog trendiranja (zapisivanja), koji omogućavaju operatorima da uzorkuju i arhiviraju podatke, dolazi sa tri programa. Historijsko doznačavanje (historical assign) je zadatak korisničke konfiguracije sa kojim korisnik radi da bi se izabrale procesne tačke za uzorkovanje. Historijsko skupljanje (historical collect) je sistemske

zadatak koji izvršava uzorkovanje podataka bazirano na konfiguracionom fajlu kojeg je kreirao zadatak historjskog doznačavanja (historical assign).

Konačno, historijski displej je korisnički zadatak koji omogućava operatoru da prikaže prikupljene podatke na trend zapisima.

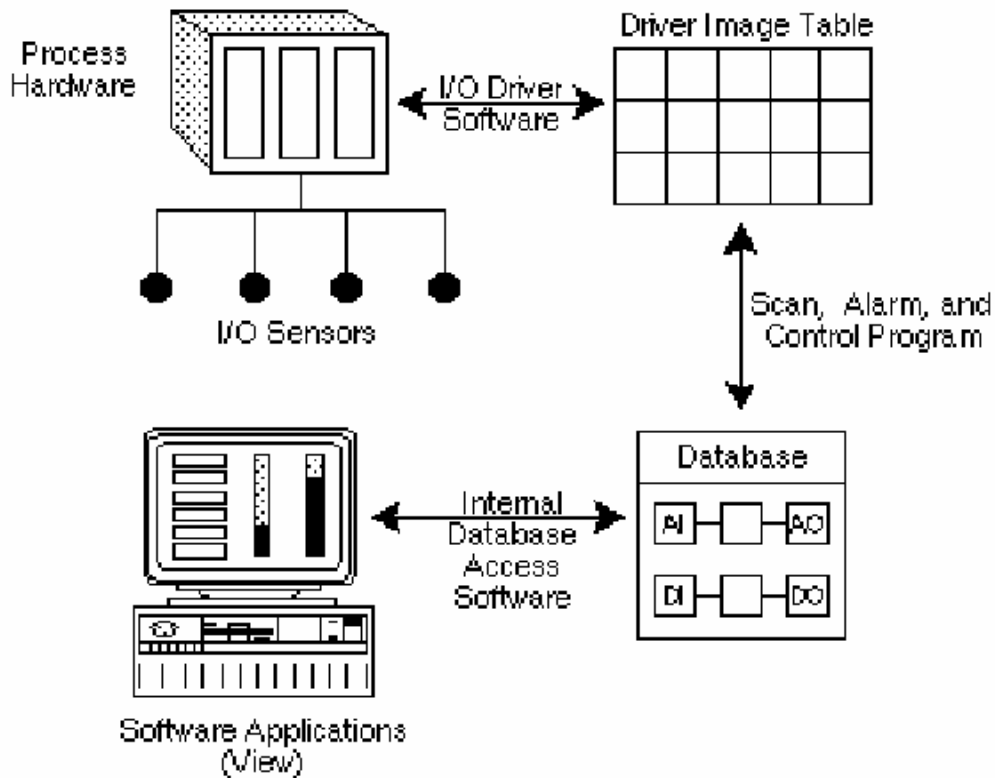
Neki programi mogu služiti za više od jedne vrste zadataka. Naprimjer, program za gradnju baze podataka (database builder) je korisnički konfiguracioni program za kreiranje procesne baze podataka. Program za gradnju baze podataka takodjer djeluje i kao korisnički program , dozvoljavajući operatorima da prikažu podatke u realnom vremenu u formi tabela (spreadsheet).

Bazna arhitektura

FIX software se gradi na instrumentaciji koja postoji u postrojenju. Tipično, postrojenje ima mrežu senzora i aktuatora povezanih sa I/O modulima uređaja kao što su programibilni kontroleri – PLC. Ovaj sistem obezbjeđuje automatsko upravljanje koje nadzire i vodi proces. Handikep je da ovi regulatori, senzori i aktuatori dobro komuniciraju medjusobno ali ne i sa operatorima koji vode proces. Tako je teško tačno znati šta se dešava u procesu u svakom trenutku vremena. FIX i FIX MMI obezbjeđuju *prozor u proces* omogućujući da podatci budu na raspolaganju operatorima i managerima u formatu koji oni razumiju.

Tok podataka

Tok procesnih podataka od procesnih hardwarea je pokazan na narednoj slici , i može se sumirati na slijedeći način:



Bazna arhitektura FIX softwarea

[1] I/O drajver software (takodjer nazvan *zadatak izbora – polling task*) čita podatke sa I/O uređaja i prebacuje te podatke na adrese u tabeli slike drajvera (*driver image table – DIT*).

[2] Skanira, alarmira i upravlja program (*Scan , Alarm and Control –SCAN*), isčitava podatke iz tabele slike drajvera (DIT), procesira ih i prebacuje podatke u procesnu bazu podataka.

[3] Funkcije sa internim pristupom bazi podataka čitaju podatke iz lokalne ili udaljene baze podataka i prebacuje ih ka softwareskim aplikacijama koje ih zahtjevaju. Ovaj transfer se odigrava bez interakcije sa operatorom.

Podatci se mogu takodjer izbacivati na procesni hardware , izvršavanjem ovih koraka u inverznom smjeru.

Sve skupa, I/O drajver, SAC i procesna baza podataka čine akvizicione i management funkcije FIX softwarea. U nastavku ovog teksta ovaj skup funkcija bit će nazivan SCADA. Dakle, SCADA čvor je onaj koji ima bazu podataka i izvršava programe I/O drajvera i SAC.

Naredne sekcije će biti posvećene ovim funkcijama sa više detaljnog opisa.

I/O drajveri

Prvi i najvažiji zadatak bili kojeg softwarea za automatizaciju procesa je da prikuplja sirove (izvorne) podatke sa procesa. Senzori i kontrolni elementi šalju podatke u registre PLC ili I/O modula. FIX software interfejsni moduli sa PLC-jevima ili ovim I/O modulima su na raspolaganju , koji mogu čitati podatke iz ovih registara. Ovaj softwareski interfejsni modul se naziva *I/O drajver*.

Intellution obezbjedjuje veliku lepezu ovih I/O drajvera za najpoznatije tipove PLC-jeva i I/O modula. Ovi drajveri obezbjedjuju takve mogućnosti kao što su : automatska detekcija komunikacionih grešaka, izvještavanje, oporavak iz stanja grešaka, ugradjeni monitoring podataka (*datascope*), i podršku za redundantni kanal komunikacije. Prijem informacija od DDE servera je takodjer podržan.

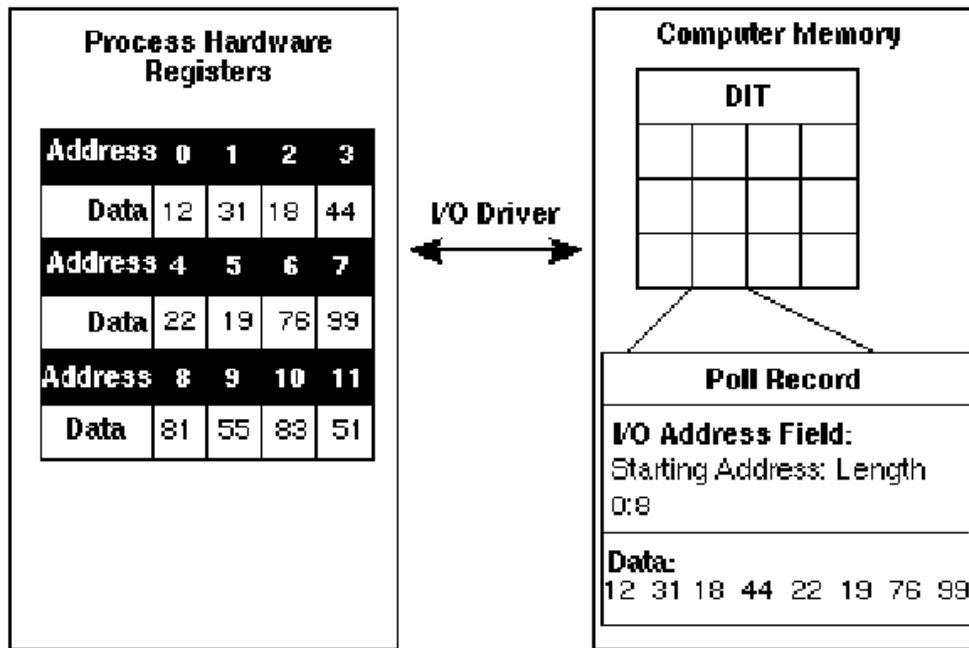
I/O drajveri su intimno povezani sa svojim hardwareom kojeg podržavaju , tako da detaljniji opis može biti nadjen uz dokumentaciju hardwarea kojeg podržavaju.

Tabela drajverske slike (driver image table)

I/O drajver je alat za pristupanje podacima u hardwarekim registrima i za specificiranje komunikacionih parametara. Kada se definišu komunikacioni parametri, I/O drajver je u stanju da formira i održava tabelu slike drajvera (*DIT*) . Neke dokumentacije nazivaju ovu tabelu kao : *izbornu tabelu (poll table)*.

Možemo zamisliti ovu tabelu kao skup mailboksova (*pretinaca za poruke*). Svaki mailbox u DIT tabeli se naziva *izborni zapis (poll record)* . Svaki *izborni zapis* može sadržavati bilo jednu tačku podatka ili niz tačaka u bloku. Da bi se specificirao *izborni*

zapis, korisnik treba da specificira startnu adresu i dužinu. Startna adresa kaže *I/O* drajveru gdje počinje opseg podataka u procesnom hardwareu. Dužina kaže *I/O* drajveru koliko podataka u bloku da uzme. Slijedeća slika ilustrira ovaj koncept:



Primjer izbornog zapisa (poll record)

Drugi zadatak *I/O* drajvera je da ažurira (update) tabelu slike drajvera (DIT), brzinom koja je specificirana za svaki izborni zapis. Vrijeme ažuriranja za DIT tabelu se naziva vrijeme izbora (*poll time*). Korisnik može specificirati ovo vrijeme od 0 (maksimalna moguća brzina), pa do 255 sekundi (otprilike 86.400 sekundi kod nekih drajvera), u inkrementima od 0.1 sekunde. Alternativno, neki *I/O* drajveri imaju mogućnost izbora procesiranja baziranog na izuzeću (exception based).

Baza podataka

Srcu FIX softwera je procesna baza podataka. Baza podataka je predstava o procesu kreirana izvršenjem upravljačke logike koja upravlja procesom. Procesna baza podataka koju kreira korisnik sa programom : graditeljem baze podataka (database Builder) koji se sastoji od blokova i lanaca. Blok (koji se naziva i tag), je kodirani skup instrukcija za kontrolu procesa koji izvršava specifični zadatak. Svaki blok zahtjeva od korisnika da obezbjedi nekoliko parametara putem dijalog boksova u okviru graditelja baze podataka (database builder). Općenito, postoji dva tipa blokova:

Primarni blokovi – čitaju podatke iz DIT tabele, upisuju podatke u DIT tabelu ili izvršavaju specijalne funkcije

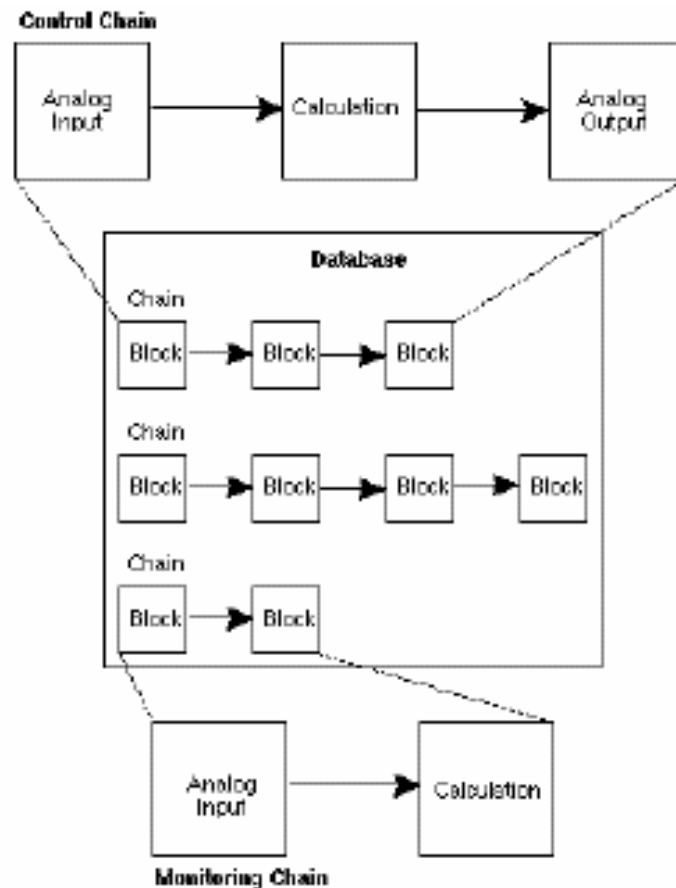
Sekundarni blokovi – manipuliraju podacima koji su im dostavljeni.

Primarni blokovi obrađuju sve ulaze i izlaze u i iz DIT tabele. Dok izborni zapis može sadržavati mnogo procesnih podataka, dotle blokovi mogu sadržavati samo jednu tačku podatka.

Lanac (*chain*) je serija spojenih blokova koji kreiraju upravljačku ili nadzornu strukturu. Naprimjer, u nekoj specifičnoj upravljačkoj konturi, treba prvo očitati tačku podatka, manipulirati sa podatkom koristeći standardne formule i zatim izbaciti na hardware dobijenu vrijednost poslije obrade. Lanac koji izvršava ovu upravljačku strategiju može uključivati blok analognog ulaza koji je spojen na kalkulacioni blok a ovaj spojen na blok analognog izlaza.

U drugom slučaju korisnik može željeti da iščita podatak i izvrši određene obrade nad njim u okviru softwareskih aplikacija. U ovom slučaju, potrebno je samo povezati analogni ulazni blok sa blokom za računanje (calculation), da bi se kreirala kontura nadzora.

Slijedeća slika ilustrira koncept blokova i lanaca.



Blokovi i lanci u okviru baze podataka

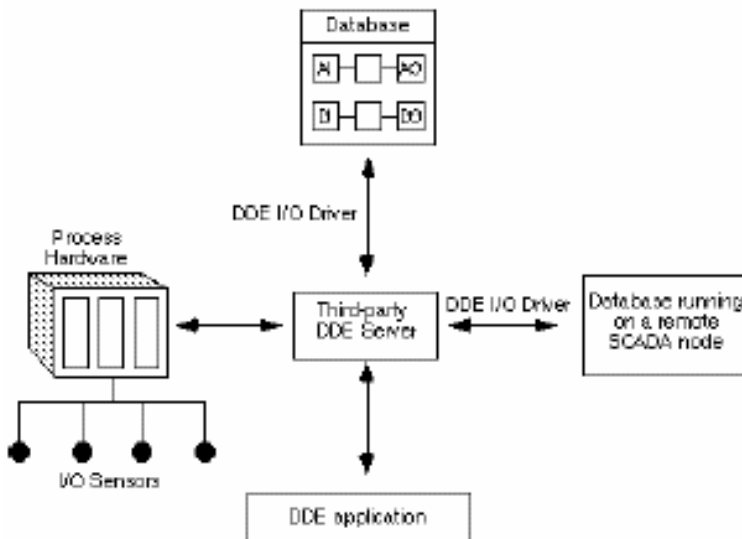
Podrška za DDE server

Procesna baza podataka može komunicirati sa DDE serverima drugih proizvođača. Koristeći ovu osobinu, moguće je čitati podatke iz i upisivati podatke u slijedeće izvore:

- I/O drajveri koji djeluju kao DDE serveri
- DDE server aplikacije (naprimjer Excel)

- Druge FIX baze podataka

Naredna slika ilustrira ovu arhitekturu:



Podrška za DDE server

Da bi se uspostavila komunikacija kroz DDE linkove, potrebno je definirati DDE adresu. Adresa se sastoji iz slijedećih stringova:

Aplikacija – definira ime aplikacije (ili servisa) sa kojim želimo da komuniciramo

Topic - definira podatak kojem želimo pristupiti. **Topic** varira zavisno od servisa. Naprimjer to može biti i ime fajla.

Item – definira jedinicu podatka. Item varira zavisno od servisa i topica. Naprimjer, item može biti čelija iz tabele ili ime taga i bazi podataka.

Nakon što korisnik definira DDE adresu, DDE I/O drajver može čitati i pisati podatke izmedju procesne baze podataka i DDE servera.

Program Skaniranja , alarmiranja i upravljanja (SAC)

Program Skaniranja , alarmiranja i upravljanja (SAC) je sistemski zadatak koji se izvršava na SCADA čvoru. SCADA je odgovoran za izvršenje logike lanaca u bazi podataka. SAC radi slijedeće:

- uzima podatke iz DIT tabele
- prevodi podatke u format koji baza podataka očekuje
- provjerava podatke na alarmne vrijednosti i generira alarmne poruke
- izvršava upravljačku logiku
- detektuje izuzeća
- pravi zahtjevane upise u DIT tabelu

Svaki lanac baze podataka takodjer sadrži informaciju o tome kako SAC treba procesirati lanac. Opcije su : vremenski bazirano procesiranje, procesiranje bazirano na izuzeću, i jednostruko procesiranje.

Vremenski bazirano procesiranje

SAC može izvršavati vremenski bazirano procesiranje u :

- subsekundnom brzinom (0.05 sec do 0.95 sec)
- sekundnom
- minutnom
- satnom

Frekvencija sa kojom SAC uzima podatke iz procesa se naziva vrijeme skaniranja (scan time) . Subsekundno skaniranje je na raspolaganju samo u nekim konfiguracijama.

Procesiranje bazirano na izuzeću

SAC može izvršavati procesiranje bazirano na izuzeću na slijedećim događajima:

- promjene podataka u DIT tabeli
- nezahtejavane poruke od procesnog hardwarea
- akcije operatora
- instrukcije od softwareskih aplikacija

Jednostruko procesiranje

SAC može izvršavati jednostruko procesiranje. Kada prvi blok u lancu ima vrijeme skaniranja jednako 0 , SAC će procesirati taj lanac samo jedanput – kad god primarni blok je skaniran.

Interfejs čovjek –sistem (MMI ili HMI)

Najvažnija FIX i FIX MMI aplikacija u procesu je je software koji obezbjedjuje *prozor na proces* . Sposobnost da može da zna šta se dešava u procesu , povezujući se sa instrumentima i računarima se često naziva : interfejs čovjek – sistem (man-machine interface - MMI) ili u nekoj literaturi HMI (human machine interface). U sistemu koji radi isključivo sa sensorima, kontrolnim elementima i procesnim hardwareom, MMI je često ograničen.

Grafičke primjene

FIX grafičke aplikacije pružaju ovaj *prozor u proces* koristeći kompjuterske ekrane visoke rezoulicije i bogate okružaje lakog za korištenje grafičkog i formata za podatke. FIX arhitektura dozvoljava računar da pristupi informacijama iz cijelog postrojenja. Rezultat je operatorska stanica koja obezbjedjuje više informacija u superiornom formatu. Grafičke aplikacije obezbjedjuju sve što je potrebno da razvijemo računarske prikaze koji omogućuju operatorima da interagiraju sa podacima u realnom vremenu. Grafičke aplikacije se sastoje se sastoje iz dva programa: crtaj i gledaj (Draw i View).

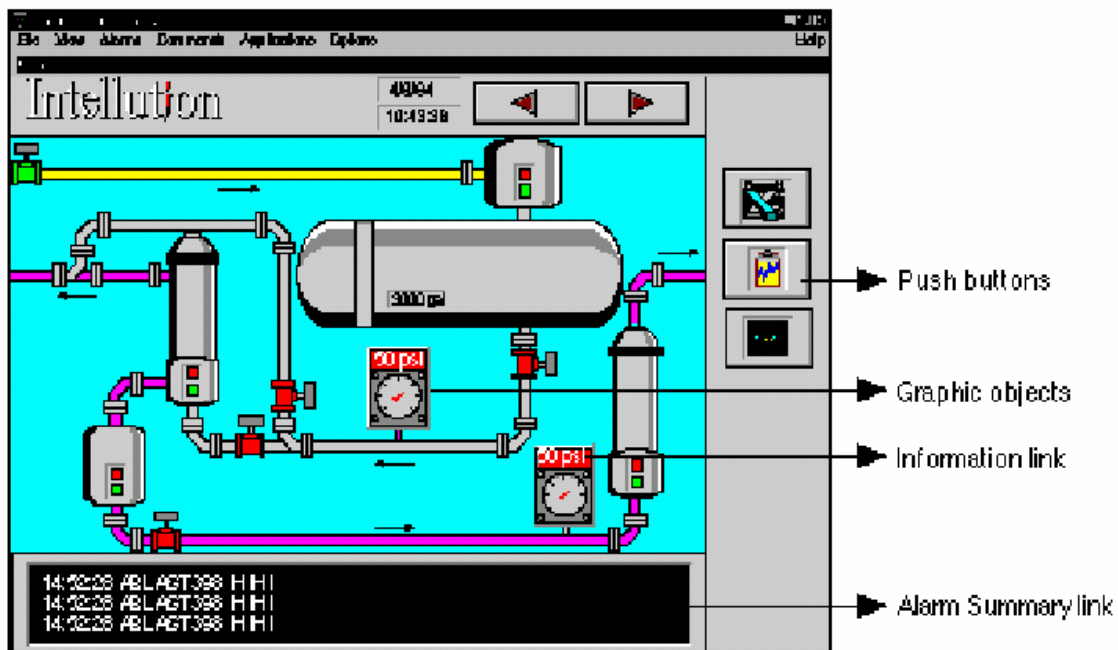
Crtaj (Draw)

Crtaj obezbedjuje dizajnera displeja sa grafičkim, tekst, podacima, animacijom, i alatom za crtanje tako da prikazi koje korisnik kreira su atraktivni, organizovani, laki za korištenje i razumljivi.

Gledaj (View)

Gledaj obezbedjuje operatore sa komandama i načinima da interagiraju sa *Crtaj* displejima u realnom vremenu.

Obadva programa imaju potpuni grafički korisnički interfejs (GUI – graphical user interface). Sa menijima , dugmadima i list boksovima i drugim familijarnim alatima GUI interfejsa , usvajanje i rad sa Draw i View programima kao Windows aplikacijama je brzo i intuitivno. Slijedeća slika pokazuje displej sa mnogim osobinama koje korisnik može uključiti u njegove displeje.



Tipovi prikaza

Linkovi

Srce grafičkih aplikacija je njihova sposobnost da pristupe informacijama iz baze podataka. Za direktne prikaze podataka, grafička aplikacija obezbedjuje raznovrsnost

linkova. Linkovi prikazuju sistemske ili procesne podatke u nizu formata i sadrže mnoge konfigurabilne opcije. Operatori mogu takodjer koristiti linkove da upisuju vrijednosti u bazu podataka.

Informacija u bazi podataka može takodjer biti korištena za kontrolu nekoliko animacionih osobina.

Svaka konfiguracija FIX korisničkog zadatka zahtjeva ulazni uredjaj tipa miša ili kugle (track ball). Pošto je Draw (crtaj) konfiguracioni zadatak (program), on zahtjeva ulazni uredjaj tipa miša. Sa druge strane View (gledaj) ne mora zahtjevati miša.

View (gledaj) podržava slijedeće ulazne uredjaje :

- miša
- kuglu (track ball)
- tastaturu
- ekran osjetljiv na dodir (touch screen)

Arhitektura distribuirane mreže

Dizajn Intellution mreže uključuje dva osnovna principa: istinsko distribuirano procesiranje i prenos podataka na zahtjev.

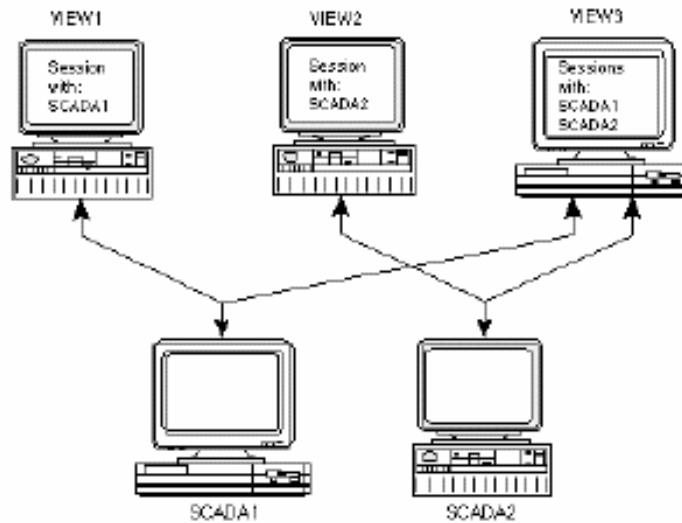
Distribuirano procesiranje

U mreži distriuiranog procesiranja, svaki čvor nezavisno izvršava zadatke koji su mu dodjeljeni. Jedna prednost ove strategije je da čvorovi mogu biti uzeti off-line (isključeni) bez da se čitava mreža mora isključiti. Kada čvor traži podatke od čvora koji je off-line (isključen), FIX mrežni software izvještava o ovome čvor koji je zahtjevao podatke, tako da čvor može da vlada ovom situacijom kada nema podataka, na elegantan način. Mada svaki čvor ima svoj integritet kao nezavisna stanica, čvorovi mogu pristupiti podacima bilo gdje u mreži. Na primjer, View (gledaj) čvor može prikazati sliku sa linkovima iz mnogih različitih SCADA čvorova.

Sesije

Svaki čvor može pristupiti podacima iz bilo kojeg drugog SCADA čvora u mreži. Korisnik selektivno konfigurira koji čvorovi komuniciraju sa SCADA čvorovima. Komunikacioni link između dva čvora preko mreže se naziva *sesijom*. Kada čvor uspostavi sesiju sa SCADA čvorom, podatci i alarmi mogu biti poslani između čvorova.

Slijedeća slika ilustrira komunikacionu sesiju



Mrežna sesija

Dinamičke konekcije

Korisnik može također za svoj čvor da automatski pravi konekcije u letu (on the fly) sa daljinskim čvorovima koji nisu specifično konfigurirani na korisničkom čvoru. Ove konekcije se zovu *dinamičke konekcije*.

Transfer podataka po zahtjevu (on demand)

Većina sistema za upravljanje procesima zahtjeva da svaki čvor koji želi da koristi podatke iz SCADA čvora, mora da ima kopiju cijele baze podataka lokalno pohranjenu. Rezultirajući saobraćaj na mreži može koristiti značajne resurse sistema. FIX i FIX MMI čitaju i pišu podatke na zahtjev i samo zahtjevani podatci se kreću mrežom. Ova strategija sačuvava resurse sistema za lokalne zadatke.

Fajl serveri

Fajl server može biti korišten da se pohrane podatci koji mogu biti potrebni u nekoliko čvorova , na neku pogodnu lokaciju. Koristeći FIX i FIX MMI , korisnik može pohraniti SCU fajlove, sigurnosne fajlove, historijske podatke, i recepte na fajl server (server fajlova).

Kada se koristi FIX sa WindowsNT operativnim sistemom, korisnik može postići sve prednosti fajl servera jednostavno implementirajući ugrađenu u operativni sistem mogućnost djeljenja fajlova (file sharing). Koristeći Windows NT File Manager, korisnik može uspostaviti konekciju sa mrežnim drajvom sa bilo kojim drugim drajvom u lokalnoj mreži. Jedanput kada se ova veza uspostavi korisnik može imati trenutni pristup bilo kojem djeljenom fajlu na tom čvoru, uključujući i FIX fajlove.

Djeljenje fajlova (file sharing)

WindowsNT bazirani čvorovi dozvoljavaju korisniku da učini fajlove u bilo kojem direktoriju raspoložive na drugim računarima na mreži putem djeljenja tog direktorija. Koristeći mogućnosti djeljenja fajlova u okviru WindowsNT, korisnik može direktno pristupiti bazama podataka, slikama, i drugim važnim FIX fajlovima iz drugih čvorova u mreži. Pristup djeljenim FIX fajlovima od strane drugih računara može biti kontroliran putem implementiranja sigurnosnih karakteristika WindowsNT sistema.

Više platformska konektivnost

FIX dozvoljava komunikaciju između čvorova koji rade na različitim platformama (operativnim sistemima). Naprimjer, možemo povezati WindowsNT, OS/2, i DOS čvorove zajedno da razmjenjuju podatke preko FIX distribuiranog sistema. Ovo omogućava korisnicima da zadrže svoj postojeći hardware kod postavljanja FIX instalacija.

Alarmiranje

FIX i FIX MMI imaju sofisticirani sistem za generiranje, prikazivanje, i pohranjivanje alarma i poruka. Korisnik može selektivno slati alarme i poruke na:

- bilo koji čvor u mreži
- printere spojene na FIX čvor
- fajlove pohranjene na diskovima
- sumarne prikaze alarma
- prozore historijskih alarma

Na lokalnom čvoru, programi koji izvršavaju ove alarmne funkcije se zovu alarmni zadatci (*alarm tasks*). Intellution također obezbjeđuje ugrađenu podršku za slijedeće funkcije:

- daljinsko potvrđivanje alarma
- suspenziju alarma (na primjer za vrijeme start procesa)
- alarmna kašnjenja

Tipovi alarma i poruka

FIX software generira jedan tip alarma i četiri tipa poruka:

Blok alarm – blokovi baze podataka generiraju alarme kada vrijednosti bloka izlaze van prethodno definiranih granica, kod promjene stanja ili kada se pojavi greška u komunikaciji.

Blok poruka – blokovi mogu također slati poruke na printere i u fajlove historije alarma da indiciraju da se neki događaj desio kod tog bloka. Ove poruke se ne pojavljuju na operatorskom prikazu i ne zahtjevaju potvrđivanje od strane operatora.

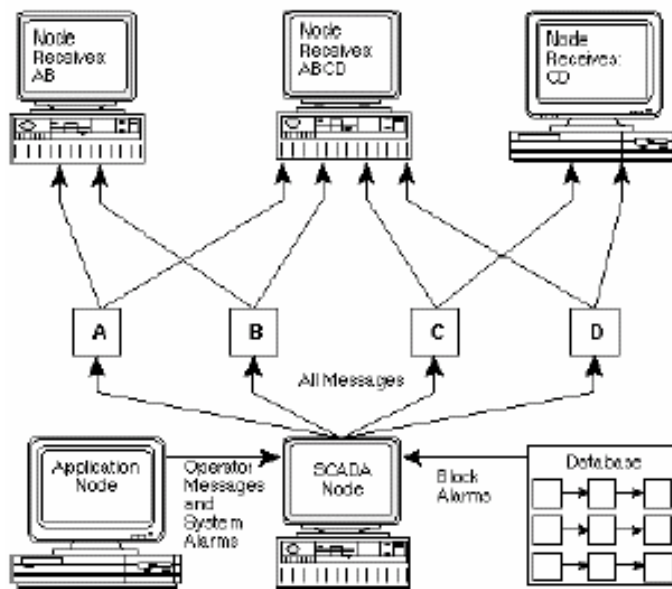
Operatorska poruka – aplikacije generiraju poruke koje kreiraju historiju važnih akcija operatora.

Sistemska poruka – FIX software generira alarmne poruke koje prate upozorenja i probleme u radu sistema upravljanja (računara).

Aplikaciona poruka- aplikacije kao što je graditelj receptata (recipe builder) i historijsko prikupljanje (historical collect), mogu takodjer slati poruke na printere i u fajn historije alarma da obezbjede zapise o aktivnosti u okviru aplikacije.

Rutiranje alarma

Intellution koristi sistem konfiguracije dvosmjernog selektivnog alarma baziranog na konceptu nazvanom alarmne oblasti (alarm areas). 16 alarmnih oblasti označenih od A do P su distribucione tačke za alarme i poruke. Svaki lokalni alarmni zadatak prima alarme iz alarmnih oblasti koje je korisnik specificirao. Slijedeća slika ilustrira alarmne oblasti:.



Alarmne zone

Rutiranje operatorskih i aplikacionih poruka

Operatorske i aplikacione poruke mogu biti nezavisno rutirane (usmjerene) ka alarmnim oblastima. Ova mogućnost dozvoljava korisniku da razdvoji operatorske i aplikacione poruke od ostalih alarma. SCADA čvorovi djeluju kao *alarmni serveri* i distribuiraju alarme i poruke širom mreže. Drugi čvorovi djeluju kao *alarmni klijenti* i primaju alarme. Kada korisnik konfigurira SCADA čvor da distribuira alarme preko mreže, on će slati alarme i poruke u svaki čvor sa kojim uspostavi sesiju.

Ne-SCADA čvor koji generira operatorske poruke i sistemske alarme upućuje ove poruke na svoje odgovarajuće SCADA čvorove.

Sigurnost

Intellution obezbjeđuje moćan i sofisticirani sigurnosni sistem koji omogućava korisniku da zaštiti:

- pristup FIX programima
- pristup kritičnim programskim funkcijama
- pristup fajlovima sa slikama i recepturama
- pravo upisa vrijednosti u blokove baze podataka

Konfiguracija sigurnosti

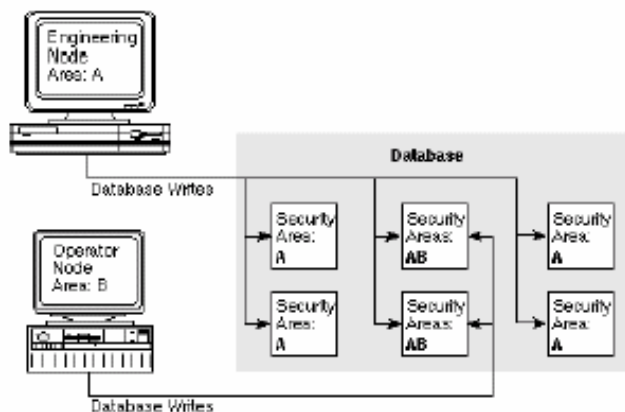
Aplikacija sigurnosti uključuje dva programa: Konfigurator sigurnosti i logiranje (login). Sa programom za konfigurisanje sigurnosti , korisnik može:

- Omogućiti ili onemogućiti sigurnost u čvoru
- Kreira račune za korisnike i grupe korisnika
- Doznačuje korisnička prava da koriste programe i programske funkcije, kao i pravo upisivanja u blokove baze podataka.
- Doznači imena korisnika i lozinke
- Doznačuje imena za sigurnosne oblasti

Sigurnosne oblasti

Da bi se zaštitili blokovi baze podataka od neautorizovanog upisivanja vrijednosti, Intellution koristi koncept oblasti sigurnosti (*security areas*). Korisnik može smatrati ove sigurne oblasti kao grupiranje blokova baze podataka sa istim nivoom sigurnosti. Korisnik može definirati do 254 nivoa oblasti sigurnosti. Operatori koji imaju prava na neku specifičnu oblast sigurnosti mogu pisati u bilo koji blok baze podataka koji je član te oblasti sigurnosti.

Naredna slika ilustrira kako FIX i FIX MMI koriste sigurnosne oblasti:



Sigurnosne oblasti

U primjeru prikazanom na slici, svi blokovi baze podataka pripadaju oblasti sigurnosti **A**. Kod konfigurisanja sigurnosti, potrebno je doznačiti prava IT inženjerima za oblast **A**. Blokovi koji sadrže zadate vrijednosti pripadaju oblasti sigurnosti **B**. Operatori imaju prava za oblast **B** ali ne i za oblast **A**. Zbog toga operatori mogu iz View programa pisati samo i blokove u zoni **B**. Inženjer može da piše u bilo koji blok baze podataka.

Logiranje

Jedanput kada čvor ima sigurnosnu zaštitu, operator mora pristupiti programu logiranja i da unese svoje korisničko ime i lozinku. Nakon logiranja, operator može pristupiti zaštićenim karakteristikama čvora za koje ima pravo pristupa.

Razvoj FIX sistema

Kako razviti specifičan sistem

U narednom poglavlju bit će opisan postupak razvoja sistema automatizacije baziranom na FIX softwareskom paketu.

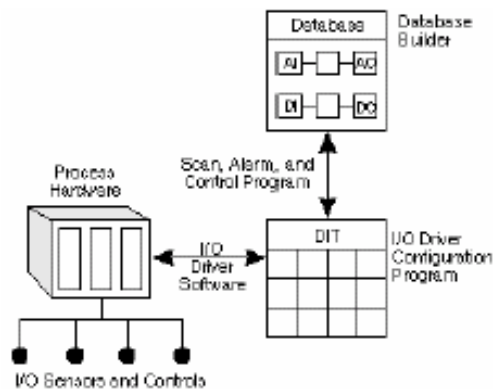
Kao što je već opisano ranije, FIX software ima tri osnovne funkcije:

- akvizicija podataka
- management podataka
- prezentacija podataka

Kao četvrti korak korisnik može razviti i svoje specifične aplikacije. FIX software ima interfejs aplikacionih programa (API) koji obezbjeđuje puni pristup ka internim softwareskim strukturama.

Prikupljanje i management podataka

Bez obzira da li automatiziramo postojeće postrojenje ili gradimo novo, korisnik može naći najviše procesnih podataka koji su potrebni za FIX i FIX MMI na P& I diagramima (process and instrumentation), tj. na dijagramima procesa sa unesenom instrumentacijom. Naredna slika pokazuje ove funkcije akvizicije i managementa podataka:



Akvizicija i management podataka

Korištenje I/O drajvera

Podatci u realnom vremenu koje FIX koristi dolaze iz registara PLC uređaja i distribuiranih I/O modula. P&I dijagrami trebaju sadržavati adrese ovih podataka koji su nam potrebni. Za svaki specifični tip I/O uređaja postoji specifični I/O drajver. I/O drajver može pristupiti podacima samo onog I/O uređaja koji taj drajver podržava. Neki I/O drajveri mogu pristupiti mrežama podržavanog tipa uređaja.

Svaki I/O drajver dolazi sa konfiguracionim programom i specifičnim za taj drajver referentnim priručnikom.

Prvi zadatak korisnika je da koristeći konfiguracion program za dati uređaj da specificira podatke koje će prikupljati.

Nakon završetka konfigurisanja i testiranja softwera I/O drajvera, gradi se tabela prikupljenih podataka u memoriji računara koju smo nazvali DIT tabela (tabela slike drajvera). Pristupajući ovoj DIT tabeli , FIX software može prikazati prikupljene podatke operatoru. Medjutim, prije nego se može pristupiti podacima, FIX software treba lokaciju gdje će pohraniti prikupljene podatke.

Zbog toga slijedeći zadatak korisnika je da izgradi procesnu bazu podataka.

Korištenje graditelja baze podataka (database builder)

FIX aplikacije čitaju i pišu podatke u procesnu bazu podataka. Baza podataka čita i piše informacije u DIT tabelu. Program gradnje baze podataka omogućava korisniku da konstruira mapu podataka dijela procesa sa kojim SCADA čvor interaktira. Baza podataka može :

- prikupiti podatke za prezentiranje podataka u drugoj aplikaciji
- generira izračunate podatke iz prikupljenih podataka
- primjenjuje procesnu kontrolnu logiku na prikupljene podatke da kreira regulacionu konturu
- primi podatke iz aplikacija za eventualno izdavanje na proces

Graditelj baze podataka je korisnikov alat da konfigurira kako da procesira svaku tačku podatka. Korisnik konstruira konture procesiranja, koje smo nazvali lancima (*chains*) povezujući konfigurabilne blokove kontrolne logike.

Korištenje graditelja receptura (recipe builder)

Graditelj receptura je FIX aplikacija koja omogućava korisniku da kreira, upravlja i izvršava recepture u procesu. Ako proces zahtjeva od operatora da često mjenjaju mnogo vrijednosti u procesnoj bazi podataka, onda korisnik može koristiti recepture da automatski realizuje ove promjene.

Graditelj receptura kreira master i kontrolne recepture. Master recepture su uzorci (templates) iz kojih se može kreirati mnogo specifičnih kontrolnih receptura. Kontrolne recepture su proizvodne verzije koje su namjenjene normalnom korištenju. Svaka

receptura može inicirati kompleksne procesne procedure koristeći matematske formule, grupe tagova, i varijable za kontrolu procesa.

Graditelj recepture obezbjedjuje dvije odvojene tabele sa poljima (spreadsheets) , da omoguće modularni sigurni razvoj recepture. Prva tabela obezbjedjuje korisniku koji razvija program pune mogućnosti editiranja za kreiranje receptura, druga tabela (spreadsheet) dozvoljava operatorima da downloaduju recepture.

Koristeći editor grupe tagova

Editor grupe tagova je aplikacija koja pojednostavljuje management slika i receptura. Ova aplikacija obezbjedjuje mehanizam da se pristupi sličnim informacijama baze podataka u različitim trenutcima vremena sa istim simbolom. Koristeći ove simbole, operatori mogu koristiti samo jednu sliku ili recepturu da nadziru dva ili više djelova procesa. Ovi specijalni simboli se kreiraju i doznačuju koristeći editor grupe tagova (tag group editor). Naprimjer, doznačujući različite tagove jednom simbolu, operatori koji nadziru proces mogu prebaciti kontrolne operacije između različitih dijelova procesa, kao naprimjer između tankova, koristeći istu sliku. Slično, jedna receptura može koristiti simbole grupe tagova da inicira slične procesne sekvence na različitim tankovima. Korištenje grupe tagova reducira zahtjev na kapacitet hard diska i pojednostavljuje razvoj procesa i rad između sličnih procesa.

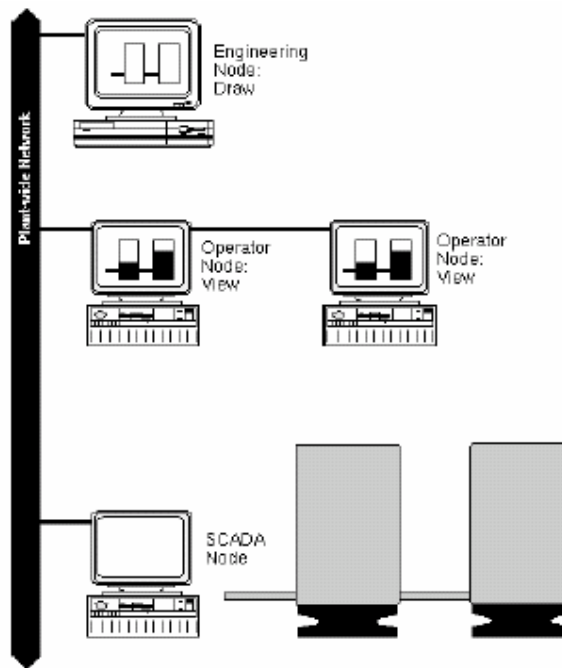
Prezentacija podatka

Jedanput kada imamo aktivnu bazu podataka, korisnik može koristiti druge FIX aplikacije da predstavlja te podatke operatorima. Zadaci predstavljanja podataka se mogu svrstati u tri kategorije:

- konstruiranje interfejsa čovjek- sistem (MMI)
- arhiviranje podataka i pristup pohranjenim podacima
- generiranje izvještaja

Kreiranje operatorskih slika

Grafičke aplikacije se sastoje od Crtaj (Draw) i Gledaj (View) programa. Draw je konfiguracioni program za kreiranje prikaza sa podacima u realnom vremenu, koji se nazivaju slikama (pictures) u nizu lako razumljivih formata. View je program koji prikazuje Draw slike sa podacima u realnom vremenu koji dolaze iz procesne baze podataka. Naredna slika ilustrira funkcije Draw i View programa:



Grafičke aplikacije

Kreiranje skripti komandnog jezika

Komandni jezik je moćni skript alat koji omogućava korisniku da automatizira operatorske zadatke putem serije instrukcija. Skript komandnog jezika pohranjuje ove instrukcije unutar serije komandi i parametara. FIX software izvršava ove instrukcije na zahtjev View programa.

Skripte komandnog jezika mogu biti i vrlo jednostavne ali i vrlo kompleksne prema potrebi korisnika. Mogućnosti komandnog jezika obezbjeđuju široku lepezu mogućnosti da se dodaju specifične funkcionalnosti strategiji upravljanja procesom koju korisnik želi implementirati.

Naprimjer, korisnik može koristiti skripte da:

- manipulira fajlovima
- manipulira alarmima
- kontrolira blokove baze podataka
- automatski izvršava druge aplikacije

Kreiranje Makroa

Makro editor omogućava korisniku da doznači jednostavnu sekvencu tastera makroima koji se grade koristeći komandni jezik. Makro je skript komandnog jezika koji sadrži seriju komandi i instrukcija koje FIX software izvršava po zahtjevu.

Kada korisnik koristi Makro editor da bi izgradio skript komandnog jezika, on definira taj zahtjev kao specijalnu kombinaciju tastera na tastaturi. Skripte komandnog jezika koje su doznačene kombinaciji tastera nazivaju se makroi tastera. Jedanput kada je makro

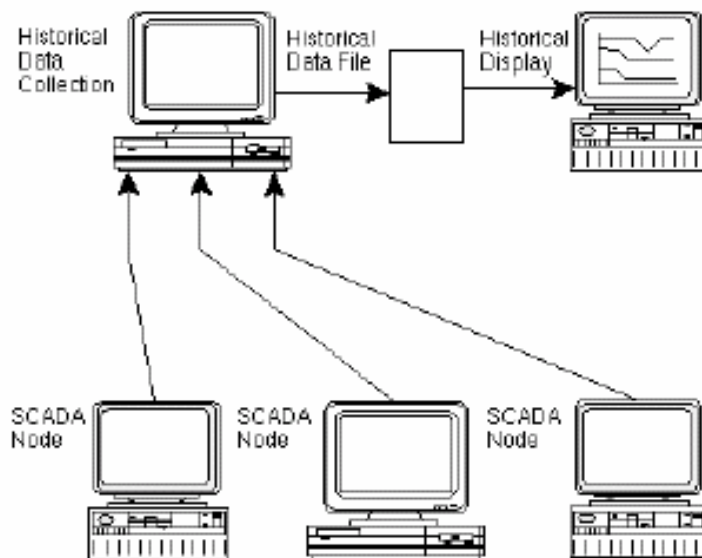
tastera definiran, FIX software izvršava pridruženu skriptu kad god operator izda specificiranu kombinaciju tastera.

FIX i FIX MMI podržavaju tasterske makroe i u Draw i View aplikacijama. Ovat alat omogućava dodavanje specifičnih funkcionalnosti za individualne, višestruke ili pak sve operatorske prikaze.

Skupljanje historijskih podataka

Aplikacija historijskog trendiranja omogućava sampliranje podataka u realnom vremenu sa brzinom uzorkovanja specificiranom od strane operatora. Korisnik može nakon toga koristiti ove podatke za analizu procesa i optimizaciju na dužem vremenskom intervalu. Naredna slika ilustrira funkciju aplikacije historijskog trenda. Program historijskog doznačavanja dozvoljava korisniku da specificira podatke koje želi prikupljati. Program historijskog prikupljanja prikuplja podatke u fajlove koji mogu biti lokalno pohranjeni ili na fajl serveru. Historijski displej omogućava operatoru da pristupi podacima u historijskim fajlovima i prikaže podatke kao trend zapise.

Kao opcija, korisnik može instalirati program historijskog displeja na čvorove nezavisno od programa doznačavanja i prikupljanja. Korisnik može izvršavati ovu opciju na čvorovima da bi prikazali podatke koji se prikupljaju na na drugim čvorovima. Dodatno, opcija historijskog displeja dozvoljava operatorima da integriraju FIX podatke sa laboratoriskim podacima iz ASCII fajlova, kao kao što su Excel spreadsheetovi (tabele).

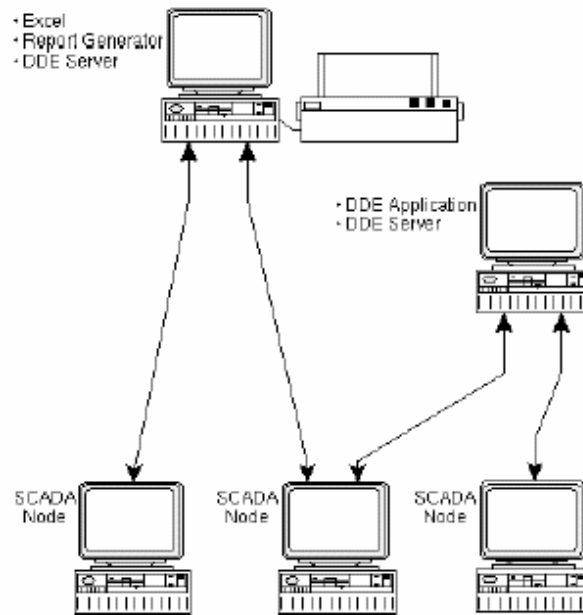


Historijsko trendiranje

Generiranje izvještaja

Većina aplikacija zahtjeva mogućnost periodičnog prikupljanja kritičnih podataka u formatu izvještaja koji inženjeri procesa mogu analizirati. FIX software obezbjeđuje dvostepeni pristup generiranju izvještaja koji se bazira na protokolu standardne razmjene podataka i spreadsheetima .

Korisnik može kreirati izvještaje koristeći Microsoft Excel i kolekciju Intellution obezbjedjenih Makroa. Sa Excelom na raspolaganju su sve snažne mogućnosti računanja i formatiranja koje jedan spreadsheet ima. Naredna slika ilustrira tipični tok podataka izmedju FIX softwarea i drugih DDE (dynamic data exchange – dinamička razmjena podataka) kompatibilnih aplikacija.



Izmjena podataka i generiranje izvještaja

Makroi Generators izvještaja omogućavaju korisniku da:

- pristupi procesnim podacima
- pristupi historijskim podacima
- kreira zapise u realnom vremenu
- raspoređuje u izvještaje za automatsku generaciju

Aplikacija Intellutionovog DDE Servera obezbjedjuje link izmedju Excela i FIX sistema. DDE (dinamička razmjena podataka – dynamic data exchange) je standardni protokol za razmjenu podataka medju Windows aplikacija. Svaka aplikacija koja može čitati podatke putem DDE može takodjer čitati FIX podatke putem DDE Servera.

Pošto Excel podržava DDE, on prenosi zahtjeve putem DDE Servera na FIX software, i FIX software obezbjedjuje podatke realnog vremena ka DDE Serveru.

Implementiranje ODBC SQL interfejsa

Aplikacija ODBC SQL interfejsa u realnom vremenu omogućava korisniku prikupljanje i pisanje procesnih podataka realnog vremena u ODBC relacionu bazu podataka. Ova aplikacija takodjer obezbjedjuje metode za čitanje podataka pohranjenih u relacionu bazu podataka i upisivanje tih podataka natrag u FIX software, omogućavajući na taj način veći nivo kontrole nad procesom.

Kreiranje specifičnih aplikacija (custom application)

Niti jedan software ne može obezbjediti svaku pojedinačnu funkciju koju operator može trebati. Ovakve specifične potrebe traže i specifične aplikacije. Zbog toga FIX software je gradjen sa otvorenom arhitekturom. FIX software obezbjedjuje dva APIja da omogući operatorima pristup svim podacima u sistemu.

- DDE Server
- Lagani pristup bazi podataka

Korištenje DDE Servera

DDE podrška dozvoljava operatorima da unesu podatke u programe koji podržavaju DDE i eksportuju podatke iz drugih aplikacija u FIX aplikacije. Sposobnost da djeluje ili kao klijent ili kao server prema drugim aplikacijama otvara širok opseg aplikacionih scenarija. Operatori su u mogućnosti da iskoriste njihovo znanje drugih programa kontrolirajući FIX aplikacije iz drugih DDE kompatibilnih aplikacija.

Dok je DDE Server veza između Intellutionovog makroa izvještaja i FIX aplikacija, korisnik može koristiti DDE Server i za druge namjene. DDE Server slijedi Microsoftov DDE protokol, tako da bilo koja aplikacija koja može pristupiti podacima kroz DDE može također pristupiti ili generirati FIX podatke kroz DDE Server.

Procesna baza podataka također obezbjedjuje DDE server podršku tako da korisnik može prenjeti informaciju direktno iz DDE aplikacije u FIX bazu podataka. Bilo koji DDE server drugog proizvođača može biti korišten, ili FIX DDE Server također.

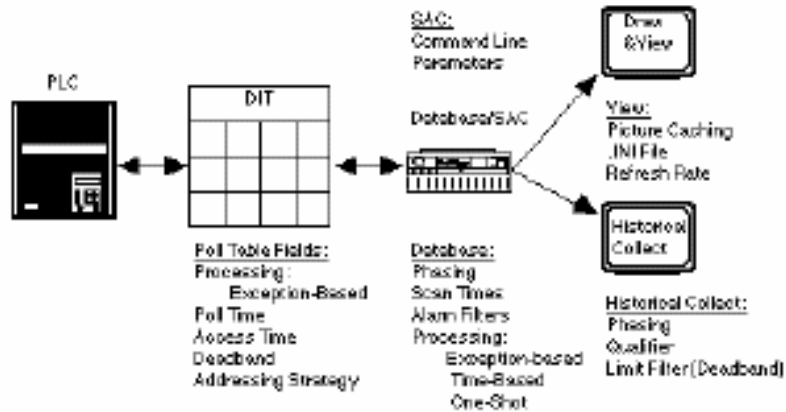
Korištenje laganog pristupa bazi podataka

Lagani pristup bazi podataka (EDA – easy database access) , je aplikacioni interfejsni program C, C++, ili Visual basic (VB) jezičkih funkcija. Funkcije dozvoljavaju operatoru da čita ili piše podatke u FIX bazu podataka. Funkcije mogu biti inkorporirane u specifične programe da omoguće razvojnim IT inženjerima da kreiraju specijalne funkcionalnosti.

Optimizacija sistema

Korisnik može maksimizirati performansu FIX softwarea podešavanjem različitih postavnih vrijednosti. Pažljivo analizirajući ukupnu arhitekturu sistema, i znajući kako podaci teku od jedne tačke do druge, korisnik može uočiti neefikasnosti i nepotrebna dupliranja u sistemu da bi rekonfigurirao sistem za optimalnu performansu. Pri tome treba imati u vidu da je specifična za svaku instalaciju, tako da treba analizirati svaki čvor posebno da bi mogao da prilagodi ono što je optimalno za svaki čvor.

Kao što je pokazano na narednoj slici postoji pet strateških zona sa kontrolom toka podataka koje korisnik može modificirati da bi poboljšao ukupnu performansu sistema. Treba koristiti ovu sliku kao referentnu da mu pomogne da identificira i rješava svaki zahtjev za povećanje performansi sistema.



Upravljanje optimizacijom toka podataka

Ovih pet strateških zona može biti referencirano u skladu sa njihovim primjenama. One uključuju:

- konfiguraciju zapisa izbora (poll record)
- bazu podataka
- SAC
- View
- Historijsko prikupljanje

Karakteristike FIX 7.0 Scada paketa

Fix 7 ima kapacitet prikupljanja do 20.000 tagova sa do 255 historijskih kolekcionih grupa.

Mogućnost konfigurisanja sigurnosti u sistemu je povećana opcijom da se koristi Windows NT autentikacija sigurnosti. Moguće je specificirati WINNT ime korisnika i lozinku kao login ime i lozinka i za FIX software. Ova mogućnost nam omogućava da koristimo postojeće NT korisničke račune kod logiranja u FIX, sinhronizirajući korisničke račune i pojednostavljujući siguronosnu konfiguraciju i održavanje.

Nadalje, koristeći WINNT sigurnost, sinhronizujući račune korisnika sa WINNT sistemom sigurnosti, dobijamo:

- lozinke koje koriste velika i mala slova (case –sensitive)
- lozinke čije trajanje je ograničeno (expire)
- online promjena lozinki

Pregled Windows NT sistema sigurnosti

Nakon sinhroniziranja WINNT i FIX korisničkih računa, operator se može logirati u FIX unoseći njegovo ili njeno WINNT login ime i lozinku. FIX šalje ovu informaciju, zajedno sa imenom domena specificiranim u Operatorovom FIX računu ka WINNT domen kontroleru za verifikaciju (authentication). Ako WINNT verificira korisničko ime i lozinku, FIX kompletira login proces. U suprotnom, sistem sigurnosti logira grešku.

Promjena vrijednosti u bazi podataka djeluje na sličan način. Kada operator unese promjenu vrijednosti u bazi podataka, pojaviće se slijedeći koraci:

[1] FIX šalje zahtjevanu promjenu , zajedno sa imenom operatora kod logiranja i njegovom lozinkom , ka SCADA serveru na kojem je rezidentna baza podataka.

[2] Nakon prijema zahtjeva, SCADA server traži ime korisnika u sigurnosnoj stazi za dati čvor (node's security path)

[3] Ako korisnički račun postoji i konfigurisan je da koristi Windows NT, SCADA server šalje operatorov login ime i lozinku ka Windows NT domen kontroleru na verifikaciju. Ako su login ime i lozinka netačni, SCADA odbacuje promjenu.

[4] Ako su korisničko ime i lozinka korektni, SCADA server ispituje zonu sigurnosti bloka baze podataka koji se mjenja. Ako operator ima prava u ovoj zoni sigurnosti, SCADA server prihvata zahtjevanu promjenu podataka. U suprotnom , promjena se odbacuje.

Zadatak (task) pregleda

Moguće je sinhronizirati Windows NT i FIX korisnički račun putem:

[1] Kreirajući Windows NT korisnički račun na kontroleru domena. Ne treba kreirati nikakve lokalne račune da bi se obezbjedio sigurnosni okružaj.

[2] Konfigurisati svaki Windows NT račun sa potrebnim pravima koristeći korisnički manager program (User Manager).

[3] Konfigurisući FIX korisnički račun omogućavajući Windows NT sigurnosnu opciju i unoseću Windows NT korisničko ime i ime o domenu u FIX korisnički račun. Korisničko ime i ime domena se moraju poklapati sa sa imenima koje koristi Windows NT korisnički račun.

Korištenje imena domena

Da bi se korektno sinhronizovala korisnička imena u Windows NT i FIX-u, potrebno je konfigurisati korisničke račune u Windows NT na slijedeći način:

[1] Izabrati korisnički Manager iz programs na slijedeći način:

[a] Izabrati programe iz **Start** menija

[b] Izabrati Adinistrativne alate (**Administrative tools**) u **Programs** podmeniju.

[c] izabrati **User manager**

[2] Izabrati **User Rights** iz **Policies** menija.

[3] Kliknuti na **Show Advanced user Rights** ček boks i selektirati **Act as Part of the Operating System** iz **Rights list**.

[4] Kliknuti na **Add**

[5] Izabrati domen koji sadrži korisničke račune koje želimo sinhronizirati iz **List Names** u okviru list boksa i kliknuti na **Show Users**.

[6] Skrolirati niz listu imena u list boks i dvaput kliknuti na korisničke račune koje želimo modificirati.

[7] Izlogirati se (log off) a zatim se ponovo prijaviti (login) tako da se promjene mogu provesti.

Konfigurisanje samostalnog čvora (stand-alone node).

Tipično, mi obezbeđujemo sve čvorove na mreži da spriječimo neautorizovane pristupe fajlovima, aplikacijama i procesnim bazama podataka. Medjutim, moguće je takodjer obezbjediti samostalne čvorove kreirajući i sinhronizujući lokalni Windows NT i FIX korisničke račune specifično za lokalni čvor.

Da bi se sinhronizovali korisnički računi:

[1] Izabrati **User Manager** iz **Programs** direktorija na slijedeći način:

[a] Izabrati **Programs** iz **Start** menija

[b] Izabrati **Administrative Tools** iz **Programs** podmenija

[c] Izabrati **User Manager** iz **Administrative Tools** ponmenija

[2] Izabrati **User Rights** iz **Policies** menija

[3] Kliknuti na **Show Advanced User Rights** ček boks i izabrati **Act as Part of the Operating System** iz **Rights** liste.

[4] Kliknuti na **Add**

[5] Izabrati ime lokalnog računara iz **List Names** iz list boksa i klik na **Show Users**.

[6] Skrolirati kroz imena i dva puta kliknuti na račune korisnika koje želimo da modificiramo.

[7] Izlogirati se iz Windows NT i logirati se u ponovno da bi se izvršene promjene mogle registrirati.

Podrška za Klijent-Server OPC

OPC (Object linking and embedding -OLE for Process Control) , tj Microsoft mehanizam implementiran u Windows Operativnim sistemima za razmjenu podataka izmedju aplikacija , definira standardne objekte , metode, i osobine za zadovoljenje zahtjeva interoperabilnosti tj. da mogu medjusobno komunicirati sa mnogo različitih tipova hardwarea koristeći isti OPC klijent software, u mnogim real-time procesnim aplikacijama.

OPC klijent djeluje takodjer kao drajver, i to omogućava FIX procesnoj bazi podataka da dobije podatke od bilo kojeg OPC servera. Ova mogućnost dozvoljava FIXu da komunicira sa bilo kojim tipom procesnog hardwarea za koji postoji OPC server i čini korisnikov izbor hardwarea sa kojim će raditi skoro neograničenim.

Intellution OPC EDA Server opcija pretvara svaku FIX 7 procesnu bazu podataka u OPC server. Ova opcija omogućava svakoj OPC klijent aplikaciji rezidentnoj na OPC EDA SCADA serveru pristup podatcima u FIX procesnoj bazi podataka i obezbjeđuje laganu vezu različitih aplikacija , pojednostavljujući integraciju sistema i omogućavajući laganu ekspanziju sistema.

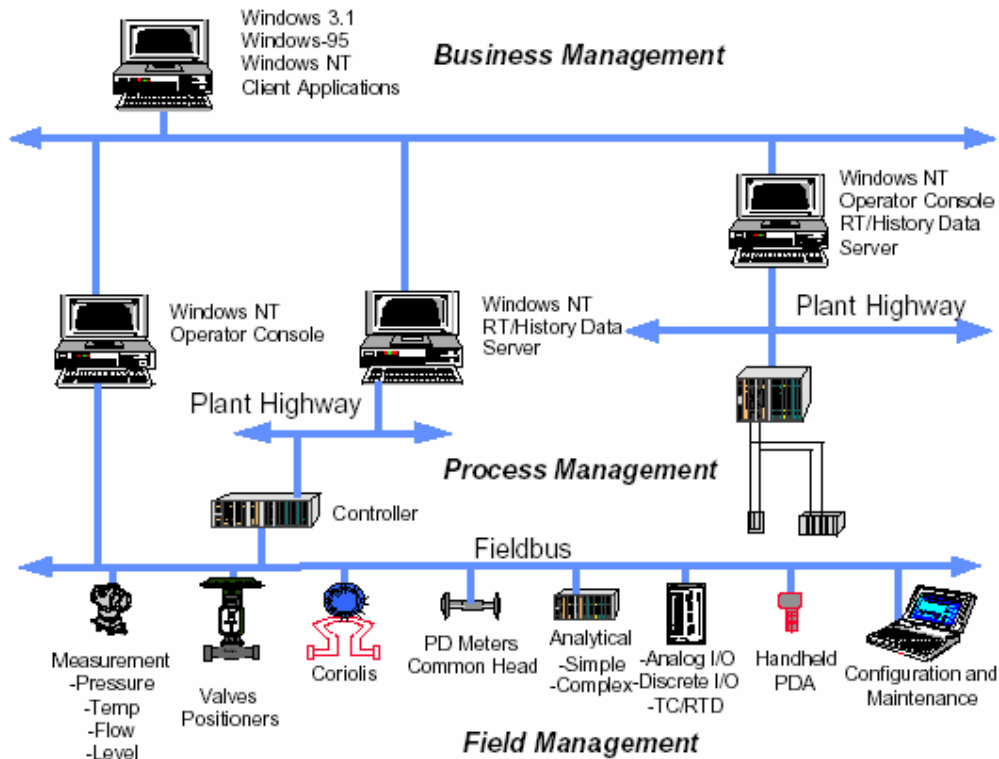
OPC - STANDARD ZA RAZMJENU PODATAKA UNUTAR SISTEMA UPRAVLJANJA I VODJENJA PROCESA

Rijetko kada bilo je toliko mnogo uzbuđenja u oblasti računarskog vođenja u upravljanja procesima kao kada se pojavio OPC kao komunikacioni standard.

OPC je akronim za “ **OLE for Process Control**” , tj. Microsoft Windows OLE (object linking and embedding) tehnologija (kasnije evoluirala u ActiveX), prilagodjena za procesno vođenje i nadzor.

Standardni mehanizam komunikacije medju brojnim izvorima podataka, bilo uradjaja rasutih diljem proizvodnih pogona jedne fabrike, ili bazama podataka u računarima u kontrolnoj sobi ili u ofisima je osnovna motivacija za OPC.

Informaciona arhitektura u procesnoj industriji pokazana je na slijedećoj slici i uključuje slijedeće nivoe:



Arhitektura informacionog sistema upravljanja

Management u postrojenju (field management)

Sa uvođenjem inteligentnih (smart) uređaja u postrojenjima kao dijelova sistema vođenja nadzora i upravljanja (inteligentni transmiteri, aktuatori, itd), pojavljuje se obilje informacija o tim uređajima i o postrojenju gdje su ugradjeni, koji nisu prije bili raspoloživi. Ove informacije obezbjeđuju podatke o zdravlju uređaja , njegovim konfiguracionim parametrima, konstrukcionim materijalima, itd. Sve ove informacije moraju se prikazati i korisniku, kao i bilo kojoj aplikaciji koja ih koristi, na konzistentan način.

Management procesa

Instaliranje distribuiranih sistema upravljanja (DCS) i SCADA sistema da nadziru i upravljaju procesima čine ove podatke raspoložive i u elektronskoj formi, za razliku od ranijih sistema kada su mnogi od njih bile ručno prikupljeni i zapisivani.

Poslovni management

Beneficije u poslovnom vođenju sistema su ogromne nakon instaliranja savremenog sistema upravljanja. Ovo se postiže integracijom informacija prikupljenih iz procesa u poslovne sisteme koji upravljaju finansijskim aspektima proizvodnih procesa.

Obezbjeđivanje ovih informacija na konzistentan način klijent aplikacijama , minimizira napor koji je potrebno uložiti da se obezbjeđi ova integracija.

Da bi se efektivno realizovali ovi ciljevi, proizvođači trebaju biti u stanju da koriste gotove (off the shelf) alate kao što su SCADA paketi, baze podataka, spreadsheetovi, itd. Ključ ovoga je u otvorenoj i efikasnoj komunikacionoj arhitekturi koja se koncentrira na pristup podacima, a ne na tipovima podataka.

Da bi ovi sistemi radili zajedno morali su se riješiti mnogi problemi koji su bili ozbiljniji nego što je to problem povezivanja mreža, rad sa različitim operativnim sistemima i povezivanje "open systems" koji su se u praksi pokazali da nisu baš tako otvoreni i koji su trebali svojom otvorenom arhitekturom da olakšaju to povezivanje.

Osnovni razlog zašto su računarski sistemi vođenja i nadzora procesa imali i svoje dodatne specifične teškoće je u tome da interfejsi između procesnih uređaja i računarskih sistema nisu bili standardizirani.

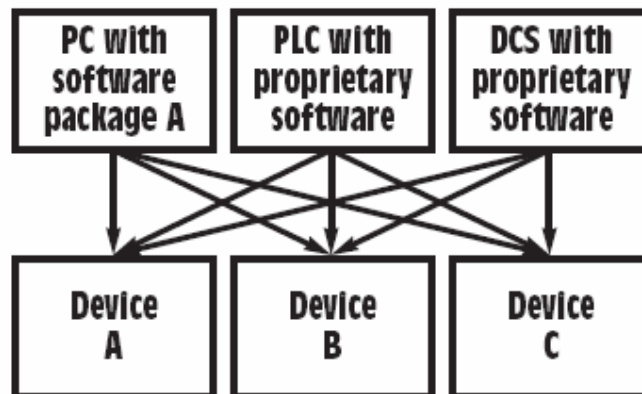
Specifični za svakog proizvođača (proprietary) sistemi koji međusobno ne komuniciraju su bili standardna praksa sve do nedavno u mnogim industrijskim postrojenjima.

Usljed odsustva standarda u ovoj oblasti , proizvođači (Vendors) su razvili specifična hardwaresko softwareska rješenja. Zbog toga mnogi sistemi procesnog i poslovnog upravljanja i vođenja su imali svoje specifične tehnike, interfejse, API module , da bi se pristupalo informacijama u takvim sistemima.

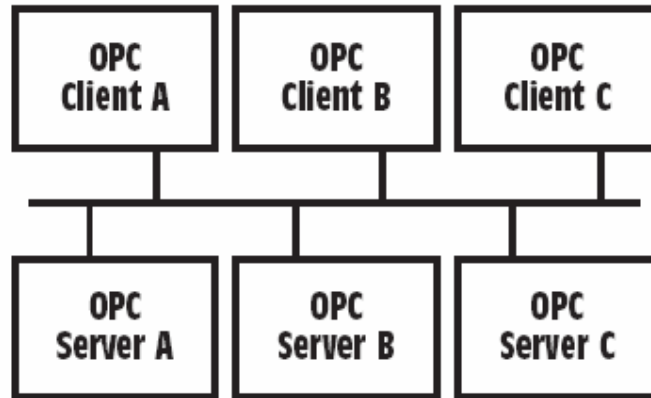
Cijena integracije takvih sistema kao i troškovi softwareskog održavanja i računarska podrška su bili značajni.

Pojavljivanje drajvera za svaki specifični tip hardwarea i njihove izmjene pri svakoj modifikaciji hardwarea bile su glavobolja za sve integratore i održavaoce ovakvih sistema.

U takvim okolnostima , za tipičan sistem procesnog i poslovnog nadzora i vođenja, 25-30% vremena IT inženjera je trošeno na razvoj i debugiranje drajverskih modula softwarea. Ovo stanje je ilustrirano i na slijedećoj slici :



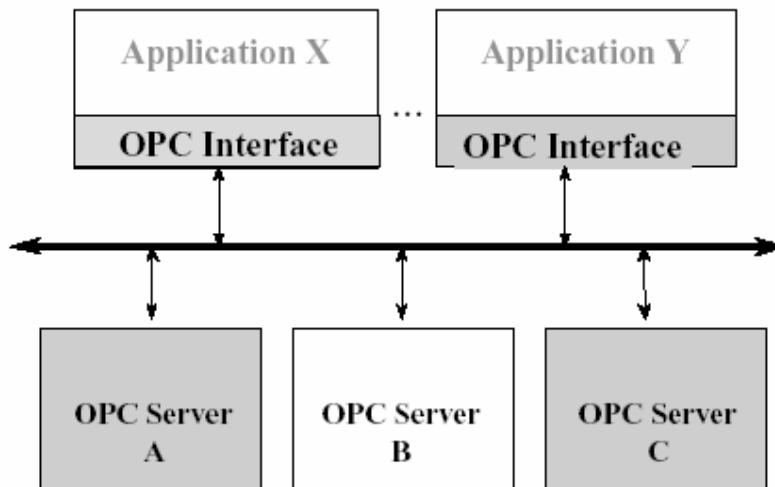
Rješenje za sve ovo je bio da se razvije standard koji će obezbjediti istinsku "plug-and-play" softwaresku tehnologiju za sisteme upravljanja i vođenja procesa, gdje će svaki sistem, svaki uređaj i svaki drajver moći se slobodno povezati i komunicirati međusobno. Ova ideja je ilustrirana na narednoj slici :



Namjena

Ono što je potrebno je zajednički način pristupa podacima od strane aplikacija ka bilo kojem izvoru podataka , bilo da je to neki uređaj ili baza podataka.

Slijedeća slika pokazuje ovu ideju :



Aplikacije koje rade sa mnogo OPC Servera

OPC Server na ovoj slici i u slijedećim poglavljima je korišten kao sinonim za bilo koji server koji obezbjedjuje OPC interfejse , kao napr.

- OPC Server za pristup podacima (OPC DataAccess Server)

- OPC Server za Alarme i događaje (OPC Alarm&Event Server)
- OPC Server za historijske podatke (OPC HistoricalData Server)

Ranija Arhitektura klijent aplikacija

Postoje mnoge klijent aplikacije koje su bile razvijene i koje zahtjevaju podatke iz izvora podataka i pristupaju tim podacima nezavisno, razvijajući "drajvere" za svoje potrebe.

Ovo je vodilo ka nizu problema , kao :

- **Dupliranje napora** od mnogo nezavisnih proizvođača softwarea, jer je svako za svoj software pisao svoj drajver za specifični tip hardwarea.
- **Nekonzistentnost između drajvera** različitih Vendra (softwareskih kuća)
- **Promjena u hardwareu** je vodila najčešće do toga da se morao pisati novi drajverski software
- **Konflikti pristupa** Dva softwarea nisu mogli istovremeno pristupiti nekom hardwareu pošto su obadva sadržavala nezavisne drajvere.

Proizvođači hardwarea su pokušavali da riješe ove probleme razvijajući drajvere za svoj hardware, ali su bili suočeni sa problemom različitih Klijent protokola.

OLE (Object Linking and Embedding) za upravljanje procesima (**OPC**), je povukao ovu liniju između proizvođača hardwarea i onih koji su razvijali software.

OLE je bio kasnije restrukturiran od objektno orijentirane na objektno baziranu arhitekturu i Microsoft mu je promjenio naziv u **ActiveX**.

OPC obezbjeđuje mehanizam da osigura podatke iz izvora podataka i prenosi ih ka bilo kojem klijentu na standardan način.

Proizvođač hardwarea je mogao sada da razvije višekoristivi , visoko optimiziran Server, koji komunicira sa izvorom podataka, i održava mehanizam da efikasno pristupi podacima iz uređaja – izvora podataka.

Dakle, ako obezbjedimo Server sa OPC interfejsom, onda će bilo koji klijent moći pristupiti tom uređaju i njegovim podacima.

Kastomizirana aplikaciona arhitektura

Sve veći broj aplikacija se razvija u okružajima kao što su : VB i VBA (Visual basic for application), Delphi, Power Builder, Visual C ++. OPC takodjer mora da uzme u obzir ovaj trend u obzir. Microsoft je uočio ovaj trend i dizajnirao je OLE/COM da bi omogućio komponentama (pisanim u C i C ++) da se mogu koristiti od strane korisnikovog prilagodjenog (kastomiziranog) programa (pisanog u VB, VBA ili Delphi).

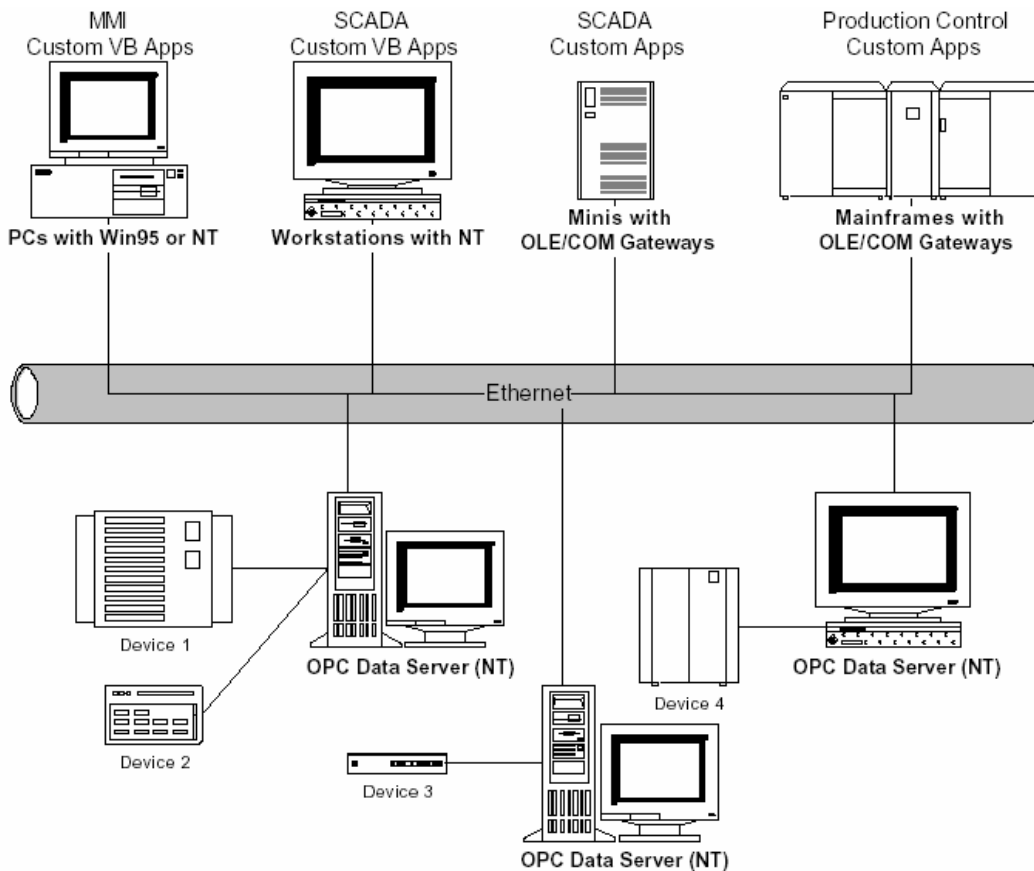
Dakle namjena svih standardnih specifikacija da olakšaju razvoj OPC Servera u C i C ++ , a da olakšaju razvoj OPC klijent aplikacija pisanim u bilo kojem jeziku koji korisnik izabere.

OPŠTE

OPC je dizajniran da omogući klijentskim aplikacijama pristup procesnim podacima na konzistentan način. OPC pruža slijedeće prednosti:

- Proizvođači hardwarea treba samo da urade jedan set softwareskih komponenti koje će korisnici koristiti u njihovim aplikacijama.
- Proizvođači softwareskih programa neće morati da uvijek ponovo pišu drajvere zato što je došlo do promjena u novoj verziji hardwarea.
- Korisnici će imati na raspolaganju više opcija hardwarea kod systemske integracije svojih sistema.

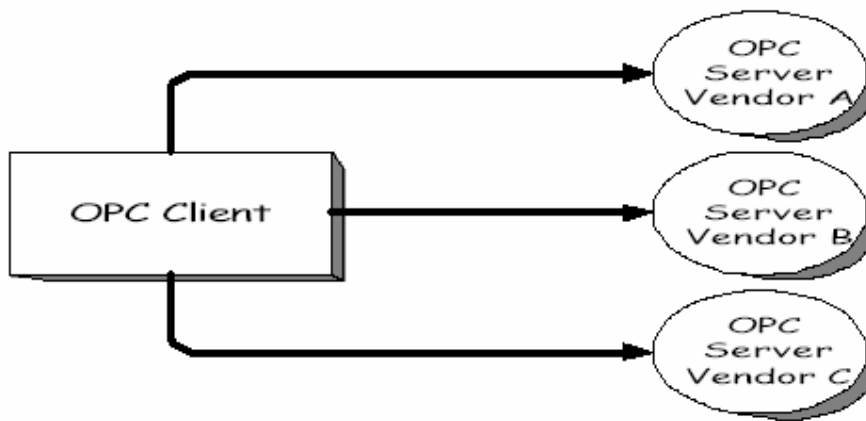
Sa OPC , integracija sistema u okružaju heterogenog kompjuterskog hardwarea će postati jednostavnija.



Heterogeni kompjuterski okružaj

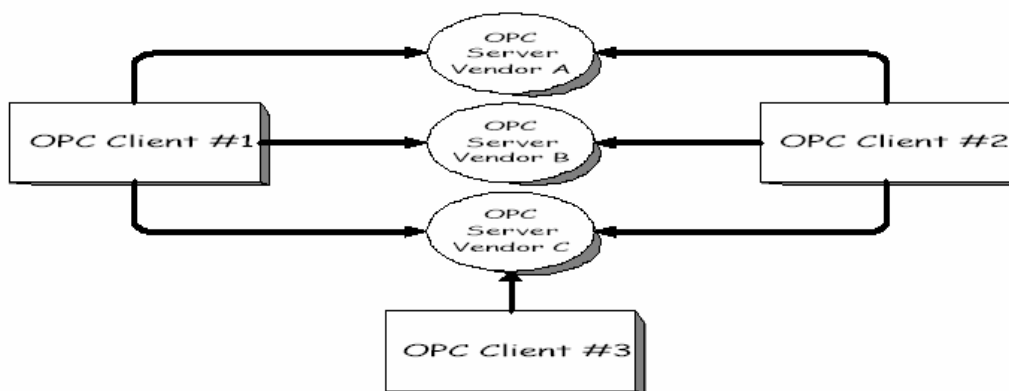
OPC objekti i interfejsi

Daćemo opis OPC COM objekata i njihovog interfejsa implementiranog sa OPC Serverima. OPC klijent se može povezati sa OPC Serverima različitih proizvođača.



OPC Klijent

OPC Serveri mogu biti napisani od različitih proizvođača hardwarea. Kod kojeg isporučuje Vendor (proizvođač hardwarea), određuje uređaje i podatke do kojih Server ima pristup, kao i detalje kako Server fizički pristupa tim podacima.



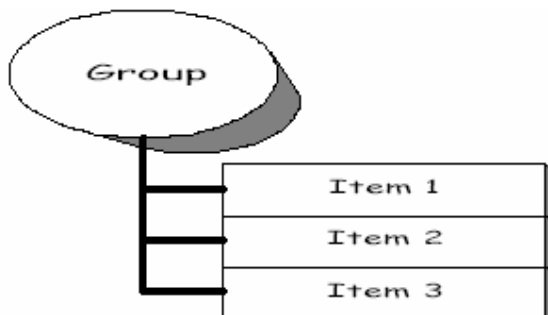
Relacija između OPC Klijenta i Servera

Pregled OPC DataAccess funkcije

Na višem nivou, OPC DataAccess Server se sastoji od nekoliko objekata: , servera, grupe i detalja (item). Objekt OPC Server sadrži informaciju o serveru i služi kao kontejner za OPC grupne objekte. OPC group objekt sadrži informaciju o samom sebi i obezbeđuje mehanizam za držanje i logičko organizovanje OPC itema.

OPC grupe obezbjedjuju način za klijente da organiziraju podatke. Naprimjer, grupa može predstavljati detalje (items) u nekom datom operatorskom prikazu ili izvještaju. Podatci mogu biti čitani ili upisivani. Konekcije bazirane na izuzeću (exception based), mogu se takodjer kreirati izmedju klijenta i detalja u grupi i mogu biti omogućene ili onemogućene. OPC klijent može konfigurirati brzinu sa kojom OPC Server treba obezbjedjivati promjene u podacima za OPC klijenta.

Postoji dva tipa grupa, javne i lokalne (ili privatne). Javna (public) grupa se koristi ako se dijeli izmedju više klijenata, lokalna grupa je lokalna i samo za jednog klijenta. Unutar svake Grupe , klijent može definirati jedan ili više OPC detalja (items).



Relacija izmedju Grupe i detalja

OPC detalji predstavljaju konekcije sa izvorima podataka unutar servera. OPC detalj, sa perspektive korisničkog interfejsa, nije pristupačan kao objekat od strane OPC klijenta. Zbog toga, nema vanjskih interfejsa definiranih za OPC Item. Svi pristupi ka OPC Itemima su preko OPC Group objekta, koji "sadrži" OPC item, ili jednostavno tamo gdje je OPC Item definiran.

Pridružena svakom Itemu je vrijednost (Value), kvalitet (quality) i vremenski otisak (Stamp) -[V,Q,T]. Vrijednost je u formi VARIANT, a kvalitet je sličnog tipa kao kako je ona specificirana i za Fieldbus (procesni bus koji povezuje u mrežu uređaje u postrojenju).

Primjetimo da Itemi nisu izvori podataka, oni su samo konekcije sa njima. Naprimjer, tagovi u DCS sistemu postoje bez obzira da li OPC klijent im pristupa. Prema tome o OPC Itemu trebamo razmišljati kao samo o specifičeru adrese podatka, a ne kao stvarnom izvoru podatka, koji je referenciran adresom u Itemu.

Pregled OPC Alarm i Event handlera

Ovi interfejsi obezbjedjuju mehanizme za OPC klijente da budu izvješteni o pojavljivanju specificiranih događaja i alarmnih uslova. Oni takodjer obezbjedjuju servise koji omogućavaju OPC klijentima da odrede događaje i uslove koji su podržani od strane OPC Servera, i da dobiju njihov tekući status.

U okviru OPC, *alarm* je nenormalno stanje (*condition*) i zbog toga je specijalan slučaj stanja. Stanje (*condition*) je stanje (named state) u OPC Event Serveru, ili jednom od njegovih sadržavajućih objekata, koji je od interesa za OPC klijente. Naprimjer, tag FC101 može imati slijedeće uslove pridružene sa njim:

visoki alarm (high alarm), vrlo visoki alarm (highhigh alarm), niski alarm (low alarm), i vrlo niski alarm (lowlow alarm).

Sa druge strane, događaj (event), je pojava koja je značajna za OPC Server, za uređaj koji predstavlja, i za njegove OPC klijente. Događaj (event) može ali i ne mora biti udružen sa uslovom. Naprimjer, prelasci u alarmna i normalna stanja su događaji koji su udruženi sa stanjima. Sa druge strane, promjene u konfiguraciji sistema, i sistemske greške su događaji koji nisu vezani za specifične događaje. OPC klijenti mogu se prijaviti da budu obavješteni od strane Servera o pojavi specificiranih događaja.

IOPCEventServer interfejs obezbjedjuje metode koji omogućavaju OPC klijentu da:

- odredi tip događaja koje OPC Server podržava
- Unese pretplate (subscription) klijenata na specifične događaje, tako da OPC klijenti mogu primiti obavjesti kada se oni dese. Mogu se koristiti da se definišu podskupovi željenih događaja.
- Uslovi pristupa i manipulacije koji su implementirani od strane OPC Servera

Pored **IOPCEventServer** interfejsa, OPC Event Server može podržavati opcione interfejse za uslove browsinga koji su implementirani od strane servera i za upravljanje javnim grupama.

Pregled OPC historijskog pristupa podacima (OPC Historical Data Access)

.Mašine za prikupljanje historiskih podataka su dodatni izvor informacija koji moraju biti distribuirani korisnicima i softwareskim klijentima koji su zainteresirani za ovu informaciju. Ranije, većina historiskih sistema je koristila njihove vlastite interfejse za prikupljanje ovih podataka.

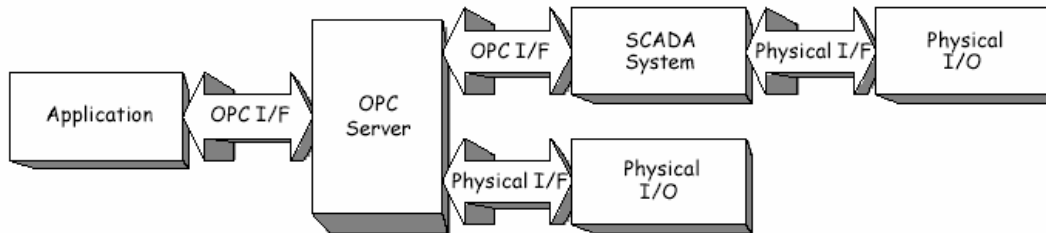
U cilju integracije podataka na svim nivoima poslovanja, historiska informacija može također biti smatrana kao još jedan tip podataka.

Postoji nekoliko tipova servera historijskih podataka (Historian server). Neki od najvažnijih tipova koji su podržani ovom specifikacijom su:

- Jednostavni serveri trend podataka. Ovi serveri obezbjedjuju u suštini samo pohranjivanje sirovih podataka. Podatci su tipično onog tipa koji je raspoloživ od OPC Data Access servera, obično u formi trojke (triple) koja uključuje : [Time , Value & Quality], tj. Vrijeme, vrijednost i kvalitet podatka. [V,Q,T].
- Serveri sa kompleksnom kompresijom podataka i analizom. Ovi serveri obezbjedjuju kompresiju podataka kao i pohranjivanje sirovih podataka. Oni su u stanju da obezbjede sumarne podatke ili funkcije analize podataka, kao što su : srednje vrijednosti, minimumi, maksimumi, itd. Oni mogu podržati ažuriranje podataka kao i historiju ovih ažuriranja. Također mogu podržavati i komentare zajedno sa historijskim podacima.

Gdje se OPC uklapa

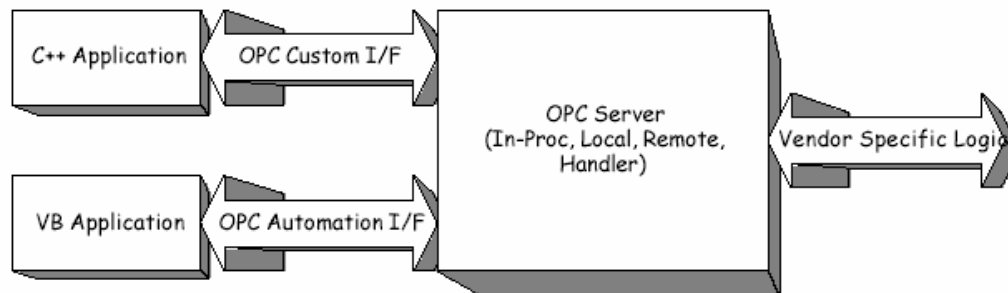
Mada je OPC primarno dizajniran za pristup podacima sa mrežnog servera, OPC interfejsi se mogu koristiti na mnogim mjestima unutar aplikacije. Na najnižem nivou oni mogu dobijati sirove podatke od fizičkih uređaja u SCADA ili DCS, ili iz SCADA ili DCS sistema u aplikaciju. Arhitektura i dizajn čine mogućim da se konstruiše OPC Server koji dozvoljava klijentu aplikacije da pristupi podacima iz mnogih OPC Servera koje obezbjeđuju razni proizvođači hardwarea priključeni na različitim čvorovima , putem jednog objekta.



OPC relacija Klijent-Server.

Opšta OPC arhitektura i komponente

OPC specifikacije uvijek sadrže dva seta interfejsa: Prilagodjeni (Custom) interfejsi i interfejsi nazvani Atomation I/F (automatizirajući interfejs) . Ovo je pokazano na slijedećoj slici:



OPC interfejsi

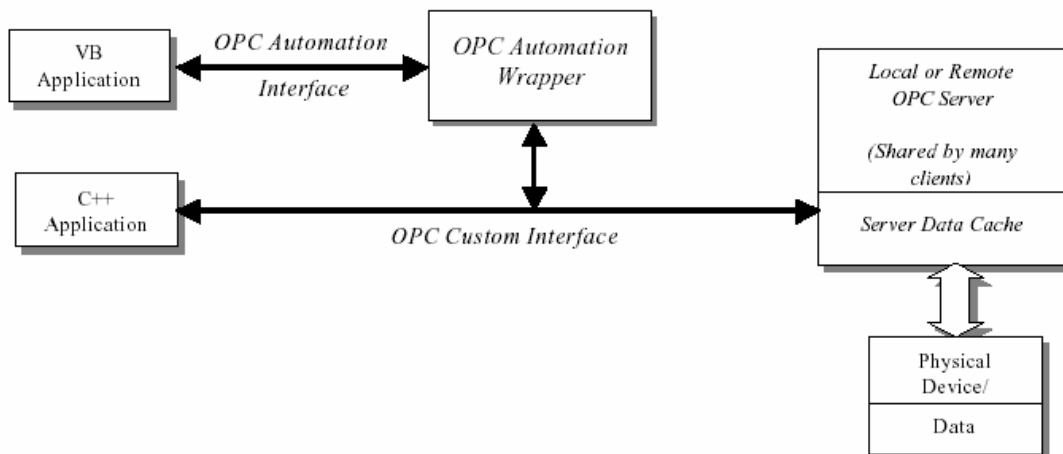
OPC specifikacija specificira COM interfejse (tj. šta su interfejsi i koje funkcije ostvaruju) , a ne njihovu implementaciju.

Dakle, ona specificira ponašanje koje se očekuje od interfejsa da obezbjede klijent aplikacije koje ih koriste.

Kao i COM implementacije, arhitektura OPC je klijent-server model, gdje OPC Server komponenta obezbjeđuje interfejs ka OPC objektima i upravlja sa njima.

Postoji nekoliko razmatranja u implementaciji OPC Servera. Glavno pitanje je frekvencija prenosa podataka preko nedjeljivih komunikacionih staza ka fizičkim uređajima ili drugim basevima podataka. Dakle, mi očekujemo da OPC serveri će biti ili lokalni ili daljinski EXE fajlovi koji uključuju kôd koji je odgovoran za efikasno prikupljanje podataka od fizikalnog uređaja ili baze podataka.

OPC klijent aplikacija komunicira sa OPC serverom preko specifičnih custom i automatizacionih interfejsa. OPC serveri moraju implementirati **custom** interfejse i opciono mogu implementirati **automatizacione** interfejse. U nekim slučajevima OPC Fondacija (tijelo koje promovira OPC standardizaciju), može obezbjediti omotnicu (wrapper) za standardni automatizacioni interfejs . Ovaj "wrapperDLL" može biti korišten za bilo koji Vendor specifični server, kao na slici :



Tipična OPC arhitektura

Lokalni i daljinski serveri

OPC Server Vendori imaju jedan od dva pristupa umreženju svojih proizvoda:

- Klijent treba uvijek da se konektira sa lokalnim serverom koji će onda koristiti već postojeće mehanizme za pristup mreži (proprietary). Ovaj pristup se koristi obično od Vendra koji dodaju OPC mogućnost već postojećem distribuiranom proizvodu kao što je bio FIX kod Intellutiona.
- Oni mogu indicirati da se klijent konektira na željeni server na ciljnom čvoru i koriste DCOM arhitekturu da obezbjede ovo umreženje.

OPC Server Browser

Interfejs za OPC Server browser (**IOPCServerList**) je specificiran kao dio **OPCCommon** dokumenta

Postavka problema realizacije ovog browsera

Problem koji se postavlja je "kako da klijent program pokaže korisniku koji OPC Serveri su na raspolaganju na datoj mašini?". OPC Serveri se registruju u sistemu preko kategorija komponenti (Component Categories).

Ovo dozvoljava Microsoftovom ICatInformation (IID_IcatInformation) Interface na StdComponentCategoriesMgr (CLSID_StdComponentCategoriesMgr), da se koristi da odredi koji OPC serveri su instalirani na lokalnoj mašini.

Problem je da ovaj mehanizam ne radi za udaljene mašine pošto Component Categories Manager je DLL i IcatInformation interfejs radi samo "in-process", tj na istoj mašini.

Kao rezultat toga, ne postoji način za klijenta da dobije listu OPC Servera instaliranih na udaljenim mašinama.

Pregled mogućeg rješenja

Server Browser OPCENUM.EXE kojeg isporučuje OPC fondacija može biti rezidentan na bilo kojoj mašini i on će pristupiti lokalnom manageru Component kategorija (Component Categories Manager), i obezbjedjuje novi interfejs IOPCServerList koji se može distribuirati (marshaling) i biti korišten od strane udaljenih klijenata.

Ovaj server ima publikovani classid , i može se instalirati po jednaput na svakoj mašini koja ugošćuje (hosting) OPC servere.

Klijent još uvijek treba da zna ime noda ciljne mašine na kojoj traži OPC servere, međutim on može sada kreirati ovaj objekat daljinski i koristiti njegov IOPCServerList interfejs da odredi koji tipovi i vrste servera su na raspolaganju na toj mašini.

Sumarni rezime

Baziran na Microsoftovoj OLE (sada ActiveX), COM (component object model) i DCOM (distributed component object model) tehnologijama, OPC se sastoji od standardnog seta interfejsa, osobina, i metoda za korištenje u sistemima vođenja i nadzora procesnih i proizvodnih sistema, i povezivanje ovih sa poslovnim sistemima.

ActiveX/ COM tehnologije definišu kako individualne softwareske komponente mogu interaktivirati i dijeliti podatke izmedju sebe. Bazirane na Microsoftovoj NT tehnologiji, OPC obezbjedjuje zajednički interfejs za komuniciranje izmedju raznorodnih procesnih uređaja i I/O podsistema, bez obzira koji operativni sistemi i MMI software koriste.

Kao što je već napomenuto cilj standarda je bio koncept "plug- and-play", koncept koji je razvio Microsoft i niz drugih kompanija. Koristeći standardni način konfiguriranja hardwarea i softwareskog interfejsa, uređaj će se lako povezivati sa ostalim djelom sistema i trenutno nakon toga raditi, bez dugotrajne instalacione procedure i kompleksnog konfiguriranja.

Korisnik, umjesto da uči 100 i više alata za povezivanje od raznih proizvođača, sada treba da nauči samo jedan set alata, jer će svi OPC drajveri raditi na isti način.

OPC standard zahtjeva od proizvođača hardwarea da obezbjede ovu kolekciju i distribuciju podatka iz svog hardwarea. Oni su zasigurno najbližiji hardwareu i znaju kako interno pristupiti podacima. Ovi uređaji sa OPC drajverom tada postaju OPC serveri, osiguravajući OPC klijent aplikacijama podatke na konzistentan i standardiziran način. IT inženjeri koji razvijaju aplikacije mogu to sada raditi u bilo kojem programskom jeziku koji im odgovara.

Šta je COM?

Component object model osigurava standardne interfejsne i medjekomponentne komunikacije. Putem COM, jedna aplikacija može koristiti osobine bilo kojeg drugog aplikacionog objekta ili operativnog sistema, ili da omogući da se ta softverska komponenta poboljša (upgrade), bez da to utiče na rad ukupnog sistema i rješenja.

COM se može koristiti od strane razvojnih IT inženjera i integratora sistema, da se kreira prilagodjena (customized) rješenja.

COM je binarni i generički standard, i nalazi se kao jezgro (core) i u DCOM, ActiveX i OLE tehnologiji.

Šta je OLE?

Objektno povezivanje i ugradjivanje (object linking and embedding), se koristi da obezbjedi integraciju medju aplikacijama, omogućujući visok nivo aplikacione kompatibilnosti, čak i izmedju različitih tipova informacija. OLE tehnologija je bazirana na COM, i omogućava razvoj višekoristivih (reusable) plug-and-play objekata koji su medjusobno operativni čak i u okviru višestrukih aplikacija. OLE takodjer obezbjedjuje višekoristivi razvoj softwarea baziran na komponentama, pri čemu se softverske komponente mogu pisati u bilo kojem jeziku, i isporučenom od bilo kojeg proizvođača softwarea.

Šta je OLE automatizacija?

OLE automatizacija i u njoj uključene COM tehnologije su razvijene od strane Microsofta da omoguće komponentama (pisanim u C i C++), da budu korištene od strane prilagodjenog programa (pisanog u napr. VB ili Delphi). Ovaj model obezbjedjuje precizno zadovoljenje potreba upravljanja i vođenja u procesnoj industriji, gdje firme koje razvijaju hardware pišu softverske komponente u C i C++, za omogućavanje pristupa podacima unutar uređaja. Kroz OPC, IT inženjeri koji razvijaju aplikacije mogu pisati kod u bilo kojem jeziku da zahtjevaju i koriste ove podatke.

Šta je DCOM?

Distribuirani komponentni objektni model proširuje COM na mreže (tj. daljinske objekte). Riječ je o novom vrlo optimiziranom protokolu, gdje se daljinske komponente pojavljuju kao da su lokalne. DCOM je prvo bio realizovan za Windows NT 4.0 u 1996 godini. Microsoft Java i VB Script podržavaju razvoj DCOM i ActiveX.

Šta je ActiveX?

ActiveX je kišobran (omotnica) za širok opseg tehnologija koje su ranije nazivane OLE Controls, a sve se baziraju na COM. Preimenovanje i restrukturiranje OLE Controls tehnologije je dovelo do toga da je postala objekt bazirana a ne objekt orijentirana.

ActiveX je otvorena, integrirana platforma koja omogućava IT razvojnim inženjerima na Web stranicama da kreiraju portabilne aplikacije i interaktivne sadržaje za WWW aplikacije. Riječ je o otvorenoj kros platformi, i podržavanoj na Mac, Windows i Unix sistemima.

Sumirajući gore iznjeto možemo reći da Microsoftov COM je softwareska arhitektura koja dozvoljava da se grade aplikacije iz binarnih softwareskih komponenata. COM je bazna arhitektura koja omogućava nadgradnju više nivovskih softwareskih servisa kao što su oni koje obezbjeđuje ActiveX.

ActiveX i COM dozvoljavaju aplikacijama da dijele "objekte", kao naprimjer spreadsheetove koji su uloženi (embedded) u dokumente za procesiranje teksta. Kada se pojave nove ažurirane vrijednosti u spreadsheetovima, ActiveX i COM će se pobrinuti da se ovi ažurirani podatci pojave i u tekstu dokumenta gdje su uloženi.

Pristup podacima je omogućen svakom u hijerarhiji

Još jedna bitna prednost primjene OPC standarda je da je omogućen pristup procesnim podacima na svakom nivou u firmi. Procesni podatci nisu više ograničeni samo na Operatore i procesne inženjere i managere procesa.

Pristup kroz VB aplikacije koje koriste OPC Data Access Specifikaciju dozvoljava IT inženjerima da pišu aplikacije koje koriste procesne podatke u poslovnim aplikacijama.

Nadalje, ove aplikacije mogu biti pisane sa malo ili nikakvog znanja o industrijskoj mreži i njenoj konfiguraciji kroz koju se prenose podatci iz procesa.

Široka primjena i usvajanje OPC standarda rezultira u većem dijeljenju informacija između više aplikacija koje se simultano odvijaju. Kako je sve više aplikacija OPC omogućeno (OPC enabled), ista informacija se može distribuirati ka mnogim aplikacijama kao što je : održavanje, skladište i nabavka rezervnih dijelova, operatorski displeji, i menagment dokumenata, korištenjem kombinacije OPC i DCOMa , tako da se tehnološki procesi i poslovni procesi mogu simultano koordinirati.

Lakoća korištenja – Autokonfiguracija tagova

Efektivno dizajnirane OPC komponente se takodjer vrlo lako koriste, i zahtjevaju vrlo malo konfiguracije. OPC serveri ne zahtjevaju od korisnika da konfigurira tagove. Server može automatizirati ovo konfiguriranje, tako da je instaliranje OPC automatizovano rješenje.

Dodavanje i brisanje procesnih tačaka bez zaustavljanja sistema

Novo procesne tačke se mogu dodati ili izbrisati bez da se server mora zaustaviti. Ovo je značajno superiornije u odnosu na mnoge vlastite firmenske proizvode koji zahtijevaju da se drajver zaustavi prije nego što se nove procesne tačke mogu dodati. Primjer korištenja ove sposobnosti je naprimjer : tačke se mogu dodati putem interfejsa baze podataka, definišući tačke konzistentne sa sintaksom servera. Podatci će se nakon toga početi trenutno pojavljivati u bazi sa OPC servera.

Sinhrono i asinhrono upisivanje na uređaj

Sinhrono ili asinhrono upisivanje na uređaj , sa potvrđivanjem (acknowledgement), je superiorna karakteristika u odnosu na ranije DDE drajvere, što je uvijek predstavljalo veliki problem IT inženjerima koji su razvijali aplikaciju. Tako naprimjer kod nekih DDE drajvera, aplikacija je pokušavala da upiše podatak na PLC uređaj. Međutim, prije nego što bi vrijednost i dospjela do PLC-ja, vrijednost bi već bila prepisana od strane drajvera u polling ciklusu.

Sa uvodjenjem DCOM, kako OPC rješava problem kada se udajeni server isključi

DCOM obezbjeđuje ugradjenu osobinu koja osigurava da OPC klijenti i serveri imaju pouzdan mehanizam da razmjenjuju podatke u realnom vremenu kroz mrežu. DCOM takodjer upravlja sa ponovnim pokušajima (retries) i isteklim vremenima (time-outs) izmedju OPC klijenta i udaljenog OPC servera , i pokušava da ponovno uspostavi izgubljen komunikaciju.

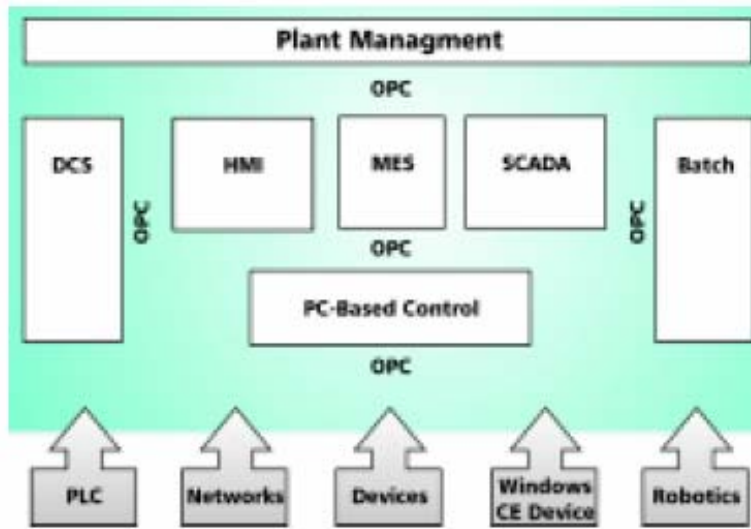
Jedna od strogih osobina OPC je da on naslijedjuje sve standardne softwareske tehnologije kao što su: ActiveX, DCOM i WindowsNT. DCOM tehnologija čini distribuirani klijent/server mrežni protokol transparentnim za OPC aplikaciju. DCOM može slati OPC poruke koristeći niz transportnih protokola kao što su: UDP, TCP/IP, IPX, koristeći istu OPC aplikaciju koja koristi DCOM.

OPC i XML

OPC je važna komponenta Microsoft Distributed Network Architecture for Manufacturing (DNA-M). [Microsoftova Internet arhitektura za proizvodnju]. Ova arhitektura je bazirana na različitim MS proizvodima (tj. Operativnim sistemima, aplikacionim serverima, programskim okružajima), i tehnologijama (DCOM; XML, Internet), koje su dizajnirane da olakšaju integraciju aplikacija automatizacije : tj. ERP (Enterprise resource planning), DCS, SCADA, HMI, MES (Management engineering systems), PC bazirano upravljanje.

Jedna od ovih tehnologija je i BizTalk , MS programski okružaj za razmjenu podataka izmedju aplikacija koje su bazirane na XML (eXtensible Markup Language, jezik za opisivanje strukture podataka), šemi i na postojećim ili budućim industrijskim standardima za razmjenu podataka (vidjeti MS Biztalk Server 2002). Korištenje BizTalka pojednostavljuje integraciju izmedju aplikacija. Sa ovom tehnologijom, nije tako važno

da se aplikacije usaglase na konvencijama pozivanja i da ih implementiraju, nego razmjena podataka se dešava pomoću samoobjašnjivih dokumenata.



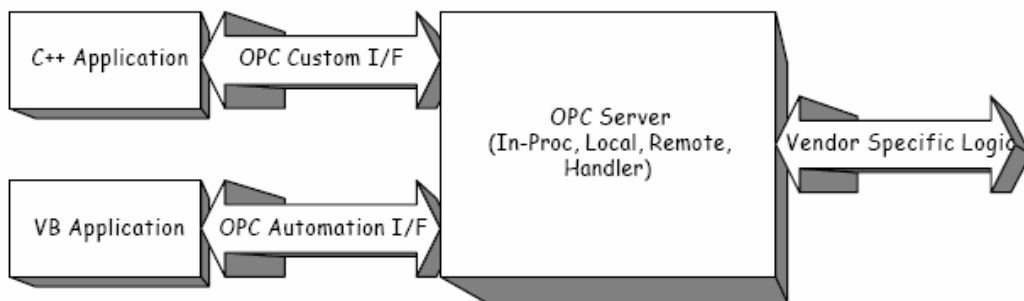
Integraciona arhitektura DNA-M

Ovo predstavlja novi izazov za OPC standard. Integracija sa BizTalkom će dovesti do porasta važnosti OPC u DNA-M, pošto će biti moguće koristiti OPC i van oblasti I/O interfejsa između procesnih uređaja i računara.

OSNOVE OPC DIZAJNA

Definicije interfejsa

OPC specifikacije uvijek sadrže dva skupa interfejsa : Prilagodjeni (custom) interfejs i automatizacioni interfejs. Ovo je pokazano i na slijedećoj slici:



OPC intefejsi

OPC klijent aplikacija komunicira sa OPC serverom kroz specifične prilagodjene i automatizacione interfejse. OPC serveri **moraju** implementirati kastomizirani interfejs i mogu opciono implementirati automatizacioni interfejs. U nekim slučajevima OPC

Fondacija obezbjeđuje standardni automatizacioni interfejsni omotač (wrapper). Ovaj "wrapperDLL" se može koristiti za bilo koji Vendor specifični server.

OPC klijent komunicira sa OPC serverom pozivajući funkcije iz OPC **zahtjevanog** (custom) interfejsa.

Proizvođači hardwarea koji za njega razvijaju OPC server mogu implementirati i funkcije **opcionog** (izbornog) interfejsa. Kada OPC server podržava opcioni interfejs, sve funkcije unutar tog interfejsa moraju biti implementirane, čak i u slučaju kada će funkcija samo vratiti **E_NOTIMPL**. OPC klijent koji želi da koristi funkcionalnosti opcionog interfejsa će pitati OPC servera za opcioni interfejs. OPC klijent ipak mora biti tako dizajniran da ne zahtjeva da ovaj opcioni interfejs i mora postojati kod OPC servera.

Općenito, OPC klijent programi koji se kreiraju koristeći skript bazirane programske jezike će koristiti automatizacioni interfejs. Klijent programi koji su kreirani u C++ će lakše koristiti kastomizirane interfejse za postizanje najbolje performanse.

OPC serveri moraju implementirati kastomizirani interfejs i opciono mogu implementirati automatizacioni interfejs.

Vlasništvo nad memorijom

Prema COM specifikaciji, klijenti moraju osloboditi svu memoriju pridruženu sa '**out**' ili '**in/out**' parametrima. Ovo uključuje i memoriju na koju su usmjereni pointeri elemenata unutar bilo koje strukture. Ovo je vrlo važno da shvate IT inženjeri koji pišu OPC klijent software, inače iskusiće curenje memorije (memory leakage) koje će biti teško otkriti. Preporučeni način da se ovo realizuje je da klijent kreira subrutinu koja će se koristiti za korektno oslobađanje memorije iz svakog tipa strukture.

Nezavisno od uspjeha ili neuspjeha poziva od strane klijenta ka serveru, server mora uvijek vratiti dobro definirane vrijednosti za 'out' (izlazne) parametre. Oslobađanje alociranih resursa za ove 'out' parametre je odgovornost klijenta.

Opaska : Ako je rezultat greške bilo koja *FAILED* greška kao naprimjer *E_OUT OF MEMORY*, OPC server treba vratiti *NULL* za sve 'out' pointere (ovo je standardno ponašanje COM modula).

Greške i povratni kodovi

OPC specifikacije opisuju interfejse i odgovarajuće ponašanje koje OPC server mora da implementira i od kojih OPC klijent aplikacija zavisi. Lista grešaka i povratnih kodova je sadržana u svakoj od tri OPC specifikacije. Za svaki opisani metod , lista svih mogućih kodova grešaka je uključena , zajedno sa najčešćim OLE kodovima grešaka.

U svakom slučaju 'E' kodovi grešaka (koje počinju sa slovom E) , će indicirati FAILED tip grešaka a 'S' kodovi grešaka će indicirati barem djelomični uspjeh.

Isključenje (Shutdown) OPC servera

Mogućnost da OPC server obavjesti klijente o svom padu je realizovana kroz zahtjev OPC servera da se svi klijenti odspoje od servera. Ova mogućnost je obezbjedjena za sve tipove OPC servera (DataAccess, Alarm&Event, Historical).

Funkcionalnost je raspoloživa putem konekcionne tačke na Server objektu i korespondirajuće tome na strani klijenta je **IOPCShutdown** interfejs.

Klijenti trebaju koristiti ovu funkcionalnost da bi mogli ostvariti na kontrolirani način prestanak dolaska podataka sa servera koji je u shutdownu.

***IconnectionPointContainer* (na OPC serveru)**

Ovaj interfejs obezbjedjuje pristup ka konekcionoj tački za IOPCShutdown.

Klijent koji je povezan sa više OPC servera (naprimjer DataAccess , Alarm&Event server, od jednog ili više Vendora), treba da održava posebne shutdown callbacks za svaki objekat pošto svaki OPC server pojedinačno može da ode u shutdown.

IOPCCCommon

Ovaj interfejs se koristi od strane svih OPC Server tipova (DataAccess, Alarm&Event, Historical Data). On obezbjedjuje mogućnost da se postavi i pošalje upit za LocaleID , koji će se koristiti za datu klijent/server sesiju.

To znači da akcije jednog klijenta ne utiču na bilo koje druge klijente.

Kao što je to slučaj i sa drugim interfejsima kao što je Iunknown, instanca ovog interfejsa za svaki server je jedinstvena. Dakle, OPC DataAccess server object i OPC Alarms&Events server object mogu obadva osigurati po jednu implementaciju od IOPCCCommon. Klijent koji održava konekcije sa obadva servera će, kao i u slučaju bilo kojeg drugog interfejsa, koristiti interfejs za ova dva objekta nezavisno jedno od drugoga.

Pitanja instalacije i registracije OPC server softwarea

Svaki proizvođač OPC server softwarea (Vendor) obezbjedjuje SETUP.EXE , da bi se instalirale potrebne komponente za njihove servere. Druga pitanja koja se tiču OLE softwarea je management Windows Registry i Component Categories.

Pitanja su:

- koji unosi trebaju biti uradjeni
- kako se oni mogu realizovati

Component Categories

Sa ogromnom količinom komponenti na svakom posebnom kompjuterskom sistemu, njihov management postaje sve teži i teži. OPC klijenti često trebaju da enumeriraju OPC servere koje žele da koriste u datom kontekstu. U svojoj prvoj verziji OPC je specificirao pod-ključ (sub-key), koji se zvao OPC , da bi tagirao sve OPC Server ulaze u registre. Klijenti su trebali da browse-ju tražeći ovaj ključ.

Ovaj metod se pokazao kao neefikasan jer je zahtijevao da se browsuju svi CLSID ulazi u registre. Mogli su se pojaviti problemi sa kolizijom imena. Konačno pristup udaljenim registrima je ograničen u NT5.0 arhitekturi (WIN 2000).

Za sve serverske aplikacije poslije DataAccess 1.0A, koriste se Component Categories kao način kategorizacije OPC Servera po njihovoj implementiranoj funkcionalnosti.

Klijenti mogu koristiti novi interfejs IOPCServerList da bi dobili listu servera sa zahtjevanom funkcionalnošću.

OPC definira " implemented categories" za svaku verziju svake OPC Interface specifikacije.

Očekuje se da će server prvo kreirati svaku kategoriju koju koristi i nakon toga će se registrirati za tu kategoriju. Pogledati IcatRegister dokumentaciju za dodatne informacije. Jedan server može pripadati u više od jedne kategorije, naprimjer može podržavati DataAccess Verzije 1.0A i 2.0 i još Alarm&Event handling.

Registracija Component Categories

Za vrijeme procesa registracije, svaki OPC Server mora se registrovati sa Managerom Component Categories, koji je COM objekat koji isporučuje MS. OPC klijenti će slati upite Components Category Manageru da nabroji (enumerate) CLSID svih registriranih OPC servera.

Registracija Servera

Da bi se registrirao sa Component Categories Managerom, server treba prvo registrirati OPC definiranu Category ID (CATID) i OPC definiranu Category Description , pozivajući

```
IcatRegister :: RegisterCategories()
```

a zatim registrirajući svoj vlastiti CLSID kao implementaciju CATID sa pozivom ka

```
IcatRegister::RegisterClassImplCategories()
```

Da bi se dobio interfejs pointer na IcatRegister, treba pozvati CoCreateInstance(), kao što je to pokazano u narednom promjeru iz Alarm&Event Servera :

```
#include <comcat.h>
```

```
CoCreateInstance(CLSID_StdComponentCategoriesMgr, NULL,  
CLSCTX_INPROC_SERVER, IID_IcatRegister, (void**)&pcr);
```

Prebrojavanje klijenta (Client Enumeration)

Klijenti će koristiti interfejs IOPCServerList da dobiju listu servera bilo lokalno ili na udaljenom hostu. Ovaj interfejs u osnovi obezbjeđuje funkcionalnost Component

Categories Managera. Definiran je u standardu OPC, pošto pristup Component Categories Manageru nije moguć na udaljenim mašinama.

Unosi u registre za Proxy/Stub DLL

Proxy/stob DLL-ovi se koriste za organizovaje interfejsa ka LOCAL i REMOTE serverima. Gernerira se direktno iz IDL koda i treba biti isti za svaki OPC server. Općenito, Proxy/Stub koristi samo registraciju.

Testirani i gotovi proxy/stub DLL je na raspolaganju na OPC Foundation Web site-u tako da nema potrebe da Vendor razvija ovaj DLL.

Kreiranje ulaza u registre (Registry Entries)

COM definira mehanizam samoregistracije koji omogućava da se uključe (enkapsuliraju) potrebe za unose u registre u DLL ili EXE, i na taj način obezbeđujući za klijente i servere lagani način da se svaki modul u potpunosti i korektno registruje.

Nadalje, COM takodjer uključuje deregistraciju , tako da server može da ukloni sve svoje registrirajuće unose , kada se DLL ili EXE otkloni iz fajl sistema, tako da se registri očiste od beskorisnih unosa.

Kada se zatraži od servera da se samoregistruje, server mora kreirati sve unose za svaku komponentu koju podržava, uključujući sve ulaze za tipske biblioteke. Kada se zatraži od njega da se deregistruje ("un-register") , server mora ukloniti one unose koje je kreirao kod samoregistracije.

Za DLL server, ovi zahtjevi se realizuju putem call-ova ka izvezenim funkcijama *DllRegisterserver* i *DllUnregisterServer*, koji moraju egzistirati unutar DLL pod tačno ovim imenima. Obadvije funkcije se pozivaju bez argumenata i vraćaju HRESULT da indiciraju rezultat.

Dva koda grešaka ako postoje su :

SELFREG_E_CLASS (neuspjeh da registruje/deregistruje CLSID informaciju) i
SELFREG_E_TYPELIB (neuspjeh da registruje/deregistruje TypeLib informaciju).

Ako je server pakovan u EXE modul, tada aplikacija koja želi da registruje server, lansira EXE server sa komandnim argumentom na liniji : /RegServer ili -RegServer (neosjetljiv na velika/mala slova).

Ako aplikacija želi da deregistruje server, lansiraće EXE fajl sa argumentom na komandnoj liniji /UnregServer ili -UnregServer. Samoregistrirajući EXE detektuje ove argumente na komandnoj liniji i poziva iste operacije kao što bi i DLL unutar **DllRegisterServer** i **DllUnregisterServer**, registrirajući svoju stazu pod LocalServer32 umjesto InprocServer32 ili InprocHandler32.

Server mora registrirati punu stazu do instalacione lokacije od DLL ili EXE modula, za njihove odgovarajuće InprocServer32 , InprocHandler32, i LocalServer32 ključeve u registrima.

Staza modula (module path) se može lako dobiti pomoću Win32 API funkcije GetModuleFileName.

Opaska: Server ne treba da registruje proxy/stub interfejsse. Oni treba da budu registrirani od strane proxy/stub DLL kako je to već ranije pomenuto.

Registri ulazi za proxy interfejsse mogu se lagano generirati kada se kompilira proxy dll. Jednostavno treba definirati konstantu REGISTER_PROXY_DLL za vrijeme kompilacije, i izvesti DllRegisterServer i DllUnregisterServer za vrijeme linkovanja.

Sada je moguće unjeti podatke u registre izvršavajući regsvr32 i prenoseći proxy.dll ime kao argumenat.

OPC DATA ACCESS SPECIFIKACIJE

OPC Data Access specifikacije sadrže informacije za dizajn:

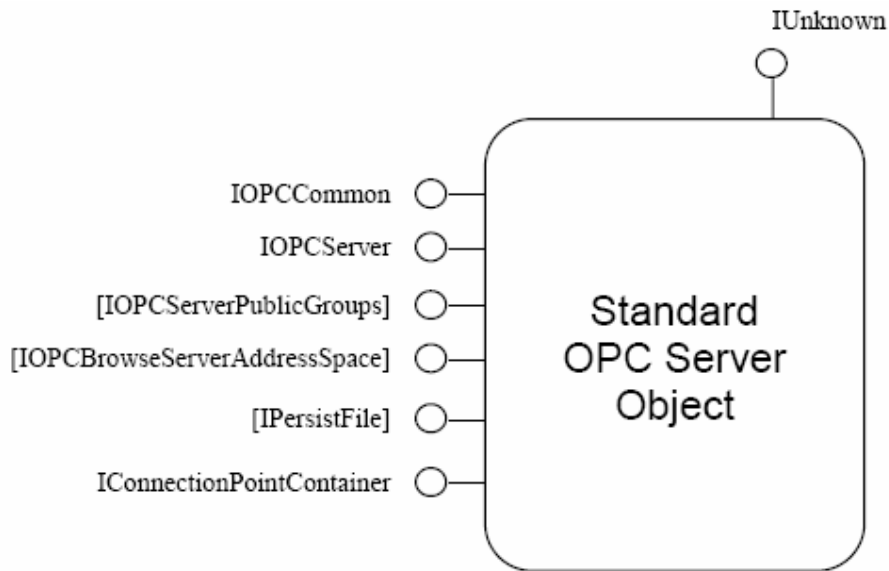
- 1. OPC DataAccess Custom Interface** - Ovaj dokument opisuje Interfejsse i Metode za OPC komponente i objekte (Components and objects).
- 2. OPC data Access Automation Interface** – poseban dokument (Verzija 2.02) koji opisuje OPC automatizacioni interfejs koji olakšava korištenje Visual basica, Delphija i drugih automation enabled softwareskih proizvoda da se interfejsiraju sa OPC Serverima.

SPECIFIKACIJE ZA OPC DATAACCESS CUSTOM INTERFACE STANDARD VERZIJA 2.04

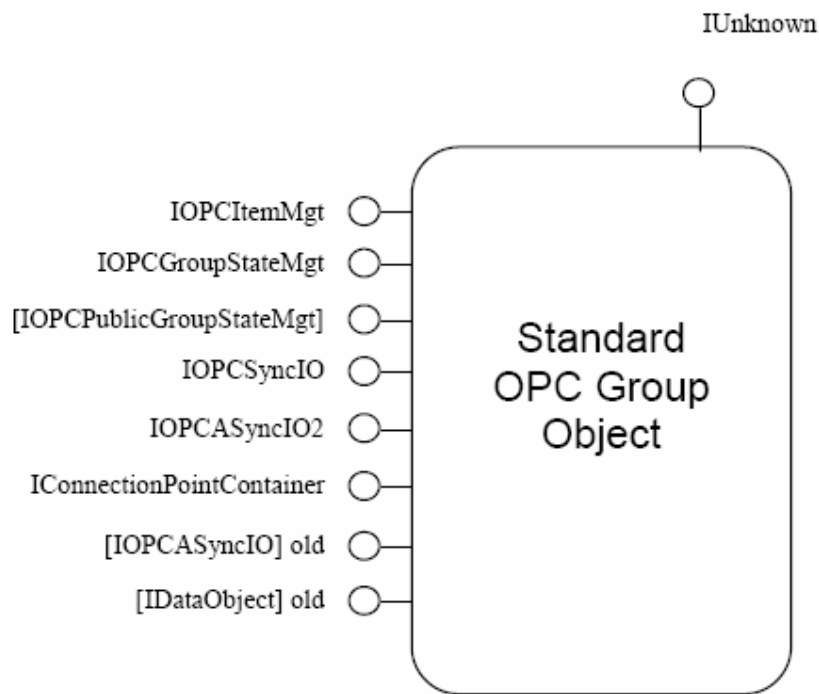
OPC Server objekat obezbjedjuje način da se pristupi (read/write) ili komunicira sa skupom izvora podataka. Tipovi izvora koji su na raspolaganju su funkcija načina implementacije servera.

OPC klijent spaja se sa OPC serverom i komunicira sa njim putem interfejsa. OPC server objekt osigurava funkcionalnost OPC klijentu da kreira i manipulira OPC group objektima. Ove grupe dozvoljavaju klijentima da organiziraju podatke kojima žele pristupiti. Grupa se može aktivirati i deaktivirati kao jedinica. Grupa takodjer obezbjedjuje način za klijenta da se 'pretplati' (subscribe) na listu itema (detalja) tako da može biti obaviješten o njihovoj promjeni.

Opaska: Svim COM objektima se pristupa kroz interfejsse. Klijent vidi samo interfejsse, Dakle objekti koji su opisani u ovom standardu su 'logičke' predstave koje ne moraju imati ništa zajedničkog sa stvarnom internom implementacijom servera. Naredna slika daje sumarnu predstavu OPC objekata i njihovih interfejsa. Neki od interfejsa su opcioni i označeni su u uglastim zagradama [].



Standardni OPC Server objekat



Standardni OPC Group objekat

OPCServer objekt je primarni objekt kojeg OPC server izlaže. Interfejsi koje ovaj objekt osigurava su:

- Iunknown

- IOPCServer
- IOPCServerPublicGroups (opciono)
- IOPCBrowseServerAddressSpace (opciono)
- IpersistFile (opciono)
- IOPCItemProperties
- IconnectioPointContainer

IOPCServer je glavni interfejs za OPC server. OPC server se registruje kod operativnog sistema na način kako je to bilo opisano ranije.

IOPCItemProperties

Ovaj interfejs može biti korišten od strane klijenata da browsuju raspoložive osobine (takodjer nazivane i atributi ili parametri) pridruženi ITEMID i da čitaju tekuće vrijednosti ovih osobina. Na neki način ova funkcionalnost je slična onoj koju obezbjedjuje **BrowseServerAddressSpace**, putem **EnumItemAttributes** i **SyncIORead** funkcija. Medjutim i razlikuje se od ova dva interfejsa u dva važna aspekta:

- a) namjenjen je da bude mnogo lakši za korištenje
- b) nije optimiziran za efikasan pristup velikom obimu podataka.

Ona je prije svega namjenjena da dozvoli aplikaciji da lagano browsira i čita male količine dodatnih informacija specifičnih za dati ITEMID.

Dizajn ovog interfejsa je baziran na pretpostavci da mnogi ITEMIDs su pridruženi sa drugim ITEMID koji predstavljaju vezane sa njima vrijednosti kao što je opseg inženjerskih jedinica , opis procesne tačke ili status alarma.

Naprimjer sistem može interno biti izgradjen od 'recorda' koji predstavljaju kompleksne objekte kao PID regulatore, tajmere, brojače, analognu ulazu , itd.

Ovi elementi rekorda će imati svoje osobine kao što su:

Tekuća vrijednost, zadana vrijednost (set point), visoka (hi) alarmna vrijednost, niska (low) alarmna vrijednost , opis , itd.)

Ovaj interfejs dozvoljava fleksibilan i pogodan način browsovanja , lociranja i čitanja ovih informacija bez da postavlja bilo kakva ograničenja na njihovu strukturu.

On takodjer dozvoljava da te informacije budu čitane bez potrebe kreiranja i manipuliranja sa OPCGroups.

U većini slučajeva sistem kao što je ovaj gore opisani (tj. Interno sastavljen od 'recorda')će takodjer posjedovati hijerarhijski adresni prostor za OPC u formi naprimjer:

A100 je grana ('branch') a A100.CV, A100.SP, A100.OUT, A100.DESC su listovi ('leafs').

Drugim riječima , osobine detalja (item) koji je u formi recorda će se mapirati u u ITEMIDS nižeg nivoa.

Tipično korištenje ovog interfejsa od strane klijenta biće da dobije ITEMID , bilo dobivši 'LEAF' putem interfejsa **BrowseServerAddress** ili putem direktnog unosa preko boksa za editiranje. ITEMID će biti prenesen do **QueryAvailableProperties()**. Rezultirajuća lista bit će pokazana korisniku. On će selektirati osobine koje želi da vidi sa liste. Klijent će prenjeti ovaj set do **GetItemProperties()** da dobije 'snapshot' snimak podataka.

Opciono, klijent je mogao poslati set do **LookupItemIDs** i koristiti rezultirajući set ITEMIDs da kreira OPCGroup da nju koristi da dobija podatke uzastopno u kontinuitetu.

Primjer

Tipični OPC ITEMID može biti FIC101.CV. Ovo će predstavljati tekuću vrijednost taga ili funkcionalni blok koji se zove FIC101. Ovaj funkcionalni blok ima i druge osobine koje su mu pridružene kao što su :

Inženjerske jedinice, opis konture , itd.

Nadalje, funkcionalni blok može takodjer uključivati granice alarma, i status alarma, parametre podešenja regulatora, kao i dokumentaciju kros referenci (tj. gdje se sve blok pojavljuje u sprezi), informacije za održavanje, help ekrane za pomoć u radu sa blokom i niz drugih osobina. Sve ove osobine su pridružene jedna drugoj putem njihove zajedničke asocijacije sa FIC101.

Ovaj interfejs obezbjeđuje pogodan shortcut da se brzo pristupi svim ovim osobinama.

MMI paket može koristiti ovaj interfejs da dozvoli korisniku da indicira da HI i LO vrijednosti za inženjersku jedinicu trebaju biti korištene kod skaliranja vrijednosti bargrafa.

Primjetimo da pošto ove asocijacije mogu biti tipa 'many to many' (tj. mnoge sa mnogima) , a mogu takodjer biti i cirkularne (prstenaste), klijent aplikacija neće ih automatski sve istraživati.

U mnogim slučajevima do ovih osobina korisnik može pristupiti i preko ItemIDs kao naprimjer. FIC101.HI_EU, FIC101.DESC, FIC101.ALMSTAT, itd. Ovi ITEMIDs se mogu naći i u OPCGroup. Ovaj interfejs omogućava alternativni način pristupa osobinama , kada je potrebno dobiti na efikasan način veliki broj informacija.

Ove osobine su podjeljene u tri skupa. OPC 'fixed' skup sadrži osobine koje su identične onima koje se dobiju natrag od poziva **OPCITEMATTRIBUTES**, preporučeni 'recommended' skup je onaj koji je zajednički za sve tipove OPC servera, i 'Vendor specific' skup sadrži dodatne osobine koje su specifične za dati tip hardwareskog uređaja.

Čitanje i Upisivanje podataka

Postoje tri načina da se klijent dobije podatke

- * IOPCSync::Read (iz cache memorije ili iz registara fizičkog uređaja)
- * IOPCAsyncIO2::Read (sa uređaja)

- * `IOPCCallback::OndataChange()` , bazirano na izuzeću , koje može također biti trigerovano sa `IOPCAsyncIO2::Refresh`

Nadalje postoje dva načina da se izbace podatci preko OPC servera na fizički izlaz:

- `IOPCSyncIO::Write`
- `IOPCAsyncIO2::AsyincWrite`

Opcenito o grupama

Svaka grupa ima ime. Za privatne grupe ime mora biti jedinstveno za sve privatne grupe koje pripadaju tom klijentu. Za javne grupe ime mora biti jedinstveno medju svim javnim grupama. Dok klijent može promjeniti ime privatne grupe, ime javne grupe ne može biti promjenjeno.

Privatna grupa i javna grupa mogu imati isto ime sve dok klijent samo ako klijent nije povezan sa javnom grupom istog imena.

Imena grupa su case sensitive. To znači da Grupa1 je različita od grupa1.

Grupe i detalji (items) unutar grupa imaju flagove aktivnosti (active flag). Aktivno stanje grupe se održava nezavisno od aktivnog stanja detalja . Promjena stanja aktiviteta grupe ne mjenja stanje detalja.

Klijenti setuju i resetuju flagove aktiviteta za grupe i detalje kao efikasan način i alternativa dodavanju i uklanjanju grupa i detalja u njima.

Tako naprimjer ako je na MMI displeju prikaz minimiziran , onda detalji koji dolaze sa servera na taj prikaz se neće prikazivati i klijent može resetovati njihove flagove tako da mu ih server ne šalje i time rastereti komunikaciju.

Drugi način minimizacije ovog saobraćaja je poziv `OnDataChange` unutar adresnog prostora klijenta, tako da kada se aktivni detalji u aktivnim grupama promjene (promjena je definirana kao promjena u vrijednosti (value) od posljednje prethodne koja je poslata klijentu) , ili promjena u kvalitetu (Q) te vrijednosti. Server će vratiti vrijednosti (V) i flagove kvaliteta (Q) za one detalje unutar aktivne grupe koji su se promjenili.

Javne grupe

IOPCServerPublicGroups (opcioni interfejs)

Ovaj opcioni interfejs dozvoljava management javnih grupa

Moguće je dizajnirati aplikaciju tako da iste grupe podataka se koriste od strane više klijenata. U ovakvim slučajevima opcionalna mogućnost servera sa javnim grupama obezbeđuje pogodan mehanizam za klijente i servere da dijele ove grupe.

Javne grupe mogu biti kreirane od strane servera ili može ih kreirati i klijent. Kada ih kreira klijent, one se prvo kreiraju kao privatne grupe a onda se konvertuju u javne grupe sa pozivom **MoveToPublic**.

Klijent može izbrojati (enumerate) raspoložive javne grupe po imenima koristeći :

IOPCServer::CreateGroupEnumerator. Može se spojiti ('connect') na javnu grupu pozivjući **GetPublicGroupName**. Može ispitati sadržaj grupe putem : **IEnumOPCItemAttributes**. Može doznačiti klijentske handles i tipove podataka koji su poželjni za specifičnog klijenta koristeći razne **IOPCItemMgt** funkcije.

Kada se klijent spoji na javnu grupu, on se ponaša vrlo slično kao i kod spajanja na privatnu grupu. On može aktivirati ili deaktivirati grupu ili detalje (items) u grupi. On može postaviti klijent handles za grupu i detalje unutar grupe. On može postaviti zahtjevani tip podataka za detalje u grupi sa kojim mu server treba slati te podatke.

Medjutim sve ove operacije se odnose samo na tog specifičnog klijenta. One ne utiču na način kako su drugi klijenti povezani na tu javnu grupu i šta oni od nje traže.

Izuzetak od ovakvog ponašanja prema javnoj grupi u odnosu na privatnu grupu je da on ne može dodavati ili brisati detalje iz te javne grupe.

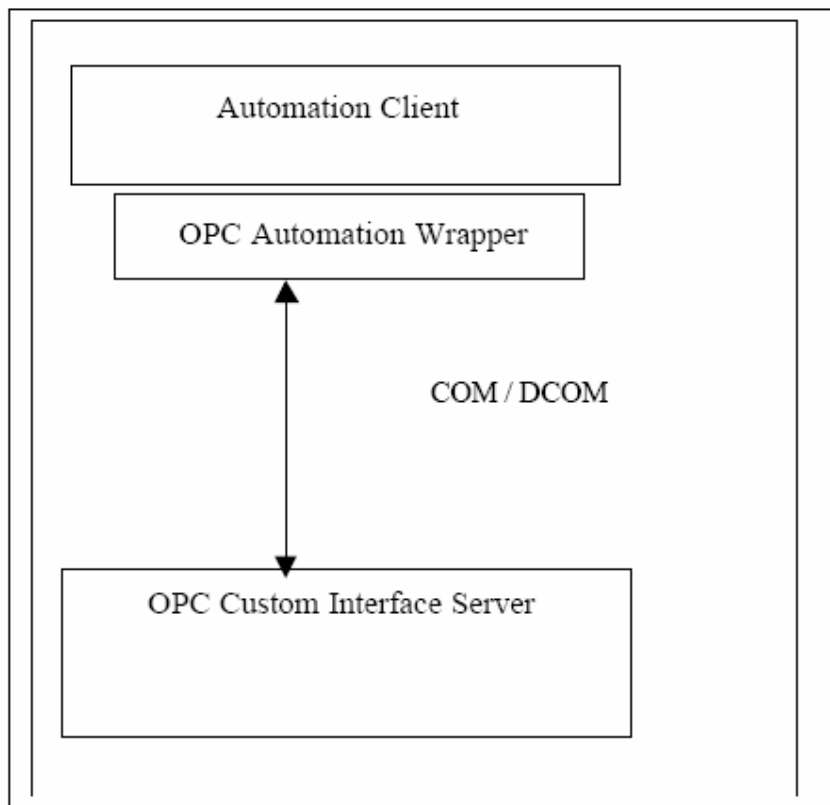
SPECIFIKACIJE ZA DATAACCESS AUTOMATION INTERFACE STANDARD VERZIJA 2.02

Namjena

Ono što ovaj standard želi da definiše je zajednički način na koji aplikacije u oblasti upravljanja i vođenja procesa mogu pristupiti procesnim podacima. Ovaj interfejs treba da obezbedi istu funkcionalnost kao i custom (custom) interfejs, ali na način koji je blizak trendu u automatizaciji programiranja.

Kao što je već omogućeno da automatizacije aplikacije mogu pristupiti drugim softwareskim okružajima kao napr. RDBMS, MS Office aplikacijama, WWW objektima, ovaj interfejs je sačinjen tako da olakša razvoj aplikacija, bez žrtvovanja funkcionalnosti definirane prolagodjenim (custom) interfejsom.

Naredna slika pokazuje automatizacioni klijent koji poziva OPC Data Access Server koristeći 'wrapper' DLL. Ovaj omotač prevodi izmedju kastomiziranog interfejsa kojeg obezbedjuje server i automatizacionog interfejsa kojeg želi klijent. Primjetimo da u opštem slučaju konekcija izmedju Automation klijenta i Automation servera bit će 'In Process' dok konekcija izmedju Automation servera i Custom servera može biti bilo In process, lokalna ili udaljena.



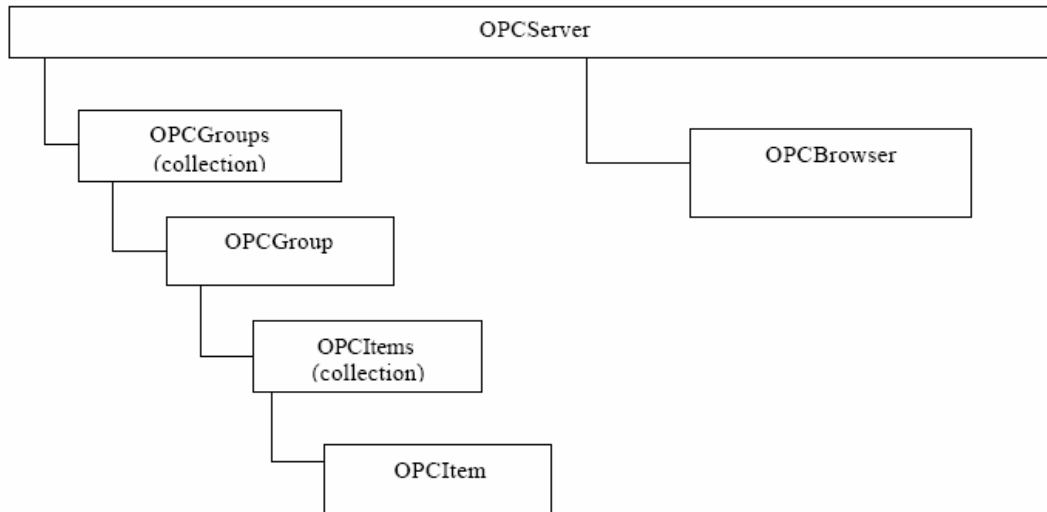
Custom i automation klijent aplikacije koje se povezuju sa OPC Serverima

Arhitektura

Osnovni cilj dizajna je da je ovaj interfejs namjenjen da radi kao 'wrapper' za postojeće OPC DataAccess Custom Interfejs servere, i da obezbjede jedan pogodan za korištenje mehanizam za funkcionalnost koju obezbjedjuje custom interfejs.

Interfejsi su u potpunosti podržavani od VC++ i VB 5.0 i kasnije verzije. Oni dozvoljavaju svakoj aplikaciji koja ima OLE automation interfejs (napr. VB, VC++ ili VBA) da pristupe OPC interfejsu.

OPC Automation Server Object model



Automation object hijerarhija

Objekt	Opis
OPCServer	.Instanca OPC Servera. Korisnik mora kreirati OPCServer objekat , prije nego se mogu dobiti reference za druge objekte. Sadrži OPCGroups kolekciju i kreira OPCBrowser objekte.
OPCGroups	Automation kolekcija koja sadrži sve OPCGroup objekte. Ovaj klijent je kreirao unutar opsega OPCServera sa kojim se Automation aplikacija povezala putem OPCServer.Connect().
OPCGroup	Instanca OPCGroup objekta. Namjena ovog objekta da održava informaciju o stanju i obezbjedi mehanizam da osigura servise akvizicije podataka za OPCItem Collection object koji OPCGroup object referencira.
OPCItems	Automation kolekcija koja sadrži sve OPCItem objekte koje je ovaj klijent kreirao unutar opsega OPCServer-a , i odgovarajućeg OPCGroup objekta koji je Automation aplikacija kreirala.
OPCItem	Automation objekt koji održava definiciju itema, tekuću vrijednost statusnu informaciju i posljednje vrijeme ažuriranja (update). Primjetimo da Custom interfejs nema ovaj posebni Item Object.
OPCBrowser	Objekt koji browsuje imena itema u konfiguraciji servera. Postoji samo jedna instanca OPCBrowser objekta za instancu OPCServer objekta.

OPC Data Access Automation Object Model

OPCServer objekt obezbjedjuje način da se pristupi (za čitanje i pisanje, tj. Read/write) ili komunicira sa skupom izvora podataka. Tipovi izvora koji su na raspolaganju su funkcija implementacije servera.

OPC Automation klijent se povezuje sa OPC Automation Serverom koji komunicira sa izvorom podataka (t.j. sa Data Access Custom Serverima) , putem funkcionalnosti koju obezbjedjuju automation objekti koji su opisani u ovom standardu.

OPCServer obezbjedjuje jedan OPCGroups objekat za automation kolekciju da bi održavao sakupljanje OPCGroup objekata.

OPCGroup objekat dozvoljava klijentima da organiziraju podatke kojima žele da pristupe. OPCGroup može biti aktivirana ili deaktivirana kao jedinica. OPCGroup takodjer obezbjedjuje način za klijenta da se 'pretplati' (subscribe) , na listu itema tako da mogu biti upoznati sa njihovim promjenama kada se dese.

OPCGroup objekat obezbjedjuje OPCItems kolekciju od OPCItema.

Sinhronizacija podataka

Postoje zahtjevi da VB klijent može da čita ili prima podatke kao što su vrijednost, kvalitet i vremensku informaciju (timestamp) u sinhronizmu.

Ako klijent dobije vrijednosti koristeći bilo koji od metoda čitanja treba da bude siguran da vrijednost (value), timestamp, i kvalitet (V,T,Q) će biti u sinhronizmu unutar njihovih samih.

Ako klijent dobija podatke registrirajući se za DataChange events (događaje) , tada V,T,Q trebaju biti u sinhronizmu unutar opsega EventHandlerler rutine.

Ako klijent mješa ova dva pristupa bit će nemoguće za klijenta da obezbjedi da su osobine itema tačno u sinhronizmu , pošto događaj koji je promjenio osobine se mogao pojaviti između vremena kada klijent pristupa različitim osobinama.

Izuzeća i događaji (exceptions and events)

Većina metoda opisanih ovdje komunicira sa OPC Custom serverom. U OLE automation, nema lakog načina da se vrati informacija o grešci kada se pristupi nekoj osobini. Najbolji način da se ovo razriješi je da automation server generira izuzeće (exception) ako se takva greška pojavi u izvoru podataka. To znači da klijent treba da ima kodiranu logiku izuzeća da bi mogao da manipuliše ovim greškama.

Događaji (events)

Automation interfejs podržava mehanizam notifikacije o događaju koji je uveden u VB 5.0 i svim narednim verzijama.

Automation server trigeruje događaje kao odziv na Async Refresh, Async Read i Async Write Method pozive (calls). Dodatno, Automation server trigeruje događaje kada se podatci promjene u skladu sa specifikacijama klijenta.

Polja (arrays)

Po konvenciji, OPC Automation interfejs predpostavlja da su polja bazirana na indeksu koji počinje sa 1 . Ako se varijabla polja prenosi ka funkciji koja je veća od Count ili NumItems parametra, samo Count ili NumItems elementi će se koristiti , počevši od indeksa 1.

Da bi se izbjegle greške sugerise se da VB kod koristi "Option base 1".

Kolekcije (Collections)

OLE Automation kolekcije su objekti koji podržavaju Count, item, i skrivenu osobinu _NewEnum. Svaki objekt koji ima ove parametre kao dio interfejsa se može zvati kolekcija.

Svi string parametri za OPC interfejs su **UNICODE**, pošto su izvorni OLE APIji svi sa UNICODE-om.

Korištenje OPC Servera

FIX OPC server je baziran na OPC 1.0a Specifikaciji za pristup podacima (OPC 1.0a Data Access Specification) , i podržava interfejs koji je opisan u ovoj specifikaciji.

FIX OPC Server takodjer podržava opcionu BrowseAddressSpace interfejs ili opcione Public groups mogućnosti.

Instalacija

Server je implementiran kao DLL process koji se zove OPCEDA.DLL. Ova DLL je rezidentna u FIX32 stazi (C:\FIX32, default). DLL se registruje kada se instalira software i može mu se pristupiti kao in-proces server. Primjetimo da ovaj server može da se izvršava samo na čvorovima na kojima je FIX instaliran, ne može se izvršavati na non-FIX čvorovima.

Umreženje

OPC server je in-proces server. DCOM za sada nije podržan. Medjutim, ovaj server kao i EDA (Easy Database Access- laki pristup podacima) , mogu pristupiti podacima od bilo kojeg FIX SCADA servera na FIX mreži. Ovo čini na taj način što koristi Intellution FIX mrežni protokol.

Ime Servera

ProgID (identifikator programa) ovoga servera je INTELLUTION.OPCEDA.1. Ovo ime treba specificirati u OPC klijentu da bi se pristupilo podacima u procesnoj bazi podataka. Pri tome OPC Klijent mora biti rezidentan na SCADA serveru da bi pristupio procesnoj bazi podataka u čvoru.

Sintaksa Item ID identifikator i brauzing

Server podržava tronivovsku hijerarhiju Item ID-ja . Najviši nivo je ime FIX čvora , drugi nivo je ime taga , i treći nivo je ime polja. Konsekventno, ItemID ima sintaksu **node.tag.field**. Ovo je slično korištenju ovih termina u FIX.

Napomena : FIX zahtjeva kolonu (:) , nakon imena čvora umjesto tačke (.)

Doznačavanje imena

Možemo doznačiti imena čvora i tagova koristeći FIX aplikacije. Koristeći **System Configuration Utility (SCU)**, možemo doznačiti ime svakom FIX čvoru. Čvorovi koji su na raspolaganju u bilo kojem čvoru su oni koji su specificirani u mrežnim konfiguraciji u SCU. Ako smo omogućili dinamičku sesiju (Dynamic Sessions) u mrežnoj konfiguraciji SCU, možemo se povezati sa bilo kojim čvorom FIX mreže , čak ako taj čvor nije vidljiv u Brauseru.

Ako je potrebno kreirati sesiju za taj detalj, može se pojaviti kašnjenje do 20 sekundi ili više zavisno od veličine i konfiguracije mreže.

Možemo kreirati imena tagova kao što kreiramo SCADA serversku procesnu bazu podataka. Da bi kreirali bazu podataka , treba koristiti **Database Builder**.

Svaki tag ima jedan ili više pridruženih polja . Ova polja postaju raspoloživa u procesu kreiranja tagova u bazi podataka.

Tipovi podataka

Server može čitati ASCII vrijednosti iz FIXa. Možemo identificirati ova polja lako jer počinju sa A _ ili F _ . OPC server takodjer podržava OPC interfejs **RequestedDataType**. Interfejs pokušava da konvertuje bilo koji podatak iz internog (kanonskog) tipa podatka u tip koji zahtjeva aplikacija. Kanonski tip podatka zavisi od polja koje se čita. Kanonski tip podatka za A _ polja je VT_BSTR, za F _ polja kanonski tip podatka je VT_R4.

Korištenje OPC klijent drajvera

Intellution OPC klijent djeluje kao I/O drajver. Možemo ga konfigurirati koristeći SCU (System Configuration Utility).

[1] Kliknuti na SCADA dugme na SCU toolboxu.

[2] Kliknuti ? dugme desno od polja imena I/O drajvera.

[3] Izabrati OPC klijent iz liste koja se pojavi.

[4] Kliknuti na **Add**

[5] Kliknuti na **Configure** da se konfigurira klijent sa **Power Tool**. Ovaj program nam omogućava da konfiguriramo klijenta kao što bi to uradili sa bilo kojim non-OPC drajverom.

[6] Kada završimo sa konfigurisanjem OPC klijenta , treba spasiti SCU i drajversku konfiguraciju.

[7] Kasnije, kada se konfiguriraju i kreiraju blokovi baze podataka, selektiraćemo OPC klijent kao blok drajver (tj. uređaj).

Potiskivanje alarma drajverske komunikacije

FIX 7 daje nam mogućnost da potisnemo alarme o komunikaciji drajvera (COMM alarme). Modificiranjem Scan, Alarm and Control (SAC program) komandne linije, možemo spriječiti COMM alarme da se nadju u repu alarma. Kao rezultat toga , COMM alarmi se neće pojaviti u SCADA alarm destinacijama. Procesna baza podataka registruje posljednje poznate vrijednosti prije nego se pojavila greška u komunikaciji, i prikazuje te vrijednosti umjesto ????????

Da bi se koristila ova mogućnost, treba da se unese **C** na komandnoj liniji.

Automatsko preključenje (failover)

Standardni FIX View čvor prikazuje procesne podatke operatorima. View čvor postiže ovo oslanjajući se na samo jednu konekciju sa SCADA čvorom da primi podatke iz tog čvora. Ukoliko taj čvor postane nedostupan procesni podatci iz tog čvora postaju također nedostupni operatoru.

Sa opcijom automatskog preključenja View čvorovi se mogu povezati sa parom SCADA čvorova preko jednostruke veze. Ova osobina obezbjeđuje dvije staze ka istim procesnim podacima umjesto samo jedne. Kao rezultat, operator može pristupiti procesnim podacima čak i kada jedan SCADA čvor nije raspoloživ. Na ovaj način, automatsko preključenje poboljšava ukupnu raspoloživost sistema za kritične operacije, jer operator će uvijek imati kontrolu nad procesom.

Karakteristike

Automatsko preključenje obezbjeđuje slijedeće osobine:

- automatski nadzor sesije za svaki SCADA čvor
- Kompatibilnost sa postojećim SCADA čvorovima
- Podrška za tag : prikaz stanja mreže (NSD – network status display)

Nadzor automatske sesije

Svaki View čvor na kojem omogućavamo automatsko preključenje, nadzire konekciju sa svojim SCADA serverima. Putem nadzora konekcije, View klijent može automatski preključiti na backup SCADA čvor kada SCADA čvor sa kojim View čvor komunicira postane nedostupan. Na ovaj način FIX obezbjeđuje da proces je kontinualno nadziran.

Primjetimo da automatsko preključenje zahtjeva TCP/IP mrežu i ne može raditi sa NetBIOSom.

Prikaz taga NSD (network status display)

Kao dio opcije automatskog preključenja , poseban tag, NSD – prikaz stanja mreže, je na raspolaganju na View čvoru.

Ovaj tag nam omogućava:

- da iniciramo ručno preključenje
- privremeno onemogućimo automatsko preključenje na jednom View čvoru
- trigeruje događaj (kao naprimjer prikaz poruke) kada se preključenje desi
- Pokazuje dijagnostičku i informaciju preključenja na ekranima operatora

Napomena

Čvor automatskog preključenja može biti bilo View (klijent) or SCADA (server) čvor. Kada se koristi na SCADA čvoru , ovaj čvor može biti lokalni čvor preključenja.

Ova opcija automatskog preključenja ne sinhronizira procesne baze podataka na SCADA čvorovima.

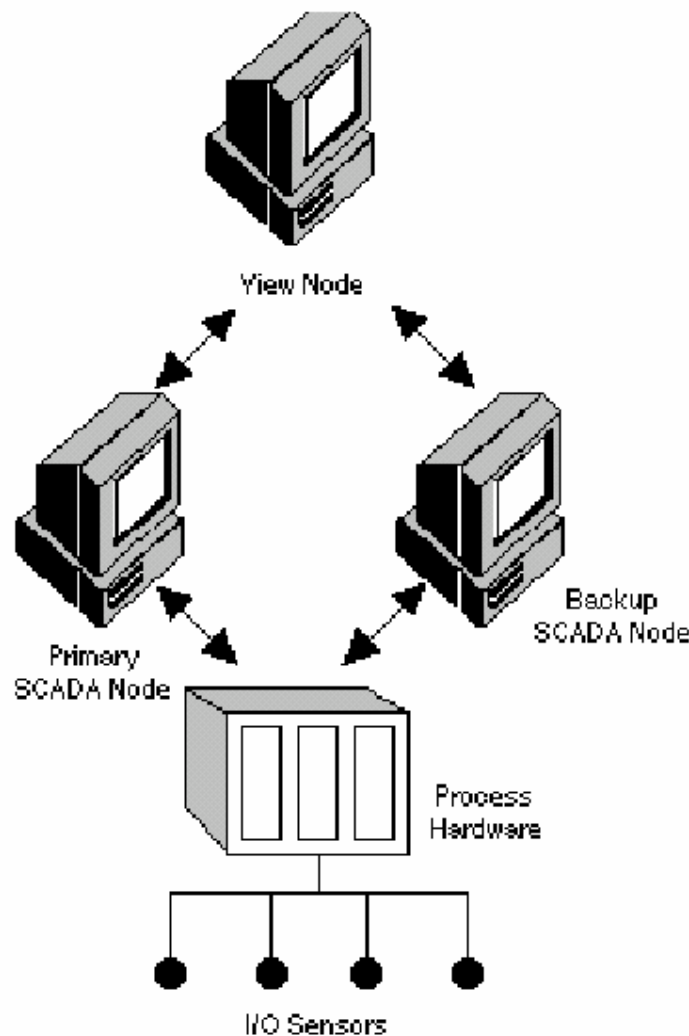
Mehanizam djelovanja automatskog preključenja

Kada se omogući automatsko preključenje , View čvor može dobiti podatke ili iz svog primarnog ili backup čvora. Primarni čvor je SCADA server sa kojim želimo da View klijent komunicira. Backup čvor je SCADA server sa kojim želimo da View klijent komunicira kada primarni čvor postane nedostupan.

Zavisno od broja preključenja koji se desio, bilo primarni ili backup čvor je u komunikaciji sa View čvorom. Čvor koji trenutno komunicira sa View čvorom je *aktivni čvor*. Drugi čvor je *standby čvor*

Preključenje sa jednog SCADA čvora na drugi se dešava kada aktivni SCADA čvor postane nedostupan.

Slijedeća slika pokazuje relaciju između View čvora sa automatskim preključenjem omogućenim kao i njegov primarni i backup SCADA server.



Automatsko preključenje kod starta

Kada FIX starta, View klijent pokušava da uspostavi komunikaciju sa svojim primarnim i backup SCADA serverima. Ako su obadva servera raspoloživa, View klijent uspostavlja konekciju sa obadva. Ako je samo jedan SCADA server raspoloživ, View klijent uspostavlja komunikaciju sa njim. Ako nijedan od servera nije raspoloživ, View klijent proziva obadva čvora dok ne uspostavi komunikaciju sa barem jednim od SCADA servera. Prvi SCADA server sa kojim uspostavi komunikaciju postaje aktivni čvor.

Jedanput kada je komunikacija uspostavljena, korist ćemo slijedeću tabelu da odredimo šta će se desiti kada se preključenje desi:

Kada komunikacija sa aktivnim SCADA čvorom je	<i>I komunikacija sa standby SCADA čvorom je</i>	I	Tada
raspoloživa	raspoloživa	Komunikacija sa aktivnim čvorom postane nedostupna	atomatsko preključenje se desi
raspoloživa	neraspoloživa	Komunikacija sa aktivnim čvorom postane nedostupna	atomatsko preključenje se neće desiti
raspoloživa	raspoloživa	Komunikacija sa standby čvorom postane nedostupna	atomatsko preključenje se neće desiti
neraspoloživa	raspoloživa	Komunikacija sa standby čvorom postane nedostupna	atomatsko preključenje se neće desiti

Prikazivanje podataka za vrijeme preključenja

U situaciji kada imamo na ekranu View čvora sliku koja referencira aktivni čvor, kada se uspostavi komunikacija sa aktivnim čvorom, View čvor počinje čitati podatke sa tog čvora. Kada aktivni čvor postane nerasploživ, View čvor će izgubiti sesiju sa tim čvorom. Kada je sesija izgubljena, pojaviće se boks poruke (message box) koji će ovo saopštiti. Prije nego što View čvor može preključiti na standby čvor, operator mora potvrditi ovu poruku.

Ako želimo da View čvor preključi automatski na standby čvor bez intervencije operatora, možemo potisnuti boks poruke, startajući View aplikaciju sa paramterom u komandnoj liniji –s1914.

Kada se View čvor preključi na na standby čvor, slijedeći događaji će se desiti:

[1] Znaci (@) biće prikazani u linkovima na otvorenim slikama View čvora da indicira da je sesija sa aktivnim čvorom izgubljena.

[2] Standby čvor postaje novi aktivni čvor i čvor koji je otkazao postaje novi standby čvor.

[3] Slike su automatski zadovoljene za linkove sa procesnim podacima iz novog aktivnog čvora. Ovo će osigurati integritet podataka.

[4] Znaci (@) će se zamjeniti sa procesnim podacima iz novog aktivnog čvora. Ime čvora koje je referencirano u ovim linkovima se ne mjenja, ono nastavlja da pokazuje ime primarnog čvora.

[5] Poruka o događaju (event message), koja indicira da se preključenje desilo se šalje kao alarm, nakon prijema procesnih podataka iz novog aktivnog čvora.

Napomena:

Da bi procesni podatci bili pokazani na operatorskom displeju nakon što se preključenje desilo, procesna baza podataka na obadva servera t.j. aktivnom i standby mora biti ista.

Izbacivanje vrijednosti na izlazima

Podatci koji se upisuju na izlaze i šalju ka procesu idu na aktivni SCADA čvor. Naprimjer, predpostavimo da je aktivni čvor SCADA1 a standby čvor je SCADA2. Prije preključenja, View čvor upisuje podatke na SCADA1. Nakon preključenja, View čvor upisuje podatke na SCADA2.

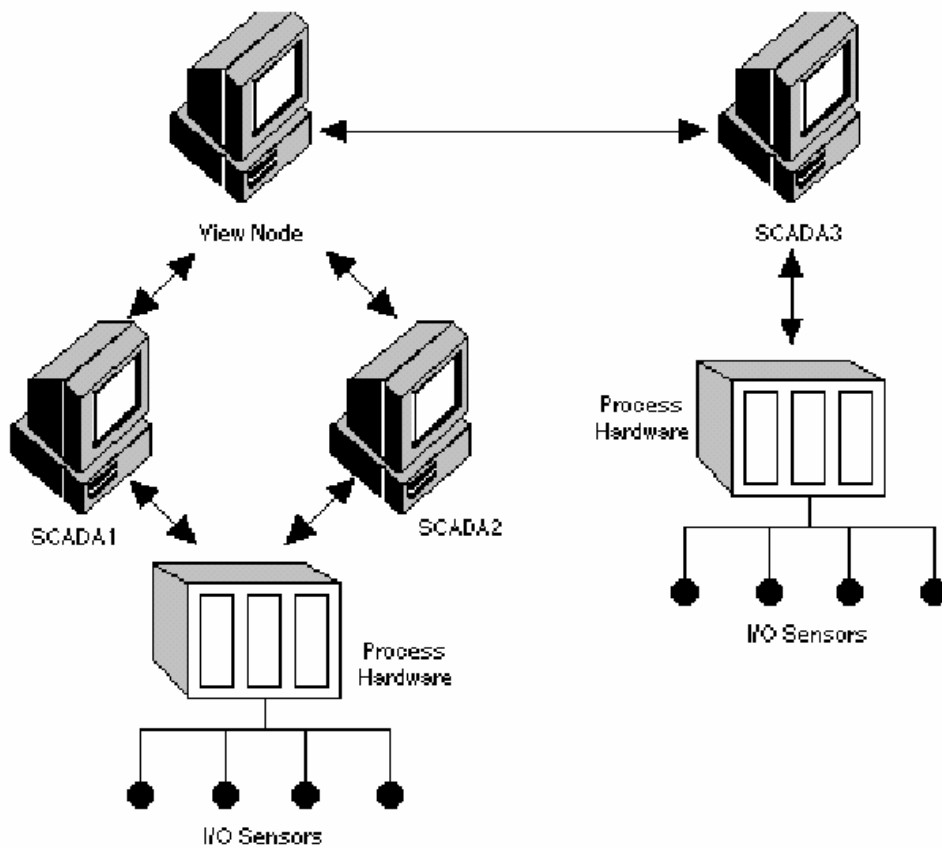
Rad sa alarmima

Kada se alarm desi na SCADA serveru on se šalje na View klijent. U opštem slučaju View klijent prihvata alarme od aktivnog čvora: međjutim ako View klijent ima sesiju sa standby čvorom, može primiti alarme iz ovoga čvora takodjer.

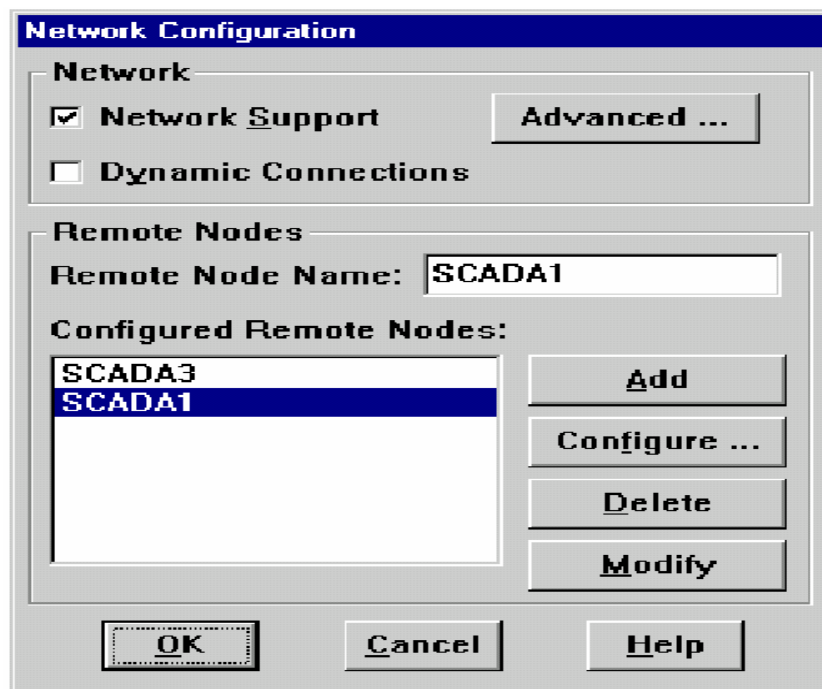
Svi alarmi primljeni iz backup čvora se modifikuju kao da dolaze iz primarnog čvora. Kada sumarni alarmi link primi alarme, izgleda kao da je isti alarm bio primljen dva puta, jedanput iz primarnog čvora i jedanput iz backup čvora. Pošto link ne prikazuje duple alarme, operator će vidjeti samo jedan alarm. Ovo ne znači da je drugi alarm izgubljen. Umjesto toga, drugi primljeni alarm zamjenjuje prvi alarm u linku i postavlja vremenski stamp alarma (vrijeme kada se desio).

Primjer konfigurisanja čvorova

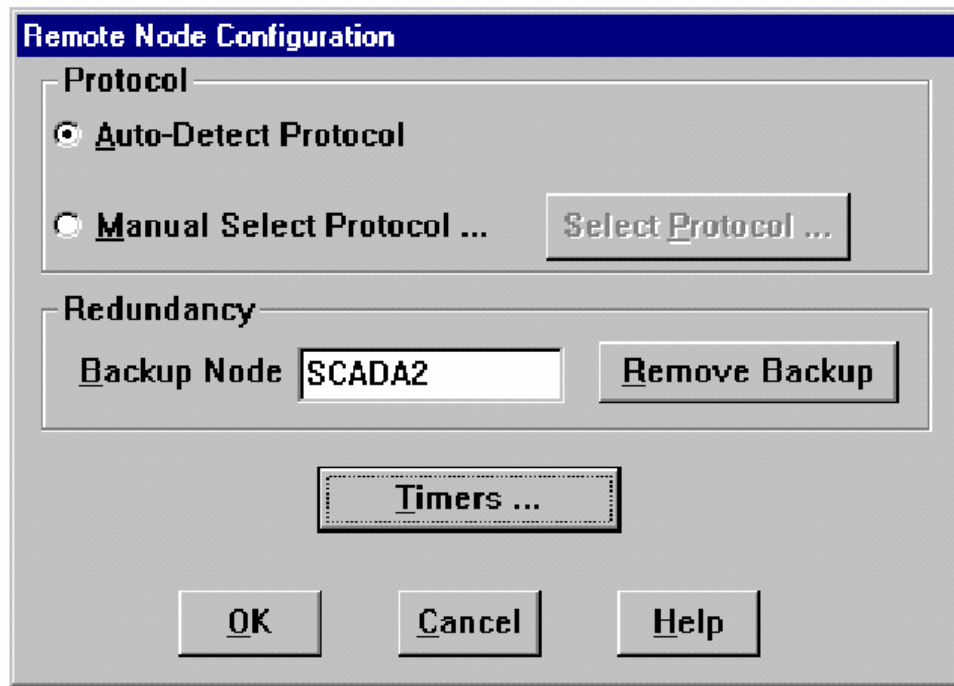
U slijedećem primjeru pokazaćemo jednostavan slučaj konfigurisanja View čvora spregnutog sa SCADA server čvorovima. Dva od tih čvorova SCADA1 i SCADA3 su aktivni čvorovi dok je čvor SCADA2 standby čvor za čvor SCADA1.



U SCU meniju u dijelu konfiguracije mreže ovo će biti uneseno na slijedeći način:



Konfiguracija za automatsko preključenje na čvor SCADA2 je unesena u SCU kao što je prikazano na slijedećoj slici:



TUMAČ POJMOVA (GLOSSARY)

?button – specijalna FIX komanda koja se koristi da se dobije dijalog boks dozvoljenih unosa u polje (legal entries)

account – sigurnosna konfiguracija za korisnika ili grupu korisnika

alarm area – fizička ili funkcionalna podjela postrojenja koja djeluje kao ruting oznaka za alarme i poruke

alarm client – čvor koji se konfigurira da prima alarme i poruke preko mreže.

alarm information – informacija generirana od strane baze podataka ili FIX programa.

alarm queue – zona podataka koja se koristi da čuva alarmne poruke koje se upućuju ka alarmnom zadatku

alarm servers – čvor odgovoran za distribuiranje alarma i poruka preko mreže. Alarmni serveri su SCADA čvorovi.

alarming - proces putem kojeg blok prima podatke , poredi podatke prema datim alarmnim granicama, i reaguje na procesne vrijednosti koje prelaze van definiranih granica.

application program interface – kolekcija programskih funkcija dizajnirana da omogućí pristup aplikacionim podacima.

base path – glavni direktorij na kojem se nalazi Fix direktorij na hard disku (drajvu)

basic node – bilo koji kompjuter koji ima podršku FIX bazičnog noda.

blind SCADA node – SCADA čvor koji ne koristi nikakvu grafiku niti neki drugi aplikacioni software. Konfiguracija dozvoljava više resursa kompjutera da se koriste za akviziciju podataka i management.

block alarm – alarm generiran od strane baze podataka baziran na odstupanju od operatorski definiranih granica ili komunikacionih grešaka.

block – individualne jedinice instrukcija koje mogu primiti, verificirati, manipulirati i izbacivati procesne vrijednosti. Blokovi mogu takodjer porediti procesnu vrijednost sa alarmnom granicom i izvršavati računanja bazirana na specifičnim procesnim vrijednostima.

centralized processing – aplikacija kada sve funkcije sistema upravljanja procesom se izvršavaju na jednom računaru.

chain – dva ili više blokova povezani jedan sa drugim. Kada korisnik kreira bazu podataka, on treba da sekvencira blokove tako da svaki blok izvršava specifični zadatak i nakon toga prenese svoju informaciju na jedan ili više odgovarajućih blokova. Kada su blokovi korektno definirani i sekvencirani u lanac, blokovi mogu verificirati, automatizirati, alarmirati, prikupiti podatke, i održavati proces.

check box – komanda koja omogućava korisniku da uključi ili isključi opciju putem izbora ili neizbora boksa (check i uncheck)

command language – moćan tekst orijentirani programski jezik koji omogućava dizajnerima prikaza procesnih slika da automatiziraju zadatke putem serije instrukcija koje mogu biti izvršene po zahtjevu.

control – sposobnost automatskog korištenja algoritama koji podešavaju procesne vrijednosti i time održavaju ove vrijednosti unutar zadatih granica

controls – objekti grafičkog interfejsa korisnika kojima korisnik može manipulirati da izvršava funkcije

data acquisition – sposobnost da se prikupe podatci iz procesa i procesiraju ti podatci u formi koristivoj u daljim obradama

data management – manipuliranje i usmjeravanje prikupljenih podataka

database builder (DBB) program – korisnički konfiguracioni program za kreiranje procesne baze podataka. Graditelj baze podataka takodjer djeluje i kao korisnički zadatak, omogućujući operatorima da prikažu podatke realnog vremena u formi tabličnih polja (spreadsheet)

database – predstava procesa kreirana ulančenjem blokova procesne kontrolne logike

DDE Server application – Intellution-ova implementacija Microsoftovog standardnog protokola za razmjenu podataka za interaktivne aplikacije.

dialog box – kolekcija komandi dizajniranih da omoguće korisniku da specificira kako da izvrši funkciju.

distributed processing – aplikacija u kojoj mnogo računara povezano mrežom dijele funkcije upravljanja procesom i kritične funkcije su raspodjeljene među tim računarima

draw program – prikazuje dizajnirani program

Driver image Table (DIT) – zona u memoriji korištena od strane I/O drajvera da pohrani u nju procesne podatke. Program skaniraj, alarmiraj i upravljaj (SAC – scan, alarm, control) uzima te informacije i ažurira bazu podataka sa njima.

Dynamic Data Exchange (DDE) - forma komunikacije razvijena od strane Microsofta koja koristi djeljenu memoriju da aplikacije mogu preko nje izmjenjivati podatke.

Easy Database Access (EDA) application – interfejsni aplikacioni program sa funkcijama pisanim u C, C++ i Visual Basic jezicima koji omogućava da čitamo ili upisujemo podatke u FIX bazu podataka.

event messaging – sposobnost zapisivanja događaja koji se pojavljuju u bloku bez zahtjeva operatoru da ih potvrdi

exception- based processing – akvizicija i manipulacija podacima nakon trigerskog događaja

field - mjesto gdje korisnik unosi podatke

filtering - sposobnost sortiranja unosa da bi se našle uparujuće kombinacije

FIX - potpuno integrirani sistem upravljanja

FIX MMI - potpuno integrirani sistem upravljanja ,sa interfejsom čovjek -sistem

GUI - grafički korisnički interfejs

historical Assign program - zadatak korisnička konfiguracije koji dozvoljava korisniku da selektira tačke podataka za uzorkovanje

Historical Collect program – sistemski zadatak koji izvršava uzorkovanje podataka na bazi konfiguracionog fajla kreiranom pomoću programa Historiskog doznačavanja

Historical Display program – korisnički zadatak koji dozvoljava operatoru da prikaže prikupljene podatke na trnd zapisima

Historical Trending application – FIX aplikacija koja obezbjedjuje mogućnost uzorkovanja podatke realnog vremena sa brzinama koje specificira operator.

I/O driver – softwareski interfejs odgovoran za prikupljanje podataka sa procesnog hardwarea i pohranjivanje u DIT tabelu.

industrial automation software – software koji obezbjedjuje procesne podatke realnog vremena osoblju kao i druge softwareske aplikacije u procesu

keyboard accelerator – taster ili kombinacija tastera koja izvršava komandu menija bez da operator mora ulaziti u meni.

link – grafička aplikacija , konekcija u realnom vremenu sa tačkom podatka u bazi podataka

list box – komanda koja obezbjedjuje klizeću listu selekcija

local node – čvor na kojem korisnik radi

logging in - process identifikacije korisnika u sigurnosnom sistemu računara tako da korisnik može dobiti prava pristupa programima za koje je autorizovan

logging out – proces obavještanja sigurnosnog sistema da korisnik neće više raditi na čvoru.

Macro Editor – FIX aplikacija koja omogućava razvojnom inženjeru da doznači jednostavnu sekvencu tastera Makrou koji je izgradjen pomoću komandnog jezika.

Man Machine Interface (MMI) – interfejs između procesnih podataka i osoblja u postrojenju. U FIX softwaru , MMI je ustvari View program

monitoring – sposobnost prikazivanja podataka realnog vremena operatorima procesa

node – bilo koji računar na kojem se izvršava FIX software

one-shot processing - procesiranje koje se pojavljuje samo jedanput kada se lanac stavi na skeniranje

Operator message – poruka generirana od FIX aplikacije. Operatorske poruke kreiraju historiju važnijih operatorskih akcija.

PLC – programabilni logički kontroler

pointing device – miš ili kugla

poll record – opseg vrijednosti podataka koji se dobijaju iz procesnog kontrolera i pohranjuju radi omogućavanja pristupa FIX programima

primary block – blok koji prima podatke iz DIT tabele i uključuje i vrijeme skaniranja. Primarni blokovi su obično udruženi sa jednim ili više I/O hardwareskih uređaja.

Process and Instrumentation Drawing (P&ID) – crtež koji opisuje fizički izgled postrojenja predstavljen simbolima sa adresama i parametrima svih tačaka podataka.

process hardware – PLC ili neki drugi I/O uređaj

pull-down menu - alternativni naziv za meni

push button – grafički korisnički interfejs koji simulira vrstu realnog kontrolnog elementa kao što je taster ili prekidač.

radio button – grafički korisnički interfejs koji korisnik koristi da izabere između međusobno isključivih opcija

remote node – bilo koji čvor za koji korisnik treba komunikacionu vezu da mu pristupi

report generator application – Microsoft Excel i kolekcija Intellution kreiranih makroa za kreiranje izvještaja od FIX podataka.

SCADA – nadzor, upravljanje i prikupljanje podataka

SCADA node – računar koji puni memoriju sa procesnom bazom podataka i prikuplja procesnu informaciju od jedne ili više kontrolnih uređaja. SCADA čvor može vratiti procesnu informaciju kontrolnim uređajima da automatizira i upravlja procesom. Svi SCADA čvorovi izvršavaju skeniranje, alarmiranje i upravljanje (SAC) program, da bi procesirao bazu podataka.

Scan time – korisnički definiran interval vremena koji određuje kada će SAC skenirati i procesirati blok u bazi podataka.

Scan, Alarm, and Control (SAC) program – program koji prikuplja procesne informacije iz DIT tabele i koristi ih da bi ažurirao blokove u procesnoj bazi podataka. SAC također vraća korekzione vrijednosti iz baze podataka na tabelu slike drajvera (DIT), odakle se šalju preko I/O drajvera na kontrolni uređaj u procesu.

secondary block – blok koji prima ulaz od uzvodnog bloka i izvršava specifičnu funkciju sa ulazom. Na primjer, sekundarni blok može izvršiti izračunavanje ili pohranjivanje narednih ulaza.

security application – FIX aplikacija koja dozvoljava operatorima da zaštite programe, blokove baze podataka, i slike od neautorizovanog pristupa.

security area – fizička ili funkcionalna podjela procesa koja dozvoljava korisniku da ograniči pristup prikazima i zaštiti blokove baze podataka od upisivanja. Sigurnosne zone mogu biti procesna oprema (napr. pumpe i peći), energenti (naprimjer gorivo, voda, para) ili funkcije održavanja.

session – komunikaciona veza izmedju dva čvora preko mreže.

Simulation I/O Driver (SIM) – simulirani I/O drajver sa kojim operatori mogu izgraditi bazu podataka prije nego što se povežu sa stvarnim I/O uređajima

spreadsheet – tabela višestrukih redova i kolona koja prikazuje informacije iz trenutno otvorene aplikacije

stand alone node – centralizirani FIX sistem

subsecond scanning – sposobnost korištenja vremena skaniranja i regionu izmedju 0.05 i 0.95 sec.

Supervisory control – sposobnost nadziranja podataka realnog vremena povezanog sa sposobnošću operatora da mijenja zadate vrijednosti i druge ključne vrijednosti direktno od računara.

System Configuration utility (SCU) – glavni konfiguracioni program FIX sistema.

System alarms – alarmi koji prate probleme u radu sistema upravljanja

System messages – poruke koje prate probleme u radu sistema

system tasks – zadatci koji rade sa procesom u realnom vremenu. Sistemski zadatci zahtjevaju vrlo malo ili nimalo interakcije sa operatorom.

tagname – informacija koja je potrebna da se pristupi podatku u bazi podataka. Ime taga se sastoji iz imena čvora(NODE), imena bloka baze podataka(TAG), i imena polja (FIELD) u slijedećem formatu : NODE: TAG. FIELD.

text box – oblast u polju gdje operator unosi podatke

time-based processing – prikupljanje i manipulacija podataka u vremenski kontroliranom procesu

user configuration task – zadatak sa kojim korisnik interaktira kada radi sa procesom

view program – program prikaza u realnom vremenu grafičke aplikacije

Window manager - software koji kreira grafički okružaj u prozorima na računaru. Na primjer, Microsoftov Windows je manager prozora za DOS.

Optimizacija mreže

Optimizacija mreže je važna ukoliko u okviru mrežne konfiguracije postoje spori linkovi (*slow links*). Sporim linkovima se smatraju komunikacione veze sa brzinom prenosa između 2400 bps i 128 kbps. U mrežama u kojima je najsporija komunikaciona veza iznad 256 kbps, nema potrebe za optimizacijom.

FIX koristi klijent-server model za obala-obala (*peer-to-peer*) komunikaciju da bi djelio podatke i alarme u realnom vremenu između čvorova u mreži. Za pristup podacima, možemo smatrati da je View čvor *klijent*, dok je SCADA čvor *server*.

Sve mrežne konverzacije su bazirane na transakcijama. Čvor klijent šalje zahtjev na serverski čvor. Nakon što server čvor djeluje na zahtjev, on šalje odgovor klijentu. Dakle *poruke* u mreži su bilo zahtjevi ili odgovori.

Maksimalna dužina poruke poslana od strane FIXa ka mrežnom interfejsu je 1400 bajta. Jedna 1400 bajtna poruka poslana od strane FIXa može biti prikazana u samoj mreži sa više mrežnih paketa. Sam protokol mrežnog transporta šalje također mrežne pakete da bi upravljao komunikacijom između dva čvora i da obezbjedi da poruke na mreži stižu na odredište bez grešaka.

Sesije

Sav mrežni promet je orijentiran prema sesijama. Transakcije između dva čvora se pojavljuju kao sesija. Prije nego što klijent može inicirati zahtjev, sesija mora biti uspostavljena sa serverom. Sesije su jednosmjerne, to jest transakcije mogu biti započete samo od strane klijent čvora. Da bi se transakcija pojavila u suprotnom smjeru, druga nova sesija mora biti uspostavljena između dva računara.

Dvije sesije se uspostavljaju kada se koristi FIX na NetBIOS mreži. Samo jedna sesija se uspostavlja kada se koristi FIX na TCP/IP mreži. Medjutim i tada se održavaju dvije logičke sesije. Sesija inicirana od strane View čvora se koristi da se dobiju podatci od SCADA čvorova. Druga sesija se uspostavlja od SCADA čvora ka View čvoru da bi se poslali alarmi ka View čvoru. U ovom slučaju, dio *zahtjeva* u transakciji se sastoji od alarmnih podataka i potvrđen je sa *odgovorom* od strane klijent čvora.

Da bi se uspostavila konekcija, adresa udaljenog čvora mora biti dobijena. Matod nalaženja ove adrese, ili adresna rezolucija, je naročito važan u WAN mrežama. NETBIOS tipično koristi broadcast zahtjeve da bi našao adresu udaljenog čvora. Ovi broadcasti se mogu nastaviti vrlo često sve dok udaljeni čvor ne odgovori. Ovi može biti vrlo ometajuće za rad WAN mreže. Zato Intellution preporučuje korištenje FIXa sa TCP/IP protokolom u WAN mrežama. Ova konfiguracija razrješuje mnogo elegantnije ovaj zahtjev za broadcastom.

Prenos podataka

Aplikacioni programi dobijaju podatke u realnom vremenu od SCADA čvora putem zahtjeva. Navažnija FIX aplikacija je View pošto ona najviše i opterećuje mrežu. Linkovi u View programu se ažuriraju birajući SCADA čvor. View ima konfigurabilnu brzinu osvježavanja slika.

Ova brzina određuje kako često svaki link u slici zahtjeva svježiju vrijednost od SCADA čvora. Ako postoji više od jednog linka u slici za neki dati SCADA čvor, svi zahtjevi za ažuriranjem linkova iz tog čvora se šalju u jednoj grupi. Ako zahtjev ili odgovor u toj grupi prelazi 1400 bajta, tada se rastavlja u pakete od po 1400 bajta maksimum. Ako je više od jedne slike otvoreno istovremeno, koje imaju linkove za isti SCADA čvor, šalje se posebna grupa za svaku sliku.

Veličina poruka

Stvarna veličina poruke koja se prenosi pomoću FIXa zavisi od tipa linka koji je konfiguriran u Draw.

Linkovi podataka obezbjeđuju korisniku opciju da zahtjeva ili numerički ili tekstualni podatak. Numerički podatak, tj. F_ polje, se uvijek prenosi kao podatak sa pokretnim zarezom i ima dužinu od četiri bajta.

Tekstualni podatak, tj. A_ polje, se prenosi kao ASCII predstava vrijednosti pokretnog zareza i ima dužinu od 12 bajta. Jedan metod da se reducira opterećenje FIX mreže je da se koriste samo linkovi numeričkih podataka.

Neki specijalni tipovi linkova koriste podatke koji su mnogo veći od 12 bajta. Na primjer, za svaki trend, statistički zapis, i XY zapis, FIX prenosi otprilike 180 bajta.

Linkovi sumarnih alarma su kombinacija niza različitih varijabli. Ove su konfigurabilne, tako da obim podataka koji se zahtjeva može biti značajno reduciran. Na primjer, deskriptorsko polje koje je 40 bajta veliko u uključeno je u default konfiguraciju, se kontinualno izabira iako nema nikakvu čestu promjenu svog sadržaja.

Prenos alarma

Alarmni uslovi se detektuju sa SAC programom koji se izvršava na svakom SCADA čvoru. SCADA čvor šalje alarmnu poruku svakom čvoru na mreži kada se alarm detektuje. Svaki alarm ima dužinu od 132 bajta. Stvarna veličina poruke koja se prenosi od strane SCADA čvora nešto je veća. Ako se nekoliko alarma detektuje u istom ciklusu skaniranja, SCADA čvor grupira što god otkrije alarma u poruku do 1400 bajta dužine, prije nego što distribuira alarme do svakog čvora.

Optimizacija FIX da se reducira saobraćaj na mreži

Neke od tehnika koje korisnik može koristiti da redukuje saobraćaj u mreži uključuju:

- da koristi samo one linkove koje doista i treba. Ako je potrebno više informacija, bolje je kreirati hijerarhiju slika. Postaviti linkove koji daju informaciju koja je važna samo pod određenim uslovima u posebnu sliku i pozivati to sliku samo onda kada se ti uslovi pojave.
- Koristiti numeričke vrijednosti (F_) umjesto tekstualnih vrijednosti (A_). Pored toga što predstavljaju više podataka na mreži, zahtjevi za ASCII vrijednostima također povećavaju obim obrade koji SCADA mora provesti pošto konverzija varijable sa pokretnim zarezom (floating point) u ASCII se također izvršava na SCADA čvoru.

- Koristiti jednu sliku umjesto višestrukih slika da bi se efikasnije grupirale poruke za jedan SCADA čvor .
- Povećati brzinu osvježanja slike da bi se smanjila frekvencija ažuriranja podataka. Default brzina osvježanja je 0.1 sec.
- Minimizirati korištenje linkova trenda , statističkog grafa i XY plota
- Ne uključivati deskriptore ili druge vrijednosti ako one nisu bitne za operatore kod linkova za sumarne alarme.

Konfigurabilni tajmeri sesija

Slijedeći uslovi mogu prouzrokovati da FIX privremeno izgubi sesiju:

- izvršenje FIXa preko mreže koja uključuje spore komunikacione linkove (manje od 256 kbps.
- otvaranje slike koja sadrži mnoge linkove na SCADA čvoru dok istovremeno drugi FIX čvorovi zahtjevaju podatke od tog SCADA čvora. U ovom slučaju FIX mrežni zadatci ne dobivaju dovoljno vremena CPUa koje oni trebaju , zbog toga što se i drugi programi natječu za resurse CPUa.

Obadva ova slučaja se mogu riješiti tako da se povećaju vrijednosti tajmera koje FIX koristi. Ove vrijednosti tajmera se zovu konfigurabilni tajmeri sesije (*configurable session timers*).

FIX tajmeri sesije

FIX koristi tajmere mrežne sesije da odredi da drugi čvor se više ne izvršava. Timeout vrijednosti (vrijednosti isteka vremena), za svaku sesijski orijentirani prenos poruke su definirane od strane FIXa kada se sesija uspostavi.

Slijedeće četiri timeout vrijednosti su definisane:

Send- iznos vremena koji View čvor čeka da bi njegov zahtjev SCADA čvoru bio potvrđen. Ako ovaj tajmer istekne, sesija se završava

Receive- iznos vremena koji View čvor čeka za odgovor od SCADA čvora. Primjetimo da TCP/IP ima svoje vlastito vrijeme timeouta koje iznosi 60 sec. Kada se FIX izvršava preko TCP/IP protokola, efektivni timeout sesije je veći od ove vrijednosti ili minimalno 60 sec. Ako ovaj timer istekne , sesija se završava.

Keep alive – iznos vremena koji, ako se nije pojavila nikakva aktivnost preko uspostavljene konekcije, View čvor čeka prije nego pošalje poruku “otkucaja srca “(heartbeat message – još sam živ).

Inactivity – je iznos vremena koji, ako se nije pojavila aktivnost preko uspostavljene dinamičke konekcije, View čvor čeka prije nego što

ukloni dinamičku konekciju iz liste izlaznih konekcija. Ako ovaj timer istekne , sesija se završava.

Odredjivanje vrijednosti za tajmere sesije

Važno je korektno izabrati timeout vrijednosti za korisnikovu aplikaciju. Vrijednosti koje su vrlo male, mogu prouzrokovati da se sesije izgube iako udaljeni čvor je ispravan i izvršava se. Vrijednosti koje su isuviše velike, mogu usporiti dobijanje obavještenja o problemima u sesiji. Kada se razmatra koje vrijednosto tajmera koristiti , primjetimo da View može “visiti ” (hung) , čitav period timeouta kada se sesija izgubi. U uslovima vodjenja brzih procesa , ovo kašnjenje može ostaviti operatorima malo vremena da reaguju u vanrednim situacijama (emergency situation).

Za većinu aplikacija, default vrijednosti za timeoute su adekvatne. Ako korisnik mora modificirati ove vrijednosti da bi korigovao gubitke kod tranzijenata u sesiji, treba koristiti podatke iz ovog poglavlja da odredi korektne vrijednosti za njegovu aplikaciju.

FIX koristi iste timeout vrijednosti sesije za svaku sesiju. Mjenjajući vrijednosti tajmera za sesiju, ima uticaja na sesije sa svakim čvorom u FIX mreži. Da bi se promjenile vrijednosti tajmera za sesiju, promjene moraju biti učinjene u svakom čvoru.

Konfigurisanje tajmera sesije

Tajmeri sesije se konfiguriraju u FIX SCU. Korisnik treba mjenjati tajmere sesije samo ako mora , i samo onda ako u potpunosti razumije implikacije ovih promjena.

Da bi konfigurirao FIX tajmere sesije:

- [1] Startati SCU
- [2] Izabrati *Network* iz *Configure menu*. Pojaviće se dajalog boks za konfigurisanje mreže .
- [3] Izabrati *Advanced* taster. *Advanced Configuration* dijalog boks će se pojaviti.
- [4] Izabrati *Timers* taster . *Timers* dijalog boks će se pojaviti
- [5] Editirati polje sekundi tajmera sesije koje želimo promjeniti. Kada završimo kliknemo na OK.
- [6] Spasiti i izaći iz SCU.

Mreže širokog obuhvata (WAN – wide area networks)

FIX radi isto tako dobro u WAN mrežama kao što radi i u lokalnim mrežama (LAN). Medjutim za korištenje u WAN mrežama kakve su i industrijske mreže na jednom kompleksu, Intellution preporučuje korištenje TCP/IP protokola.

Daljinski pristup

Korisnik ima na raspolaganju nekoliko izbora da omogući računarima na različitim geografskim lokacijama da međusobno komuniciraju. Medju ovim izborima su programi daljinskog upravljanja i daljinskog pristupa i u novije vrijeme nova generacija FIXa sa Internet mogućnostima ili **iFIX**.

Daljinski kontrolni programi

Daljinski kontrolni programi preuzimaju totalnu kontrolu udaljenog računara i šalju kompletan ekranski prikaz udaljenog računara preko modema , što može rezultirati u sporom izvršenju programa. Primjetimo da je ekran i kod lokalnog i daljinskog računara isti. Unošenje tastera i kretanje miša koje pravimo na lokalnom računaru se preslikavaju i ponavljaju na daljinskom računaru (naprimjer vnc –virtual network connection program)

Programi daljinskog pristupa

Programi daljinskog pristupa, kao što je to Microsoftov servis daljinskog pristupa (remote access service –RAS) , tretiraju čvor koji je lociran na udaljenoj geografskoj lokaciji kao da se nalazi na lokalnoj mreži. Sami podatci realnog vremena se prenose preko modema. Ovaj metod daljinskog pristupa obično je adekvatniji za daljinski nadzor FIX sistema.

Microsoftov RAS ne zahtjeva da neki posebni računar ispunjava ulogu *gateway-a* u ovom povezivanju. Mnogi drugi softwareski proizvodi za ovaj daljinski pristup zahtjevaju neku vrstu posvećenog računara da izvršava gateway funkcije između LANa i asinhronih linije.

Servis daljinskog pristupa (RAS – remote access service)

Microsoftov RAS obezbjeđuje NetBEUI i TCP/IP protokole preko telefonske linije , tretirajući modem kao mrežni adapter. Na taj način NetBIOS aplikacije koje se izvršavaju preko mreže mogu se izvršavati i preko serijske konekcije. FIX može koristiti NetBEUI interfejs koji je uključen u RAS da komunicira sa drugim FIX čvorovima preko asinhronih linija.

RAS klijent i server software su standardni sa Windows NT.

Općenito, RAS klijent čvor bira RAS server čvor. RAS klijent može birati i pristupiti resursima na NT RAS server mreži. Windows NT Workstation software dozvoljava jednog RAS klijenta. Windows NT 4,0 Server software dozvoljava više RAS klijenata da budu istovremeno spojeni.

Mada RAS dozvoljava i do 256 klijenata da simultano biraju Server, praktična granica kada se RAS koristi sa FIXom je značajno manja. Ovo je uzrokovano time što FIX zahtjeva dodatne resurse da bi komunicirao. Svaka simultana konekcija zahtjeva poseban modem na serveru. Prije nego što korisnik inkorporira RAS u njegov proizvodni okružaj, preporučuje se testiranje konfiguracije za višestruke konekcije.

Korištenjem FIXa u sprezi sa RAS serverom, pružaju se slijedeće mogućnosti:

- sumarni displeji alarma su na raspolaganju
- podatci i alarmi su na raspolaganju sa jednim parom alarma
- rep alarma kod starta (alarm startup queue)

- transfer fajlova korištenjem Microsoftovog mrežnog softwera preko RASa
- Višestruki daljinski čvorovi mogu simulatano birati i priključiti se na server

Korisnik treba koristiti slijedeće preporuke kod inkorporiranja RAS u FIX mrežu:

- koristiti modeme sa minimalnom brzinom od 9600 bauda. Verificirati da kada se uspostavi RAS konekcija da su modemi spojeni sa velikom brzinom. Što je veća brzina, može se očekivati bolja performansa.
- Koristiti status porta i na RAS klijentu i serveru da se istraže problemi konekcije i nadzire komunikacija koja je u toku.
- Kada se koristi podrška ? koju pruža FIX software, na RAS klijentu da bi pristupio udaljenom SCADA čvoru, može se poboljšati performansa da se kopira SCADA fajl čvora, *nodename.TAG*, iz SCADA čvora (u stazi PDB) u View čvor u istu stazu (direktorij) PDB.
- Slijediti korake koji su dati u nastavku ovog teksta da se omogući FIX RAS klijent čvoru da bira u Windows NT RAS server i pristupi drugim FIX čvorovima. Primjetimo da ovi koraci vrijede samo za FIX koji koristi NetBEUI .

Na RAS serveru:

[1] Konfigurirajte na Network control panelu korištenje NetBEUI sa mrežnim adapterom kao LAN Adapter 0. Takodjer konfigurirajte Network Control panel da koristi NetBEUI sa modemom kao LAN adapter sa brojem različitim od 0. (ovo je opisano u nastavku).

[2] postaviti Remote listen parametar u registru na 2. (opisano u nastavku)

Na RAS klijentu:

[1] konfigurirati na network Control panelu da koristi NetBEUI sa modemom kao LAN adapter 0.

[2] Povećati brzinu osvježavanja na slici na 3 sec ili više. (opisano u nastavku).

Povećanje brzine osvježavanja

Da bi se povećala brzina osvježavanja na svim slikama, uraditi slijedeće:

[1] Startati sa brzinom osvježavanja od 3 sec na svim slikama

[2] Otvoriti sliku u View da se utvrdi da da sesije se ne gube i da se alarmi primaju na vrijeme.

Ako se neke od sesija gube ili se alarmi ne primaju u razumnom intervalu vremena, povećati brzinu osvježavanja u inkrementima od 1 sec sve dok nema više izgubljenih sesija i alarmi se primaju bez velikog kašnjenja.

RAS primjeri korištenjem FIX preko NetBIOSa

Ovo poglavlje ilustrira mogućnosti prenosa podataka i alarma u FIXu u slijedećim tipovima RAS komunikacija:

- point to point (tačka – tačka)
- point to LAN (tačka- lokalna mreža)
- LAN to point
- LAN to LAN

U svakom od ovih slučajeva NODE A je RAS klijent a NODE B je RAS sever

Point to point

Slijedeća slika ilustrira komunikaciju sa daljinskim pristupom između dva samostalna čvora.



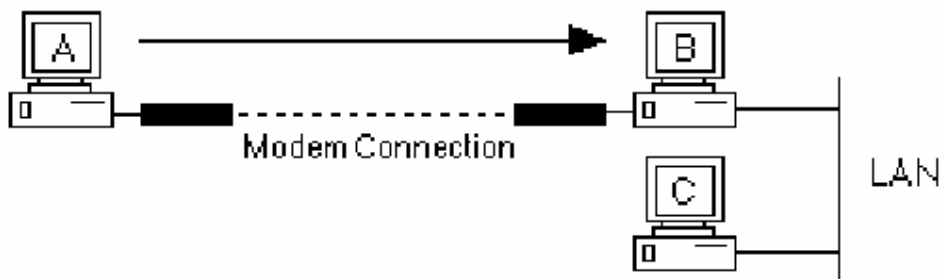
Point to point komunikacija

Naredna tabela ilustrira komunikacione mogućnosti RASa u ovom scenariju:

Čvor	Može primiti podatke i alarme iz čvora
A	B
B	A

Point to LAN

Naredna slika ilustrira udaljeni samostalni čvor A koji bira u čvor B u LAN-u.



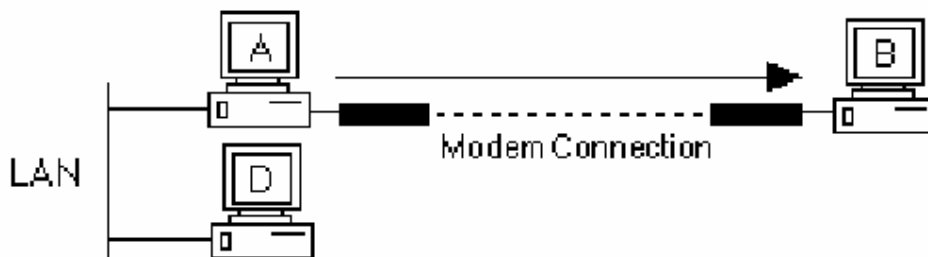
Point to LAN komunikacija

Naredna tabela pokazuje komunikaciju koja postoji u ovom scenariju.

Čvor	Može primiti podatke i alarme iz čvora
A	B,C
B	A,C
C	A,B

LAN to point

Naredna slika ilustrira umreženi čvor A koji bira u udaljeni samostalni čvor B.



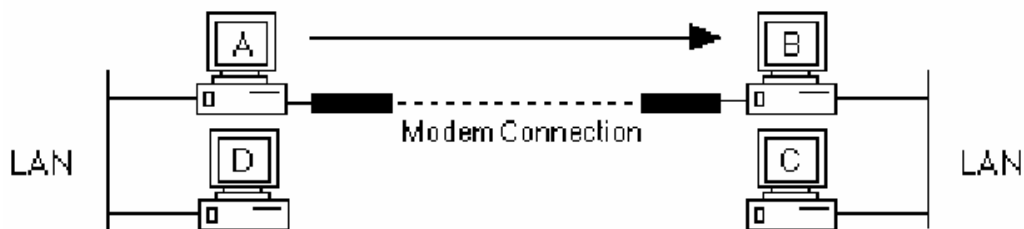
LAN to point komunikacija

Naredna tabela pokazuje komunikaciju koja postoji u ovom scenariju.

Čvor	Može primiti podatke i alarme iz čvora
A	B
B	A
D	Nema

LAN to LAN

Slijedeća slika ilustrira komunikaciju između dva LAN-a.



LAN to LAN komunikacija

Naredna tabela pokazuje komunikaciju koja postoji u ovom scenariju.

Čvor	Može primiti podatke i alarme iz čvora
A	B, C
B	A, C
C	A, B
D	Nema

Integracija mreže

Korisnik može imati komunikacione zahtjeve pored FIX komunikacija koje također moraju biti razmatrani. Ovi mrežni i komunikacioni zahtjevi su obično u slijedećim kategorijama:

- dodatni mrežni software zahtjevan od strane I/O drajvera koji korisnik koristi
- Fajl server komunikacioni software da bi se omogućila centralna arhiva i pohranjivanje fajlova
- Komunikacioni software relacije baze podataka

Integracija svih ovih zahtjeva u jedinstven softwareski sistem može biti dosta zahtjevan zadatak.

Procesna baza podataka

FIX omogućava korisniku da konfigurira sistemski okružaj koji obezbjeđuje:

- Nadzorno upravljanje, šaržno upravljanje procesom (batch processing), kontinualno upravljanje , statističko upravljanje procesima
- Procesne informacije za operatore i inženjere procesa, nadzornike, u obliku izvještaja, prikaza, arhiviranih podataka, alarmnih poruka, i statističkih grafova.

Izvor svih ovih informacija je proces (regulatori, transmiteri, senzori, motori, prekidači, itd.). Izvor informacija na distribuiranoj mreži je jedan ili više FIX baza podataka koje su rezidentne na SCADA čvorovima.

Baza podataka ima integrativnu ulogu u strategiji gradnje sistema upravljanja, ona je primarni izvor procesnih informacija u realnom vremenu za sve aplikacione programe. Bilo da prikupljamo historijske podatke o procesu ili generiramo smjenske izvještaje, FIX software omogućuje da kreiramo bazu podataka koja će podržavati svaku od specifičnih potreba .

Graditelj baze podataka (database builder)

Korisnik može kreirati i upravljati bazama podataka za SCADA čvorove koristeći *Graditelj baze podataka*. Ovaj program omogućuje korisniku da čini slijedeće:

- kreira i modificira bazu podataka

- kreira i spašava zahtjeve za pretraživanje (queries)
- sortira polja baze podataka
- automatski generira višestruke blokove baze podataka
- uvozi i zvozi iz baze
- kustomizira (kroji po mjeri korisnika) prikaze

Graditelj baze podataka može takodjer prikazati informacije u tabličnom prikazu (spreadsheetu). Ovaj spreadsheet omogućava korisniku da lagano prolazi kroz bazu podataka i nadzire višestruke tačke baze. Dodatno, spreadsheet je idealan za upite , sortiranja, i pretraživanja baze.

Razumjevanje baze podataka

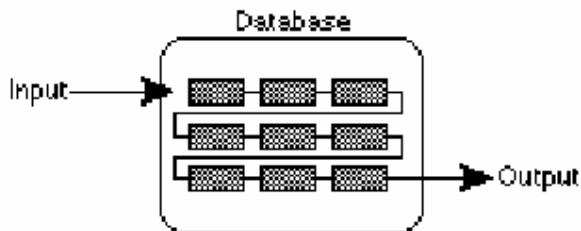
Prije nego što korisnik koristiti softwareski alat : Graditelj baze podataka da bi kreirao bazu, on treba da razumije funkciju procesne baze podataka, njene blokove, i njene lance.

Kako radi baza podataka

Srce svakog SCADA čvora je njegova baza podataka. Namjena baze podataka je da:

- primi ulazne vrijednosti od drajvera tabele slike (DIT)
- manipulira procesnim vrijednostima u skladu sa instrukcijama korisnika sadržanim u kontrolnoj strategiji.
- Poredi ulazne vrijednosti sa alarmnim granicama koje zada korisnik
- Izbaci podesne vrijednosti na DIT tabelu
- Pošalje alarmne signale na operatorske displeje, printere, fajlove i alarmne uređjaje u mreži.

Naredna slika ilustrira kako vrijednosti ulaze u bazu podataka , prolaze kroz sekvencu blokova i izlaze iz baze kao izlazi.



I/O podatci i baza podataka

Kako blokovi funkcioniraju

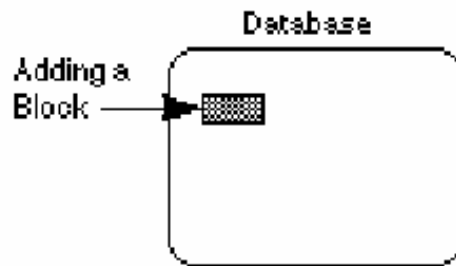
Blokovi baze podataka

Baza podataka je sačinjena od *blokova*. Blokovi su individualne jedinice instrukcija sposobne za neke ili sve od slijedećih mogućnosti :

- primaju vrijednosti bilo od drugog bloka ili direktno od DIT tabele
- manipuliraju vrijednostima u skladu sa instrukcijama korisnika
- porede vrijednosti sa njihovim alarmnim granicama
- skaliraju procesne vrijednosti u date opsege inženjerskih jedinica
- izvršavaju kalkulacije nad podacima
- izbacuju vrijednosti natrag u DIT tabelu nakon obrade

FIX software obezbjeđuje različite tipove blokova, svaki sposoban da izvršava jedinstvenu funkciju.

Kada se FIX software instalira na SCADA čvoru, on kreira praznu bazu podataka. Koristeći graditelj baze podataka, korisnik može dodati blokove koji su mu potrebni da izvršavaju jedan ili više koraka u upravljanju procesom.



Nakon toga, korisnik definira specifične instrukcije za blok koristeći dijalog boks koji Graditelj baze pokaže. Ove informacije uključuju:

- ime bloka (koje se naziva *Tag*)
- kako on prima vrijednosti ili informacije iz DIT tabele ili od nekog drugog bloka
- gdje on šalje informaciju
- da li manipulira sa vrijednostima
- kako reagira na kritične vrijednosti ili njihove promjene(alarmne vrijednosti)

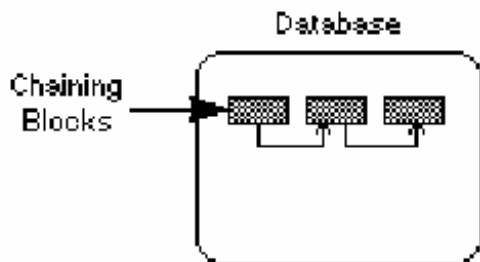
- kako skalira vrijednosti radi prikaza na operatorskim displejima

Kako funkcionira lanac

Lanci baze podataka

Blokovi sami od sebe izvršavaju specifične zadatke u bazi podataka. Cilj korisnika kod kreiranja baze je da poveže blokove u sekvencu tako da svaki blok izvršava specifični zadatak a zatim prosljedjuje informaciju na jedan ili više narednih blokova. Povezivanjem dva bloka , korisnik kreira lanac. Kada su blokovi konfigurirani i sekvencirani u lanac, blokovi mogu verificirati, automatizovati, alarmirati, prikupljati podatke, i održavati proces.

Kod automatizovanja velikih procesa, koji uključuju veliki broj I/O uređaja, baza podataka može sadržavati jako mnogo lanaca, od kojih je svaki dizajniran da automatizira i održava specifičnu funkciju ili procesni korak.



Svaki lanac može sadržavati do 30 blokova, od kojih je svaki blok konfiguriran da izvršava specifičnu procesnu funkciju. Primjetimo da neki blokovi su dizajnirani da rade u lancu dok su drugi dizajnirani da rade samostalno.

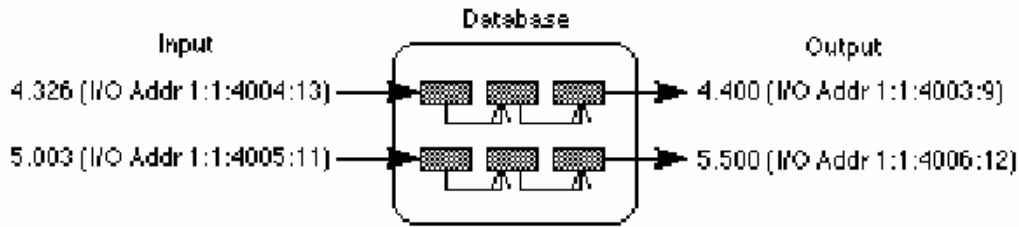
Procesiranje lanaca

SAC program (skanira , alarmira, upravlja)

Nakon što je korisnik izgradio lanac, SAC program procesira instrukcije svakog bloka u specifičnim intervalima vremena koje je definirao korisnik. Glavna namjena SAC programa je da:

- skanira individualne blokove u lancu, uzimajući nove I/O vrijednosti i vraćajući izračunate i obradjene vrijednosti na I/O uređaj
- provjerava na alarme ulazne vrijednosti da vidi ima li uslova za alarme koji su definirani u blokovima
- kontrolira proces automatizacije obezbjedjujući da svaki blok verificira ili manipulira ulaznim vrijednostima u skladu sa instrukcijama i prenosi odgovarajuće vrijednosti na slijedeći blok u lancu

Naredna slika pokazuje kako SAC uzima ulazne podatke, prolazi sa njima kroz lance baze podataka i vraća ih kao izlazne vrijednosti:



SAC procesiranje I/O podataka

DIZAJNIRANJE BAZE PODATAKA

Prikupljanje informacija o procesu

Prije nego što počne da koristi graditelj baze da bi kreirao bazu podataka, korisnik treba da prikupi slijedeće podatke:

- Flowchart procesa (dijagrame toka procesa)
- Listu adresa DIT tabele za dati I/O drajver
- Zahtjeve na alarme
- Tipove kondicioniranja signala koje će koristiti i opsege vrijednosti za I/O hardware koje će on mjeriti. Opseg vrijednosti se naziva granice blokova (*block limits*) ili takodjer kao EGU opseg .

Skupljajući ove informacije na jednom mjestu , korisnik može uštedjeti vrijeme kada bude spreman za dizajniranje njegove baze podataka.

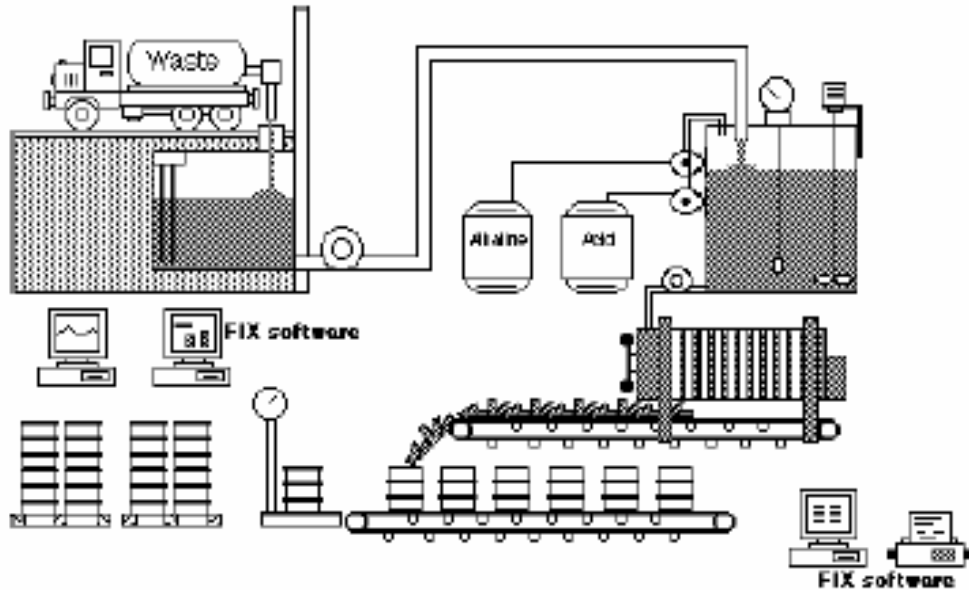
Kada ima gornje podatke prikupljene, korisnik može početi sa dizajniranjem baze podataka. Da bi ovo ilustrirali, mi ćemo u nastavku ovog poglavlja proći kroz postupak dizajniranja lanca za ENVIRO kompaniju. Koristeći ove primjere, korisnik će biti u stanju da kreira svoje blokove i lance u njegovoj bazi podataka.

Primjer aplikacije

Kompanija : ENVIRO kompanija skuplja mulj koji se generira od strane postrojenja za preradu gradskih otpadnih voda i pretvara ovaj mulj u različite tipove poljoprivrednog đubriva u formi briketa. Prema tome ENVIRO kompanija uključuje nekoliko tipova procesa:

- kontinualno upravljanje
- šaržno (batch) upravljanje
- diskretno upravljanje

Naredna slika prikazuje konfiguraciju ENVIRO postrojenja i lokaciju FIX čvorova na djelovima postrojenja:



Primjer procesne aplikacije

Aplikacija: ENVIRO kompanija želi da automatizira proizvodnju đubriva i nadzire potrošnju kemikalija i korištenje opreme. Naredna tabela izlistava ove proizvodne zadatke , objašnjava kako FIX software može izvršavati ove zadatke i identificira ekonomske dobiti.

Osnovni elementi u ovoj aplikaciji mogu se primjeniti na procese u drugim industrijama kao što su kemijski procesi, procesiranje hrane i čak diskretni proizvodni procesi (proizvodnja djelova za automobile i slično).

Ekonomske beneficije:	Prozvodni taskovi:	FIX Aplikacija:
<ul style="list-style-type: none"> ■ Povećanje proizvodnje putem planiranja. ■ Proizvodnja kvalitetnijeg proizvoda ■ Povećana raspoloživost postrojenja kroz planirano preventivno održavanje. ■ Nizi operativni troškovi zbog boljeg korištenja materijala i nize troškove rada ■ Povećan odziv na tržišne uslove kroz variranje veličine proizvoda, sastava, boje i sadržaja vode. ■ Povećani profit. 	<ul style="list-style-type: none"> • Planiranje proizvodnje sa narudjbama . ■ Procesiranje mulja u đubrivo ■ Proizvodnja različitih vrsta đubriva po zahtjevu. ■ Nadzor korištenja kemikalija i održavanje skladišta. ■ Održavanje opreme. ■ Održavanje evidencije proizvodnje. ■ Arhiviranje procesnih podataka. 	<ul style="list-style-type: none"> ■ Planiranje receptura za batch procesiranje. ■ Izvršava nadzorno upravljanje , direktno digitalno upravljanje (DDC) , i alarmiranje ■ Download recepture za svaku vrstu đubriva. ■ Trendiranje kemijskih nivoa i stampanje izvještaja o korištenju kemijskih reagenasa. ■ Trendiranje rada pumpi i njihovog korištenja i izvještaja o održavanju. ■ Historijsko trendiranje procesa. ■ Totaliziranje proizvedenih količina po svakom batchu i proizvodnom ciklusu radi statističke kontrole kvaliteta.

Automatizacija procesa sa FIX (primjer)

Dizajniranje lanca

Jedanput kada znamo zadatke koje treba FIX da izvršava , možemo dizajnirati lance za bazu podataka. Najlakši i najefikasniji način da dizajniramo funkcionalni lanac je da koristimo slijedeće korake dizajna:

Koraci dizajna

[1] Analizirati proces i napraviti zapis podataka koji su nam potrebni, uključujući:

- * I/O adrese kontrolera I/O uređaja (listu ožičenja)
- * tipove uređaja
- * kondicioniranje signala
- * granice blokova (EGU opsege)

[2] Dizajn i strategija automatizacije koja objašnjava kako želimo da proces bude automatiziran. Naprimjer:

- Koje tipove alarma i alarmnih granica želimo uspostaviti
- Kako često želimo da kontura ili lanac budu procesirani

- Kada želimo da operatori budu informirani o događajima u procesu
- Kako želimo da se rješava problem prekida u radu procesa

[3] Kreiranje algoritma koji kombinira analizu procesa i strategiju automatizacije. Ovo će odrediti kako će FIX software automatizirati proces i obezbjeđuje specifične instrukcije koje ćemo unjeti u blokove za dijalog boksove kada se budu pojavljivali.

[4] Uparuje broj koraka u algoritmu sa blokovima koji su stanju da izvršavaju ove korake.

[5] Crta flowchart , izlistavajući sve tipove blokova koji izvršavaju svaki korak zajedno sa specifičnim procesnim instrukcijama za svaki blok.

Primjer prikazan u slijedećem poglavlju ilustrira proces razmišljanja uključen u dizajniranje jednostavnog lanca.

Opis jedne procesne aplikacije

Jedan korak u procesu konverzije mulja u ENVIRO kompaniji je neutralizacija kiselina i lužina u mulju. Proces tretmana uključuje četiri koraka:

[1] Dodavanje i mješanje vode sa muljem

[2] Podešavanje pH vrijednosti mulja da se neutraliziraju kiseline i lužine.

[3] Dodavanje sodijuma, azota i potaše da se proizvedu različiti tipovi đubriva

[4] Pumpanje preradjenog mulja do filterskih presa gdje se otklanja suvišna voda iz briketa.

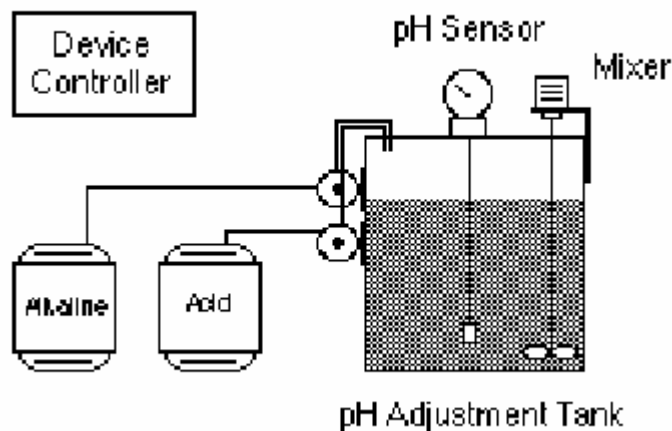
Nastavak ovog poglavlja će se fokusirati na drugi korak: podešavanje pH vrijednosti.

Mulj iz tanka za prikupljanje se periodično pumpa u veliki tank gdje mu se dodaje voda da bi se razrijedio i bio lakši za mješanje.

Senzor u tanku registruje pH vrijednost mulja. Ako je mulj kiseo (acidic), uređaj za upravljanje uključuje pumpu koja dodaje lužasti rastvor (alkaline), da bi podigla pH vrijednost ($\text{pH} = 7$ za neutralne rastvore, manji od 7 za kiseline i veći od 7 za lužine).

Ako je mulj pak lužast ($\text{pH} > 7$), kontrolni uređaj uključuje pumpu koja dodaje kiselkast rastvor, da bi se smanjila pH vrijednost. U oba slučaja , pH vrijednost mulja bit će korektno podešena do vrijednosti koja je potrebna za dati tip đubriva koji se proizvodi, prije nego što se mulj ispumpa do filterske prese.

Naredna slika pokazuje tank sa uređajem za podešenje pH vrijednosti:



Ono što sada treba da uradi korisnik je da dizajnira lanac koji ispituje i podešava pH vrijednost mulja. Zadatci upravljanja koje treba izvršiti su:

- Akvizicija podataka (da se dobiju podatci o pH vrijednosti od senzora)
- Direktno digitalno upravljanje (DDC) da se sekvenciraju pumpe i na taj način održava pH vrijednost.

Kada se proces automatizira, informacija o fluktuaciji pH vrijednosti, korištenje kemikalija, korištenje pumpi, vrijeme kvara (van pogona), i ostale statističke informacije mogu se trendirati kao historijski trend, štampati u izvještajima, i nadzirati od strane operatora. Podešavanje procesa može biti izvršeno direktno sa bilo kojeg FIX čvora koji ima operatorski interfejs.

Analiza demo procesa

Prvi korak u automatizaciji procesa je analiza rada procesa. Korisnik treba da izolira i registrira sve specifične zadatke koji moraju biti izvršeni da se kompletira proces. Na primjer, zadatak može biti : “ ako je pH veća od 8.5 , dodati kiselinasti rastvor dok ne dosegne 7.5”.

Treba imati u vidu da ovaj korak određuje da li jednostavni lanci ili konture u bazi podataka mogu automatizirati proces. Pokušaj automatizacije kompleksnog procesa traži uključanje velikog broja koraka sa jednim lancem i dobro poznavanje FIXa i njegovih blokova. Razvojem detaljne analize procesa omogućava korisniku da izolira one procese koji se lagano mogu adaptirati na jednostavne lance i da identificira one koji će zahtijevati dizajn kompleksnijih lanaca ili kontura.

Koraci koji su potrebni da se podešava pH vrijednost su:

[1] testirati pH vrijednost svaki sat

[2] Ako je pH vrijednost ispod 5.5 , postepeno dodavati lužasti rastvor, ako je iznad 8.5, postepeno dodavati kiseli rastvor.

[3] Prestati dodavanje rastvora kada je pH vrijednost između 6.5 i 7.5

Mada ova analiza opisuje kako proces radi, njoj nedostaje važna informacija o kontrolnom uređaju. Korisnik treba da doda detaljnu informaciju o adresama u DIT tabeli, tipu kontrolnog uređaja, kondicioniranju signala, i maksimalnom opsegu koji uređaj može prihvatiti. Ukratko, treba registrovati sve informacije o kontrolnom uređaju. Nakon što su ove informacije dobijene, korisnik može produžiti sa detaljnom analizom svog procesa. Naredna tabela predstavlja jedan primjer ovakvog uređaja koji uključuje mogućnost mjerenja i kontrole pH vrijednosti:

Analogna adresa pH senzora	1:1:30001
Digitalna adresa pumpe za kiselinu	1.1:40004:8
Digitalna adresa pumpe za lužinu	1:1:40004:5
Tip uređaja	XYZ
Kondicioniranje signala	XYZ
Granice bloka	0-65535

Analiza procesa opisuje postupak podešenja pH vrijednosti, koji uključuje:

- I/O adrese
- Tip kontrolnog uređaja
- Tip kondicioniranja signala
- Granice broja blokova

Dizajniranje strategije automatizacije za demo proces

Dizajniranje lanaca

Slijedeći korak u dizajniranju lanca u bazi je planiranje uspješne strategije automatizacije. Ovo zahtjeva od korisnika da odredi kako da automatizira korake koje je odredio u analizi procesa. Pošto su pH senzor i pumpa upravljani električnim signalima putem uređaja kontrolera, oni mogu biti manipulirani preko njihovih adresa u DIT tabeli.

Znanje o tome kako proces radi je prvi korak u njegovoj automatizaciji. Znati kako želimo da automatiziramo proces je slijedeći logički korak u razvoju strategije automatizacije. Naredna tabela prikazuje strategiju automatizacije za naš demo proces podešenja pH vrijednosti.

- primi pH informaciju od senzora u podešenim intervalima
- provedi alarmiranje. Postaviti vrijednosti za alarmne granice : HIHI, HI, LO, LOLO i ROC (rate of change – brzina promjene
- sekvenca pumpi da bi se održala pH vrijednost
Ako je pH ispod 5.5 , uključiti pumpu za lužinu

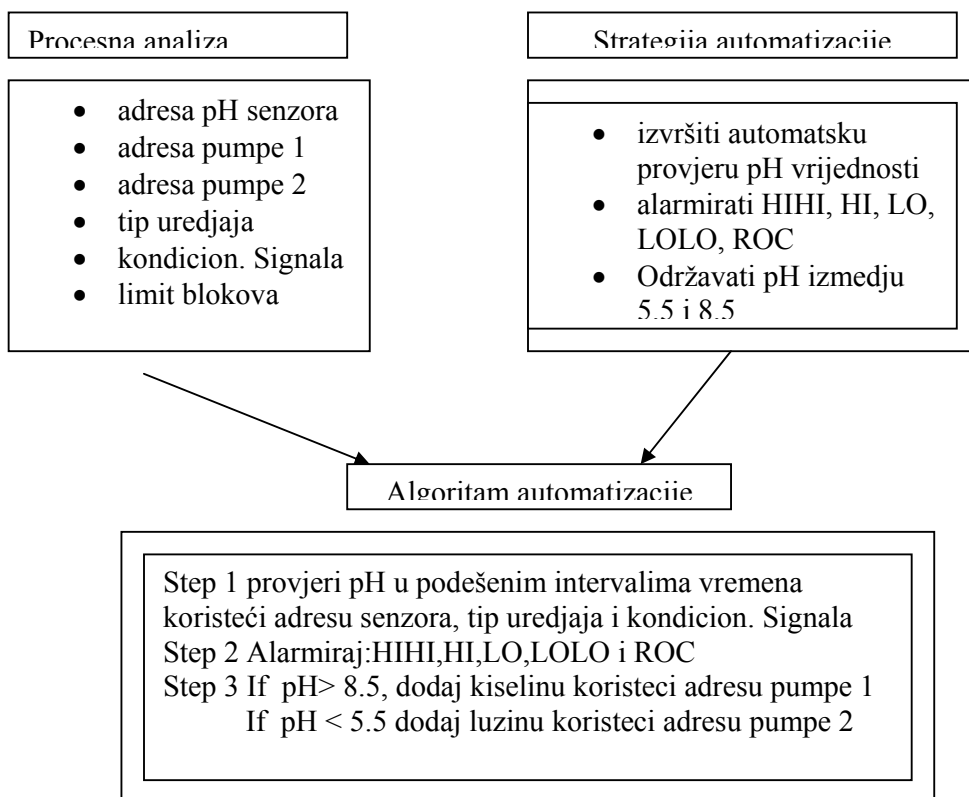
Ako je pH iznad 8. , uključiti pumpu za kiselinu

- Ponavljati prethodne korake u podešenim intervalima vremena.

Primjetimo kako ova strategija automatizacije uključuje i plan alarmiranja za manipulaciju u slučaju procesnih poremećaja, kao što su ekstremne vrijednosti fluktuacije pH vrijednosti (ROC – brzine promjene). Ovo omogućava operatorima da primjete kada su bačve sa rastvorima kiselina i lužina prazne i treba ih zamjeniti kao i da će otkazi u radu opreme biti registrovani od strane operatora i reagovati na njih.

Pisanje algoritma automatizacije za demo proces

Slijedeći i najvažniji korak u dizajniranju lanca za podešenje pH vrijednosti je pisanje algoritma koji kombinuje i procesnu analizu i strategiju automatizacije. Naredna slika pokazuje kako tehničke specifikacije definirane u analizi procesa su kombinirane sa potrebama automatizacije nevedenim u strategiji automatizacije.



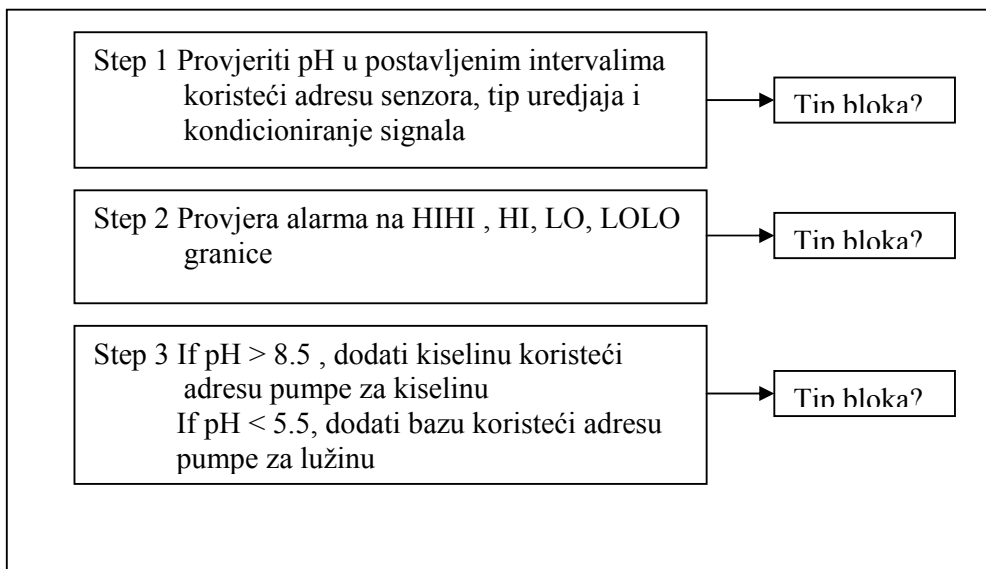
Algoritam čini bazu lanca, on identificira red u kojem se izvršava proces, gdje svaki korak prikuplja informaciju, i kako se informacija prikuplja, procesira ili verificira.

Uparivanje procesnih koraka sa tipovima blokova

Biranje blokova

Slijedeći korak je nalaženje korektnih blokova baze podataka da izvrše korake definisane u algoritmu. Proces transformira algoritme od pisanih koraka u šematiku lanca. Šta više, ovaj korak osigurava da su mogućnosti softwera inkorporirane u procesne planove. Jedanput kada je korak u dizajnu kompletiran, informacija se može lagano formatirati u flowchart i unjeti u SCADA bazu za čvor.

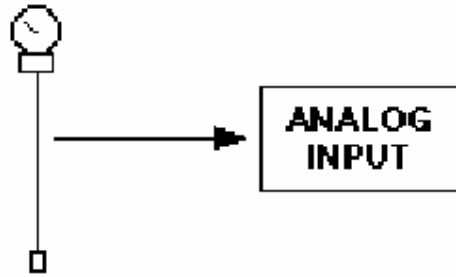
Automatizacioni algoritam



Kada uparujemo blokove sa koracima, treba izabrati blok koji može izvršiti funkciju koraka. U mnogim slučajevima, jedan tip bloka može manipulirati sa više od jednim korakom, dok neki koraci mogu uključivati suviše mnogo funkcija ili isuviše kompleksnih operacija za bilo koji blok.

Za proces podešenja pH vrijednosti, slijedeći blokovi zadovoljavaju algoritam:

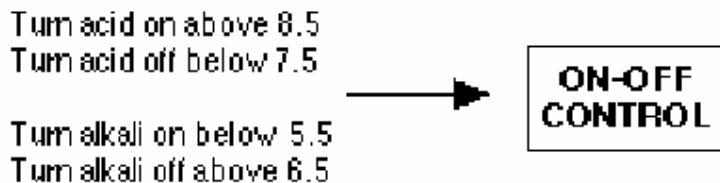
Step 1: Analogni ulazni blok može primiti analogne signale od adrese pH Senzora.



Step 2: Analogni ulazni blok može također realizovati zahtjeve alarmiranja, tako da ćemo i ovaj zadatak doznačiti ovome bloku:



Step 3: Pumpe se uključuju ili isključuju zavisno od vrijednosti koje analogni ulazni blok prima od pH senzora. Blok koji se zahtjeva za step 3 mora biti u stanju da nadzire analogni ulaz i , zavisno od vrijednosti tog ulaza , otvori ili zatvori digitalni izlaz.



Digitalni izlazi se šalju direktno na adrese pumpi za kiselinu i lužinu (bazu). Kontrolni blok On-Off može izvršiti ovu funkciju.

Pošto On-Off kontrolni blok može uključiti i isključiti pumpe, bez korištenja bilo kojeg drugog bloka, ovo kompletira treći korak.



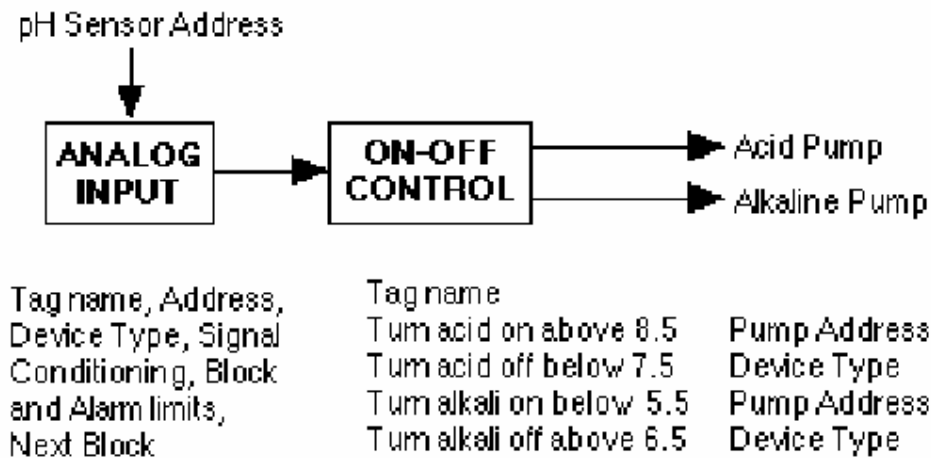
U ovom trenutku smo identificirali blokove koji mogu izvršiti sve korake u našem algoritmu.

Crtanje flowcharta za jednostavni lanac

Posljednji korak je crtanje flowcharta koji pokazuje :

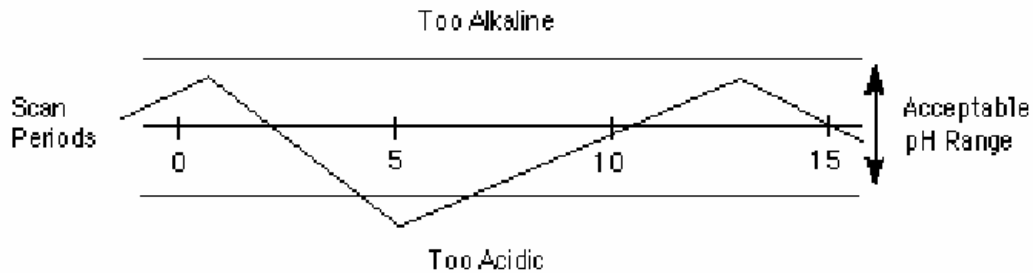
- ulančene blokove
- imena svakog bloka
- I/O adrese
- Dodatne instrukcije o tome kako svaki blok manipulira sa informacijom

Flowchart je vizuelna predstava lanca i pokazuje instrukcije koje će korisnik unjeti u svaki blok kako ih bude dodavao u bazu podataka. Naredna slika pokazuje flowchart za proces podešavanja pH vrijednosti.



Flowchart lanca

Demonstracioni problem: Testiranje i podešavanje pH svake 4 minute radi zadovoljavajuće većinu vremena, ali sa vremena na vrijeme proces podešavanja naruši prihvatljive granice za pH vrijednosti , upumpavajući previše kiselog rastvora , što dovodi do smanjenja pH vrijednosti van dopuštenih granica. Naredna slika ilustrira ovaj problem:



Primjer pH problema

Mi želimo da imamo mogućnost upumpavanja kiseline u kraćim intervalima vremena od perioda skaniranja od 4 minute. Ovo će dati kiselinu više vremena da reagira sa muljem prije nego što kiselina udje u tank. Jedno dodatno poboljšanje bilo bi da skratimo vrijeme skaniranja lanca, obezbjeđujući na taj način više pH smplova da bi odredili koji rastvor treba biti upumpan u tank.

Ponovno ispitujući algoritam automatizacije, možemo odrediti gdje su moguća poboljšanja u lancu. Reprudokovaćemo u nastavku originalni algoritam koji je doveo do On-Off kontrolnog bloka.

Step 3

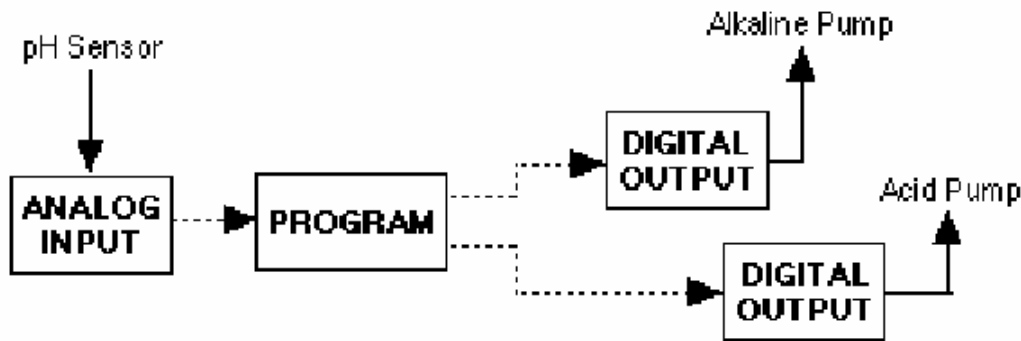
- Ako je pH ispod 5.5 , uključiti pumpu da doda bazni rastvor. Ako je pH iznad 8.5 , uključiti pumpu koja dodaje kiseli rastvor.
- Digitalna adresa pumpe za kiselinu je : 1:1:40004:8
Digitalna adresa pumpe za bazu je 1:1:40004:5
- Ako je pH između 5.5 i 8.5 isključiti obadvije pumpe

Sa ovim mi želimo da modifikujemo algoritam tako da pumpa za kiselinu bude uključena kraće intervale vremena, dajući vremena kiselinu da smanji vrijednost pH , prije nego se pumpa za kiselinu ponovo uključi ako smanjenje nije bilo dovoljno.

Da bi implementirali ovaj algoritam, moramo ispitivati i druge blokove u okviru biblioteke FIXa koji mogu izvršavati ovakav zadatak.

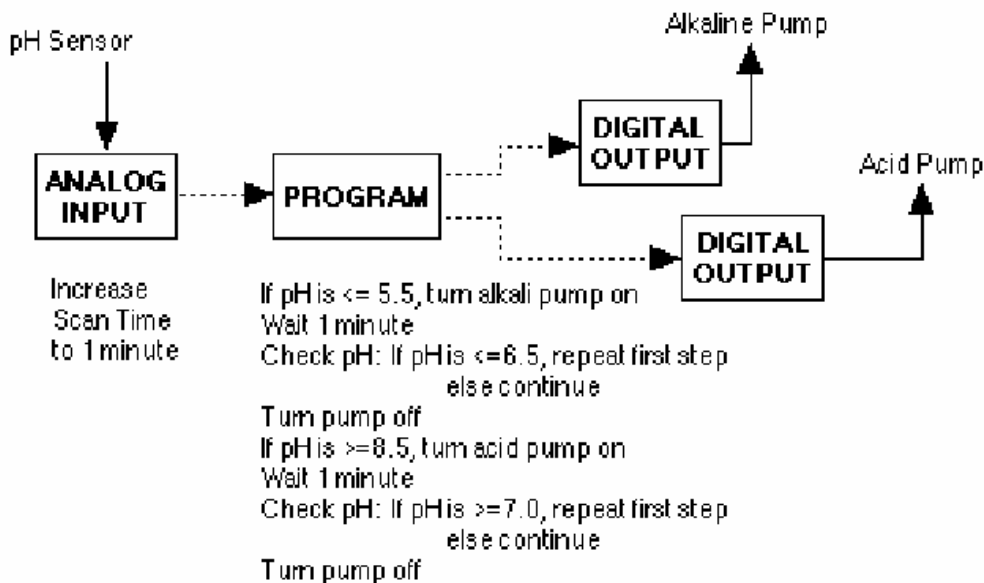
Jedan blok koji može da uključi i isključi pumpu je programski blok. Možemo zamjeniti On-Off kontrolni blok sa Programskim blokom. Medjutim, programski blok ne može uspostaviti direktno digitalno upravljanje sa pumpama za kiselinu i bazu. Za to moraćemo koristiti dva Digitalna izlazna bloka, po jedan za svaku pumpu.

Naredna slika pokazuje zamjenu On-Off kontrolnog bloka sa Programskim blokom i dva bloka Digitalnog izlaza, i prikazuje kako informacija prolazi iz jednog bloka u naredni.



Modifikacija lanca

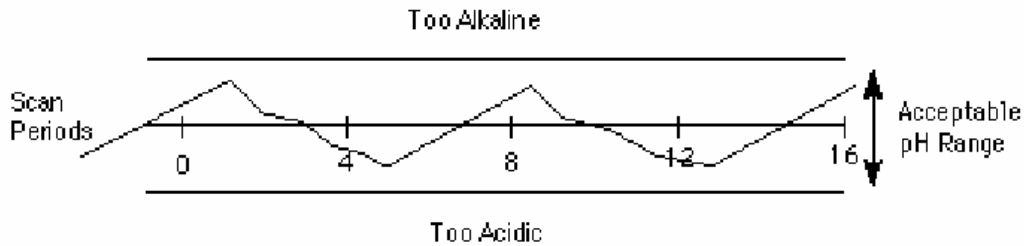
Slijedeća slika prikazuje nove programske instrukcije potrebne da se izvršavaju programski blok i blokovi digitalnih izlaza:



Modifikovani flowchart

Sa ovom modifikacijom i sa blokom Analognog ulaza, i vremenom skaniranja reduciranom na jednu minutu, lanac kontrolira podešavanje pH vrijednosti mnogo efikasnije, dozvoljavajući Programskom bloku da izvršava kratki program koji podešava pH vrijednost.

Naredna slika pokazuje kako modificirani lanac kontrolira pH podešavanje:



Trend zapis korektnog podešenja pH vrijednosti

Sada bi možda željeli da razmatramo mogućnost kako da obezbjedimo i više kontrole nad procesom nadzora. Na primjer, ako operator treba da nadzire pH fluktuacije, možemo koristiti blok trendiranja sa više pera (Multi-Pen trend). Ili Multi-Bar trend. Ovi blokovi obezbjedjuju kako trendiranje u realnom vremenu.

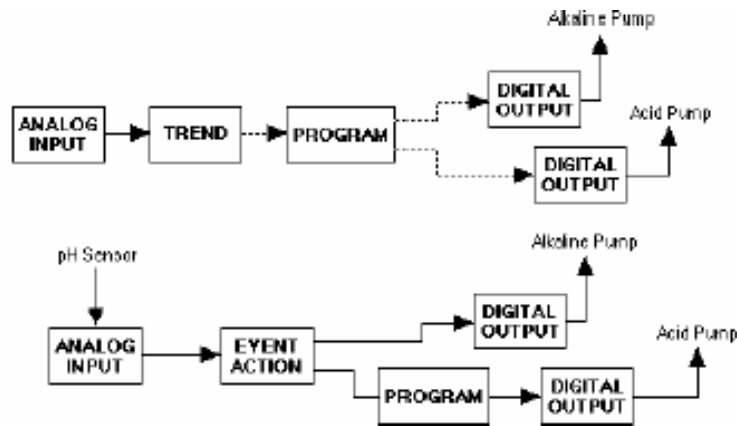
Naredna slika pokazuje kako pH trend može da izgleda operatoru:



Izgled trenda na operatorskom displeju

Podatci koji se trendiraju se ovim linkom ne pohranjuju. Da bi se pohranili podatci iz bloka Analognog ulaza, i koristili u historijskom trendu, potrebno je dodati Trend blok u lanac. Sa Multi-Pen trend li Multi-Bar trend linkom, podatci pohranjeni u trend blok mogu biti direktno prikazani operatoru u realnom vremenu.

Slijedeća slika pokazuje isti kontrolni lanac sa Trend blokom dodatim u lanac:



Unošenje Trend bloka u lanac

Dodajući ostale blokove kao što je Statistički data blok, dozvoljava korisniku da prikaže statističke podatke u grafu.

Opisi blokova

U ovom poglavlju dat je kratak opis blokova koji su na raspolaganju u FIX softwareu.

Općenito, FIX software obezbeđuje set standardnih blokova koji su na raspolaganju u svakom SCADA čvoru. Postoje dva tipa standardnih blokova: primarni i sekundarni. Glavna razlika između ovih tipova blokova je da primarni blokovi imaju definisano vrijeme skaniranja i nalaze se prvi u lancu. Sekundarni blokovi nemaju vremena skaniranja i nisu nikada prvi u lancu.

FIX software također obezbeđuje set izbornih blokova (opcija), koji su dati u narednoj tabeli:

Opcija	Obezbeđuje
Upravljanje	Kontinualno, PID, direktno, i digitalno upravljanje.
Statističko upravljanje procesom	Statistička analiza i obrada podataka, alarmiranje, nadzorno upravljanje, i prikaz statističkih podataka.
Batch(šaržno)	Upravljanje stanjima procesa, blokade i zaštite (interlocks) i batch upravljanje.
SQL	Pristupi čitanja i pisanja relacionoj bazi podataka na udaljenom serveru.

Primarni blokovi

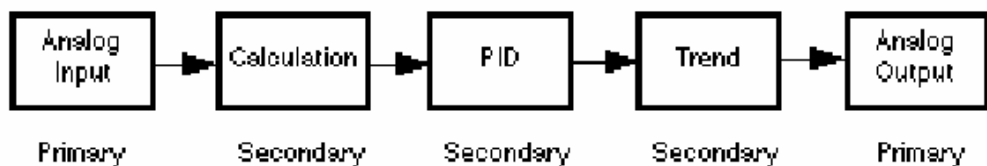
Primarni blokovi mogu primiti podatke iz DIT tabele i generirati alarme na osnovi ovih podataka. Primarni blokovi su obično udruženi sa jednim ili više djelova I/O hardwarea. Većina primarnih blokova također uključuje i vrijeme skaniranja . Ovo vrijeme skaniranja kontrolira kada SAC program skanira blokove u bazi podataka.

Naredna tabela izlistava ove primarne blokove po njihovim funkcijama

Blok	Oznaka	Funkcija
Analogni alarm	AA	Uzima podatak iz DIT tabele svaki put kada se blok skanira i obezbjedjuje mogućnost da setuje i potvrdi alarm
Analogni ulaz	AI	Uzima podatak iz DIT tabele svaki put kada se blok skanira i obezbjedjuje mogućnost da setuje granice alarma
Analogni izlaz	AO	Šalje analogne podatke u DIT tabelu kada mu blok prije njega pošalje vrijednost ili kada operator , programski blok ili program lakog pristupa bazi (EDA – easy database access) setuje vrijednost.
Analogni registar	AR	Obezbjedjuje piši/čitaj pristup u DIT tabelu samo kada je link koji je spojen na blok vidljiv na operatorskom displeju.
Boolov blok	BL	Izvršava boolove (binarne) operacije do na ukupno 8 ulaza
Digitalni alarm	DA	Uzima digitalni podatak iz DIT tabele svaki put kada se blok skanira i obezbjedjuje mogućnost da setuje i potvrdi alarm
Digitalni ulaz	DI	Uzima podatak iz DIT tabele svaki put kada se blok skanira i obezbjedjuje mogućnost da setuje alarmne granice
Digitalni izlaz	DO	Šalje digitalni podatak na DIT tabelu kada mu blok prije njega u lancu pošalje vrijednost ili kada operator , Programski blok ili EDA program ga setuje
Digitalni registar	DR	Obezbjedjuje čita/piše pristup ka DIT tabeli samo kada link koji je spojen na blok je vidljiv na operatorskom displeju.
Višestruki (multistate) digitalni ulaz	MDI	Uzima podatak do 3 ulaza iz DIT tabele svaki put kada se blok skanira , kombinira ulaze u direktnu vrijednost , i obezbjedjuje sposobnost da postavi alarmne granice
Tekst	TX	Obezbjedjuje mogućnost čitanja ili upisa tekstualne informacije u uređaj.

Sekundarni blokovi

Sekundarni blokovi manipuliraju podacima u skladu sa instrukcijama korisnika. Sekundarni blokovi obično primaju ulaz od prethodnog ili primarnog bloka i izvršavaju specifičnu funkciju sa tim ulazom, kao što je neko izračunavanje ili pohranjivanje nekoliko sukcesivnih ulaznih vrijednosti. Zbog toga, sekundarni blok ne može biti prvi blok u lancu. Ipak, korisnik može ulančiti nekoliko sekundarnih blokova. Naredna slika pokazuje kako se spajaju sekundarni blokovi:



Lanac koji povezuje sekundarne blokove

Primjetimo da je prvi blok u lancu primarni blok. Ovaj blok je primarni izvor podataka za naredni blok u lancu i određuje vrijeme skaniranja za cijeli lanac. Naredna tabela izlistava sekundarne blokove po njihovoj funkciji;

Tabela; Standarne funkcije Sekundarnih blokova

Blok	Oznaka	Funkcija
Izračunavanja [Calculation]	CA	Izvršava matematske kalkulacije koristeći vrijednosti iz prethodnog bloka i do sedam drugih konstanti iz bloka ili vrijednosti koje su pohranjene u bloku
Akcija na događaj [Event Action]	EV	Nadzire vrijednosti ili alarmne uslove prethodnog bloka i izvršava akcije bazirane na izlazu iz prethodnog bloka.
Fanout	FN	Prenosi vrijednost koju prima od prethodnog bloka prema, do četiri bloka, koja slijede.
Selekcija signala [Signal Select]	SS	Sampluje do šest ulaza, manipulišući ulazima u skladu sa modom koji selektira korisnik, i prosljeđuje izlaz ka narednom bloku
Timer	TM	Funkcionira kao brojač vremena inkrementirajući ili dekrementirajući njegovu vrijednost.
Totalizer	TT	Održava totalnu vrijednost kao vrijednost sa pokretnim zarezom , koju mu dostavljaju prethodni blokovi. Blok prenosi vrijednost sa preciznošću do šest digita ka drugim blokovima. Može prikazati preciznost do petnaest digita u View.
Trend	TR	Dozvoljava korisniku da prikuplja do 80 vrijednosti u realnom vremenu od prethodnog bloka. Podatci se mogu prikazati u obliku grafa u View.

Kontrolni blokovi

Kontrolni blokovi obezbjeđuju direktne ili digitalne mogućnosti upravljanja. Naredna tabela izlistava kontrolne blokove po njihovoj funkciji:

Tabela: Funkcije kontrolnih blokova

Blok	Oznaka	Funkcija
Mrtvo vrijeme [Dead Time]	DT	Zakašnjava do 256 sekundi prenos ulazne vrijednosti do slijedećeg bloka u lancu. Može pohraniti do 60 vrijednosti ulaznih varijabli i prenjeti dalje vrijednosti na principu FIFO (prvi ulazi-prvi izlazi)
Lead Lag	LL	Obezbjeđuje mogućnost simuliranja dinamike procesa i uključuje digitalnu aproksimaciju eksponencijalne jednačine za lead-lag. Blok je koristan u strategijama feed-forward upravljanja (sa vodjenjem signala unaprijed).
PID	PID	Poredi analogne ulaze sa zadatom vrijednošću (set-point), definisanom od strane korisnika i šalje na izlaz inkrementalna podešenja da približi varijablu procesa što bliže zadatoj vrijednosti.

On-Off upravljanje [On-Off Control]	BB	Prima analognu vrijednost i na izlazu izbacuje digitalne vrijednosti upravljanja
Ramp	RM	Povećava ili smanjuje vrijednosti ka ciljnoj vrijednosti u specificiranoj brzini promjene. Ciljne vrijednosti mogu biti ručno unesene ili dobijene iz drugih blokova. Tri odvojena stepena se mogu definisati za ramp proces.
Ratio/Bias	RB	Obezbjedjuje mogućnost da promjeni dolazne signale dodavajući konstantu (bias) i/ili množeći sa konstantom (ratio) nakon oduzimanja offseta od signala. Blok koristi manje memorije i izvršava se brže nego blok izračunavanja (Calculation)

Statistički blokovi procesnog upravljanja

Statističko procesno upravljanje (SPC) obezbjedjuje statističku analizu podataka i izračunavanja, alarmiranja, nadzorno upravljanje i prikazivanje statističkih podataka. Naredna tabela izlistava statističke procesne blokove upravljanja po njihovim funkcijama:

Blok	Oznaka	Funkcija
Histogram	HS	Prikazuje frkvenciju pojavljivanja ulazne vrijednosti u formi bar grafa
Pareto	PA	Prihvata, izračunava i sortira frekvencije do 8 ulaznih vrijednosti. Ove vrijednosti mogu biti prikazane u View kao bar čart.
Statističko upravljanje [Statistical Control]	SC	Upareno sa statističkim blokom podataka, ovaj blok podešava procesnu varijablu na bazi izračunavanja srednjeg offseta (otklona) i brzine devijacije od ciljne vrijednosti XBARBAR. Blok se aktivira ako je alarm generiran od strane statističkog bloka podataka.
Statistički podatci [Statistical Data]	SS	Posmatra podatke od strane operatorskog ulaza ili drugih blokova i izvršava statistička izračunavanja. Ovaj blok dozvoljava alarmiranje bazirano na standardnim SPC tehnikama. XBAR, R, i S čartovi (zapisi) se mogu generirati u View.

Batch blokovi (bač)

Bač blokovi su specifično dizajnirani za diskontinualno (stanjima upravljano – state driven, sekventno, međublokade – interlocks i bač) upravljanje procesima. Naredna tabela izlistava bač blokove po njihovim funkcijama .

Tabela: Funkcije bač blokova

Blok	Oznaka	Funkcija
Kontrola uredjaja [Device Control]	DC	Koordinira otvaranje i zatvaranje digitalnih uredjaja u pogonu na bazi uslova koje definira korisnik.
Program	PG	Izvršava kratke programe za bač operacije ili da poveća stepen automatizacije u nekoj aplikaciji.

SQL Blokovi

SQL blokovi čitaju i upisuju podatke u relacionu bazu podataka na udaljenom serveru. Naredna tabela izlistava SQL blokove po njihovim funkcijama.

Tabela : Funkcije SQL blokova

Blok	Oznaka	Funkcija
SQL podatci [SQL Data]	SQD	Identificira koje podatke će skupljati u FIX bazu podataka za transfer ka ODBC relacionoj bazi podataka ili gdje da upisuje vrijednosti natrag u FIX bazu podataka.
SQL trigger [SQL Trigger]	SQT	Trigeruje (starta) prikupljanje i unošenje procesnih podataka. Također definira frekvenciju i prirodu FIX transakcija sa ODBC relacionom bazom podataka.

Dinamoi baze podataka (database dynamos)

FIX software može procesirati informacije od jednog ili više objekata koji su nazvani database dinamoi. Svaki dinamo baze podataka je opcioni blok koji dodaje funkcionalnost ka procesnoj bazi podataka. Koristeći database dinamo, korisnik može kreirati nove blokove skrojene po mjeri njegove aplikacije. Na primjer, korisnik može kreirati Dinamo koji obezbjeđuje specifične PID ili druge kontrolne blokove.

Kreiranje dinamo se vrši pomoću alata za database dinamo (database dynamo toolkit). Jedanput kada se kreira , database dinamo se tretira kao i bilo koji drugi blok u procesnoj bazi podataka. Ova karakteristika omogućava FIX softwareu da procesira alarme iz Dinamo, zajedno sa drugim alarmima iz sistema. Database dinamoi također omogućuju korisniku da :

- pristupi Dinamovim podacima iz bilo koje FIX aplikacije
- Koristi Graditelj baze podataka da kreira, konfigurira, i manipulira sa operacijama nad Dinamoima u procesnoj bazi podataka.

Jedan takav Dinamo database koji je pripremljen od strane originatora FIX paketa-Intellutiona , je Blok proširenog trendiranja. Ovaj blok dozvoljava korisniku da prikuplja

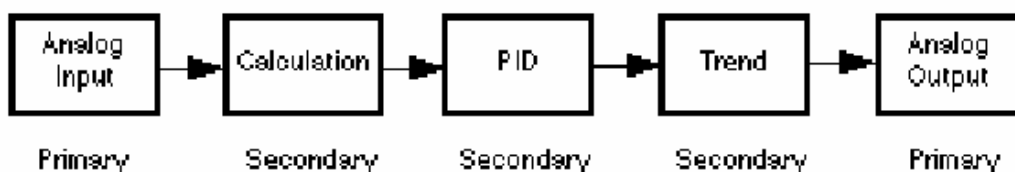
do 600 vrijednosti u realnom vremenu iz prethodnog bloka na koji je priključen (upstream bloka). Podatci se nakon toga mogu prikazati u formi grafikona u View paketu.

Tabela : Sumarni pregled blokova

Tip bloka	Oznaka	Primarni	Sekundarni	Standardni	Opcioni
Analogni alarm	AA	•		•	
Analogni ulaz	AI	•		•	
Analogni izlaz	AO	•		•	
Analogni registar	AR	•		•	
Bulov	BL	•		•	
kalkulacioni	CA		•	•	
Mrtvo vrijeme	DT		•		• (upravljanje)
Kontrola uređaja	DC	•			• (bač)
Digitalni alarm	DA	•		•	
Digitalni ulaz	DI	•		•	
Digitalni izlaz	DO	•		•	
Digitalni registar	DR	•		•	
Akcija na događaj (event action)	EV		•	•	
Prošireni trend	ETR		•	• (Dinamo)	
Fanout	FN		•	•	
Histogram	HS		•		• (SPC)
Lead Lag	LL		•		•(Upravljanje)
Višestepeni digitalni ulaz	MDI	•		•	
On-Off upravljanje	BB	•			• (upravljanje)
Pareto	PA	•			•(SPC)
PID	PID		•		•(upravljanje)
Program	PG	•			•(bač)
Ratio/Bias	RB		•		•(upravljanje)
Selekcija signala	SS		•		•(upravljanje)
SQL podatci	SQD		•		•(SQL)
SQL Trigger	SQT	•			•(SQL)
Statističko upravljanje	SC		•		•(SPC)
Statistički podatci	SD	•			•(SPC)
Tekst	TX	•		•	
Timer	TM		•	•	
Totalizer	TT		•	•	
Trend	TR		•	•	

Formati lanaca

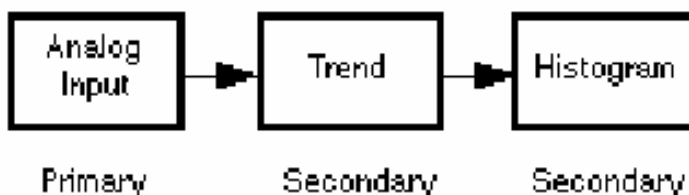
Dizajn lanaca zavisi od zadatka kojeg izvršavaju. Za lance za koje se očekuje da izvršavaju prikupljanje podataka i upravljanje obično primaju ulaze iz tabele slike drajvera (DIT) preko ulaznog bloka , manipuliraju ulazima sa sekundarnim blokovima i vraćaju podešene vrijednosti u DIT tabelu preko analognog izlaza, digitalnog izlaza ili On-Off kontrolnog bloka. Naredna slika ilustrira dizajn prikupljanja uzorka podataka i upravljanje:



Lanac prikupljanja i obrade podatka

Lanci za koje se očekuje da prikupljaju podatke iz DIT tabele za potrebe nadzora obično primaju ulaze iz DIT tabele putem ulaznog bloka ali ne moraju vraćati ulaze u DIT tabelu, pošto se ne zahtjeva procesno podešenje podatka.

Naredna slika ilustrira dizajn lanca u ovome slučaju:



Lanac za nadzor

Broj lanaca koje korisnik može koristiti je ograničen samo sa raspoloživom memorijom.

Razumjevanje formata polja i parametara.

Svaki blok u bazi podataka ima jedan ili više polja. Ova polja sadrže procesne informacije kao što su: trenutna vrijednost podatka, alarmni status bloka, adresa bloka, ime bloka, koji je slijedeći blok u lancu. Polja bloka sadrže takodjer specifične za svaki blok informacije koje variraju od bloka do bloka.

Neke od informacija u poljima se unose preko dijalog boksa koje se unose kod kreiranja bloka. Ove informacije uključuju podatke kao što su : ime bloka, opis bloka, i I/O adresa. Ostale informacije dolaze iz samog procesa ili iz drugih blokova. Naprimjer tekuća vrijednost primarnog bloka dolazi iz procesa. Ipak, sekundarni blokovi primaju podatke iz prethodnog (uzvodnog) bloka.

Ime svakog polja bloka se sastoji iz dva elementa : *format* i *parametar*. Sintaksa ovih polja je data kao:

format_parameter

Format indicira tip podatka koji se pohranjuje u tom polju. Naredna tabela daje formate koji su raspoloživi:

Format	Opis	Koristi se u :
A	ASCII Format	Data linkovima
F	Floating point format	U data linkovima i blok-blok referenciranju
G	Graphic format	Linkovima zapisa (chartova)

Parametar indicira specifičnu informaciju u polju. Na primjer, tekuća vrijednost bloka se identificira sa :

CV

Kombinacija formata polja i parametra polja obezbjeđuje korisnika sa informacijom koja mu je potrebna. Na primjer, ako želimo tekuću vrijednost bloka da bude prikazana kao broj, izabraćemo F_CV parametar. Ako želimo da tekuća vrijednost parametra bude prikazana kao tekst , izabraćemo A_CV parametar.

Izlistavanje parametara

Help sistem parametara polja obezbjeđuje listu svih raspoloživih parametara polja za svaki blok. Koristeći ovu informaciju, možemo pokazati podatke u poljima na operatorskom displeju, kreirajući jedan od linkova izlistanih u prethodnoj tabeli i izabirući odgovarajuće polje.

Alternativno, ako korisnik kreira EDA (**E**asy **D**atabase **A**ccess – lak pristup bazi podataka), on će biti u stanju da pristupi poljima bloka u svojim programima.

Inicijalni modovi blokova

Pojedini blokovi uključuju osobinu koja se naziva **inicijalni mod**. Ova osobina određuje kako blokovi primaju podatke. Blokovi mogu primiti podatke iz slijedećih izvora:

- drugih blokova
- iz DIT tabele (driver image table) nekog I/O uređaja
- od tastature (preko displej linka)
- receptura
- EDA programa

Automatski mod

Može se postaviti inicijalni mod da bude automatski ili ručni. Primarni blok postavljen na automatski mod prima podatke iz DIT tabele ili od drugih blokova. Blokovi koji

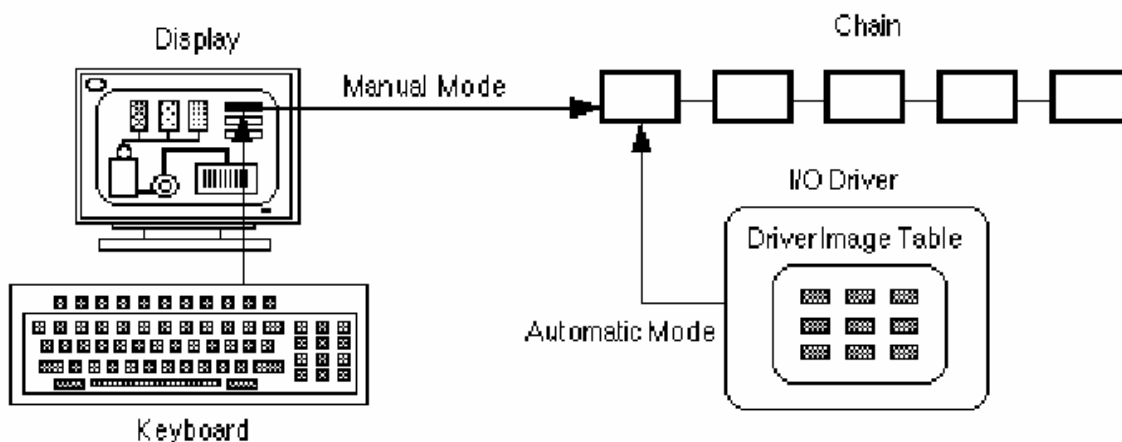
izvršavaju programske iskaze , kao što je programski blok, su sposobni da izvrše svoje iskaze bez prekida dok su u Automatskom modu.

Ručni mod

Primarni blok postavljen na (ručni) Manual mod može primiti podatke i iz drugih izvora osim DIT tabele. Ovi izvori mogu biti programski blokovi (koristeći SETOUT iskaz), Fanout blokovi, Operatori koji koriste tastaturu da unesu vrijednosti u linkovima, ili EDA programi.

Postavljajući blok u ručni mod ne postavlja blok van skaniranja, SAC još uvijek procesira blok koristeći posljednju vrijednost iz bloka. Ručni mod samo spriječava blok da uzima podatke iz DIT tabele. Naredna slika pokazuje kako lanac prima podatke u obadva ova moda. Tabela koja slijedi identificira kako specifični blokovi funkcioniraju u obadva ova inicijalna moda.

Primjetimo da blok izvršava alarmiranje u obadva i automatskom i ručnom modu rada.



Lanac u automatskom i ručnom modu rada

Tabela: Funkcije blokova u automatskom i ručnom modu rada

Blok	Automatski mod	Ručni mod
Analogni ulaz	Prima podatke iz DIT tabele u svakom periodu skaniranja	Prima podatke od Operatora koji unose vrijednosti u linkove, od Programskih blokova ili EDA programa
Kontrola uredjaja [Device Control]	Izvršava sve iskaze bez prekidanja	Suspendira izvršenje dok se blok ne prebaci u automatski mod.
Digitalnu ulaz	Prima podatke od DIT tabele svaki period skaniranja	Prima podatke od Operatora koji unose vrijednosti u linkove ili od programskih blokova ili od EDA programa
PID	Izvršava PID podešavanja. Setpoint se može mjenjati putem	Suspenduje automatske PID izlaze. Ovo dozvoljava korisniku da promjeni izlaz iz bloka. Vrijednosti bloka kao što su setpoint i parametri

	displej linka u ovom modu.	podešenja regulatora, mogu se promijeniti pomoću tastature.
Program	Izvršava sve iskaze bez prekidanja	Suspendira izvršenje iskaza u programu sve dok blok se ne vrati ponovno u automatski mod. Kada se blok prekluči ponovno u automatski mod rada, on nastavlja tamo gdje se zaustavio.
Statističko upravljanje [Statistical Control]	Izvršava on-line statističko upravljanje procesom na osnovu dolaznih podataka iz drugih blokova.	Suspenduje on-line osmatranje podataka. Koristi se kod off-line statističkog upravljanja procesima.

PMAN i PAUT modovi

Kada je blok u automatskom modu, korisnik može ga prebaciti u ručni mod koristeći sekvencu <Ctr> <M> ili A_AUTO polje u linku koji referencira blok. Tekst u linku se mijenja na PMAN (pending manual). Ovaj tekst indicira da je promjena moda potvrđena i da će se desiti (pending). Sama promjena moda u bloku će se desiti u trenutku kada SAC skanira taj mod. SAC skanira blok kada se unese ručno od strane Operatora ili Programskog bloka, ili kada se pojavi naredni interval skaniranja.

Kada se promjeni mod bloka sa ručnog na automatski mod, tekst u A_AUTO polju se promjeni na PAUT (pending auto). Ovaj tekst indicira da promjena moda je potvrđena i da čeka izvršenje. Mod bloka će se promijeniti na automatski kada SAC skanira blok.

Ako korisnik promjeni mod bloka koristeći Graditelj baze podatak (Database Builder), spreadsheet će pokazati novi mod (bilo MANL ili AUTO), kada SAC skanira blok.

Opaska: Da bi se promjenio Automatski/Ručno mod iz View-a , korisnik treba koristiti A_AUTO polje u linku koji referencira taj blok.

Postavljanje blokova u i van skaniranja (on and off scan)

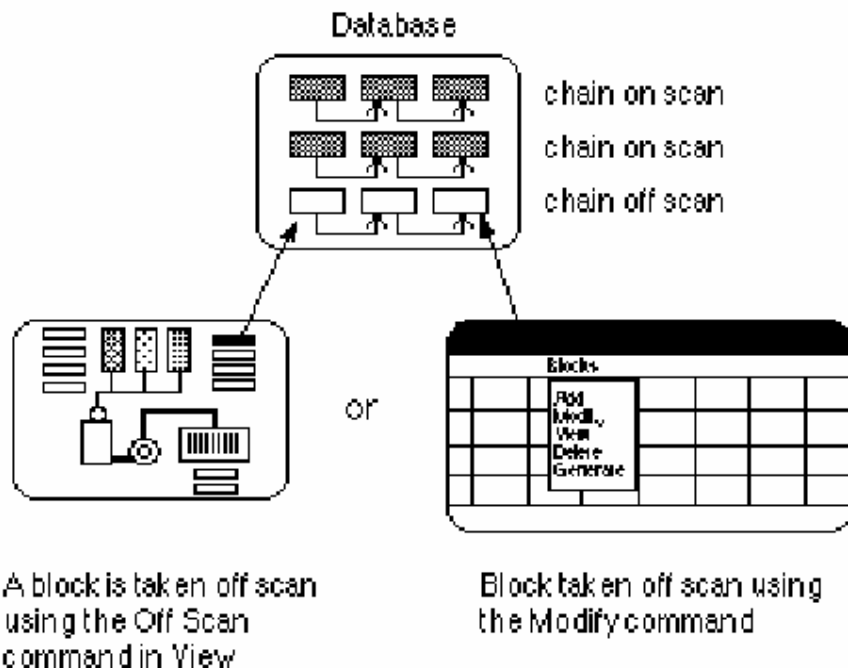
Postavljanje blokva van skaniranja

SAC je odgovoran za procesiranje blokova baze podataka koji su stavljeni u skaniranje. Kada SAC starta ili se puni (load) baza podataka, SAC automatski procesira sve primarne blokove koji imaju selektiran ček boks : Startaj blok u skanu. SAC prestaje procesiranje primarnog bloka ili lanca pod slijedećim uslovima:

- Iz View programa, korisnik selektira link koji je spojen na primarni blok i onda selektira : *Off Scan* iz komandnog menija. Ova akcija će postaviti blok van skaniranja. Ova komanda će se izvršiti samo onda ako je selektiran ček boks koji se kontroliše linkom.
- Korisnik modificira blok u lancu koristeći komandu Modify iz Graditelja baze podataka. Modificirajući blok dok SAC procesira lanac u kojem se nalazi taj blok, prebaciće cijeli lanac van skaniranja.
- Ako je izbrisan blok unutar lanca.

- Iskazi u programskom bloku ili blokovima akcije na događaj (event action blocks), ili EDA program može prebaciti primarni blok u lancu van skaniranja.
- Ako su blokovi nekorektno uvezani u lanac, SAC ne može skanirati lanac. Koristite Verify komandu da se utvrdi koji su blokovi nekorektno ulančeni.
- Ako programski blok završi procesiranje svojih iskaza i izadje, tada Programski blok neće biti ponovno izvršen sve dok se ponovno ne napuni (loading) baza podataka ili se programski blok isključi a onda ponovno vrati na skaniranje.
- Sa kolonom : Poljem Statusa skaniranja prikazanom u spreadsheetu, korisnik može postaviti blok van skaniranja koristeći Graditelj baze podataka mjenjajući tekst u koloni sa ON na OFF.

Naredna slika ilustrira ova dva prva uslova:



Postavljanje blokova van skaniranja

Postavljanje blokova u skaniranje

Ako korisnik postavi primarni blok (napr. Analogni ulaz) van skaniranja, tada se time čitav lanac u kojem se nalazi taj blok smješta van skaniranja. Da bi se lanac ponovno vratio u skaniranje, treba uraditi nešto od slijedećih opcija:

- Otvoriti Operatorski displej u View koji ima link spojen na primarni blok u lancu i selektirati taj blok. Nakon toga treba selektirati : On Scan iz komandnog menija da bi se postavio lanac na skaniranje. Prednost korištenja

ovog metoda je da to dozvoljava korisniku da brzo postavi lanac natrag na skaniranje bez da poremeti normalno procesiranje baze podataka.

- Postaviti programski ili blok akcije na događaj (event action block), koji će vratiti lanac ponovno na skaniranje. Ova procedura je korisna ako imamo planirane periode kada znamo da će lanac biti postavljen van skaniranja. On-line help daje mnogo više informacija o ovoj mogućnosti.
- Promjeniti tekst u koloni statusa skana sa OFF na ON.

Korisnik može također postaviti lanac u ili van skaniranja dodavajući blok iz Draw (crtaj) programa. Zavisno od izabrane opcije, Draw automatski postavlja blok na skaniranje, prebacuje blok van skaniranja, ili promptira korisnika da postavi blok u ili van skaniranja.

Blokovi sa dugim vremenima skaniranja

Blokovi koji imaju duga vremena skaniranja reaguju različito na promjene on/off skaniranje, nego blokovi sa kratkim vremenom između dva uzorkovanja. Ako korisnik promjeni skan status bloka, on će prikazati bilo PON ili POFF (pending on ili pending off). Ovaj tekst indicira da je bila zahtjevana promjena statusa skaniranja i da se čeka izvršenje te promjene, ali da blok još uvijek nije izuzet sa ili vraćen na skaniranje. Kada je blok u PON, svaka nova vrijednost biće ignorirana.

Po defaultu, blok mijenja status skaniranja kratko nakon što udje u stanje čekanja izvršenja. Međutim, ako je SAC startao sa "S" parametrom na komandnoj liniji, blok ostaje u stanju čekanja sve dok SAC je spreman da ga skanira. Naprimjer, pretpostavimo da imamo blok sa vremenom skaniranja od 1 sata. Blok je već 45 minuta u tom vremenu za skaniranje, kada ga korisnik prebaci iz OFF scana na ON scan. Blok će ostati u PON stanju 15 minuta da bi se sinhronizovao sa SAC i nakon toga će blok biti prebačen na skaniranje.

Ako korisnik promjeni mod bloka koristeći graditelj baze podataka (Database Builder), spreadsheet će prikazati bilo PON ili POFF. Sa uzastopnom selekcijom komande osvjeđenja (refresh), korisnik može ažurirati spreadsheet da prikaže novi status skaniranja (bilo ON ili OFF), kada SAC skanira blok.

Vremena skaniranja

Svaki primarni blok koji korisnik dodaje u bazu podataka zahtjeva od njega da unese vrijeme skaniranja. Vrijeme skaniranja određuje kako često SAC procesira instrukcije u bloku i prenosi tekuću vrijednost na slijedeći blok u lancu. Svi sekundarni blokovi u lancu se procesiraju u skladu sa vremenom skaniranja primarnog bloka.

SAC može procesirati lanac koristeći jedan od slijedećih metoda:

- procesiranje na bazi vremena
- procesiranje na bazi izuzetnosti
- jednostruko (one-shot) procesiranje

Kod vremenski baziranog procesiranja, SAC procesira blok u postavljenim trenucima vremena. Naredna tabela ilustrira opsege vremena skaniranja koji su mogući u FIX software-u za vremenski bazirane lance.

Opsezi vremena skaniranja

Opseg	Inkrementiran svakih
0.05 do .95 sekundi	0.05 sekundi (0.05, 0.10, 0.15, 0.20, itd.)
1 do 60 sec	1 sekunda
1 do 60 min	1 min
1 do 24 sata	1 sat

Subsekundno skaniranje

Ako se unese subsekundno vrijeme skaniranja, SAC će automatski se izvršavati dovoljno brzo da može da ovo ostvari. Nije potrebna neka specijalna konfiguracija.

Satni i minutni skan je baziran na sistemskom satu (čvora) noda na kojem lanac egzistira. Vremena skaniranja se postavljaju relativno u odnosu na pola noći (00: 00: 00 sati). Sekundno i subsekundno skaniranje su bazirani na vremenu startanja sistema.

Tabela ; Primjeri vremna skaniranja kod vremenski baziranih procesiranja

Ako je vrijeme skaniranja	SAC će procesirati blok svakih;
1 sat	Iz sata u sat
1 min	Iz minuta u minut
10 sec	10 sec nakon startup vremena sistema.

Procesiranje bazirano na izuzeću

Procesiranje na bazi izuzeća dozvoljava da blok ili lanac blokova budu procesirani sa izuzećem (exception) a ne kod planiranih intervala vremena. Slijedeće akcije čine izuzeće:

- promjena vrijednosti I/O u DIT tabeli, veća od definirane mrtve zone (dead band) za izuzeće
- jedna neplanirana poruka od PLC-a je primljena

Blok koji je baziran na izuzetnosti se takodjer skanira kad mjenja mod rada (sa automatskog na ručno i obratno). Dok je u ručnom modu, SAC prihvata bilo kojim ulaz koji je unesen i koji je unutar EGU (dopuštenog) opsega, i trenutačno skanira blok. Dok je u ručnom modu ne vrše se provjere zone neosjetljivosti (dead band).

Koristeći procesiranje bazirano na izuzeću zahtjeva manje CPU vremena i poboljšava performansu sistema , jer SAC ne mora da skanira blokove u definiranim intervalima. Medjutim, ako se podatci u bloku često mjenjaju, procesiranje bazirano na vremenskom intervalu bit će efikasnije.

Ako korisnik želi da koristi procesiranje bazirano na izuzetnosti, drajverski modul I/O uređjaja to mora omogućavati.

Opaska : SIM (simulation) drajver ne podržava procesiranje bazirano na izuzeću.

Kao i kod vremenski baziranog i jednostrukog procesiranja, procesiranje bazirano na izuzetnosti dozvoljava korisniku da koristi primarne blokove unutar lanca, ne samo kao prvi blok u lancu. Međutim, izvjesni blokovi se izvršavaju bolje pri procesiranju kod izuzetnosti nego kod drugih tipova procesiranja. Moguće je koristiti slijedeće blokove u lancima koji se procesiraju na bazi izuzetnosti:

Analogni ulaz	On-Off upravljanje
Analogni izlaz	Pareto
Digitalni ulaz	Odnos ili bajes
Digitalni izlaz	SQL podatci
Fanout	Tajmer
Histogram	Totalizator

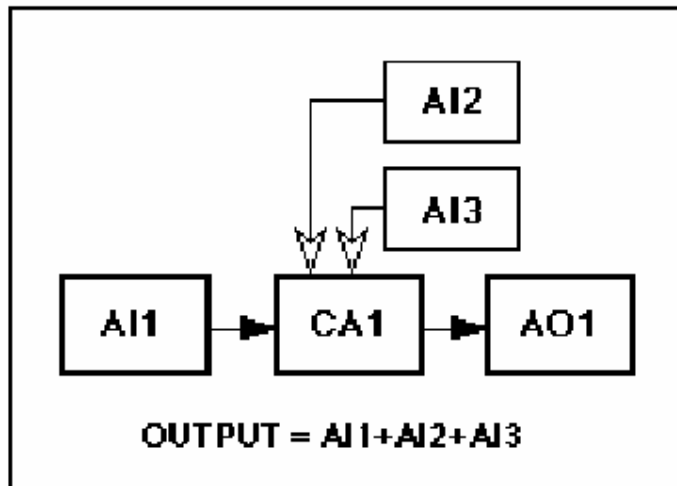
Tabela: Analiza programskih i statističkih data blokova

Blok	Ne treba koristiti slijedeće:
Program	CALL, DELAY, WAITFOR, WAITSTAT
Statistički podatci	WAIT TIME polja

Slijedeći blokovi mogu koristiti vrijednosti iz multiplih blokova, ali se procesiraju samo kod specificiranog trenutka skaniranja njihovog primarnog bloka. Zbog toga je potrebno koristiti ove blokove sa oprezom kod lanaca baziranih na izuzeću. :

- Bool-ovi
- Račuski blok (calculation)
- Akcija po događaju (event action)
- Selekcija signala (signal select)

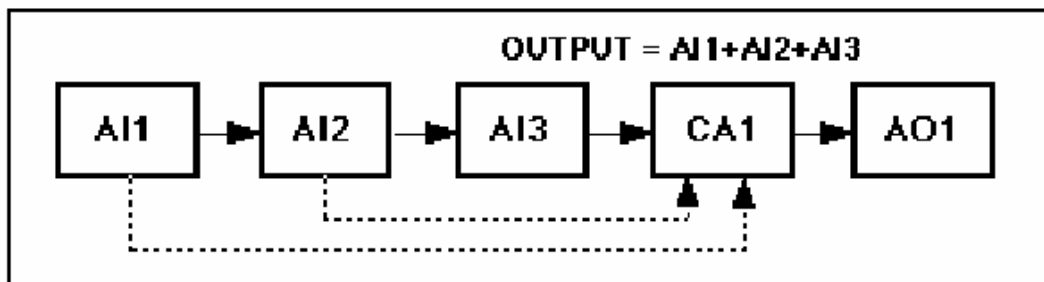
Naprimjer, slijedeća slika prikazuje primjer lanca koji sadrži računski blok koji prima vrijednosti iz blokova analognog ulaza van lanca:



Procesiranje bloka lanca

Bez obzira da li CA1 primi vrijednost od AI1, ako je AI1 setovan da bude skaniran na bazi izuzeća, CA1 će pristupiti samo vrijednostima preostalih analognih ulaznih blokova kada se izuzeće pojavi za AI1. CA1 ne pristupa vrijednostima baziranim na specifikacijama skaniranja analognih ulaznih blokova izvan lanca. To znači da, bez obzira na promjene vrijednosti u AI2 i AI3, CA1 neće ponovno računati svoj izlaz sve dok nije trigerovan sa izuzećem i skaniranjem u AI1.

Naredna slika prikazuje jedan poboljšani dizajn lanca koristeći blokove iz prethodne slike. Ovaj lanac dozvoljava CA1 da bude ponovno izvršen kad god se pojavi izuzeće za bilo koji analogni ulazni blok i obezbjeđuje da se svi blokovi procesiraju prije izračunavanja izlaza.



Poboljšano procesiranje bloka lanca

Blokovi koji će biti izlistani u narednoj tabeli koriste sistemsko vrijeme čvora (node) na kojem se izvršavaju da definiraju vremensku konstantu. Zbog toga Intellution preporučuje da se ne koriste kod lanaca baziranih na izuzeću.

OPREZ : Nemojte doznačavati iste I/O adrese za blokove bazirane na vremenskom i one na izuzeću baziranom procesiranju. Ako se to uradi, onda će blokovi bazirani na izuzeću povremeno pustiti nove dolazne vrijednosti da se na njih ažuriraju.

Tabela: Analaza procesiranja baziranog na izuzeću

Blok	Polje
PID	RATE
Lead Lag	LEAD, LAG
Dead Time	DEAD TIME

Blokovi analognog i digitalnog alarma

Blokovi analognih i digitalnih alarma podržavaju procesiranje bazirano na izuzeću. Međutim, korisnik mora da ostavi nepromjenjenim polja Re-alarm i Delay Time. Ako to ne učini, SAC će postaviti ove blokove (i njihove lance) u off scan stanje, kada FIX software starta ili se reloaduje baza podataka.

Kada su blokovi analognog ili digitalnog alarma bazirani na izuzeću, potvrđivanje alarma u View paketu će trenutno trigerovati procesiranje ovog bloka. Potvrđivanje alarma sa daljinskim potvrđivanjem (remote acknowledge) poljem, neće trigerovati SAC da procesira blok analognog ili digitalnog alarma.

Jednostruko procesiranje (one shot processing)

Svaki primarni blok ili lanac može biti trigerovan samo jedanput sa postavljanjem vremena skaniranja na vrijednost 0 i postavljanjem bloka na skaniranje. Ovaj blok neće više biti trigerovan sve dok se SAC ponovo ne starta, ili baza podataka se ponovo napuni (reload), ili pak blok se stavi van skana a onda se vrati ponovno na skaniranje.

Isto ograničenje vrijedi i za jednostruke (one-shot) bazirane lance. Drugi način na koji se može trigerovati jednostruki lanac uključuje postavljanje bloka na skan iz:

- View, koristeći On Scan komandu
- Programskih ili akcija na događaj (event action) blokova

Svaki blok sa parametrom vremena skaniranja može biti postavljen na jednostruko procesiranje.

Doznačavanje vremena skaniranja

Doznačavanje vremena skaniranja za jednostruko bazirane blokove

Korisnik kontrolira kada SAC skanira lanac putem vrijednosti koju unese u polje vremena skaniranja bloka. Ako želi da blok ili lanac u kojem se nalazi bude skaniran samo jedanput, treba da unese vrijednost **0** u ovo polje.

Doznačavanje vremena skaniranja za blokove bazirane na izuzeću

Ako želimo da SAC procesira lanac samo kada se pojavi izuzetni događaj (exception), treba unjeti **E** u polje vremena skaniranja. Ako koristimo višestruke primarne blokove u jednom lancu koji je baziran na izuzeću, korisnik treba da unese vrijednost 0 u polje vremena skaniranja za sve ostale primarne blokove sa kojima ne starta lanac. Također, check box : Start block on Scan treba ostaviti nesektiranim. Konfigurirajući primarne

blokove tako da ne startaju lanac, na ovaj način, obezbjeđuje da SAC procesira korektno lanac.

Doznačavanje vremena skaniranja za vremenski bazirane blokove

Ako želimo da specificiramo vremenski bazirano procesiranje, treba unjeti vrijeme skaniranja primarnog bloka u polje vremena skaniranja. Kada koristimo višestruke primarne blokove u lancu baziranom na vremenu, potrebno je unjeti 0 u polje vremena skaniranja svakog primarnog bloka kojim ne počinje lanac. Također je potrebno obezbjeđiti da ček boks primarnog bloka sa kojim starta lanac je neselektiran.

Koristiti slijedeći format da se definira vrijeme skaniranja kada koristimo vremenski bazirano procesiranje:

time unit

Naredna tabela izlistava sve dopuštene vremenske jedinice i njihove skraććenice . Ako korisnik ne unese jedinicu vremena FIX pretpostavi da su jedinice sekundi.

Tabela : Vremenske jedinice

Jedinica	Unos
Sekunde	S
Minute	M
Sati	H

Primjer: Da se skanira blok svaka 3 sata , treba unjeti 3H

Pošto je vrijeme skaniranja linkovano sa sistemskim satom, blok će se skanirati u 0:00, 3:00, 6:00, 9:00 i 12:00, bez obzira kada se blok stavio na skaniranje.

Potrebno je slijediti slijedeće preporuke:

- Doznačiti vremena skaniranja koja su veća od vremena izbora (polling time) I/O drajvera. Ovo će obezbjeđiti da I/O drajver je imao vremena da ažurira DIT tabelu prije nego što SAC skanira ponovno svaki blok. Pogledati manual I/O drajvera da bi se ustanovila vremena izbora.
- Fazno pomjeriti (stagger) vremena skaniranja da se smanji rizik od preopterećenja CPU.
- Omogućiti kritičnim procesnim lancima da imaju češća vremena skaniranja nego nekritični lanci. Ako je potrebno da se lanac skanira svake 2 minute, doznačite mu 2 minute za vrijeme skaniranja a ne 5 sekundi. Potrebno je imati na umu da vrlo kratka vremena skaniranja zahtjevaju više CPU vremena i SAC procesiranja nego duža vremena skaniranja.
- Ako lanac ne treba da bude procesiran u datom vremenu, treba mu doznačiti procesiranje bazirano na izuzetku. Čineći ovo bit će potrebno manje CPU vremena i poboljšaće se performansa sistema.

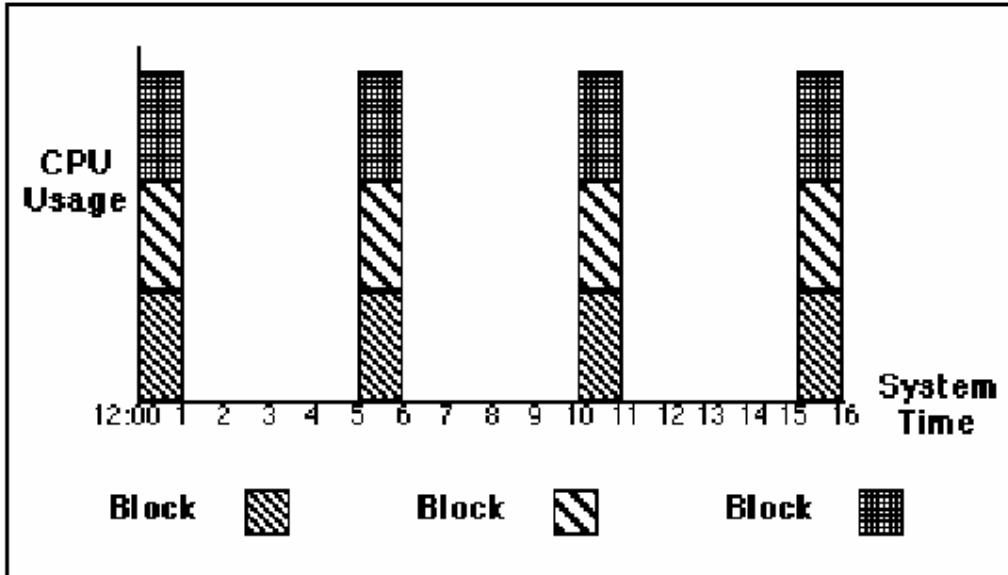
Faziranje

Faziranje znači zatezanje (staggering) vremena kada će blokovi biti skanirani, i time bolje iskorišćavajući CPU vrijeme.

Naprimjer, ako imamo 3 bloka koji se skaniraju svakih 5 sekundi, i mi ne specificiramo nikakvo vrijeme faziranja, SAC će pokušati da procesira sva tri bloka u istom trenutku

vremena. Naredna slika pokazuje distribuciju korištenja kada SAC procesira sva tri bloka simultano.

Napomena : Faziranje je naročito važno kod velikih baza podataka kada može dramatično da poboljša performansu.

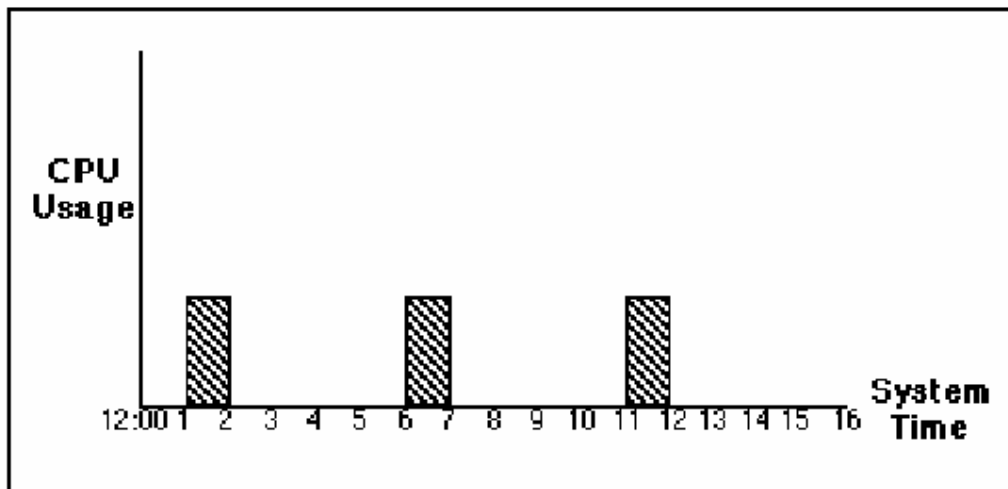


Procesiranje blokova bez faziranja

Faziranje dozvoljava da korisnik specificira kada će SAC procesirati svaki blok. Ako želimo da procesiramo prvi blok sa 1 sekundom faziranja, potrebno je unjeti slijedeću vrijednost u polje vremena skaniranja:

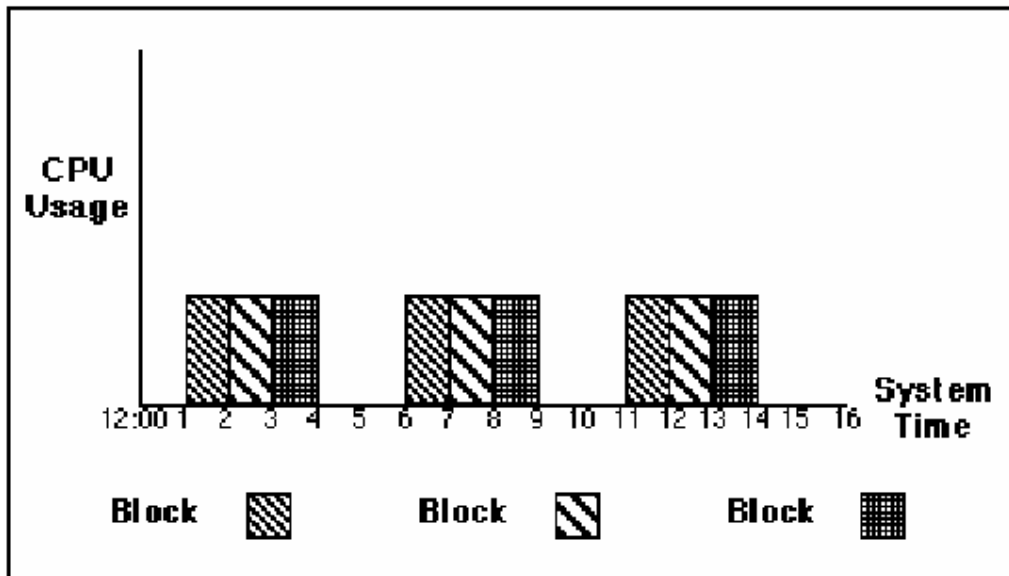
5S;1

Slijedeća slika pokazuje kada je Blok 1 skaniran:



Faziranje Bloka 1

Ako specificiramo Blok 2 da bude procesiran sa 2 sekunde faziranja, a Blok 3 sa 3 sekunde, izbjećemo da blokovi budu simultano procesirani. Naredna slika pokazuje distribuciju korištenja CPU kada SAC procesira ove blokove sa zategnutim vremenima.



Faziranje vremena skaniranja blokova kod multiplih blokova

Za vrijeme prvog skana, SAC procesira Blok 1 nakon 1 sekunde, Blok 2 kod 2 sekunde, i Blok 3 kod 3 sekunde. Kod drugog skaniranja, SAC procesira Blok 1 kod 6 sekundi, Blok 2 kod 7 sekundi i Blok 3 kod 8 sekundi. Zatezanjem vremena skaniranja blokova i faziranjem, možemo jednakomjerno raspodjeliti procesiranje blokova i time rasporediti uniformno i opterećenje CPU.

Faziranje sekundnih i subsekundnih blokova

Kada specificiramo fazu za blok sa sekundnim ili podsekundnim vremenom skaniranja, SAC ofsetuje inicijalni skan bloka za vrijeme faziranja kada FIX software starta ili kada se baza podataka reloaduje. Međutim, SAC će ignorirati fazni pomak u vremenu pri slijedećim uslovima:

- Program ili akcija na događaj (event action) postavlja fazirani blok na skaniranje
- Operator postavlja blok na skaniranje iz View-a
- EDA postavlja blok na skaniranje.

Doznačavanje vremena faziranja

Korisnik može definirati vrijeme faziranja , dodajući ga vremenu skaniranja u polje vremena skaniranja. Treba koristiti slijedeću sintaksu da se definira vrijeme skaniranja i faziranja za vremenski bazirane lance:

scan_time unit; phase

Korisnik može specificirati fazu koja je sa istom vrmenskom jedinicom ili jednu jedinicu vremena niže od vremena skaniranja. Redoslijed jedinica vremena je : sati, minute, sekunde, i podsekunde.

Naravno , jedinica vremena faziranja ne može nikada biti veća od jedinice vremena skaniranja.

Tabela : Formati vremena faziranja

Ako je vrijeme skaniranja u ...	Vrijeme faziranja može biti u..
Satima	Satima: minutama
Minutama	Minute: sekunde
sekundama	sekunde
subsekundama	subsekunde

Opaska: FIX software podržava Windows Internacionalni format. Zato je potrebno konfigurirati ovo i u dijalog boksu Kontrolnog panela. Preporučeni separator koji odvaja vrijeme skaniranja od vremena faziranja je polukolona (;).

Prefaziranje

Možemo se susresti sa situacijama kada pojedini blokovi zahtjevaju fazu koja je veća od njihovih vremena skaniranja. U takvom slučaju blokovi će biti prefazirani. Naprimjer, pretpostavimo da imamo blok računanja (calculation) koji referencira dva analogna ulazna bloka, kao na prethodnoj slici.

U ovom lancu AI2 i i AI3 imaju 5 sekundi vremena skaniranja. AI1 će biti prefaziran ako koristi vremena skaniranja 5;10. Ovo će zakasniti AI1 u izvršenju za 10 sekundi, i obezbjeđuje da je SAC skanirao AI2 i AI3 prije AI1. To će obezbjeđiti da blok računanja će primiti najsvježije vrijednosti nad kojima će izvršiti svoju funkciju.

Prefaziranje i duga vremena izvršenja

Moguće je prefazirati samo blokove sa sekundnim i podsekundnim vremenima skaniranja. Blokovi sa vremenima skaniranja od minute ili duže ne mogu se prefazirati. Naime, ukoliko unesemo vrijeme faziranja koje je duže od vremena skaniranja, kreiraćemo početni pomak (offset) od vremena od kojeg bi SAC normalno skanirao taj blok. Naprimjer, ako želimo da skaniramo blok svakih 6 sati, ali želimo početni offset od 2 sata i 10 minuta , unjećemo:

6H;2:10

Pošto je vrijeme skaniranja linkovano sa sistemskim satom, blok će uvijek biti skaniran u 2:10, 8:10, 14:10, 20:10 , bezobzira kada je postavljen na skaniranje.

Blokovi sa vremenima skaniranja od 1 minute ili duže se trenutačno skaniraju kada starta FIX software ili kada se ponovo napuni baza podataka.

Skaniranje blokova u PMAN modu

Kada blok prelazi iz automatskog moda u ručni onda on prvo ulazi u stanje čekanja izvršenja (pending) , PMAN mod. Blok ostaje u PMAN modu sve dok SAC ne skanira ponovno blok.

Dok je u PMAN modu, SAC nastavlja da skanira blok u skladu sa svojim vremenom skaniranja i fazom. Jedanput kada je SAC skanirao blok , on ulazi u ručni mod rada. Ručni unosi od strane Operatora ili Programskog bloka takodjer trigeruju trenutačno skaniranje.

SAC skanira blokove u ručnom blok modu na isti način kako ih skanira u PMAN modu. Naredna skaniranja nakon ručnog unosa zavise od vremena skaniranja bloka. Za blokove

sa kratkim vremenima skaniranja , slijedeće vrijeme skaniranja će se pojaviti relativno u odnosu na vrijeme ulaska u ručni mod. Naprimjer, pretpostavimo da blok A11 ima 30 sekundno vrijeme skaniranja i skanira blok u trenutku 1:15:30. Ako Operator unese vrijednost za A11 u 1:15:45 , SAC će skanirati blok neposredno nakon tog unosa i nakon toga skanirati blok u 1:16:15, 1:16:45, 1:17:15, 1:17:45 i tako dalje.

Ako Operator nakon toga unese novu vrijednost u 1:15:50, SAC će skanirati A11 neposredno nakon unosa i nastaviti skanirati blok u trenucima 1:16:20, 1:16:50, 1:17:20 itd.

Za blokove sa vremenom skaniranja od minute i duže, slijedeći skan je sinhroniziran sa sistemskim satom. Naprimjer, pretpostavimo da A12 ima vrijeme skaniranja od 1 sat. Ako Operator unese vrijednost u trenutku vremena od 15 minuta nakon punog sata, SAC će skanirati blok neposredno nakon unosa, a nakon toga opet na puni sat.

Skaliranje blokova u PAUT modu

Kada blok prelazi iz ručnog u automatski mod rada, on ulazi u stanje čekanja izvršenja , PAUT. Blok će ostati u tom stanju dok SAC ne skanira blok.

Dok je u PAUT modu, SAC skanira blok kao da je još uvijek u automatskom modu. Za blokove sa kratkim vremenom skaniranja, ovo znači da će se slijedeći trenutak skaniranja pojaviti relativno u odnosu na trenutak posljednjeg skaniranja. Naprimjer, pretpostavimo da blok A11 ima 30 sekundno vrijeme skaniranja i da je skaniran u trenutku 1:15:30. Ako Operator promjeni mod rada bloka na automatski, blok će ući u PAUT mod. U 1:16:00, SAC će skanirati blok ponovno i prebaciti blok u automatski mod rada.

Skaliranje procesnih veličina za Operatore

Skaliranje akviziranih vrijednosti iz procesa je postupak koji konvertira podatke primljene od I/O drajvera u format koji je lako prepoznatljiv Operatoru. Moguće je skalirati procesne vrijednosti u tri koraka:

1. Izvršiti I/O drajver konfiguracioni program. Kada se izvršava ovaj program, definirajte format koji koristi I/O drajver da prenese prikupljene vrijednosti u DIT tabelu. Specifični formati koje podržava I/O hardware su obično navedeni u referentnoj dokumentaciji za taj drajver.
2. Definirati granične vrijednosti za blok i kondicioniranje signala koristeći Graditelj baze podataka (DBB). Ove granične (EGU- engineering unit) vrijednosti koje se unose kroz dijalog boks kod konfiguriranja bloka, određuju kako se vrijednosti skaliraju.
3. Sa programom Draw (Crtaj), kreirati displeje koji pokazuju skalirane procesne vrijednosti Operatorima koristeći Data linkove.

Primjer iz procesa

Predpostavimo da imamo 70 galonski tank sa vodom i želimo pokazati Operatorima koliko ima vode u tanku. Možemo prikazati nivo vode u tanku kao:

- nescaliranu cjelobrojnu (integer) vrijednost koju primimo od I/O uređaja

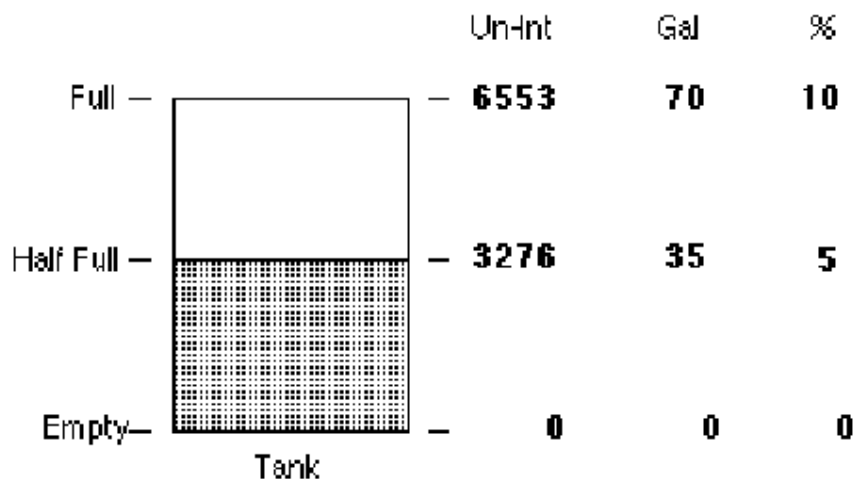
- u galonima
- u procentima (0 -100%) popunjenosti

Za ovaj primjer ćemo predpostaviti da I/O drajver prenosi nesigniranu cjelobrojnu vrijednost (bez +/- znaka) u DIT tabelu (tj. 0 – 65535). Naredna tabela izlistava donje i gornje granice uzorka koje se mogu doznačiti ulaznom bloku. Ove vrijednosti će skalirati dolazne vrijednosti da prikažu nivo vode u tanku u procentima popunjenosti i u galonima.

Tabela: Primjer graničnih vrijednosti bloka

Operatorski displej:	Granične vrijednosti:
Cjelobrojni broj bez znaka	0- 65535
Procentno	0-100
Galonima	0-70

Slijedeća slika pokazuje vrijednosti koje će biti prikazane kada je tank pun, polupun i prazan:



Skaliranje vrijednosti

Promjena preciznosti graničnih vrijednosti bloka

Ako odlučimo da je potrebno promijeniti preciznost graničnih vrijednosti bloka, možemo promijeniti granične vrijednosti i Graditelj baze podataka (DBB) će modificirati sve reference ka ovom bloku kroz čitavu bazu. Naprimjer, ako kreiramo analogni ulazni blok sa granicama bloka od 0.00 i 100.00 umjesto 0.0 do 100.0 , Graditelj baze će pretražiti bazu za sve reference ka ovom bloku i provesti odgovarajuće promjene. U ovom slučaju , Programski blok koji sadrži slijedeći korak :

SETOUT AI1 50,0

Bit će prepodešen na:

SETOUT AI1 50,00

Promjena opsega graničnih vrijednosti bloka

Ako promjenimo opseg vrijednosti granica bloka, morat ćemo promijeniti ručno sve reference na ovaj blok. Naprimjer, ako kreiramo analogni ulazni blok sa graničnim vrijednostima 0.0 i 100.0 i nakon toga odlučimo da promjenimo opseg na 0.0 do 70.0, sve reference na granice ovog bloka moraju biti ručno promjenjene. U ovom slučaju, pretpostavimo da imamo programski blok koji izbacuje na izlaz vrijednost koja je jednaka polovini originalnog opsega bloka, kao što je pokazano u nastavku:

SETOUT AI1 50.0

Iskaz SETOUT programskog bloka se mora modificirati da odrazi ovaj novi opseg, tj.

SETOUT AI1 35.0

Raspoloživi formati za granične vrijednosti blokova

Graditelj baze omogućava slijedeće formate za granične vrijednosti blokova:

- Standardni cjelobrojni (integer)
- Proširena decimalna notacija (expanded decimal notation)
- Naučna notacija (scientific notation)

Svaki od formata ima tačnost do 6 decimala. Zbog ograničenja kompajlera, moguće je pojavljivanje grešaka zaokruživanja na sedmoj decimali.

Naredna slika pokazuje koja decimalna mjesta se smatraju tačnim, Mjesta koja su zasjenjena se smatraju tačnim:

1.234567
0.234567
1.234567.654321

Tačnost graničnih vrijednosti bloka

Korištenje simulacionog drajvera (SIM)

FIX software je opremljen sa simulacionim drajverom (SIM) koji omogućava da se odredi kako dobro će lanci reagirati na kritične prekide iz procesa (kao naprimjer na gubitke napona napajanja ili visoke vrijednosti iz procesa), prije nego što priključimo sistem na stvarne I/O vrijednosti iz procesa.

SIM drajver može također generirati ponavljajući patern slučajnih ili definiranih vrijednosti da pomogne testiranje lanaca i detektuje i prikaže potvrđene i nepotvrđene alarme u bazi podataka.

SIM drajver je ustvari matrica adresa. Blokovi baze podataka čitaju vrijednosti iz i upisuju ih natrag u adrese. Ako jedan blok upisuje u specifičnu adresu, ostali blokovi mogu čitati iz te iste adrese tu vrijednost.

Da bi se koristio SIM drajver , unesite SIM u polje uređaja primarnog bloka. Format adresnog polja je kako slijedi :

register:bit

Za analogne vrijednosti , opseg registra je od 0-2000. *Bit* se ne koristi .

Za digitalne vrijednosti, registar ima isti opseg od 0-2000. Bit ima opseg 0-15. Puni opseg za setovanje *register:bit* je 0:0 do 2000:15.

Primjeri

0:0 , 50:2 , 63:15

Samo jedan set registara se dijeli izmedju analognih i digitalnih blokova. Zbog toga , možemo adresirati svih 2000 registara kao analogne ili digitalne vrijednosti. Naredna tabela pokazuje kako se setuju biti registara za digitalne vrijednosti, kada SIM registar sadrži analognu vrijednost.

Tabela : SIM analogne i digitalne vrijednosti

Kada je analogna vrijednost ...	Biti digitalne vrijednosti su ...	
65535	Biti 15 do 0 su setovani na 1	
32768	Bit 15 =1, Biti 14 do 0 su setovani na 0	
32767	Bit 15=0, Biti 14 do 0 su setovani na 1	
255	Biti 15 do 8 su setovani na 0. Biti 7 do 0 su setovani na 1	

Hardwareske opcije i polja kondicioniranja signala nisu podržavana kada se koristi SIM drajver, i sistem ne prihvata bilo kakav unos u ova polja, SIM drajver također ne podržava procesiranje bazirano na izuzeću. Nadalje , SIM drajver ne podržava preciznost sa sedam decimalnih mjesta, on koristi preciznost sa 5 decimalnih mjesta.

Opaska: Ako su granične vrijednosti blokova setovane od 0 do 100, View program neće dozvoliti Operatoru da unese bilo koju vrijednost koja je manja od 0 ili veća od 100.

SIM drajver ne izbacuje pogrešne vrijednosti niti na jedan blok. Ako testiramo sistem na toleranciju greški (fault tolerance), trebamo imati u vidu da komunikacione greške, izvještavane kao loše vrijednosti (BAD values) neće biti primljene od strane bilo kojeg ulaznog bloka.

Generiranje slučajnih vrijednosti

Da se pomogne korisniku da testira bazu podataka sa simuliranim ulazima, SIM drajver obezbjeđuje set registara koji generira ponavljajući patern slučajnih i prethodno definiranih vrijednosti. Naprimjer, možemo rampirati (inkrementalno povećavati ulaz sa datim nagibom), da vidimo kako specifični lanci se ponašaju kod izvršavanja ili možemo generirati seriju slučajnih brojeva da testiramo kompletnu bazu podataka.

Da doznačimo jedan od ovih registara bloku, potrebno je unjeti **SIM** u polje uređaj (device) za blok. U polju I/O adrese koristiti slijedeću sintaksu:

register:bit

Naredna tabela izlistava registre koji su na raspolaganju:

Tabela: SIM registri za generiranje signala

Registar	Opis	Validan unos
RA	Rampira vrijednost od 0 do 100% od EGU opsega sa brzinom kontroliranom vrijednošću u RY registru	Read only (samo iščitavanje)
RB	Inkrementira od 0 do 65535 sa brzinom od 20 brojeva po sekundi.	Read only (samo iščitavanje)
RC	Pomjera po jedan bit kroz 16 bitnu riječ sa brzinom kontroliranom sa vrijednošću u registru RZ	Read only (samo iščitavanje)
RD	Generira sinusni valni oblik od 0 do 100% od graničnih vrijednosti opsega (EGU), sa brzinom kontroliranom sa vrijednošću u RY registru	Read only (samo iščitavanje)
RE	Generira sinusni valni oblik od 0 do 100% od graničnih vrijednosti opsega (EGU), sa brzinom kontroliranom sa vrijednošću u RY registru. Valni oblik sinusa je zakašnjen za 90 stepeni relativno u odnosu na RD registar.	Read only (samo iščitavanje)

Registar	Opis	Validan unos
RF	Generira sinusni valni oblik od 0 do 100% od graničnih vrijednosti opsega (EGU), sa brzinom kontroliranom sa vrijednošću u RY registru. Valni oblik sinusa je zakašnjen za 180 stepeni relativno u odnosu na RD registar.	Read only (samo iščitavanje)
RG	Generira slučajne vrijednosti između 25% i 75% od EGU opsega	Read only (samo iščitavanje)
RH	Rampira vrijednost od 0% do 100% od EGU opsega i nakon toga rampira nadole do 0% sa brzinom kontroliranom sa vrijednošću u RJ registru.	Read only (samo iščitavanje)
RI	Kontrolira smjer rampe u RH registru. Kada je vrijednost 0 , registar RH rampuje nadole, kada je 1, RH rampuje nagore. Vrijednost se automatski mijenja kada RH dostigne vrijednost 0 ili 100% od svoje EGU vrijednosti.	Numerička vrijednost 0 ili 1
RJ	Kontrolira brzinu rampe (u ciklusima po satu), za vrijednost u registru RH. Default vrijednost je 60 (1 ciklus po minuti)	Numerička vrijednost (2 do 3600)
RK	Omogućava ili onemogućava generiranje vrijednosti u RH registru. Unesite vrijednost 0 da zamrznete (onemogućite) rampu a ne nultu vrijednost (1) da omogućite rampu.	Numerička vrijednost 0 ili 1
RX	Omogućava ili onemogućava generiranje vrijednosti u drugim registrima. Unesite vrijednost 0 da zamrznete (onemogućite) registre a ne nultu vrijednost da omogućite sve registre	Numerička vrijednost 0 ili 1
RY	Kontrolira brzinu (u ciklusima po satu)sa kojom se generiraju RA, RD, RE, ili RF. Po defaultu RY registar je setovan na 60 (1 ciklus u minuti).	Numerička vrijednost (2do 3600)
RZ	Kontrolira brzinu (u ciklusima po satu)sa kojom registar RC mijenja svoju vrijednost. Po defaultu , RZ registar je setovan na 180 (3 bita pomjeranja po sekundi)	Numerička vrijednost (2do 1200)

Svi SIM registri podržavaju blokove :Analogne ulaze , Analogne registre, Digitalne ulaze i Digitalne registre.

Blok	Register
Analogni ulaz	RA,RD,RE,RF,RG i RH
Analogni izlaz	RJ,RY iRZ
Analogni registar	RA,RD,RE,RF,RH,RI,RJ,RK,RX,RY i RZ
Digitalni ulaz	RB i RC
Digitalni Registar	RB,RC , RI, RK i RX

Opaska: Registri RB i RC podržavaju ofsete Digitalnig registara od A_0 do A_15.

Prikazivanje potvrđene i nepotvrđene informacije o alarmu

Pored pomoći u testiranju lanaca u bazi podataka, SIM drajver može otkriti alarme u bazi podataka i prikazati broj potvrđenih i nepotvrđenih alarma u FIX sistemskom softwaru. Koristeći ove mogućnosti, korisnik može, naprimjer da aktivira zvučni alarm u PLC-u kada god se otkrije nepotvrđeni alarm.

SIM drajver broji i obezbjedjuje slijedeće metode da prikaže vrijednost svakog brojača alarma Operatoru:

- Doznači blok SIM drajveru i kreira Data link za taj blok u Draw programu
- Kreira Data link u Draw da pristupi brojaču direktno iz SIM drajvera.

Metod koji će biti primjenjen zavisi od potreba korisnika. Ako on planira da poduzme neku akciju baziranu na jednom ili više brojača alarma, tada treba da doznači jedan ili više blokova za SIM drajver. Medjutim, ako samo želi da prikaže ukupni broj alarma , (naprimjer broj potvrđenih alarma niskog nivoa prioriteta), može pristupiti ovom brojaču direktno iz SIM drajvera.

Doznačavanje bloka SIM drajveru

Da bi se doznačio blok SIM drajveru, potrebno je unjeti **SIM** u polje uređaja (device) za blok. U polju I/O adrese treba koristiti slijedeću sintaksu:

C:field

Field je jedno od polja brojača alarma.

Primjer

Predpostavimo da u čvoru egzistira 20 potvrđenih alarma niskog nivoa prioriteta. Da bi se pohranila ova vrijednost u blok analognog ulaza (Analog Input), unesite slijedeću adresu u polju I/O adrese:

C:ACKL

Da bi se prikazala ova vrijednost Operatoru, kreirajte Data link za blok analognog ulaza i specificirajte A_CV ili F_CV polje za ime taga linka.

Direktno pristupanje brojačima alarma

Da se prikaže ovaj broj alarma direktno iz SIM drajvera, koristite slijedeću sintaksu imena taga u Data linku:

node:SYSTEM.field

Primjer

Predpostavimo da je ime čvora (node), SCADA01. Koristeći gornji primjer, možemo direktno pristupiti i prikazati broj potvrđenih alarma niskog nivoa prioriteta iz SIM drajvera , kreirajući Data link i unoseći slijedeće ime taga:

SCADA01:SYSTEM.A_ACKL

Primjetimo da A_ dio polja je uključen kada pristupamo SIM drajveru direktno iz Draw programa.

Pristupanje sumarnoj informaciji

Pored brojača alarma, SIM drajver takodjer vodi i sumarni pregled svih potvrđenih i nepotvrđenih alarma izlistanih po alarmnim oblastima i prioritetima. Ove sumarne vrijednosti indiciraju da li potvrđeni ili nepotvrđeni alarmi postoje u bazi podataka.

Pristupanje sumarnim informacijama direktno iz SIM drajvera je identično pristupanju brojačima alarma. Doznačavanje bloka da pristupi sumarnoj informaciji je slično pristupanju brojaču alarma. Unesite SIM u polje za uređaj (device) a za polje I/O adrese koristite slijedeću sintaksu:

C:field:bit

Field je jedno od polja brojača alarma. *Bit* dio je potreban jedino kada se koristi digitalni blok.

Kada se koristi analogni blok, SIM drajver vraća jednu od slijedećih vrijednosti:

Tabela : Globalne vrijednosti alarma

Vrijednost	Ima bit patern	I indicira..
0	00	Nema alarma

1	01	Postoji prethodno potvrđen alarm
2	10	Postoji nepotvrđen alarm
3	11	Postoje i potvrđeni i nepotvrđeni alarmi

Kada se koristi digitalni blok, moguće je maskirati desni ili lijevi bit u prethodnoj tabeli. Svaki bit indicira tip alarma koji postoji (potvrđen ili nepotvrđen).

Rad sa DDE Drajverom

FIX software uključuje i DDE drajver koji omogućava čitanje vrijednosti kao i njihovo upisivanje na slijedeće izvore:

- I/O drajvere koji djeluju kao DDE serveri
- DDE serverska aplikacija (kao naprimjer Excel)
- Druge FIX baze podataka

Čitanje podataka iz ovih izvora je locirano u procesnim bazama podataka. Pošto su podaci raspoloživi u procesnim bazama podataka, korisnik ih može uključiti u svoju strategiju da:

- prenese podatke kroz lance
- alarmira na osnovu ovih DDE podataka
- trendira na bazi ovih DDE podataka

Koristeći ulazne blokove, kao što je blok analognog ulaza, možemo čitati iz *DDE adrese*. DDE vrijednosti dobivene iz DDE adrese se pohranjuju u bazu podataka na isti način kao i informacije od standardnih I/O uređaja. Zbog toga, standardne alarmne konfiguracije u blokovima se mogu koristiti da trigeruju alarme. Pošto je DDE informacija sadržana u bazi podataka, može se prenjeti do bilo kojeg bloka u lancu i korištena da izvršava i historijsko i trendiranje u realnom vremenu.

Korisnik može upisivati na DDE lokacije koristeći izlazne blokove kao što je naprimjer blok analognog izlaza. Adresirajući izlazni blok na DDE izvor, izlaz će biti poslat na DDE adresu kada se procesira izlazni blok.

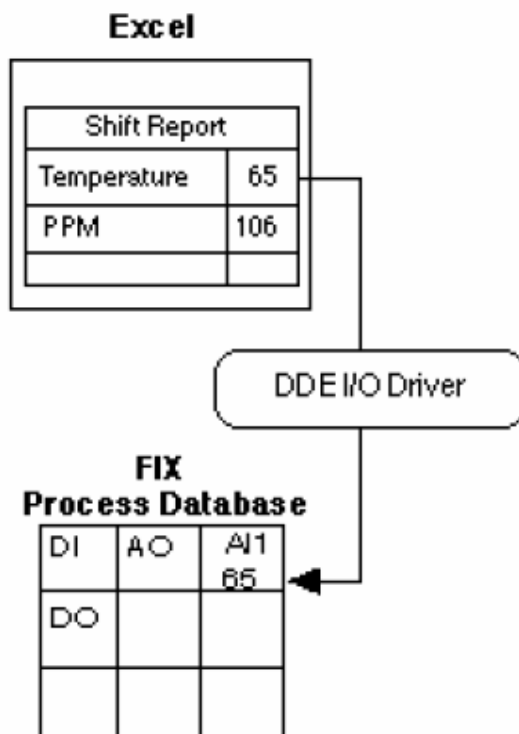
Korištenje DDE Klijent podrške u bazama podataka

FIX DDE klijent podrška dozvoljava korisnicima da čitaju sa i pišu na DDE adrese. Koristeći DDE I/O drajver i DDE adresu u blok konfiguraciji, korisnik može inkorporirati informaciju u procesnu bazu podataka iz druge aplikacije, iz DDE drajvera za uređaj ili iz drugog SCADA čvora. Pošto je podatak na raspolaganju u bazi podataka, može se uključiti u lanac i izvršiti alarmiranje i trendiranje.

1. The user enters a new value in the spreadsheet.

2. The DDE I/O Driver automatically transfers the data from the server to the client.

3. The database block updates with the new value.

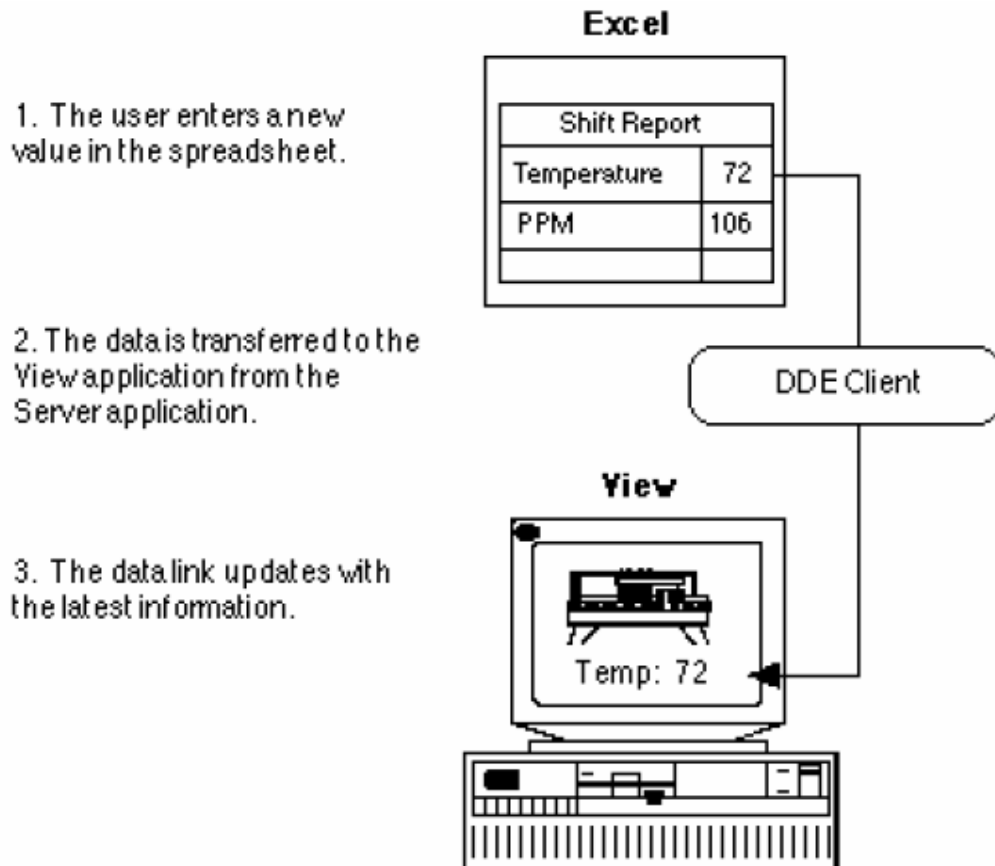


DDE klijent podrška u bazi podataka

Korištenje DDE klijent podrške u slikama

Koristeći DDE klijent podršku, korisnik može prikazati DDE informaciju na svojim slikama koristeći DDE adresu umjesto imena taga u linkovima. Ovo omogućava prikazivanje informacija iz drugih izvora, kao što je statistički program, sa procesnim informacijama iz baze podataka. Možemo također uključiti DDE podatke u CLS (skript komandnog jezika).

Naredna slika ilustrira kako su podatci prebačeni iz DDE Server aplikacije i linkovani sa slikom.



DDE klijent podrška u slikama

FIX DDE Server podrška

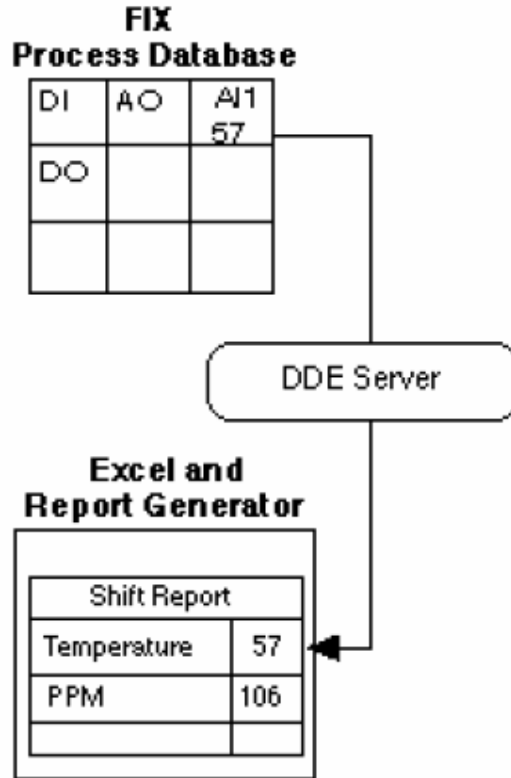
FIX DDE Server podrška omogućava korisniku da prebacije podatke iz FIX baze ka drugim DDE klijent aplikacijama. Korisnik može koristiti linkove koji su mu na raspolaganju u opciji softwarea Report Generatora da brzo kreira standardne izvještaje u Excelu. Možemo koristiti DDE adrese da prebacimo podatke ka drugim DDE klijent aplikacijama.

Naredna slika ilustrira kako Excel izvještaj je ažuriran sa real time podacima da bi se dobili ažurni izvještaji.

1. The I/O device updates the FIX database with new values.

2. The DDE Server automatically transfers the data to the client as the value changes.

3. The pre-defined links update with the latest process information.



DDE Server podrška za izvještaje

Korištenje DDE adresa

DDE klijent zahtjeva podatke od DDE servera koristeći DDE adrese. Sintaksa koja se koristi kod DDE adresa je različita zavisno da li je Server lokalni ili daljinski. Lokalni DDE server je DDE aplikacija koja je locirana na istom računaru kao i klijent aplikacija. Daljinski DDE server je DDE aplikacija koja je locirana na različitom računaru od klijent aplikacije.

DDE adresiranje

Da bi se pristupilo DDE informaciji sa lokalnog DDE Servera, unesite **DDE** u polje uređaja (device) i unesite DDE adresu u polje I/O adrese primarnog bloka. Da bi se pristupilo lokalnom DDE serveru, koristićemo dakle slijedeću sintaksu:

=application|topic!item

Lokalna DDE adresa se sastoji od slijedećih stringova

Application – definira ime aplikacije (ili servisa) sa kojom želimo da komuniciramo. Ime aplikacije može biti dugačko do 26 karaktera.

- Topic** – definira podatke kojima želimo da pristupimo. Topic varira zavisno od aplikacije. Naprimjer, to može biti radni list (worksheet) u Excelu. Topic može biti dug do 20 karaktera.
- Item** – definira jedinicu podatka. Item varira zavisno od servisa i topica. Naprimjer, item može biti ćelija u Excelovom radnom listu ili ime taga u bazi podataka. Ime itema može biti dugo do 79 karaktera.

Excel Primjer

Predpostavimo da želimo konfigurirati blok da čita sa lokalnog DDE servera. Specifikacija hardwareske grupe u bloku može da izgleda kao :

Device : **DDE**
 Hardware options:
 I/O address: = **EXCEL|[FILE.XLS]SHEET1!R1C1**
 Signal conditioning:

U ovom primjeru, ulaz za drajver uredjaja , DDE, indicira da podatci dolaze iz DDE servera. DDE adresa zahtjeva podatke iz prvog reda i kolone Excel radnog lista sa imenom file.xls. Kondicioniranje signala se ne vrši na DDE podatku. Excel nam dozvoljava da kreiramo multiple sheetove unutar radne knjige (workbooka).

NetDDE adresiranje

FIX software može koristiti i NetDDE Server i klijent podršku koja je obezbjedjena kao dio Operativnog sistema. NetDDE dozvoljava DDE omogućenim aplikacijama da zahtjevaju ili primaju podatke od druge DDE omogućene aplikacije na udaljenom čvoru unutar mreže.

Ako pristupamo informaciji na udaljenom DDE serveru koristeći NetDDE, sintaksa DDE adrese je različita od maloprije opisane za lokalni čvor. NetDDE adresa se sastoji iz četiri dijela kako slijedi:

= \\Ime računara\NDDE\$\DDEShare.DDE!Item

gdje :

Ime računara – je ime računara radne stanice ili Servera. Ime je definirano u polju imena računara u okviru mrežnog setinga, u Kontrolnom panelu

NDDE\$ - je rezervirano ime koje indicira da je NetDDE korišten za prenos podataka

DDEShare – je rezervirano ime koje se konfigurira za svaku aplikaciju i topic sa kojim ćemo komunicirati. DDEShare ime je alias koji definiramo. Potrebno je paziti da dodamo .DDE kao suffix na Share ime kada specificiramo adresu.

Item ime – je individualni dio podatka koji se prenosi. Ime itema zavisi od od metoda pomoću kojeg je podatak pohranjen u aplikaciju servera.

Primjetimo da je sintaksa NetDDE osjetljiva na velika i mala slova (case sensitive).

Primjer

Predpostavimo da imamo Excel workbook , FILE.XLS, sa dva worksheets , SHEET1 i SHEET2. Nadalje predpostavimo da je DDEShare setovan za svaki od sheetova. Da bi pristupili informaciji na sheetu SHEET2 workbooka FILE.XLS, koristićemo slijedeću sintaksu:

= \\NODE1\NDDE\$\\$SHIFT2.DDE!R1C1

gdje NODE1 je ime računara na udaljenom čvoru servera, NDDE\$ indicira da se koristi NetDDE , \$SHIFT2 je DDEShare ime za SHEET2, a R1C1 je item u ćeliji prve kolone i reda.

Broj karaktera koji možemo koristiti u NetDDE adresi zavisi od toga da li specificiramo blok baze podataka, ili ime taga u Draw linku ili Komandnom jeziku (CLS). Naredna tabela pokazuje koji je maksimalni broj karaktera koji može biti korišten.

Kada unosimo I/O adresu...	Kada unosimo ime Taga..
Do 26 karaktera za ime računara i rezervirano ime NDDE\$	Do 31 karakter može biti korišten za ime računara i rezervirano ime NDDE\$
Do 20 karaktera može biti korišteno za DDEShare ime	Do 31 karakter može biti korišten za DDEShare ime
Do 79 karaktera može biti korišteno za item	Do 25 karaktera može biti korišteno za item

Ako pokušavamo da pristupimo FIX podacima iz Excel spreadsheeta koristeći NetDDE, moramo zatvoriti Ime kompjutera i NDDE\$ ključnu riječ kao i DDEShare ime u jednostruke znake navodnika (' ...'). Naprimjer:

= "\\NODE1\NDDE\$|"DMDATA.DDE!" SCADA1.AI1.A_CV"

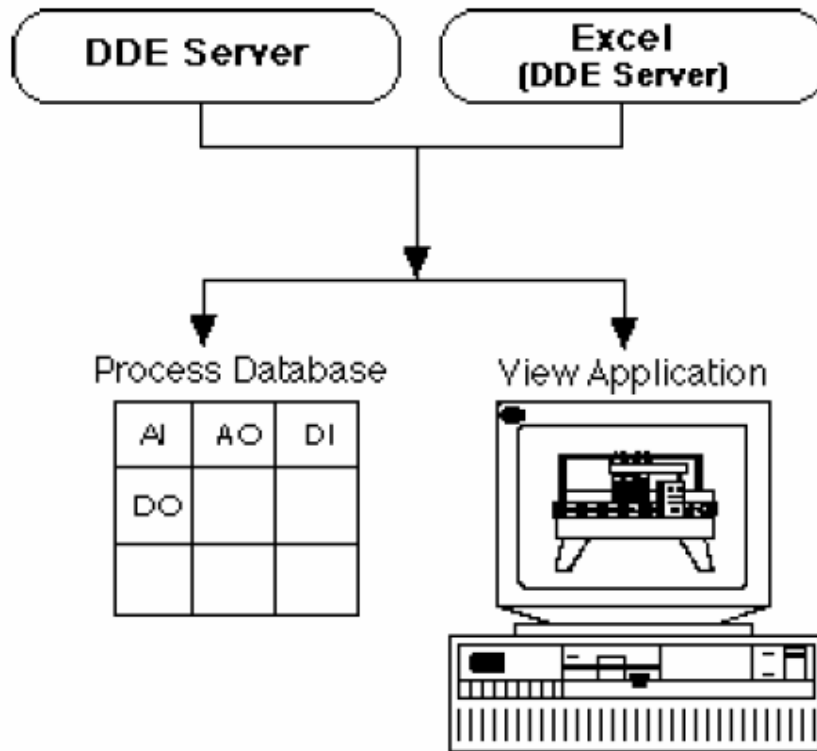
Korištenje DDE klijent podrške

Naredni tekst će opisati DDE klijent podršku koja je na raspolaganju u FIX softwareu. On takodjer objašnjava kako da konfiguriramo DDE klijent podršku i kako da tražimo i otklanjamo greške kod DDE konekcija.

Razumjevanje DDE klijent podrške

DDE klijent podrška nam omogućava da prenesemo podatke u FIX software i da ih koristimo na slikama ili procesnoj bazi podataka.

Naredna slika ilustrira kako FIX DDE klijent podržava rad sa Excelom i drugim aplikacijama koje koriste DDE mehanizam.



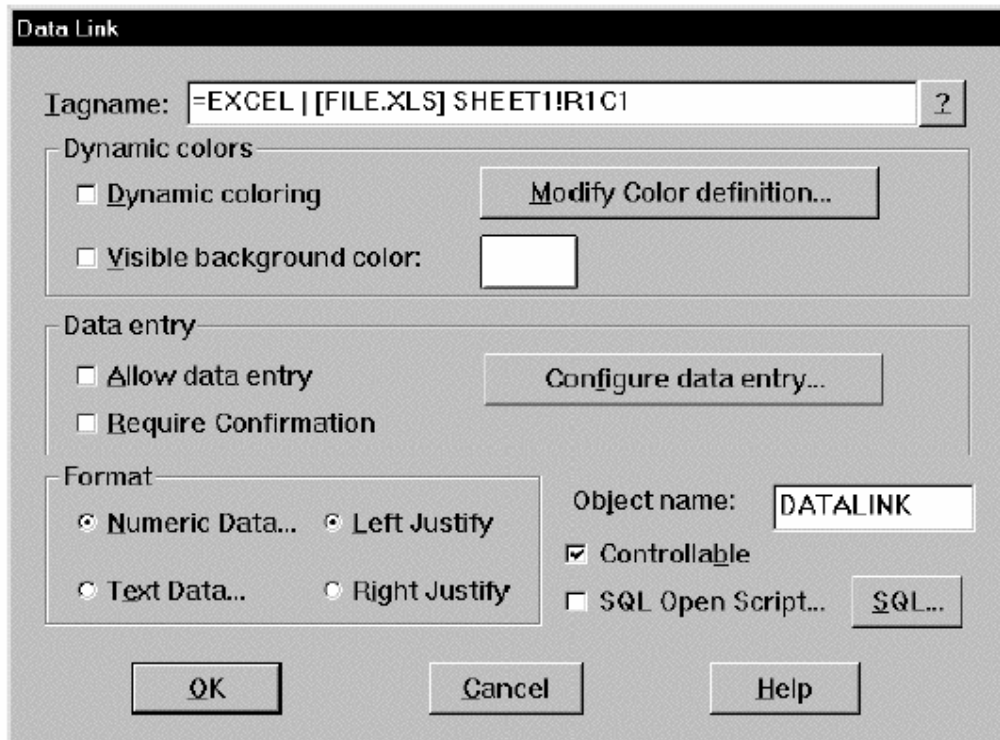
Ilustracija DDE klijenta

Korištenje DDE klijent podrške u slikama

DDE klijent podrška dozvoljava korisniku da prikaže DDE informaciju na svojim slikama u View , koristeći DDE adrese umjesto imena Tagova. DDE informacija zaobilazi bazu podataka i direktno se prikazuje na slici. DDE može niti korišten u slijedećim situacijama u Draw:

- data linkovima
- dinamičkim osobinama (prednja boja, boja ivice (edge color), X pozicija, Y pozicija, horizontalno punjenje bojom (fill), vertikalni fill, vidljivost (visibility), skala, i rotiranje.)
- X/Y plot, više kanalno zapisivanje (multipen), i multi-bar zapisi.
- komandnom jeziku

Naredna slika prikazuje kako koristiti DDE adrese u data linkovima



Primjer data linka

Umjesto unošenja imena taga da bi se prikazala informacija iz procesne baze podataka, FIX dozvoljava da unesemo DDE adresu. U ovom primjeru, informacija pohranjena u prvom redu i koloni na sheetu 1 Excelove workbook , FILE.XLS, je prikazana. Kako se informacija u spredu sheetu bude ažurirala tako će se ažurirati i link na slici.

Korištenje DDE klijent podrške u bazi podataka

Korisnik može uključiti DDE podatke u bazu podataka koristeći ulazne blokove, kao što je analogni ulazni blok. Standardna konfiguracija alarma u bloku se može koristiti da se trigeruju alarmi. Pošto je informacija pohranjena u bazi podataka , može se prenjeti na ostale blokove u lancu kao i koristiti za real time i historijsko trendiranje.

Korisnik također može pisati u DDE lokacije koristeći izlazne blokove, kao što je analogni izlazni blok. Afdresiranjem izlaznog bloka, izlaz će se poslati na DDE adresu kada se izlazni blok bude procesirao od strane SAC-a u lancu.

Informacija koja se unosi u FIX software koristeći DDE I/O drajver je u ASCII formatu. Ovo može predstavljati problem kada koristimo digitalne ulazne blokove pošto ASCII verzija tekuće vrijednosti bloka (CV), je korisnički definirani tekst string. Da bi se vrijednost korektno pohranila u bazu podataka, korisnik treba da obezbjedi da vrijednosti koje koristi u DDE server aplikaciji su iste kao *OPEN* i *CLOSE* labela koje on konfigurira u ciljnom bloku baze podataka.

Višestepeni digitalni ulazni blok (multistate DI -MDI), koristi OPEN i CLOSE kao defaulte za 0 i 1 vrijednosti respektivno. Korisnik može promijeniti ove labela koristeći VIEW linkove ili EDA (Easy data base) program pristupa.

Djeljenje podataka izmedju FIX DDE baza podataka

Koristeći DDE I/O drajver i FIX DDE server, korisnik može prebacivati podatke izmedju SCADA čvorova. Ovaj metod automatski ažurira obadvije baze podataka kada se informacije promjene, eliminirajući na taj način nepotrebne duple konekcije na I/O uredjaj koji obezbjedjuje te informacije.

Naredna tabela ilustrira konfiguraciju koju moramo izvršiti na izvornom i prijemnom čvoru da bi koristili ovaj metod dijeljenja podataka.

FIX software izvorni čvor	FIX software prijemni čvor
[1] Konfiguriraj blok baze podataka da pristupi I/O uredjaju	[1] Konfiguriraj blok baze podataka da pristupi DDE adresi na izvornom čvoru
	[2] Startaj DDE I/O drajver software
	[3] Startaj DDE Server

Konfiguriranje SCADA – SCADA prenosa podataka

Naprimjer, ako imamo dva SCADA čvora u mreži izmedju kojih želimo da uspostavimo prenos podataka, treba provesti slijedeće korake :

[1] Konfigurirati blokove baze podataka na izvornom SCADA čvoru, NODE1, koji je spojen sa PLC da bi pristupio I/O uredjaju.

[2] Konfigurirati blokove u čvoru NODE2 za DDE prenos koristeći slijedeće unose kao primjer:

Device: **DDE**
 Hardware Options:
 I/O Address: = **DMDDE| DATA!NODE1:AI1.A_CV**
 Signal Conditioning:

Koristeći ovu konfiguraciju, blokovi u NODE2 se ažuriraju kada se podatci ažuriraju u čvoru NODE1. Ovo dozvoljava korisniku da koristi istu informaciju od I/O uredjaja u dvije različite baze podataka bez da kreira dvije odvojene konekcije ka I/O uredjaju.

Opaska: Koristeći FIX DDE Server , samo ASCII polja (A_) mogu biti korištena za upisivanje podataka. Obadva ASCII(A_) i floating point (F_) polja mogu biti korištena za čitanje podataka.

Traženje i otklanjanje grešaka (troubleshooting) za DDE klijent podršku u bazi podataka

FIX software obezbjedjuje tri metoda pristupa informacija trobleshootinga o DDE klijent komunikacionim sesijama. Ove su:

- DDE klijent task poruke
- DDE klijent task dijalog boksovi
- DDE klijent task itemi

Svaka od ovih metoda je opisana u slijedećim podsekcijama:

DDE klijent task poruke

FIX DDE klijent task može biti konfiguriran da prikaže poruke kod startovanja i za vrijeme runtime (izvršenja).

DDE klijent task prikazuje slijedeće tipove poruka:

- Problemi startovanja
- Neuspjeh uspostavljanja DDE konverzacije
- Neuspjeh uspostavljanja *DDE advise* konture.
- Neuspjeh kompletiranja izlaza

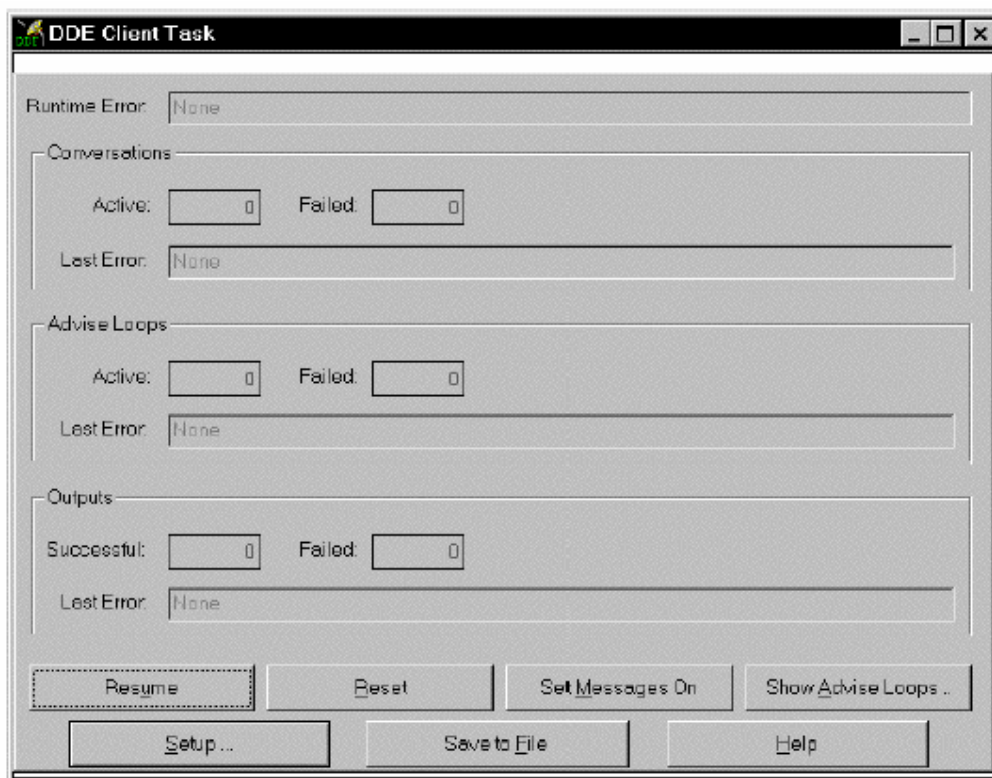
Opaska: FIX poruke se ne šalju na link sumarnih alarma (Alarm Summary). Mogu se konfigurirati da idu u file, na štampač, ili u prozor historije alarma.

DDE klijent task dijalog boks

DDE task dijalog boks prikazuje statističku informaciju za DDE klijent sesije. Ova informacija može biti korištena za :

- Nadzor DDE ažuriranja i izlaza
- Omogućavanja i onemogućavanja DDE klijent poruka
- Posmatranja poruka grešaka kod izvršenja (runtime)

Kada se maksimizira DDE klijent task, pojavljuje se DDE klijent task dijalog boks, kao što je pokazano na slijedećoj slici:



DDE kljent task dijalog boks

Dijalog boks sadrži nekoliko brojača i poruka grešaka koji se mogu koristiti za nalaženje grešaka. Brojači se resetuju na 0 (nulu), ili na riječ None pod slijedećim uslovima:

- startanjem DDE kljent taska
- ponovnim punjenjem (reload) baze podataka
- ručnim resetovanjem brojača koristeći Reset dugme

Naredna lista opisuje polja i dugmad u dijalog boksu :

Runtime error – prikazuje bilo koji problem koji je prouzrokovan DDE kljent taskom, koji dovodi do suspenzije rada. Većina runtime grešaka se pojavljuje za vrijeme DDE kljent task startanja i indicira problem sa resursom ili memorijom.

Konverzacije

DDE konverzacije su konekcije na DDE server putem *Application* i *Topic* dijelova DDE adrese. Ako koristimo DDE komunikaciju , kolona aplikacija prikazuje ime računara, a kolona Topic prikazuje DDEShare ime i NetDDE adresu.

Active – prikazuje broj trenutno aktivnih DDE konverzacija

Opaska: Ako koristimo NetDDE komunikaciju izmedju dva FIX čvora, i DDE server software se ne izvršava, broj aktivnih konverzacija će se mjenjati izmedju 1 i 0. Ova

vrijednost prvo indicira aktivnu sesiju sa lokalnim NetDDE. Vrijednost nakon toga se mijenja na 0 da označi da je daljinski konverzacioni status NetDDE neaktivan.

Failed – prikazuje broj neuspjelih pokušaja da se uspostavi DDE konverzacija nakon što je startovan i resetovan. Ovaj broj raste ako je DDE server neaktivan, ili su prisutni mrežni problemi, ili je *Topic* pogrešan.

Last Error - prikazuje posljednju grešku koja se pojavila kada se pokušala uspostaviti DDE konverzacija.

Advise kontura

Advise kontura je konekcija na pojedinačni detalj (item). Mogu postojati višestruke advise konture unutar jedne konverzacije.

Active - prikazuje broj trenutno aktivnih advise kontura.

Failed – prikazuje broj neuspjelih advise kontura nakon što je advise task startovao ili je resetovan. Ovaj broj će rasti ako je item nekorektan.

Last Error – prokazuje posljednju grešku koja se pojavila pri pokušaju uspostavljanja advise konture.

Izlazi

Successful – prikazuje broj uspješnih upisa iz FIX softwarea u DDE aplikaciju nakon što je DDE klijent task startovao ili je resetovan.

Failed – prikazuje broj neuspjelih upisa iz FIX softwarea u DDE aplikaciju nakon što je DDE klijent task startovao ili je resetovan.

Last Error – prikazuje posljednju grešku koja se odnosi na izlaz koja se pojavila.

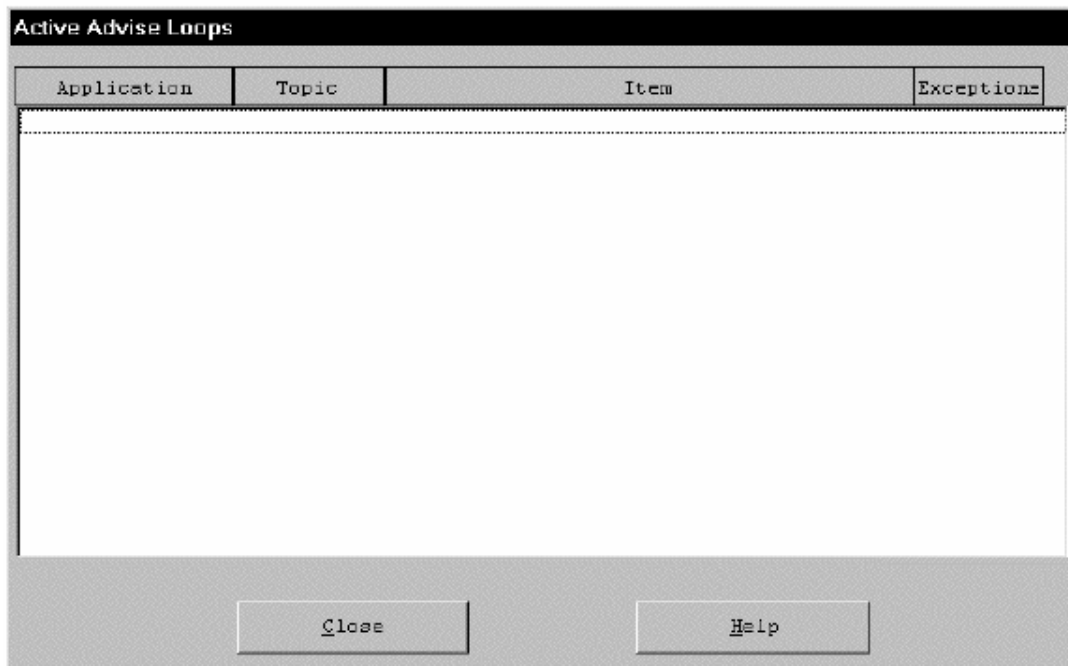
Dugmad

Pause/Resume – dozvoljava korisniku da pauzira ili ponovo nastavi ažuriranje svih brojača i grešaka u dijalog boksu. Kada selektiramo *Pause* dugme, tekst na dugmetu se mijenja na *Resume* i brojači se zaustavljaju. Kada ponovo kliknemo na *Resume* dugme, brojači će nastaviti da se ažuriraju sa dodatkom brojenja koje se pojavilo za vrijeme pauze.

Reset – resetuje sva polja u dijalog boksu. Brojači se setuju na broj 0 , i polja posljednje greške se resetuju na riječ *None*.

Set Messages On/Off – omogućava ili onemogućava slanje DDE poruka na njihova odredišta. Ove poruke se mogu koristiti da se traže greške u DDE konekcijama.

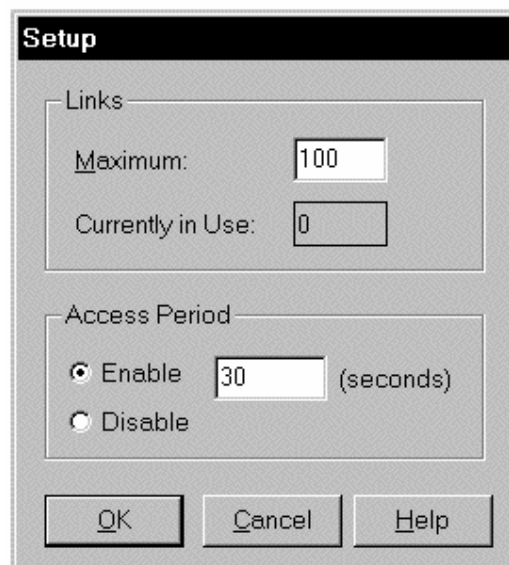
Show advise Loops – prikazuje diajlog boks aktivnih advise kontura kao na slijedećoj slici:



Dijalog boks aktivnih advise kontura

Ovaj dijalog boks prikazuje sve aktivne advise konture u čvoru. Svaka trenutno aktivna DDE adresa, ili advise kontura je izlistana. Dijalog boks također prikazuje broj ažuriranja koji je registriran za svaku aktivnu advise konturu u koloni izuzeća (exceptions).

Setup - dozvoljava da se modifikuju konfiguracioni parametri za DDE klijent task. Dva parametra koja se mogu modificirati su period pristupa (access period) i broj maksimalnih linkova. Dijalog boks prikazan na narednoj slici će se pojaviti kada se izabere Setup dugme.



Setup dijalog boks

Maximum links – definira maksimalni broj DDE adresa, koje mogu biti aktivne na ovom čvoru. Default vrijednost je 100. Validan opseg ulaza je od 1 do 10000. Poruka upozorenja će biti poslata na sve konfigurirane destinacije poruka kada se svi slotovi budu koristili. Ako se ovo desi, dodatni linkovani blokovi ne idu na skaniranje ili poslate izlaze.

Opaska: Ako modificiramo vrijednost Maksimalnog broja linkova, moramo ponovo startati FIX software da bi promjene bile izvršene.

Currently in use – prikazuje broj DDE adresa ili linkova koje su trenutačno aktivne u čvoru. Ovaj broj može biti upoređen sa brojem maksimalnih linkova za svrhe optimizacije.

Access Period - kontrolira dužinu vremena koji advise konture ostaju aktivne poslije posljednjeg zahtjeva za podacima. Period pristupa (*access period*) je definiran jedanput za sve *Advise* konture na čvoru. Period pristupa se definira izabirući *Enable* dugme i unoseći vrijednost u polju *Access time*. Default period pristupa je 30 sec. Validan opseg unosa je od 0 do 172800 sekundi. Izabirući *Disable* radio dugme dozvoljava *Advise* konture da ostanu neograničeno otvorene.

Konfiguracija Setup dijalog boksa se pohranjuje u DDE klijent inicijalizacioni file, DDECLNT.INI, svaki put kada se OK dugme izabere.

Opaska : *Advise* konture sa vremenom skaniranja, kao naprimjer analogni ulazni blok (*AI*), ne postaju neaktivne bez obzira na period pristupa.

Save to file – pohranjuje tekuće vrijednosti brojača, poruke grešaka, i aktivne *Advise* konture u tekst file. Kada izaberemo "Save to file" dugme, "Save As" dijalog boks će se pojaviti i možemo definirati ime filea i stazu gdje će biti pohranjen tekst file.

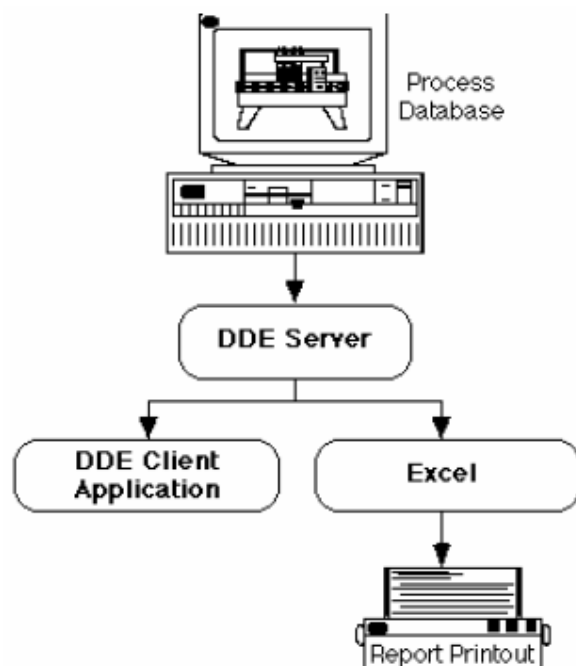
Korištenje DDE Servera

DDE Server nam omogućava da prenosimo real-time i historjske podatke iz FIX softwarea drugoj DDE omogućenoj aplikaciji, kao što je MS Excel.

MS Excel, u sprezi sa DDE Serverom i generatorom reporta opcijom softwarea, dozvoljava nam da kreiramo i generiramo izvještaje koristeći prethodno definirane macroe izvještaja. Ovi macroi nam omogućuju da izvršavamo komplicirane taskove koristeći jednostavne sekvence tastera ili klik miša, bez da budemo eksperti za DDE komande i Excel funkcije.

Nadalje, DDE server i generator izvještaja nam omogućuju da kreiramo i generiramo izvještaje u MS Excelu radi praćenja procesa proizvodnje.

Naredna slika ilustrira kako FIX DDE Server radi sa MS Excelom i drugim aplikacijama koje koriste DDE mogućnosti.



Ilustracija DDE Servera

Konfiguracija NetDDE Server podrške

FIX podržava NetDDE mogućnosti Operativnog sistema. Da bi se konfigurirala NetDDE server podrška na FIX čvoru, Operativni sistem na računaru mora bit i NetDDE Server čvor i klijent čvor. Ovi čvorovi moraju sadržavati slijedeće aplikacije:

NetDDE Server čvor	NetDDE klijent čvor
32 bitni FIX software Ver. 6.0 ili veća FIX DDE Server Jedan ili više konfiguriranih DDEShares	DDE omogućena aplikacija koja će zahtjevati i primati podatke od FIX softwarea. (naprimjer MS Excel)

NetDDE adrese koriste DDEShare imena da pristupe DDE aplikaciji na udaljenom serveru. Da bi kreirali DDEShare na WindowsNT čvoru , mi moramo koristiti DDEShare program. Ovaj utility program nam omogućava da kreiramo svaki DDEShare koji nam je potreban sa doznačavanjem različitih nivoa dozvola korisnicima koje omogućimo.

Predpostavimo naprimjer da smo kreirali DDEShare koje smo nazvali \$TEST. Takodjer predpostavimo da smo doznačili da su korisnici Senad i Ferid za ovaj share. Ako izaberemo read-access za ove vlasnike, kada Senad ili Ferid pristupe share \$TEST, njima će biti doznačen samo read-access.

Nakon doznačavanja dozvola za svaki DDEShare koji smo kreirali, treba da damo povjerenje (trust) za share. Share kojoj se vjeruje (trusted share) omogućava aplikaciji da dijeli podatke i da trenutno logovanim korisnicima da pristupe dijeljenoj aplikaciji. Ako se više operatora lokalno loguje na isti računaru, treba doznačiti trusted share za svakog operatora pojedinačno.

Kreiranje DDEShare za WinNT

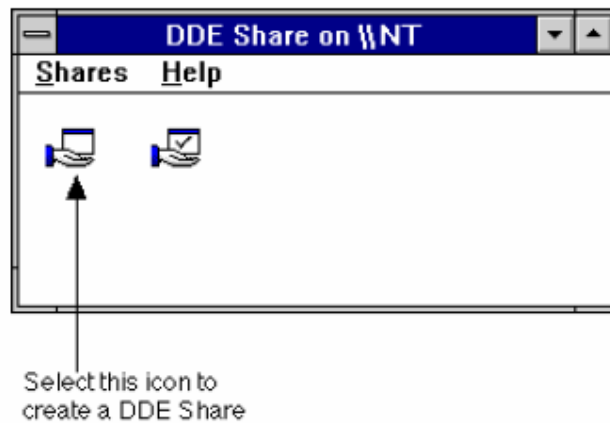
Da bi ovo kreirali potrebno je uraditi slijedeće:

[1] Iz program Managera , izabrati Run iz File Menu-a

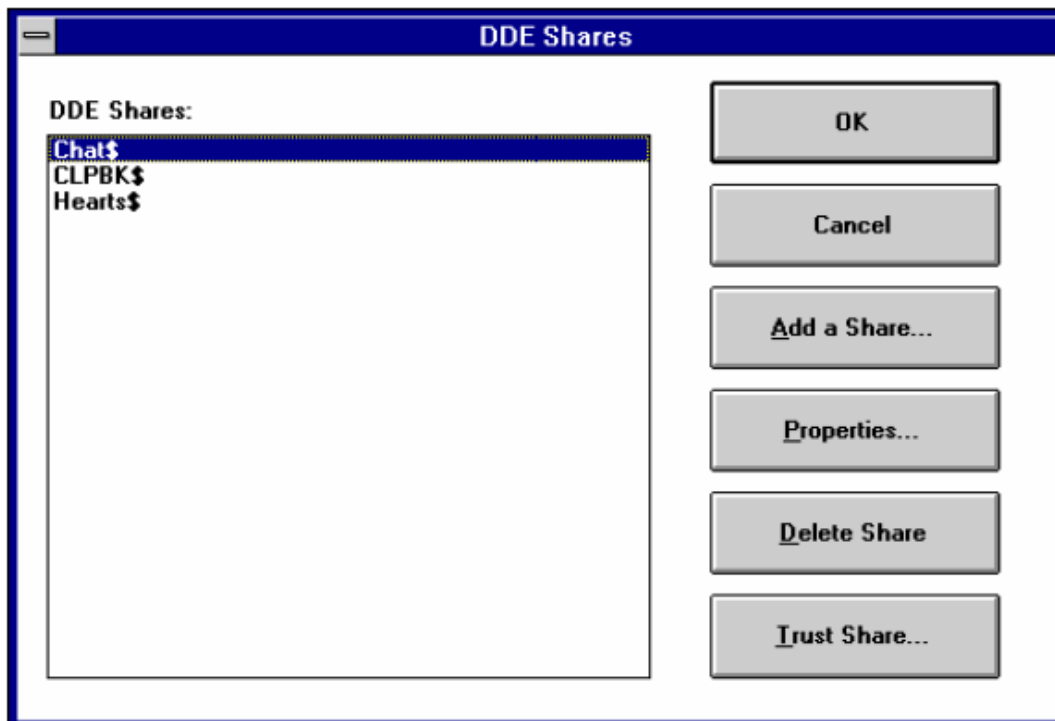
[2] Unjeti slijedeći tekst u Run dijalog boks

DDESHARE

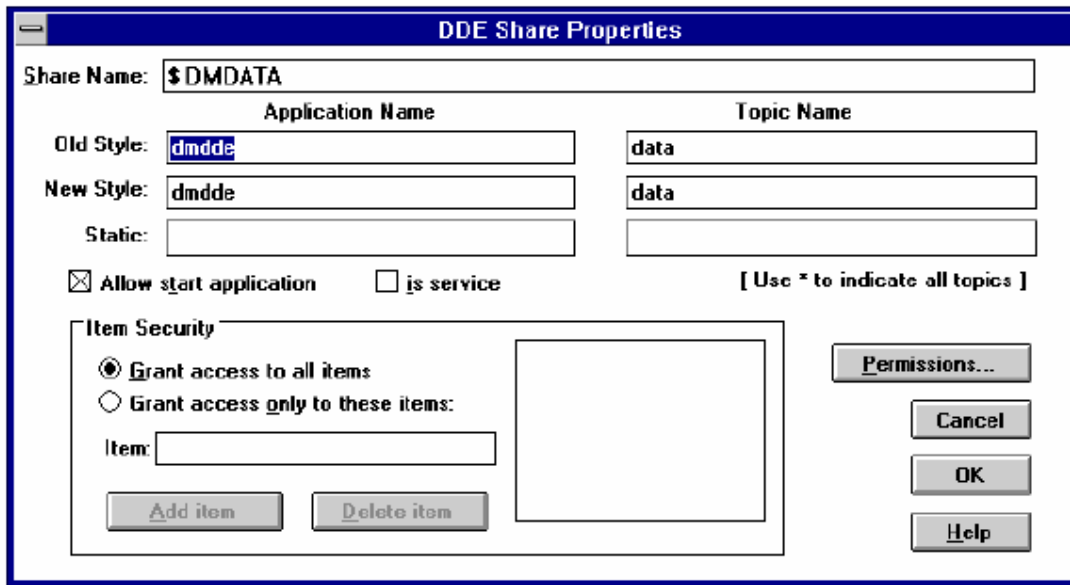
[3] Selektirati ikonu kao na slijedećoj slici :



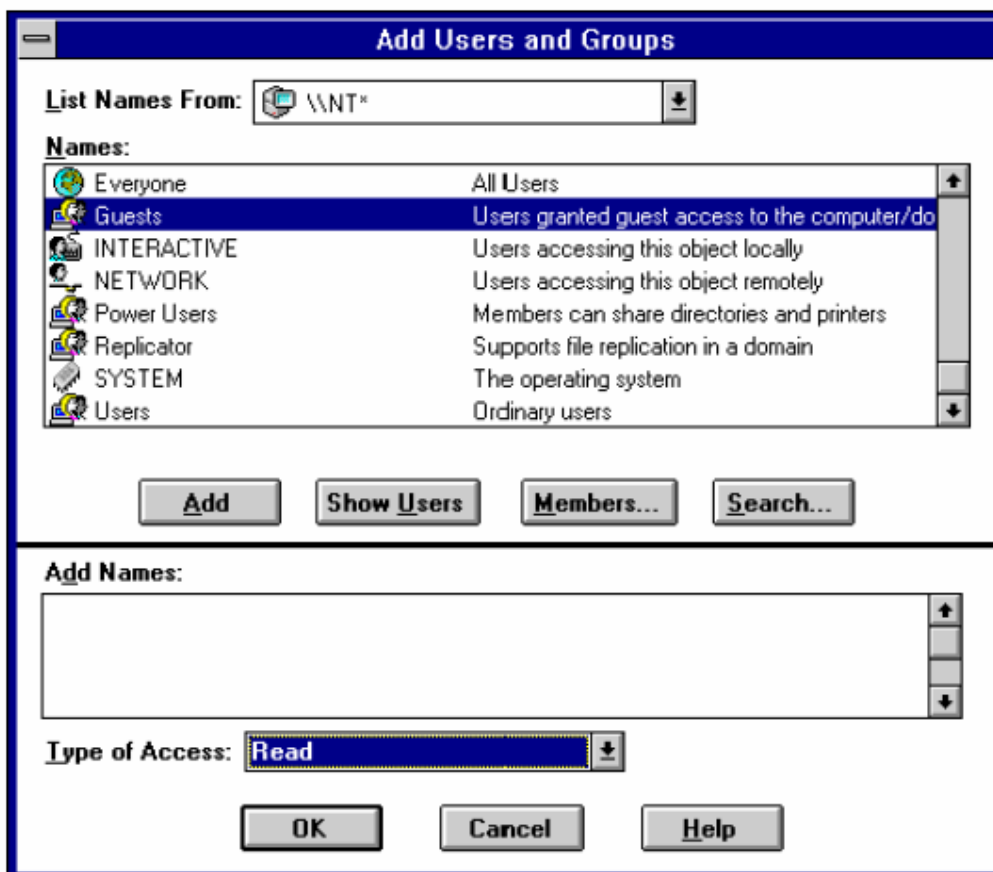
DDEShare dijalog boks će se pojaviti kao što je pokazano na slijedećoj slici



[4] Izabrati Add i Share dugmad i kompletirati dijalog boks kako je pokazano na slijedećoj slici :



[5] Izabrati *Permissions* dugme. Pojaviće se slijedeći dijalog boks:

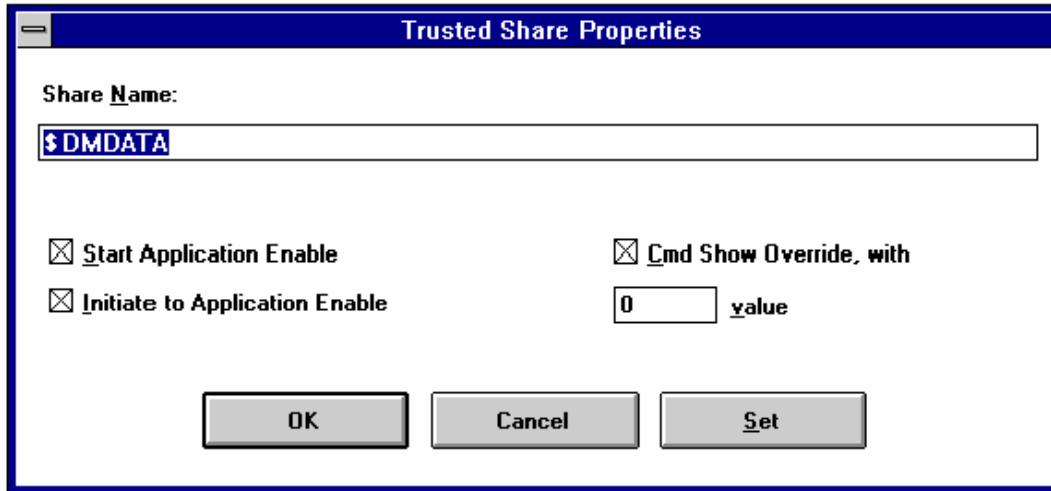


[6] Izabrati users i groups za koje želimo da imaju pristup DDEShare iz list boksa imena.

[7] Izabrati tip pristupa koji želimo da izabrani korisnici imaju u polju "Type of access".

[8] Kliknuti na OK da zatvorimo "Add users and groups" dijalog boks, i kliknuti na OK da zatvorimo DDEShare dijalog boks.

[9] Kliknuti na "Trust Share" dugme na DDEShare dijalog boks i kompletirati dijalog boks kako je pokazano na narednoj slici:



[10] Kliknuti na Set dugme da se setuje Trust i kliknuti na OK da se pohrane promjene.

Da bi se setovao DDEShare za historijske podatke, koristiti gornju proceduru i kompletirati DDEShare Properties dijalog boks, sa slijedećim vrijednostima:

Share Name	SDMHTR
Application Name, old style	DMDDE
Topic name, old style	HTR
Application Name, new style	DMDDE
Topic name, new style	HTR

Potrebno je takodjer kreirati Share name (djeljeno ime) za svaku od DDE aplikacija koje će se izvršavati na serveru i koje će obezbjedjivati FIX software sa podacima. Slijedeći primjer pokazuje DDEShare ime za Excel koje se može konfigurirati:

Share name	\$NETEXCEL
Application name, old style	\$NETEXCEL
Topic Name, old style	[FILE.XLS] SHEET1
Application name, new style	EXCEL.SHEET.5
Topic Name, new style	FILE.XLS

FIX DDE ADRESE

Možemo koristiti FIX podatke u drugim DDE klijent aplikacijama bilo da su rezidentne na lokalnom ili udaljenom računaru. U narednom tekstu opisaćemo koje DDE adrese su potrebne da pristupimo FIX podacima. Ako koristimo report generator opciju, adresiranje je ugrađeno u link makroe.

FIX DDE lokalno adresiranje

Sada ćemo definirati DDE adresnu sintaksu koja se koristi za prenos podataka iz FIX softwarea do drugih aplikacija na istom računaru.

String dio	Real-time	Historijski
Ime aplikacije ili servisa	DMDDE	DMDDE
Topic name	DATA	HTR
ITEM	Node:Tag.Field(A_)	Pen group name Time group name Display node (Y ili N) Display tag (Y ili N) Display date(Y ili N) Display time (Y ili N) Number of samples

Sintaksa za Real-time podatke

Primjer sintakse koja se koristi za prenos real-time podataka je:

= DMDDE|DATA! NODE1.AI1.A_CV

gdje: DMDDE je ime aplikacije ili servisa, DATA je ime topica, a NODE1.AI1.A_CV je ime Itema.

Sintaksa za historijske podatke

Primjer sintakse koja se koristi za prenos historijskih podataka je :

= DMDDE|HTR!Group1.time1,Y.Y.Y.Y.25

Ovaj string vraća ime fajla koji sadrži historijske podatke. Historijske vrijednosti se pohranjuju u (CSV – comma separated values) fajl u FIX brzom stazi (Fast path).

FIX NetDDE adrese

NetDDE adresiranje koje se koristi da se pristupi real-time i historijskim podacima u FIX softwarea je dato u narednoj tabeli:

String dio	Real-time	Historijski
Ime aplikacije ili servisa	'\\Server Name\NDDE\$'	'\\Server Name\NDDE\$'
Topic name	'\$DMDATA.DDE'	'\$DMHTR.DDE'
ITEM	Node:Tag.Field(A_)	Pen group name

		Time group name Display node (Y ili N) Display tag (Y ili N) Display date(Y ili N) Display time (Y ili N) Number of samples
--	--	--

Sintaksa za real-time podatke

Primjer sintakse koja se koristi za prenos podataka u realnom vremenu je :

```
='\ABC\NDDE$'!$DMDATA.DDE!'NODE1.AI1.A_CV'
```

gdje je : ABC ime računara ili radne stanice, NDDE\$ indicira da se koristi NetDDE , DMDATA je DDEShare ime za podatke u realnom vremenu i NODE1.AI1.A_CV je ime litema.

Sintaksa za historijske podatke

Primjer sintakse za prenos historijskih podataka je :

```
='\ABC\NDDE$'!DMHTR.DDE!'Group1.time1.Y.Y.Y.Y.25'
```

Ovaj string će vratiti ime fajla koji sadrži historijske vrijednosti. Historijske vrijednosti se pohranjuju sa CSV formatom u fajl u FIX brznoj stazi.

Ako pokušamo da pristupimo FIX podacima iz Excel spreadsheeta koristeći NetDDE, moramo uključiti ime računara i ključnu riječ NDDE\$ kao i DDEShare ime u znake jednostrukih navoda.

Startanje DDE i Excela

Nakon startanja FIX softwarea , startati DDE server na slijedeći način:

[1] Izabrati Run command dijalog boks.

[2] Unjeti slijedeći tekst u komandnu liniju na Run dijalog boks:

```
C:\FIX32\DMDDE
```

[3] Kliknuti OK. DDE task prozor će se otvoriti da indicira da se DDE izvršava. Sada možemo startati Excel.

Podešenje vremena ažuriranja

DDE automatski ažurira radne listove Excela svake dvije sekunde. Ovo je moguće podesiti iz DDE task prozora.

Da bi se promjenilo vrijeme ažuriranja , pritisnuti rastavnicu (space bar) kada je prozor DDE taska aktivan.

Pojaviće se opcije pokazane u slijedećoj tabeli:

Pritisnuti ..	Da se postigne
+	Poveća vrijeme ažuriranja. Svaki put kada se pritisne + taster, brzina DDE ažuriranja se poveća za 0.2 sec do minimalne brzine ažuriranja od 0.2 sec.
-	Smanjuje brzinu ažuriranja. Svaki put kad kada pritisnemo – taster, brzina DDE ažuriranja se smanjuje za 0.2 sec do maksimalne brzine ažuriranja od 40 sec.
w	Minimiziraj ili maksimiziraj DDE task prozor.
x	Izadji iz DDE programa

Opaska: Opcija ažuriranja je validna samo za tekuću sesiju. Vrijeme ažuriranja se vraća na 2 sekunde kada izadjemo ili se ponovo vratimo u DDE program.

ODBC arhitektura u FIXu

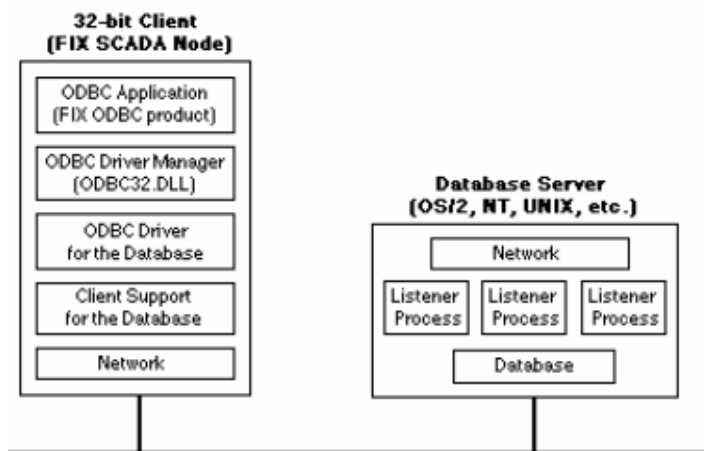
Postoje u osnovi dva tipa ODBC arhitekture: ona koja uključuje višenivovske ODBC drajvere i jedno nivovske ODBC drajvere.

Višenivovski drajveri su češći i tipično se koriste sa daljinskim Serverom baze podataka kao što je Oracle i SQL Server. Višenivovski ODBC drajveri procesiraju ODBC pozive (calls) koje čine aplikacije, ali prenose stvarnu SQL komandu na sistem baze podataka. Jedno nivovski drajveri, kao što je MS Access drajver, često djeluju direktno na fajl ili fajlove baze podataka.

Jedno nivovski ODBC drajveri procesiraju i ODBC pozive ali i stvarne SQL komande. U većini slučajeva, konfiguracija koja koristi jedno nivovski drajver može biti rezidentna samo na jednom računaru.

Tipična ODBC arhitektura

Naredna slika ilustruje tipičnu arhitekturu Servera relacione baze podataka i ODBC drajver.



Tipična ODBC arhitektura

Definicije ODBC termina

Client support – sloj podrške klijentu baze podataka obično sadrži jedan ili više fajlova dinamički linkovanih biblioteka (DLL) , u sprezi sa konfiguracionim fajlovima. Podrška klijentu baze podataka je najčešće obezbjeđena od isporučioaca baze. ODBC drajverski sloj (layer) , komunicira sa slojem podrške klijentu , kao što je to pokazano na prethodnoj slici. Mnogi isporučioaci baza podataka nude i podršku za mrežu (za mrežne protokole kao TCP/IP, Microsoft NetBEUI, Novel Netware , itd). Specifični fajlovi koji su potrebni za komunikaciju sa jednom od ovih mreža su obično dio sloja za podršku klijenta.

Database layer – sloj baze je sastavljen od pokretačkog jezgra baze podataka (database engine) i fajla ili fajlova gdje su podatci stvarno i pohranjeni.

Listener processes – proces slušaoca (listenera) veže mrežni protokol sa database engine. Ovo je ustvari serverski dio servera baze podataka. Prethodna slika je pokazala tri odvojena procesa slušaoca da demonstrira jednu moguću konfiguraciju koja opslužuje tri odvojena klijenta. Ovaj sloj značajno varira u zavisnosti od isporučioaca baze i Operativnog sistema.

Network layer – mrežni sloj je potpuno odvojen od ODBC slojeva i specifičan je za Operativni sistem. Obično je dio Operativnog sistema ili je isporučen od strane mrežnog provajdera. Obadva, i Klijent kompjuter kao i računar Servera baze podataka sadrže ovaj sloj.

ODBC Administrator program- program koji se koristi da se konfiguriraju ODBC izvori podataka. Zavisno od Operativnog sistema, obično se može naći u Control panelu, ali može biti instaliran i kao neovisni exe fajl (ODBCAD32.EXE).

ODBC Application - aplikacija koja generira ODBC pozive (calls). Pošto aplikacija komunicira sa sa ODBC slojem, ona je nezavisna od tipa baze podataka. To znači da ODBC aplikacija može biti napisana i, plagovanjem različitih ODBC drajvera, može pristupiti bilo kojoj bazi. FIX ODBC proizvod je ODBC aplikacija.

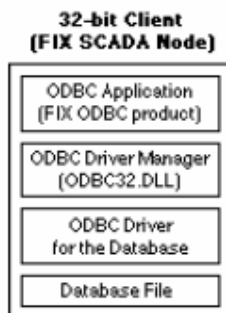
ODBC driver- ODBC drajver prevodi ODBC poziv koji izdaje aplikacija u specifični poziv (e) , za određeni tip baze. ODBC drajveri su na raspolaganju iz različitih izvora. Neke softwareske kompanije su specijalizovane u pisanju database drajvera. One nude pakete koji nude tuce ODBC drajvera za različite baze. ODBC drajveri su često raspoloživi i od proizvođača softwarea baze podataka. ODBC drajver komunicira sa slojem podrške klijentu.

ODBC Driver Manager- modul napisan od MS koji se isporučuje sa većinom ODBC drajvera. Djeluje kao sloj između aplikacije i bilo kojeg ODBC drajvera. Ustvari, on puni drajver kada aplikacija zahtjeva konekciju.

Opaska: Ne postoji ODBC software na računaru database servera. ODBC drajver na klijentu prevodi ODBC pozive u izvorne (native) database pozive koje sloj podrške klijenta (client support layer) može razumjeti. Zbog toga, u trenutku kada zahtjev ka bazi napusti klijent mašinu, on je već upotpunosti transformiran u izvorni poziv (native call) za tu bazu. Slušalac (listener) i mašina (engine) na serveru ne znaju da li je zahtjev došao iz ODBC aplikacije ili izvorne database aplikacije.

Microsoft Access ODBC arhitektura

ODBC arhitektura MS Accessa je jednostavnija nego tipična serverska arhitektura. Naredna slika ilustrira ODBC arhitekturu Accessa kao i druge jednonivovske (single tier) ODBC drajvere.



Jedno nivovska ODBC arhitektura

Primjetimo da nema računara database servera, sloja podrške klijentu, ili mreže na ovoj slici. Access ODBC drajver djeluje direktno na fajl baze podataka. U ovoj jednostavnoj konfiguraciji database fajl je lociran na istom računaru kao i aplikacija. Koristeći MS mrežni software, fajl baze podataka može biti lociran na drugom računaru baš kao i bilo koji drugi djeljeni fajl. Na ovaj način, Accessova baza podataka se može djeliti između više kompjutera i aplikacija.

Konfiguriranje ODBC izvora podataka

Svaki specifični drajver za svaku bazu zahtjeva različitu informaciju da bi se uspostavila konekcija. Jedna stvar koja im je zajednička je ime izvora podataka. Svaki izvor podataka koji konfiguriramo za spajanje na specifičnu bazu je identificiran sa imenom izvora podataka. Kada jedna aplikacija treba da se poveže na bazu, aplikacija pošalje ime izvora podataka, koje se mapira na informaciju koja je bila konfigurirana za tu bazu.

Sistemske i korisničke izvori podataka

Bilo koji korisnik može kreirati sistemski izvor podataka, i taj izvor podataka će postati raspoloživ bilo kojem korisniku koji se logira u WinNT operativni sistem. Korisnički izvor podataka sa druge strane je samo raspoloživ korisniku koji ga je kreirao.

ODBC Administrator nam omogućava da kreiramo sistemski izvor podataka jednostavnim klikanjem na DSN taster i dodajući ime izvora podatka. Sistemski izvor podataka će se pojaviti u sistemskoj DSN listi. Međutim, kada pozovemo listu raspoloživih izvora podataka u FIX SCU, i sistemski i korisnički izvori podataka će biti izlistani.

Konfiguriranje za Microsoft Access

Konfiguriranje ODBC izvora podataka

Za naredni primjer , predpostavimo da imamo Access bazu koja se zove PROD_REC u Access direktoriju. Neka je staza za ovaj file C:\ACCESS\PROD_RCP.MDB.

Da bi se konfigurirao izvor ODBC podataka uraditi:

[1] Startati ODBC Administrator program (iz Control panel)

[2] Kliknuti na Add dugme i selektirati Access drajver

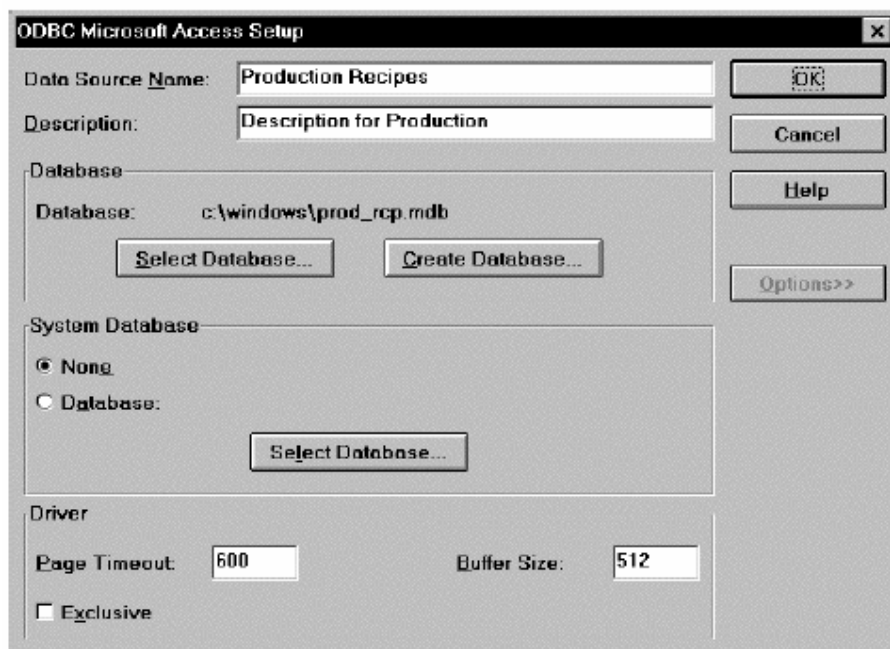
[3] Unjeti ime izvora podataka , kao što je naprimjer *Production Recipes* , u polje "Data Source Name". Opciono, možemo unjeti i duži opis ovog imena u polje " Data Source Description".

[4] Kliknuti na "Select database" taster i selektirati " PROD_RCP" fajl.

[5] Kliknuti na Database radio dugme , unutar System database grupe.

[6] Kliknuti na Select (ili System) database dugme i selektirati sistemsku bazu (obično C:\ACCESS\SYSTEM.MDA).

[7] Provjeriti da su "Exclusive" i "Read only" nečekiirani u Option grupi. Naredna slika ovo pokazuje :



ODBC MS Access setup diajlog boks

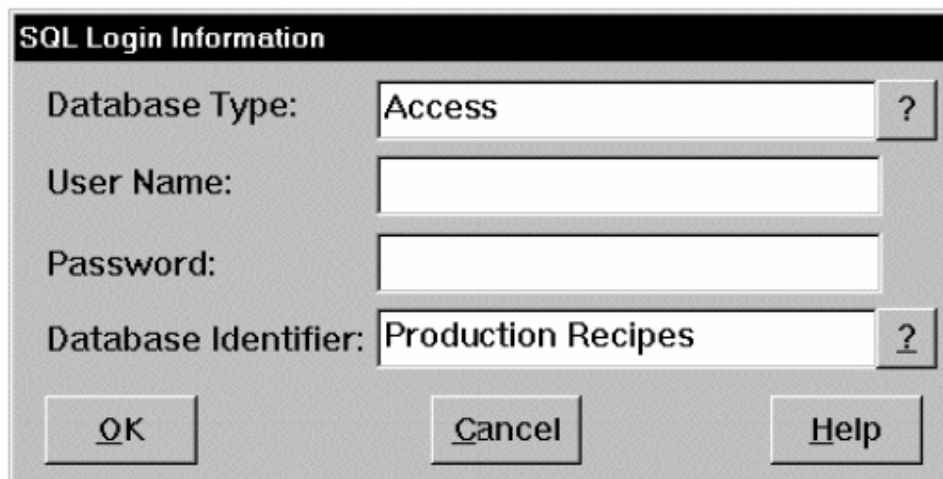
[8]Kliknuti na OK. Time je uspješno završeno konfiguriranje ODBC izvora podataka za PROD_RCP bazu podataka

Konfiguriranje SCU

Da bi se FIX spojio na Access bazu podataka, unos se mora dodati u SQL konekcioni listu u SCU.

- [1] Startati SCU i selektirati SQL iz konfiguracionog menija.
Pojaviće se SQL račun diajlog boks. Lista konfiguriranih računa, *Database identifiers*, je prikazana. (može biti i prazna).
- [2] Kliknuti na "Add" dugme da se doda Access izvor podataka.
- [3] Kliknuti na ? i izabrati Access tip baze,
- [4] Unjeti *Production recipes* u *Database identifier* polje ili kliknuti na ? taster za listu konfiguriranih identifičera baze.
- [5] Ako je Access security onemogućena , ostaviti polja "user name" i "Password" nepopunjena.

Naredna slika pokazuje kako ova polja trebaju izgledati



The image shows a dialog box titled "SQL Login Information". It contains the following fields and buttons:

- Database Type:** A text box containing "Access" and a "?" button to the right.
- User Name:** An empty text box.
- Password:** An empty text box.
- Database Identifier:** A text box containing "Production Recipes" and a "?" button to the right.
- At the bottom, there are three buttons: "OK", "Cancel", and "Help".

Dijalog boks za SQL login informaciju.

- [6] Kliknuti na OK. Time je unošenje u SQL dijalog boks SCU završeno.
- [7] Pohraniti i izaći iz SCU.

Krerenje tabela biblioteke i grešaka (library and error tables)

Konfiguriranje Access baze podataka

Ako koristimo FIX ODBC opciju da komuniciramo sa FIX bazom, moramo kreirati tabele u Accessu da pohranimo SQL komande i greške. Da bi kreirali tabelu u Accessu treba uraditi slijedeće:

[1] Startati Access

[2] Izabrati "Open" iz File menija i selektirati bazu koju želimo otvoriti (naprimjer C:\ACCESS\PROD_RCP.MDB).

[3] Kliknuti na ikonu za tabelu

[4] Kliknuti na New da se doda nova tabela.

Na spreadsheetu koji će biti pokazan unjeti slijedeće podatke:

MS Access **SQLLIB** tabela

Field name	Datatype	Notes
sqlname	Text, size 8	Izabrati Yes za indeksirani atribut. Takodjer kliknuti na Key ikonu na toolbaru da ovo bude primarni ključ (primary key) za tabelu
sqlcmd	Text, size 100- 255	Ako je bilo koja od SQL komandi duža od 255 karaktera , napraviti ovo polje kao Memo polje. Onda je moguće imati dužinu do 64.000 karaktera.

MS Access **SQLERR** tabela

Field name	Datatype
td	Date-Time
node	Text, size 8
tag	Text, size 10-30
sqlname	Text, size 8
Fix_err	Text, size 100
Sql_err	Text, size 250
Prog_err	Text, size 100

Podržavani tipovi podataka u kolonama

Moguće je prenositi podatke u i iz FIX baze koristeći FIX ODBC. Naredna tabela prikazuje FIX i Access tipove podataka koji su podržani. Takodjer indicira i smjer transfera.

Smjer transfera (I/O) je određen unosom u polje smjera u SQL bloku podataka kada definiramo FIX bazu. **I** indicira smjer od IN (za izabrane komande) i podržan je za Access i FIX field tipove. **O** indicira smjer OUT (za markere parametara) i takodjer je podržan.

Access field type	DATE keyword	TIME keyword	TMDT keyword	F_CV	A_
Date-time	O	O	O		I
Broj (single, double, integer, long integer, byte)				I, O	I
Counter (brojač)				I, O	I
Tekst					I, O

FIX obezbjedjuje tekst file (SQLERR.TXT) , da pomogne traženju komunikacionih grešaka za relacionu bazu podataka. Ovaj file sadrži brojeve komunikacionih grešaka za Oracle baze.

Pregled ODBC SQL softwarea u FIX-u

FIX real-time ODBC SQL interfejsni software dozvoljava korisnicima da:

- skupljaju i pišu real-time procesne podatke u jednu ili više relacionih baza podataka
- Čitaju podatke pohranjene u relacionu bazu i pišu ih natrag u FIX procesnu bazu podataka
- Pobrišu podatke u tabelama relacione baze.
- Pohranjuju (back up) podatke i SQL komande na disk ako mreža otkáže i ne može da održava konekciju sa serverom ili ako Server otkáže.
- Izvrši automatski spašene (backed up) SQL komande kada se ponovo uspostavi konekcija sa Serverom.

Komponente FIX ODBC SQL softwarea

FIX ODBC proizvod obezbjedjuje komunikaciju izmedju relacionih baza i FIX baze podataka. FIX baza može biti konfigurirana da komunicira na bazi događaja (event), vremena ili kombinacije jednog i drugog.

FIX ODBC proizvod se sastoji od:

- SQL taska
- SQL trigerskog bloka baze (SQT)
- SQL bloka baze za podatke (SQD)

SQL zadatak izvršava slijedeće funkcije:

- Izvršava SQT blokove koji trigeruju
- Uzima (retrieves) procesne podatke iz SQD blokova i umeće podatke u relacionu bazu.
- Pohranjuje podatke (back up) u slučaju otkaza mreže (back up se nastavlja sve dok primarna ili sekundarna staza nije puna).

- Automatski restaurira podatke u relaciju bazu kada se mrežna komunikacija ponovno uspostavi.

SQL triggerski blok definira:

- Koje SQL komande u SQL bibliotečkoj tabeli se koriste da manipuliraju sa podacima
- Da li SQL komande se backupiraju u slučaju da aplikacija izgubi konekciju sa serverom
- Vrijeme ili odgadaj koji trigeruje transfer podataka.

SQL blok podataka definira:

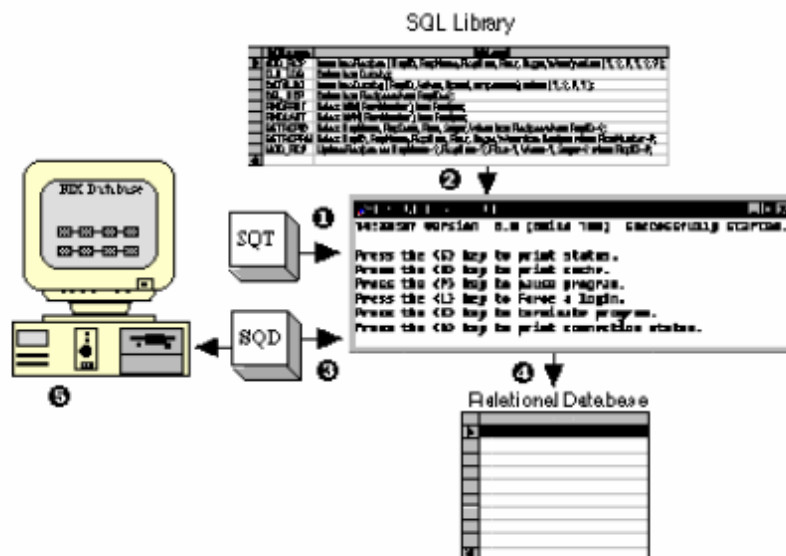
- Podatke koji će se skupljati i transferisati
- Smjer prenosa podataka

Obadva bloka baze podataka komuniciraju sa SQL taskom , WSQLODC.EXE. Ovaj task se izvršava na SCADA čvoru i upravlja komunikacijom sa ODBC drajverom.

Ako je konekcija sa relacionom bazom izgubljena SQL komande i podatci se mogu backupirati. Kada se konekcija ponovno uspostavi, SQL komande i podatci se izvršavaju u redoslijedu u kojem su bili backupirani.

Razumjevanje komunikacionog procesa

Naredna slika ilustrira proces prikupljanja i unošenja procesnih podataka u relaciju bazu. Tabela ilustrira korake uključene u ovaj proces:



Prikaz kako FIX ODBC prikuplja i unosi podatke

Tabela : Transfer podataka iz FIX baze u relacionu bazu

Step	Akcija
1	SQL trigerski blok (SQT) trigeruje. SQL task čita SQL ime komande
2	SQL task uzima pridruženu komandu iz SQL bibliotečke tabele
3	SQL task čita imena tagova specificirana u SQL komandi iz SQL bloka podataka (SQD) i čita vrijednosti pridružene sa ovim tagovima iz FIX baze
4	SQL task izvršava SQL komandu i unosi podatke u izabrane podatke iz relacione baze.
5	Ako je SQL komanda SELECT komanda, uzeti podatci se upisuju u FIX tagove definirane u SQD bloku.

SQL task WSQLODC.EXE , je odgovoran za:

- izvršenje SQT blokova koji trigeruju
- Dobijanje procesnih podataka iz SQD blokova i unošenje podataka u relacionu bazu.
- Selekciju podataka iz relacione baze i upisivanje podataka natrag u FIX bazu podataka.
- Backup podataka u slučaju otkaza mreže.
- Automatske restauracije podataka u relacionu bazu kada mrežna lomunikacija ponovno uspostavljena.

Modifikacija startup opcija

Kada je SQL podrška omogućena u SCU, SQL task se automatski dodaje u SCU (System configuration utility). Komandni parametri koji su mogući za ovaj task su pokazani u slijedećoj tabeli:

Tabela : SQL task parametri na komandnoj liniji

Parameter	Opis	Validni unosi
/LDn	Definira kako često task pokušava da se logira na server nakon što je izgubio konekciju. Ovaj parametar je opcioni. Sistem defaultira na 300 sec ako ovaj parametar nije definiran.	60 sec (min). Nema maksimalne granice. Ipak, najveći broj aplikacija ne trebaju više od 1 sata (3600 sec)
/CAN	Pohranjuje specificiran broj SQL komandi u memoriju (cache) da poveća performansu sistema. Parametar je opcioni. Poboľšanje performanse varira zavisno od relacione baze.	Specificira korisnik. Nema maksimalne granice. Ipak, najveći broj aplikacija ne trebaju više od 10 do 100.
/CLn	Povećava maksimalno dozvoljenu dužinu SQL komande od default od 255.Naprimjer unošenje /CL1000 dozvoljava dužinu komande od 1000 karaktera	Korisnik specificira

/LM	Uključuje ODBC drajver informaciju zajedno sa porukama grešaka relacione baze. Kada se koristi ovaj parametar, treba voditi računa da drajverska informacija može biti dosta dugačka i možda da ne bude dovoljno prostora u stvarnoj poruci greške u omogućenoj destinaciji alarma.	Ne zahtjeva se
/Dn	Definira startup kašnjenje prije pokušaja logiranja u SQL bazu podataka. Treba koristiti ovaj parametar ako se uoče problemi kod konekcije sa bazom za vrijeme starta.	0 do 65535 sec.

Primjer: Ako želimo da definiramo login kašnjenje od 5 minuta (300 sec), maksimalnu SQL komandnu dužinu od 1000 i komandni cache od 10, treba unjeti slijedeće parametre u SQL task polja parametara:

/LD300 /CL1000 /CA10

Korištenje kaširanih (caching) komandi

Zavisno od relacione baze koja se koristi, može se poboljšati komunikaciona performansa dodavajući parametar kaširanja komandi, /CAN , u SQL task.

Kaširanje komandi omogućava svakoj komandi koja je bila pozvana i izvršena da se pohrani u radnoj memoriji. Performansa će se poboljšati kada slijedeći put komanda bude trebala biti izvršena. Pošto je komanda u memoriji, nema potrebe da se ponovo traži sa diska.

Komanda se može izvršiti bez stvarnog pristupa bazi koja pohranjuje komandu. Time, konekcija sa tabelom komandi u bazi se može prekinuti a da task još uvijek može koristiti komandu.

Koristeći kaširanje komandi , moguće je čitati komandu iz baze podataka, izvršiti je u okviru druge baze specificirane u SQL triggerskom bloku, i pohraniti njen spojnicu (handle).

Konfiguriranje SQL database blokova

FIX ODBC proizvod se sastoji od dva database bloka pored SQL taska. Ova dva bloka su:

- SQL triggerski blok (SQT)
- SQL data blok (SQD)

Kada se ulanče ova dva bloka definiraju:

- podatke koji se skupljaju i transferišu
- smjer prenosa
- SQL komande koje manipuliraju sa podacima

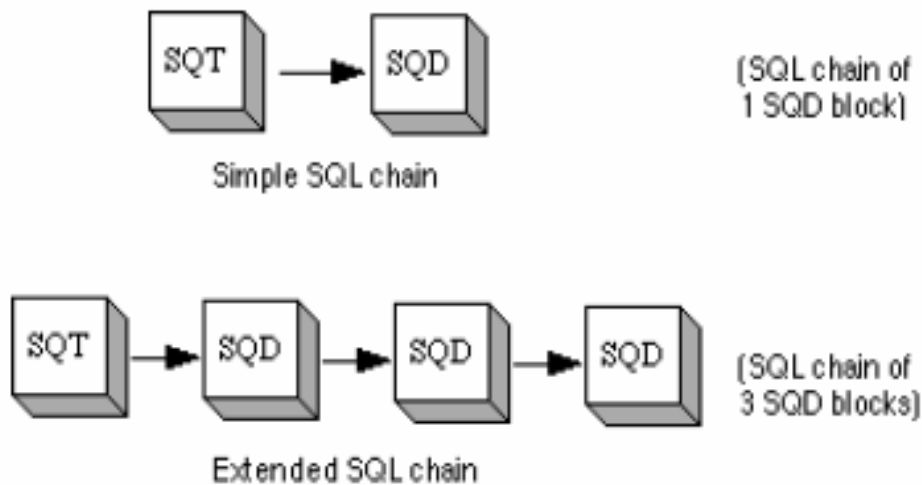
SQL lanci

Svaki SQL lanac baze koji korisnik kreira počinje sa SQT blokom. SQT blok može se koristiti samostalno, ili mu se može odati jedan ili više SQD blokova u lancu.

SQD blok može identificirati 20 tagova baze podataka. Kada se koriste SQD blokovi, SQL komanda specificirana u SQT bloku bilo vadi procesne podatke iz ovih tačaka procesnih podataka ili upisuje podatke iz relacione baze u njih.

SQT blok ulančen sa SQD blokom predstavlja jednostavan SQL lanac. Treba imati u vidu kada se dizajniraju SQL lanci da SQD blokovi se mogu ulančavati jedan na drugoga, i na taj način formirati proširene lance koji mogu identificirati veći broj tačaka u bazi.

Specifična aplikacija određuje koliko tačaka baze se zahtjeva od SQL lanca. Naredna slika ilustrira dva tipa SQL lanca:



Tipovi SQL lanca

Samostalni SQT blokovi

Može se definirati SQT blok kao samostalni blok. Kada to učinimo, SQT blok je dizajniran da izvršava SQL komandu koja nema definirane nikakve markere parametara. Parametri omogućavaju da ažuriramo relacionu bazu koristeći vrijednosti FIX baze. Koristeći samostalni SQT blok, možemo izvršiti slijedeću SQL komandu:

" delete from DATATABLE;"

Ova komanda briše sve unose u tabelu relacione baze, DATATABLE.

SQL triggerski blok

Namjena

SQL triggerski blok (SQT) obezbjedjuje sredstvo triggerovanja izvršenja SQL komande. Blok definira kako se FIX interfejsira sa relacionom bazom podataka.

SQT blok radi slijedeće:

- starta prenos procesnih podatka u relacionu bazu ili u blokove u bazu podatka SCADA čvora.

- Potvrđuje prenos podataka
- Prima ili obezbeđuje operatorsku informaciju putem linkova u View. Vrijeme, datum, triggerski parametri događaja , i SQL komande koje su na raspolaganju za ovaj blok mogu biti redefinirane sa Operatorskog displeja.
- Može biti ručno triggerovano sa operatorskog displeja.

Karakteristike SQT bloka

SQL triggerski blok je primarni blok koji:

- može postojati kao samostalni blok , ali je obično iza njega barem jedan SQL blok podataka
- Određuje koja SQL komanda u SQL bibliotetskoj tabeli će se koristiti u obradi procesnih podataka
- Definiira mod selekcije
- Dodjeljuje mu se 312 bajta memorije.

Dijalog boks SQL triggerskog bloka je pokazan na narednim slikama. Prvo je pokazana prva strana SQL triggerskog bloka a zatim slijedeće:

Dijalog boks SQL triggerskog bloka (strana 1)

Druga strana SQL trigerskog bloka data je na narednoj slici:

SQL Trigger Block (Page 2)

Tag Name: SQT1

Alarms

Enable Alarm

Alarm Areas: ALL

Alarm Priority

Low Medium High

Security Areas

1: NONE

2: NONE

3: NONE

OK Cancel Help

Dijalog boks SQL trigerskog bloka (strana 2)

Kliknuti na "Alarms and Security" taster da bi se pristupilo sekundarnom dijalog boksu od "Alarms and Security", koji je prikazan na narednoj slici:

SQL Trigger Block (Page 2)

Tag Name: SQT1

SQL Name: INS_1

Database ID: Access_DMACS

Command Type

SQL cmd Procedure

Select parameters

Single row Multiple rows Array mode

Enable backup

Columns: 0

Rows: 0

OK Cancel Help

Dijalog boks SQL parametara

Kliknuti na taster SQL parametara da bi se pristupilo sekundarnom dijalog boksu SQL parametara.

Razumjevanje Identifikatora baze (database ID)

Polje ID-a SQT baze definira relacionu bazu koja će se koristiti. ID-ovi baza su definirani u SQL konfiguracionom dijalog boks u SCU.

Opaska: ODBC izvor podataka definiran za vrijeme setupa relacione baze je ekvivalentan ID baze u FIXu.

Definiranje ID-a baze u SCU je opciono. Ako ne definiramo ID baze podataka u SCU za SQL komandnu tabelu, ID baze specificiran u SQT bloku se koristi i za ekstrakciju (retrieval) komande kao i za njeno izvršenje. Ako je ID baze bio specificiran u SCU za komandnu tabelu, komanda se ekstrahuje iz te baze i šalje na i izvršava se u bazi specificiranoj u SQT bloku.

SQT blok podataka

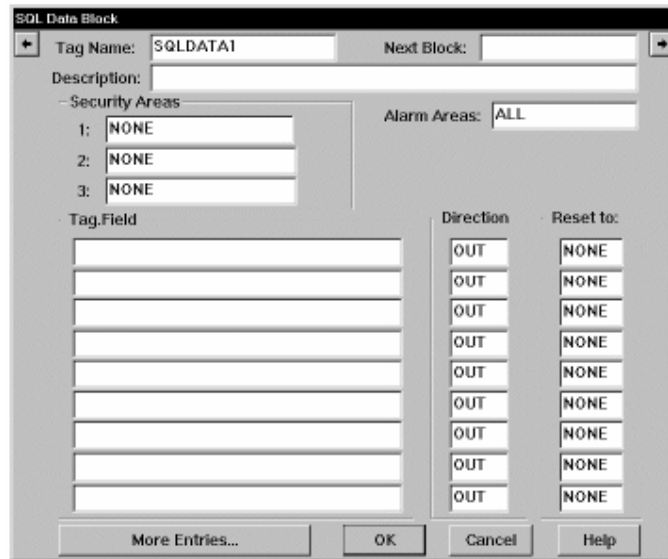
SQL blok podataka (SQD) identificira gdje treba skupljati podatke u FIX bazu za prenos ka relacionoj bazi, ili gdje upisivati vrijednosti natrag u FIX bazu.

Karakteristike

SQL blok podataka (SQD) je sekundarni blok koji:

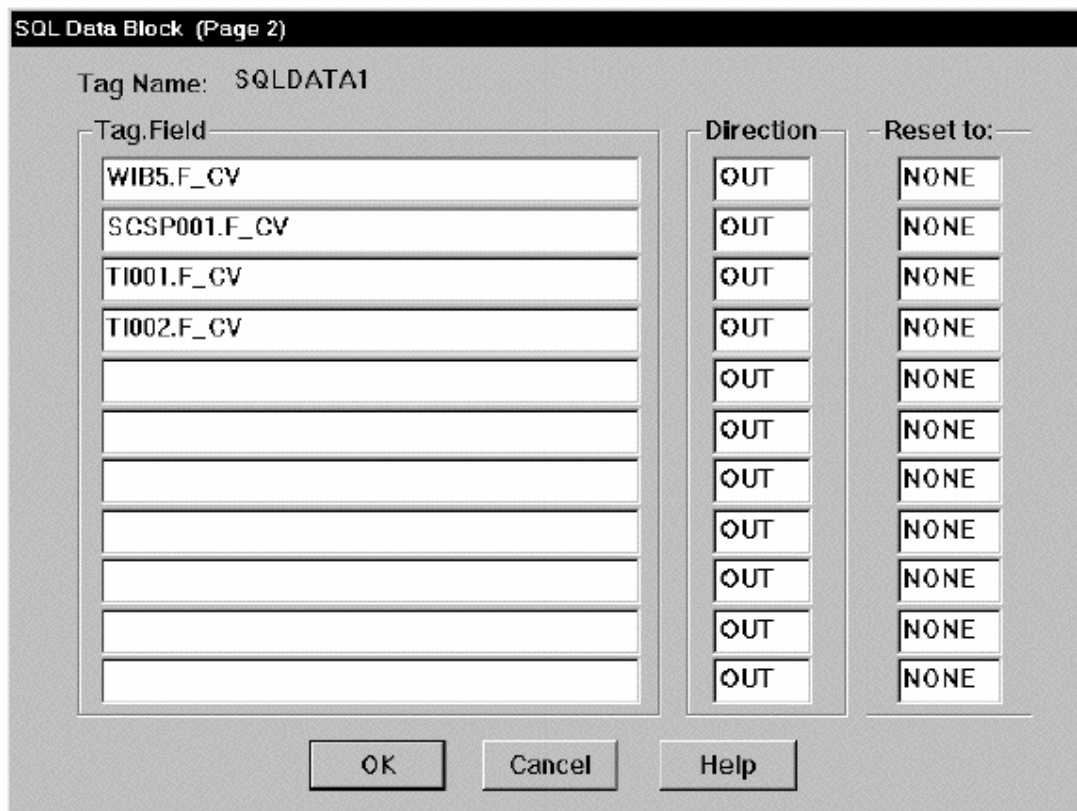
- Može se pojaviti samo poslije SQL trigerskog bloka (SQT).
- Može identificirati do 20 tagova i kombinacija imena polja.
- Može se ulančiti sa drugim SQL blokovima podataka (SQD).
- Odredjuje smjer prenosa podataka.
- Dodjeljeno mu je 738 bajta u memoriji

Dijalog boks SQL bloka podataka je raspoređen na dvije strane koje su prikazane na naredne dvije slike:



Dijalog boks SQL bloka podataka (Strana 1)

Naredna slika pokazuje drugu stranu dijalog boksa:



Dijalog boks SQL bloka podataka (strana 2)

Tipični primjeri primjene

SQL blok podataka :

- Identificira imena tag/field kombinacija (tačaka podataka) da prikupi podatke iz ili upisuje na njih
- Postavlja smjerove prenosa podataka (čita vrijednosti iz ili upisuje vrijednosti u relacionu bazu)
- može resetovati tačke podataka kada se izvršavaju blokovi podataka
- spaja se sa drugim SQL blokovima podataka da formira prošireni lanac.

Tipične aplikacije

Ako SQD blok izvršava UPDATES i INSERTS za tačke podataka koje imaju smjer OUT, tekuća vrijednosti u polju je prvo upisana u relacionu bazu a zatim se postavlja na nulu ili prazno za tekstualne stringove.

Ako SQD blok izvršava SELECT za tačke podataka koje imaju smjer IN, polje se čisti a zatim prihvata dolazeću vrijednost.

Opaske o skupljanju podataka

Kada se koriste FIX ODBC proizvodi, slijedeće situacije mogu uticati na proces prikupljanja:

- skupljanje podataka od bloka koji je van skaniranja
- nadzor SQT bloka koji je u ručnom modu

Blokovi van skaniranja

SQD blok identificira gdje treba skupljati podatke u FIX bazi za prenos ka relacionoj bazi, ili gdje upisivati vrijednosti natrag u FIX bazu . SQD blok, međjutim, ne nadzire status ili mod bloka.

Ako je blok van skaniranja on pokazuje ???????? u linku kao svoju tekuću vrijednost. Kada se SQT blok trigeruje, SQL task će zamjeniti tekuću vrijednost bloka sa vrijednošću nula (0). Ako ciljna kolona ne može prihvatiti vrijednost nule, umetanje vrijednosti ili njeno ažuriranje ne uspjeva i SQT blok će generirati alarm.

Blokovi u ručnom modu

SQT blokovi koji su u ručnom modu ne mogu trigerovati sve dok se ne prebace u Automatski mod. Ipak, Operator može ručno trigerovati SQT blok putem linka koji koristi parametar A_TRIP polja ili preko Graditelja baze (DBB).

Ako imamo SQT blok koji nadzire CLOSE i OPEN stanje digitalnog bloka, kada se digitalni blok prebaci u OPEN stanje, SQT blok će trigerovati. Ako je SQT blok

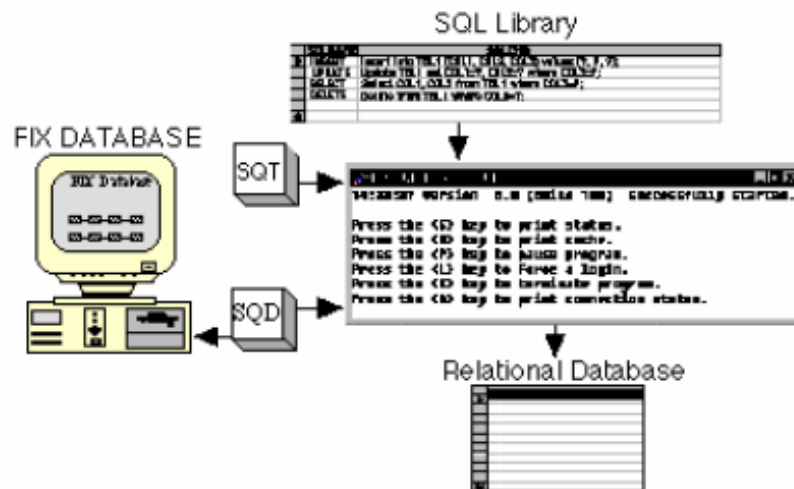
postavljen u ručni mod prije nego se stanje digitalnog bloka promjeni na OPEN, SQT blok neće trigerovati dok se ne prebaci u Automatski mod.

Korištenje SQL komandi

Sve SQL komande su podržavane od strane FIX ODBC proizvoda. Ipak, mi ćemo se koncentrirati na najčešće komande i to:

- INSERT
- UPDATE
- SELECT
- DELETE

SQL komande se pohranjuju u SQL tabele biblioteka. SQL trigerski blok definira koje SQL komande treba koristiti u ovoj tabeli, kao što je pokazano na narednoj slici:

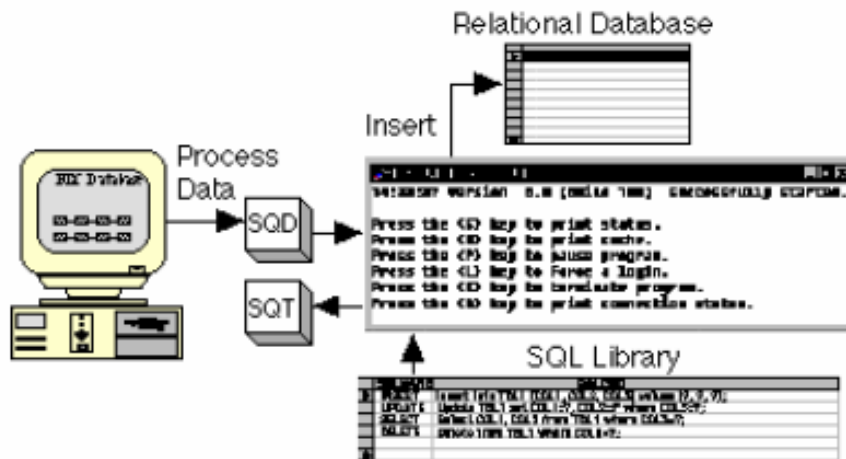


Izvršenje SQL komandi

U nastavku ćemo opisati kako ove komande rade sa FIX ODBC-om.

INSERT komanda

INSERT komanda dodaje podatke iz FIX tagova u novi red relacione baze. Naredna slika pokazuje operaciju koja se izvršava kada sistem izvršava umetanje (insert):



Primjer INSERT komande

Opaska : INSERT iskaz može imati samo jedan record pridružen sa komandom.

SQT1 – SQL trigerski blok definira SQL ime i komandu koja će se koristiti. U ovom primjeru, slijedeća INSERT komanda u SQLLIB tabeli izvršava se kada SQT blok trigeruje.

INSERT into tbl1 (col1, col2, col3) values (?, ? , ?) ;

SQD1 - SQL blok podataka referencira tri taga i kombinaciju imena polja u bazi, i postavlja smjer za prenos podataka ka OUT.

NAME (TAG, FIELD) DIRECTION

- T01 - . A11.A_DESC OUT
- T02 - . AO1.F_CV OUT
- T03 - . DO1.A_ADI OUT

TBL1 - je tabela referencirana od strane INSERT komande koja se koristi u SQT bloku.

COL1	COL2	COL3
Pumpa korak 101	99.7	Hello
Temperaturna zona 2 (C)	-2.1	Bye Bye
Ručno prebacivanje	.004	what

Objašnjenje

Kada software izvrši ovu INSERT komandu , kreiraće novi red u TBL1 koji sadrži vrijednosti FIX tagova i imena polja izlistanih u SQD1 bloku. Rezultirajuća tabela je pokazana kako slijedi:

COL1	COL2	COL3
Pumpa korak 101	99.7	Hello
Temperaturna zona 2 (C)	-2.1	Bye Bye
Ručno prebacivanje	.004	what
AI1 deskriptor	21.04	ABDFG

AI1 Descriptor – dolazi iz AI1 bloka iz polja A_DESC.

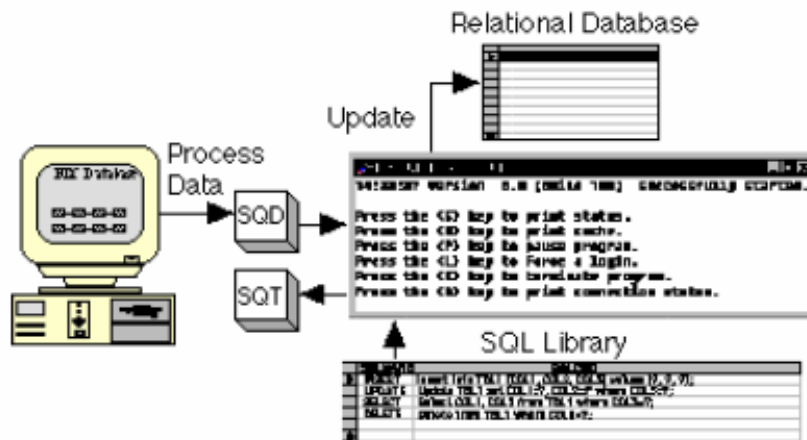
21.04 - dolazi iz AO1 bloka koristeći F_CV polje

ABDFG – dolazi iz bloka koji koristi A_ADI polje.

Opaska: Ako SQL task ne može čitati vrijednost iz bloka (na primjer, pokušava da čita tekuću vrijednost iz analognog ulaznog bloka, dok je blok van skeniranja), SQL task zamjenjuje nul vrijednost umjesto vrijednosti bloka. Ako ciljna kolona ne prihvata nul vrijednosti , novi red se umeće i SQL task generira grešku. SQT blok takodjer generira alarm.

UPDATE komanda

UPDATE komanda mjenja vrijednosti u relacionoj bazi podataka da odrazi tekuće vrijednosti FIX tagova. Naredna slika pokazuje operaciju koja se izvršava kada sistem izvršava ovo ažuriranje.



Primjer UPDATE komande

SQT1 – SQL trigerski blok koristi slijedeću UPDATE komandu definiranu u SQLLIB tabeli:

Update TBL1 set COL1=?, COL2=?, where COL3=? ;

SQD1 – SQL blok podataka referencira kombinaciju tri taga i imena polja u bazi, i setuje smjer prenosa podataka ka OUT.

NAME (TAG.FIELD) DIRECTION

T01-. AI1.A_DESC OUT
T02-. AO1.F_CV OUT
T03 -. DO1.A_ADI OUT

TBL1 – je tabela referencirana od strane UPDATE komande koja je korištena u SQT bloku.

COL1	COL2	COL3
Pumpa korak 101	99.7	Hello
Temperaturna zona 2 (C)	-2.1	Bye Bye
Ručno prebacivanje	.004	what
AI1 deskriptor	21.04	ABDFG

Objašnjenje

Kada software izvršava ovu komandu, on gleda u svaku vrijednost u COL3 tražeći vrijednost koja se podudara s A_ADI poljem u DO1 bloku. Pošto je posljednji red u tabeli taj koji se podudara, sistem ažurira taj red. COL1 i COL2 primaju nove vrijednosti iz AI1 deskriptora i AO1 tekuće vrijednosti respektivno. Rezultirajuća tabela je dakle:

COL1	COL2	COL3
Pumpa korak 101	99.7	Hello
Temperaturna zona 2 (C)	-2.1	Bye Bye
Ručno prebacivanje	.004	what
New deskriptor	-23.09	ABDFG

New Descriptor - dolazi iz A_DESC polja AI1 bloka.

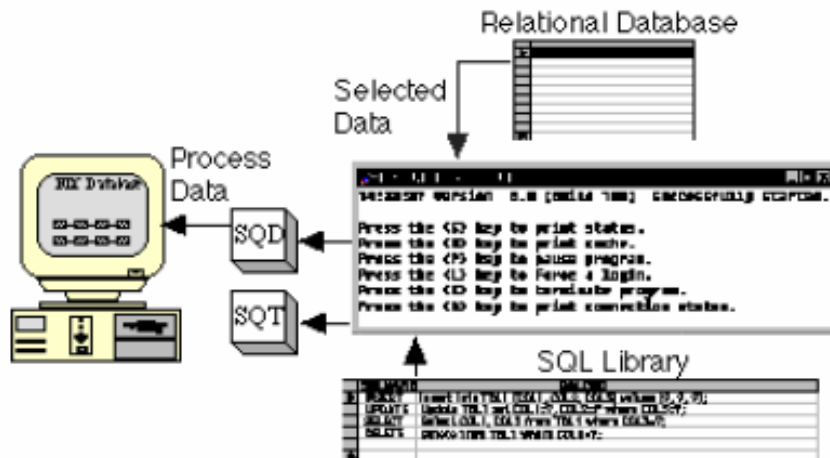
-23.09 - dolazi iz F_CV polja AO1 bloka

ABDFG – dolazi iz DO1 bloka koristeći A_ADI polje

Opaska : FIX ne podržava ažuriranja u koloni datuma (dates). Ako SQL task ne može čitati vrijednost iz bloka , on zamjenjuje nul vrijednost umjesto vrijednosti bloka. Ako ciljna kolona ne prihvata nul vrijednosti, redovi se ne ažuriraju i SQL task generira grešku. SQT blok generira alarm.

SELECT komanda

SELECT komanda vadi (retrieves) podatke iz relacione baze na bazi kriterija selekcije. Naredna slika pokazuje operaciju koja se izvršava kada sistem selektira podatke:



Primjer SELECT komande

SQT1 – u ovom primjeru, SQL trigerski blok koristi slijedeću SELECT komandu definiranu u SQLLIB tabeli:

Select COL1, from TBL1 where COL3= ? ;

SQD1 - u ovom primjeru , SQL blok podataka referencira kombinaciju imena tri taga i polja u bazi , i setuje smjer za dva od njih ka IN i jedan za OUT.

NAME (TAG, FIELD) DIRECTION

TO1-. AI1.A_DESC IN
 TO2-. AO1.F_CV IN
 TO3-. DO1.A_ADI OUT

TBL1 - je tabela referencirana sa SELECT komandom koju koristi SQT blok.

COL1	COL2	COL3
Pumpa korak 101	99.7	NONE
Temperaturna zona 2 (C)	-2.1	Bye Bye
Ručno prebacivanje	.004	NONE
Deadband	9	ALL

Objašnjenje

Kada software izvrši ovu komandu, vrijednost za DO1.A_ADI je očitana, pošto je to izlazno polje. Vrijednost za polje je ALL: Ova komanda vadi samo zadnji red pošto je to jedini red koji se poklapa sa selektiranim kriterijima. Vrijednosti iz COL1 i COL2 u zadnjem redu su selektirane iz TBL1 i upisane su u AI1.A_DESC i AO1.A_CV.

Novi deskriptor za AI1 je sada Deadband (zona neosjetljivosti). Ako SQL task čita null vrijednost iz SQL tabele, neće se upisati nikakva vrijednost u polje ciljne baze. Dodatno,

SQT blok generira alarm i grešku upisa u polje. Ipak, bilo koja druga selektirana nenulta vrijednost će se upisati u procesnu bazu podataka.

Selektiranje višestrukih redova

FIX ODBC proizvod se može koristiti da pristupi više nego jednom redu sa SELECT komandom i upisuje vrijednosti u višestruke setove tagova ili višestruke ofsete registar blokova.

U nastavku ćemo opisati kako možemo koristiti FIX ODBC proizvod da selektiramo višestruke redove SQL podataka. Ovaj selekcionni mod je određen sa konfiguracijom SQL trigerskog bloka.

Koristiti slijedeću tabelu da se konfigurira SQL trigerski blok:

Da se selektira...	Definirati polje select parametara kao ...	Koristiti polje redova da se odredi ...
Jedan red od nekoliko selektiranih redova (rezultat postavlja)	Jednostruki red	Korišteni red
Mnogo redova	Višestruki redovi	Startni red
Mnogo redova i kolona	Mod polja (array)	Granica koliko redova se koristi iz rezultirajućeg seta

SQT polje kolone bloka se koristi da odredi broj kolona koje će se koristiti. Ovo vrijedi samo kada se koriste višestruki redovi i izabrani parametri array moda.

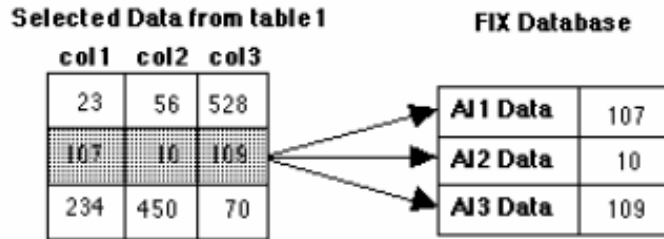
Jednostruki red

Ako SELECT komanda vrati višestruke redove, možemo koristiti mod jednostrukog reda (single row mode) SQT bloka i polje redova da odredimo koji red je upisan u SQD blok. Kada je SQT blok konfiguriran za mode jednostrukog reda , SQD blok prihvata samo jedan red podataka bez obzira na broj redova koje vrati SELECT komanda.

Polje redova SQT bloka određuje broj reda u rezultatu setovan da se koristi. Polje redova normalno defaultira na vrijednost nula kada selektiramo mode jednostrukog reda u SQT bloku. Ako je nula vrijednost u modu jednog reda, a vrati se više od jednog reda, greška će se pojaviti i podaci se neće upisati u FIX tagove. Ako se koristi neka druga vrijednost osim nule, odgovarajući broj koji se vrati će se koristiti.

Naprimjer, ako je polje reda setovano na jedan, prvi red selektiranih podataka će se koristiti.

Na ovaj način možemo identificirati koji će se red koristiti:



Primjer moda jednostrukog reda

Polje kolona se ignorira u ovom modu. Broj kolona je definiran brojem tagova u SQD bloku koji imaju smjer IN. Ako broj IN tagova se ne podudara sa brojem vraćenih kolona, pojavice se greška i podatci se neće upisati.

Višestruki redovi

Kada je SQT blok konfiguriran za mode rada višestrukog reda, SQT blok će prihvatiti više od jednog reda. Skup FIX tagova se koristi za svaki vraćeni red. Vraćene vrijednosti se upisuju kolona po kolona, red po red.

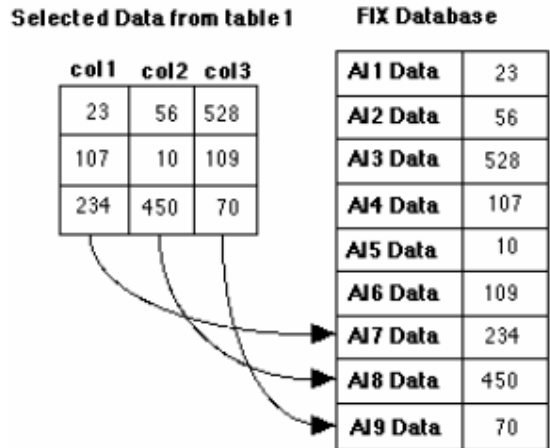
Polja kolona moraju se konfigurirati sa korektnim brojem kolona u SELECT komandi. Ova informacija mora biti definirana prije nego što je komanda izvršena.

Kao primjer, posmatrajmo slijedeći SELECT iskaz:

```
Select col1, col2, col3 from table1
```

Ova će komanda vratiti tri kolone iz baze podataka. SQD blok mora sadržavati multipl od tri FIX taga. Ako SELECT komanda vrati dva reda podataka, šest tagova treba biti specificirano u SQD bloku. Ako je više redova vraćeno sa SELECT komandom nego što ima tagova definiranih u SQD bloku, dodatni redovi se odbacuju. Ako se vrati manje redova, viška tagovi u SQD bloku će ostati neupisani.

Polja redova određuju startni red koji će se koristiti u selektiranim podacima. Na primjer, polja redova mogu biti definirana kao 3. Ako SELECT komanda vrati 10 redova, pojavice se greška i podatci se neće upisivati u tagove.



Primjer sa mode sa višestrukim redovima

Možemo također koristiti SELECT komandu da uključimo markere parametara. Na primjer.

```
Select col1, col2, col3, from table1 where col4 = ?
```

U ovom slučaju tri kolone će biti vraćene. Komanda zahtjeva jedan tag sa smjerom OUT za marker parametra.

Array mode.

Array mode se koristi sa blokovima registara. Registar blok se specificira za svaku kolonu koja se vrati od SELECT iskaza. Svaki registarski blok prima višestruke redove od kolone.

Polje redova se koristi da se postavi granica koliko će se redova upisati.

U sljedećem primjeru, SELECT iskaz vraća tri kolone i potreban je jedan marker parametara:

```
Select col1, col2, col3, from table1 where col4 = ?
```

U array modu, SQD blok sadrži jedan IN registarski blok za svaku vraćenu kolonu. Tag sa OUT smjerom je specificiran u SQD bloku za marker parametra.

Svaki registarski blok prima jedan red podataka startujući sa registarskim ofsetom specificiranim u SQL bloku podataka. Podatci se upisuju sve dok se ne dostigne granica redova (specificirana u polju redova), ili dok se podatci ne iscrpe.

Polje F_n se koristi za registarski blok u SQD bloku, gdje je n ofset od bazne adrese koja je referenca bloka. Na primjer, AR1.F_10, AR2.F_0, i AR3.F_0 su pokazani na narednoj slici. Vrijednosti selektiranih podataka se upisuju u blokove startujući od specificiranog ofseta, 10,0 i 0 respektivno.

Selected Data from table 1

col1	col2	col3
23	56	528
107	10	109
234	450	70

FIX Database

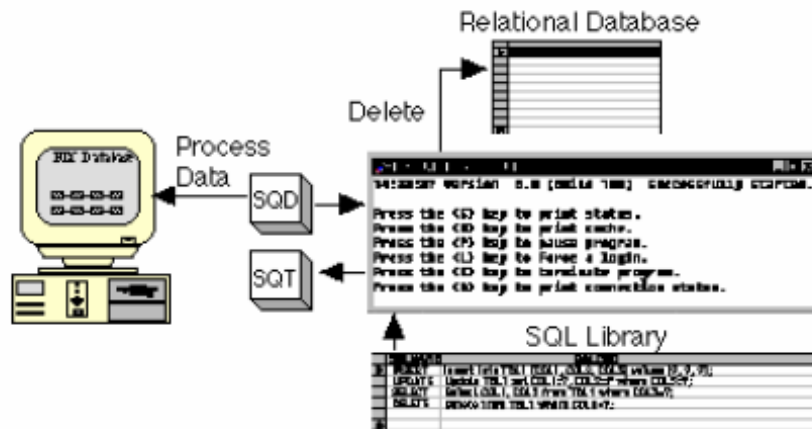
Offsets	10	11	12
AR1 Data	23	107	234
Offsets	0	1	2
AR2 Data	56	10	450
Offsets	0	1	2
AR3 Data	528	109	70

Primjer array moda

Polje kolona se ignorira u ovom modu pošto broj kolona je isti kao i broj u definiranim SQD tagovima.

DELETE komanda

DELETE komanda odstranjuje zapise (records) iz relacione baze na bazi kriterija selekcije. Naredna slika pokazuje operaciju koja će se izvršiti kada sistem izvrši delete komandu



Primjer DELETE komande

SQT1 - SQL trigerski blok koristi DELETE komandu u SQLLIB tabeli. DELETE komanda izvršava slijedeći iskaz kada je SQT blok trigeruje:

Delete from TBL1 where COL3 = ? ;

SQD1 - SQD blok podataka referencira kombinaciju taga i ime polja u bazi, i setuje smjer prenosa podataka ka OUT.

NAME (TAG.FIELD) DIRECTION

T01-. DO1.A_ADI OUT
 T02-.
 T03-.
 T04-.

TBL1 - je tabela referencirana od DELETE komande koja se koristi od strane SQT bloka.

COL1	COL2	COL3
Pumpa korak 101	99.7	NONE
Temperaturna zona 2 (C)	-2.1	Bye Bye
Ručno prebacivanje	.004	NONE
Deadband	9	ALL

Objašnjenje

Vrijednost DO1.A_ADI je jednaka NONE. Kada software izvršava komandu, vrijednost se učitava i zamjenjuje u SQL komandu. Komanda je tada:

Delete from TBL1 where COL3=' NONE' ;

Pošto i red jedan i 3 imaju 'NONE' kao vrijednost u koloni 3, obadva ova reda će biti pobrisana.

Rezultirajuća tabela će biti:

COL1	COL2	COL3
Temperaturna zona 2 (C)	-2.1	Bye Bye
Deadband	9	ALL

Pohranjene procedure

Pohranjene procedure su kompilirani blokovi koda u relacionoj bazi. One su korisne pošto mogu sadržavati kondicionalne i flow (programskog toka) iskaze. Pohranjene procedure mogu izvršavati INSERT, UPDATE, DELETE i SELECT komande. Osim toga, pohranjene procedure mogu uzimati argumente i vraćati rezultate. Argumenti mogu biti vrijednosti koje se unose ili vrijednosti koje se koriste kod 'where' iskaza.

Procedure mogu biti mnogo brže od SQL komandi iz slijedećih razloga:

- Izvršavanje pohranjene procedure zahtjeva samo jedan poziv. Da se izvrši SQL komanda, dva poziva (calls) su potrebna prema relacionoj bazi: jedan da se dobije komanda a drugi da se izvrši. Koristeći pohranjenu

proceduru, referencira se po imenu i onda se ime šalje u bazu podataka , uključujući i sve parametre ako postoje.

- Pohranjena procedura je već kompilirana u bazi podataka. Kada koristimo SQL komandu koja nije pohranjena, obadva poziva su *ad-hoc* upiti , tako da ih baza mora kompilirati kod runtime-a (izuzev ako se koristi kaširanje).

Da bi se konfigurirali SQL blokovi da koriste pohranjene procedure, koristiti slijedeće informacije:

- selektirati proceduru u boksu 'Command type group' , u okviru dijalog boksa SQT bloka za SQL parametre.
- Unjeti ime procedure u polje SQL imena, koristeći ne više od 8 karaktera.
- Definirati bilo koji ulazni argument koji se zahtjeva od pohranjene procedure, koristeći polje smjera u SQD bloku.

Naprimjer, ako pohranjena procedura treba dva ulazna argumenta i vrati podatke iz SELECT iskaza, konfigurirati SQD blok sa OUT tagovima za parametarske markere i IN tagovima za rezultate iz SELECT iskaza.

Opaska: MS Access ne podržava korištenje pohranjenih procedura. Ako koristimo MS Access kao relacionu bazu, ne smijemo koristiti PROCEDURE komandu u skriptu komandnog jezika i ne smijemo izabrati 'Stored procedure' radio dugme u konfiguratoru SQL komandi.

Korištenje podrške za multiple relacione baze

Podrška multiplim bazama dozvoljava FIX ODBC proizvodu da izvršava komande u različitim bazama. Implementacija podrške za multiple baze omogućava nam da:

- koristimo različite kombinacije 'user name/password' u bazi
- komuniciramo sa nekoliko različitih relacionih baza
- pohranjujemo sve SQL komande u jednu bazu i izvršavamo komande u nekoliko različitih relacionih baza.

Opaska: Metode pohranjivanja SQL komandi i korištenja multiplih relacionih baza koje će biti diskutovane u nastavku ne mogu se mješati sa prethodnim

Rad sa multiplim SQL konekcijama

SQL task radi sa multiplim konekcijama koristeći slijedeće:

Kada ...	Tada SQL task ...
Sistem starta	Čita iformaciju o konfiguraciji SQL taska iz SCU-a i loguje se u ID baze (ako je definiran) u konfiguraciji SQL taska. Ovaj

	izvor podataka je default login.
SQT blok trigeruje	Loguje se u bazu koristeći slijedeći patern traženja za ID baze: [1] SQT blok ID baze [2] SCU ID baze (lokacija iz tabele komandi i grešaka) [3] Prvi ID baze koji je izlistan u dijalog boksu SQL računa
Konekcija je uspostavljena	Održava konekciju sve dok SQL task se ne zaustavi.
Konekcija je izgubljena	Ponovo pokušava konekciju u intervalima specificiranim od strane /LD parametra i nastavlja procesiranje SQL blokova iz drugih izvora podataka. Komande se backupiraju u backup file specificiran u SCU sve dok SQL task ne uspostavi ponovno konekciju sa bazom podataka.
Konekcija je ponovno uspostavljena	Traži kroz backup file za ulaze. Ulazi u baze koje nisu spojene se ignoriraju sve dok se konekcija ponovno ne uspostavi.

Korištenje multiplih korisničkih računa (accounts)

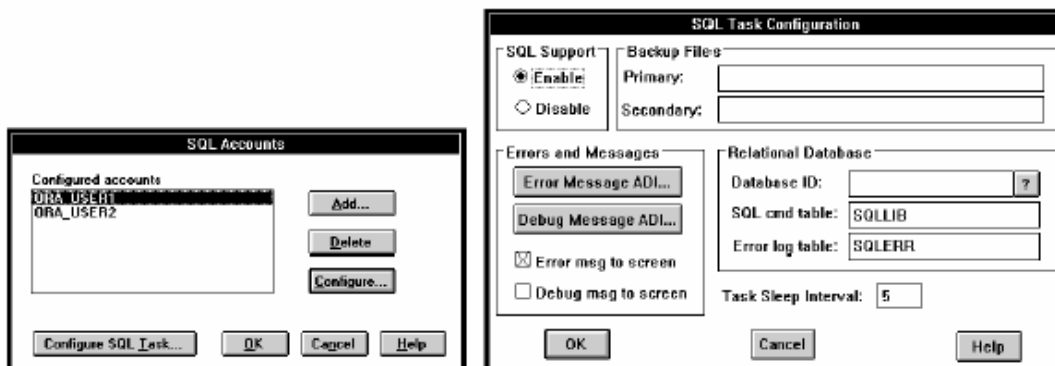
Korisnik može željeti da koristi istu bazu za sve transakcije, ali ima različite kombinacije 'user name/password'. na primjer, Oracle baza može biti konfigurirana za dva korisnika. Jedan korisnik ima pristup jednom setu tabela, a drugi korisnik drugom setu. U ovom slučaju, treba koristiti slijedeće korake da se konfigurira FIX ODBC:

[1] Kreirati dva ODBC izvora podataka koristeći ODBC administrator u Control panelu. Naprimjer, izvori podataka mogu biti nazvani ORA_USER1 i ORA_USER2. Ovi izvori podataka su identični izuzev njihovih imena (pošto su iz iste baze).

[2] Konfigurirati dva SQL login računa (po jedan za svaku bazu) u FIX SCU i dati svakom računu različito ime i password korisnika.

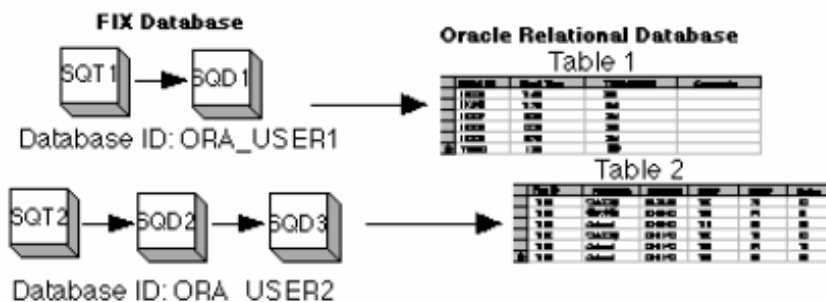
[3] Kreirati FIX lanac baze, definirajući korektne accounte, ORA_USER1 i ORA_USER2 u u SQT blokovima koristeći ID polje baze.

Naredna slika ilustrira informaciju koja je potrebna da se definira u FIX SCU da se koriste dva različita Oracle računa u FIX bazi. Konfigurator SQL taska ne uključuje referencu na ID baze. Umjesto toga, korisnik mora definirati Oracle account u svakom SQT bloku koristeći polje ID- ja za bazu.



Konfiguracija za multiple korisnike

Kada SQT blok trigeruje da izvrši komandu koristeći ORA_USER1, SQL task se loguje u relacionu bazu sa ovim imenom i passwordom korisnika. Kada SQT blok se trigeruje da izvrši komandu koristeći ORA_USER2, SQL task se logira u istu bazu sa različitom kombinacijom 'user name/password'. Naredna slika ilustrira komunikaciju izmedju lanca u FIX bazi i tabela unutar Oracle relacione baze:



Višestruki korisnici

Korištenje multiplih baza

Korisnik može koristiti FIX ODBC da pristupi nekoliko različitih relacionih baza koje je instalirao u konfiguraciji. U ovom modu, tabela za komande i greške može takodjer biti locirana u istom accountu. Međutim, tabele moraju koristiti konzistentnu konvenciju njihovog imenovanja. Na primjer, ako nazovemo SQL komandnu tabelu kao SQLLIB u jednoj relacionoj bazi, moramo koristiti isto ime i drugim relacionim bazama.

Korisnik može pohraniti recepture da ih downloaduje iz Access baze a logira procesne vrijednosti u Oracle bazu. Da bi se definirala ova konfiguracija, koristiti slijedeće korake:

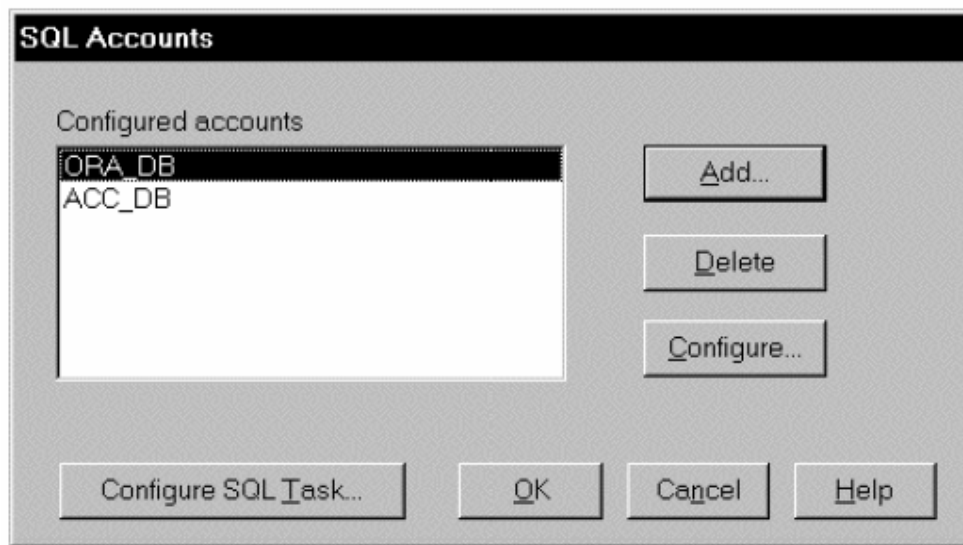
[1] Kreirati ODBC izvor podataka za svaku relacionu bazu koristeći ODBC Administrator u Control panelu. Na primjer, izvori podatka mogu biti nazvani ACC_DB i ORA_DB.

[2] Konfigurirati dva SQL accounta (po jedan za svaki izvor podataka) u FIX SCU. Polje ID baze treba biti sa imenima izvora podataka definiranim u prethodnom koraku: ACC_DB i ORA_DB.

[3] Ostaviti polje ID baze u diajlog boksu konfiguracije SQL taska bez unosa teksta (blank).

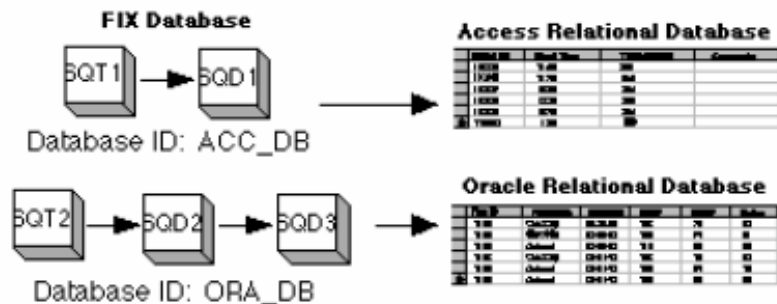
[4] Definirati polje ID baze u svakom SQT bloku da bi se pristupilo korektnoj relacionoj bazi , koristeći odgovarajući ID baze.

Naredna slika ilustririra informaciju koja je potrebna da se definira FIX SCU da bi koristio dvije različite relacione baze. Konfiguracioni task SQL ne uključuje reference ka identifikatoru (ID) baze. Umjesto toga, definirajte relacionu bazu u polju ID baze za svaki SQT blok.



Konfiguracija za multiple baze

Kao što je prikazano na narednoj slici, kada SQT1 trigeruje, konfigurirana komanda se izvršava u bazi, ACC_DB , definiranoj u konfiguraciji bloka. U ovom primjeru, lanac trigeruje download recepture. Blokovi definirani u SQD bloku prihvataju downloadovane vrijednosti recepture. Lanac FIX baze za SQT2 logira vrijednosti iz procesa u Oracle bazu koristeći ORA_DB kao ID za bazu.



Multiple baze

Centralno pohranjivanje komandi

Podrška za multiple baze može pohraniti sve SQL komande i greške u jednu lokaciju. Korisnik potom može izvršavati ove komande da bi pristupio podacima u drugim relacionim bazama. Na primjer, on može pohraniti sve konfigurirane komande u jednu Access bazu, recepte za download u drugu Access bazu, i logirati vrijednosti iz procesa u Oracle bazu.

Da bi se definirala ova konfiguracija , provesti slijedeće korake:

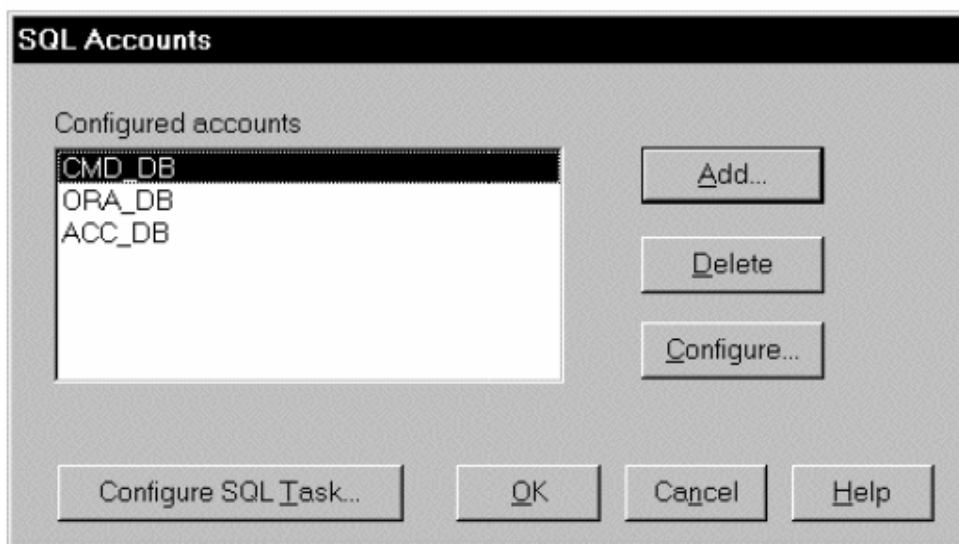
[1] Kreirati ODBC izvor podataka za svaku relacionu bazu koristeći ODBC administrator u Control panelu. Na primjer, izvori podataka mogu biti nazvani CMD_DB, ACC_DB i ORA_DB.

[2] Konfigurirati tri SQL login accounta (po jedan za svaki izvor podataka) u FIX SCU. Polje ID baze trebaju biti imena izvora podataka definirana u prethodnom koraku: CMD_DB, ACC_DB i ORA_DB.

[3] Definirati CMD_DB kao polje ID baze u dijalog boks u konfiguratoru SQL taska. Lista konfiguriranih ID baza u accounts dijalog boks se pokaže kada se selektira ? dugme.

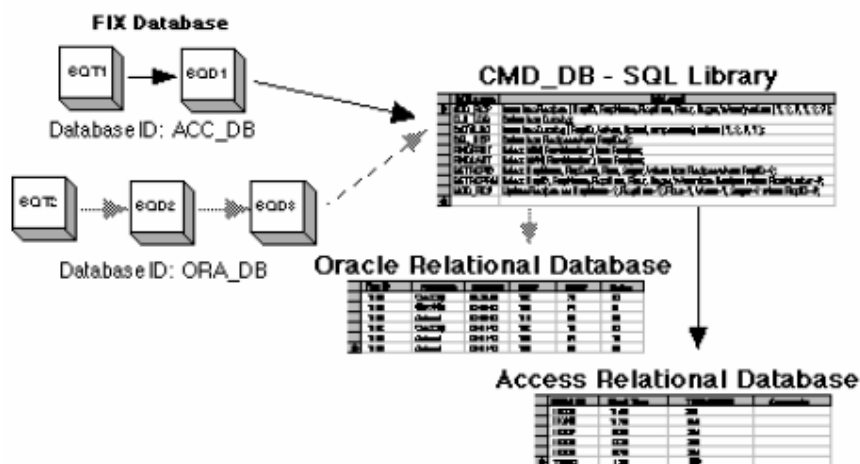
[4] Definirati polje ID baze u svakom SQT bloku da bi se pristupilo korektnoj relacionoj bazi koristeći odgovarajući ID baze. Na primjer, ako se komanda izvršava u Access bazi, tada ID baze u SQT bloku je ACC_DB.

Naredna slika ilustrira informaciju koja je potrebna da bi se definirao FIX SCU da koristi dvije različite relacione baze. ID baze u SQL task konfiguratoru referira na tabelu relacione baze koja se koristi da pohrani sve SQL komande koje se koriste sa svim relacionim bazama. Polje ID baze SQT bloka definira gdje se komanda izvršava. Slika ilustrira informaciju koja je potrebna da se definira FIX SCU da koristi relacionu bazu pohranjenu u sve SQL komande.



SCU konfiguracija za centralno pohranjivanje SQL komandi

Kako se vidi sa naredne slike, kada se izvršava SQL komanda, komanda se vadi (retrieved) iz CMD_DB baze i izvršava u relacionoj bazi specificiranoj u SQT bloku.



Centralno pohranjivanje SQL komandi

Nadzor i kontrola komunikacije sa bazom podataka

U ovom poglavlju biće opisano nekoliko načina nadzora i kontrole FIX ODBC za vrijeme runtime rada.

Pošto možemo nadzirati proces sa Operatorskog displeja, procedura opisana u ovom poglavlju se odnosi na opis šta možemo uraditi preko linkova displeja. Kompletan opis

link polja koja su na raspolaganju za korištenje sa dva SQL bloka i kojima se može pristupiti koristeći dugme parametra u DBB i Draw.

Mjenjanje setinga blokova putem linkova.

Korisnik može koristiti polja SQL blokova u linkovima da obezbjedi veću fleksibilnost u komunikaciji sa relacionom bazom. Korisnik može kreirati slike koje mu omogućuju da:

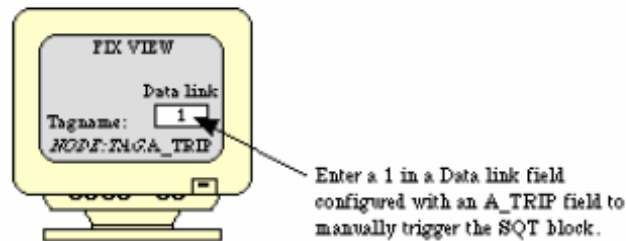
- ručno trigeruje SQT blokove
- resetuje triggerske specifikacije
- resetuje specifikacije SQD bloka

Stvarna snaga softwarea je u njegovoj mogućnosti da izvršava procesne operacije skrojene po mjeri korisnika. Slijedeći primjeri ilustriraju nekoliko primjera.

Ručno trigerovanje aplikacije

FIX ODBC proizvod dozvoljava trigerovanje SQT blokova bilo putem vremena, datuma, ili događaja na bazi tagova. Međutim, korisnik može ručno trigerovati izvršenje SQL komande putem linka, obezbjeđujući direktnu kontrolu putem FIX ODBC, dok istovremeno testira novi proces ili dok nadzire performansu aplikacije.

Definišući link podataka sa A_TRIP poljem, korisnik može prvo promijeniti mode SQT bloka na ručno, a zatim unjeti 1 da ručno trigeruje SQT blok.



Ovo obezbjeđuje moćno sredstvo izvršenja SQL komandi po zahtjevu.

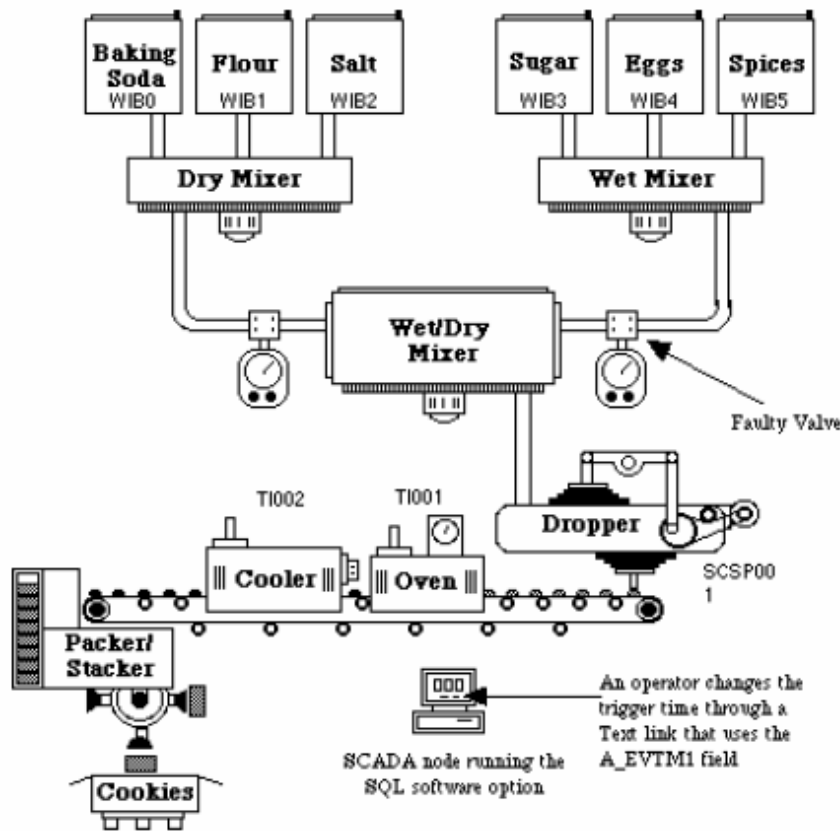
Specifikacije resetovanja time/date/events

Čak i ako imamo definiranu vremensku, datumsku ili baziranu na događaju šemu trigerovanja, možemo promijeniti ove setinge putem linkova. Ovo omogućava korisniku da redefiniira rad sistema bez modifikovanja SQT blokova. To omogućava korisniku da promjeni izvršenje batcha, da uploaduje procesnu informaciju, ili suspendira rad dok se ne obavi održavanje kritične opreme.

Na primjer, u aplikaciji gdje FIX ODBC downloaduje batch recepte koji su pohranjeni u relacionoj bazi, rad može biti suspendiran i reprogramiran (rescheduled) za neki budući datum putem linka displeja.

Opaska: Kada se koristi tekst blok za trigerovanje događaja , ODBC SQL opcija može samo čitati prvih 40 karaktera iz polja A_CV bloka. Ako korisnik uključi više od 40 karaktera u ovo polje, preostali tekst biće ignoriran.

Naredna slika pokazuje liniju proizvodnje kolača koja je pretrpila mehanički otkaz. Slijedeći download recepture biće reprogramiran promjenom trigerskog vremena putem linka displeja.



Resetovanje vremena izvršenja putem linkova

Resetovanje SQD blokova

SQD blokovi se mogu modificirati sa Operatorskih displeja. Putem linkova, korisnik može redefinirati slijedeće:

- Doznačavanje imena tagova i polja, bilo dodavajući nove ili brišući postojeće (koristeći polja A_TF01-20).
- Smjer prenosa podataka (koristeći polja A_DIR01-20)
- Resetovanjem polja u bloku (koristeći A_RST01-20).

Praćenje transakcija

Slijedeći parametri polja u SQT bloku , prikazuju informaciju koja može pomoći u debugiranju setupa i pomoći u nadzoru real – time transakcija:

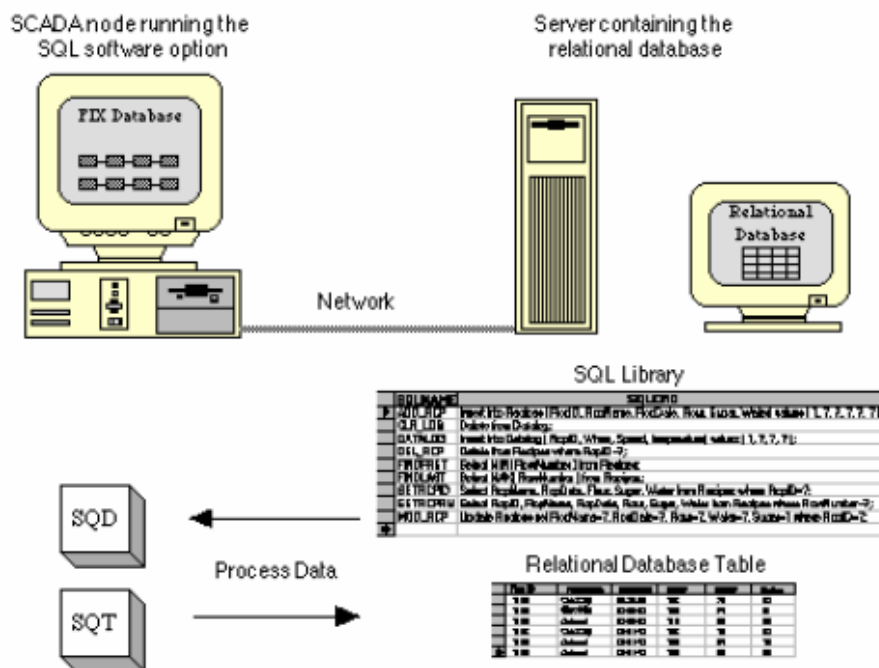
- A_SEQ pokazuje broj transakcije indicirajući da je blok prepoznao događaj. Brojevi transakcija se inkrementiraju od 1 do 255.
- A_TIME pokazuje vrijeme posljednje transakcije
- A_DATE prikazuje datum posljednje transakcije
- A_STATE prikazuje tekuće stanje rada svakog SQT bloka
- A_DBERR prikazuje izvorni broj greške baze, od posljednje izvršene SQL komande.
- A_SQLST prikazuje SQL stanje posljednje izvršene SQL komande.

Kompletan opis link polja koja su na raspolaganju za korištenje sa dva SQL bloka može biti dobijen koristeći dugme Parametara u DBB i u Draw.

Backupiranje i restauracija podataka

Za vrijeme normalnog rada FIX ODBCa, komunikacija podataka se uspješno odvija kroz mrežu kako je to zahtjevano i aplikacijom.

Naredna slika pokazuje normalnu sesiju između SCADA čvora i servera u kojoj su procesni podaci unošeni i ažurirani u relacionu bazu, a procesni podaci se downloaduju u procesnu bazu iz tabele u relacionoj bazi.

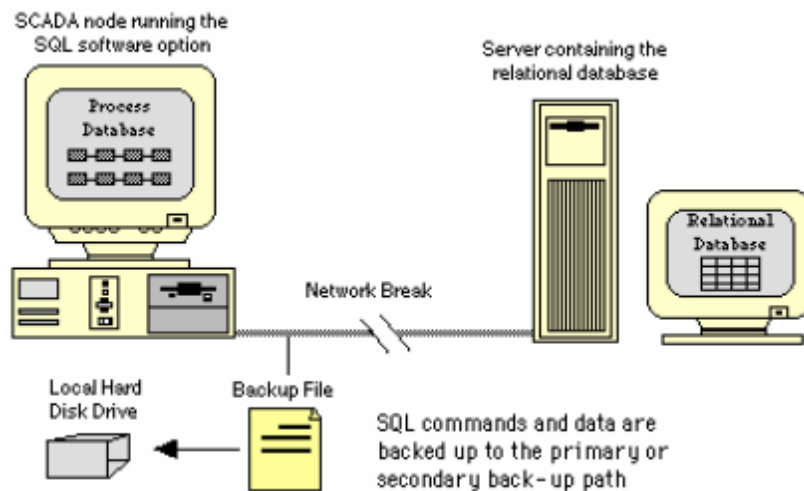


Normalan rad mreže

Backup podataka

Ako SCADA čvor na kojem se izvršava FIX ODBC proizvod izgubi konekciju sa relacionom bazom, svaka INSERT, UPDATE i DELETE komanda koja je omogućena za backup se automatski šalje bilo na primarni ili sekundarni backup file.

Naredna slika pokazuje podatke koji se backupiraju na lokalni disk SCADA čvora. Primjetimo da drugi diskovi na drugim čvorovima se mogu koristiti kao sekundarni backup. Svaki udaljeni file server se može koristiti kao primarna ili sekundarna staza (path).



Backup procedura za vrijeme greške u komunikacionom kabelu

U situacijama kada je konekcija sa serverom izgubljena za duži period vremena, FIX ODBC backupira podatke sve dok:

- primarna ili sekundarna backup staza (path) su pune
- konekcija sa serverom je ponovno uspostavljena
- SQL task je zaustavljen od strane Operatora.

Restauracija backup podataka

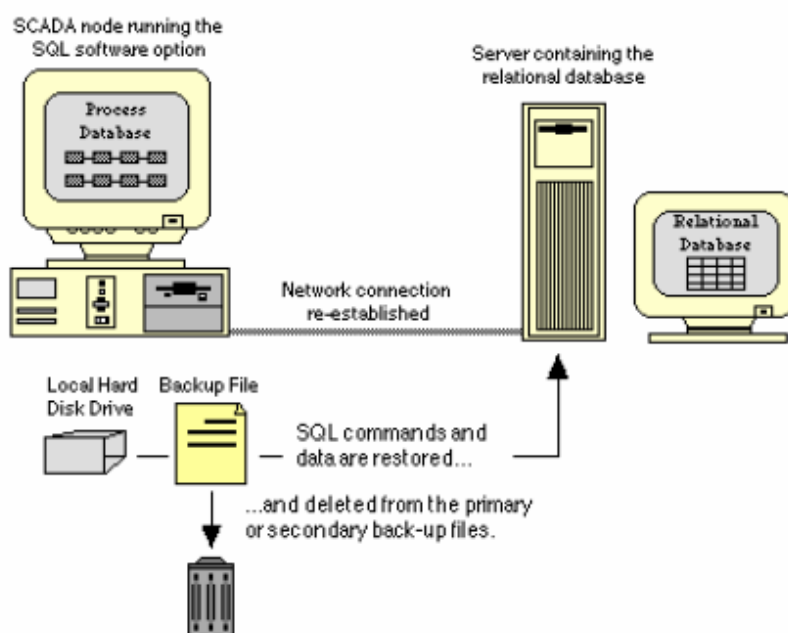
Kada FIX ODBC proizvod ponovno uspostavi konekciju sa serverom, on automatski restaurira sve backupirane SQL komande u kronološkom redosljedju, startujući sa onom koja je prva backupirana . FIX ODBC proizvod restaurira backupirane podatke koristeći slijedeće korake:

- sve backupirane SQL komande se restauriraju iz primarnih i sekundarnih backup fileova.

- FIX ODBC procesira backupirane SQL komande u redosljedu u kojem su backupirane. Ovo znači da backupirane SQL komande se procesiraju na bazi first in , first out (FIFO).
- nakon restauracije svih SQL komandi, sistem će pobrisati backup fileove.

Opaska: Ako su krirani veliki backup fileovi, restauracija SQL komandi i podataka može trajati dosta dugo. Ipak, dok trje ova restauraciija , istovremeno se procesiraju i sve nove SQL komande.

Naredna slika pokazuje backupirane podatke koji se restauriraju u relacionu bazu a zatim se brišu.



Restauracija backup podataka.

Zaštita procesa koji se nadzire i upravlja

Zbog znatnog obima vremena potrebnog za kreiranje baze podataka, korisnik će željeti da zaštiti blokove u bazi podataka od neautorizovanih izmjena. Da bi se spriječio neki Operator da može unjeti neautorizovane promjene, FIX software posjeduje sistem sigurnosti koji dozvoljava administratoru sistema da doznači pristup samo onim aplikacijama koji svaki Operator treba. Svaka aplikacija koja nije doznačena tom Operatoru ne može biti dostupna tom Operatoru.

Administrator doznava jednu ili više aplikacija svakom Operatoru kreirajući račun (Account) za svakog korisnika. Ovaj račun kontrolira programe koje Operator može izvršavati. Da bi aktivirao račun, Operator se mora logirati u sistem unoseći ime korisnika u njegovu lozinku.

Primjer

Predpostavimo da administrator sistema kreira račun za Senada i doznači mu pristup ka View programu. Kada se Senad uloguje u sistem, on može samo pokrenuti View program.

Razumjevanje sigurnosnih oblasti

Osim računa za korisnike, i ograničenja za pristup specifičnim programima, korisnik može da zaštiti i svaki pojedinačni blok u bazi podataka. Sa zaštitom bloka od pisanja, mi možemo spriječiti neautorizovanu osobu da mijenja procesne vrijednosti bloka. Na ovaj način mi štitimo i sam proces koji upravlja postrojenjem.

Zaštita od pisanja (write protection)

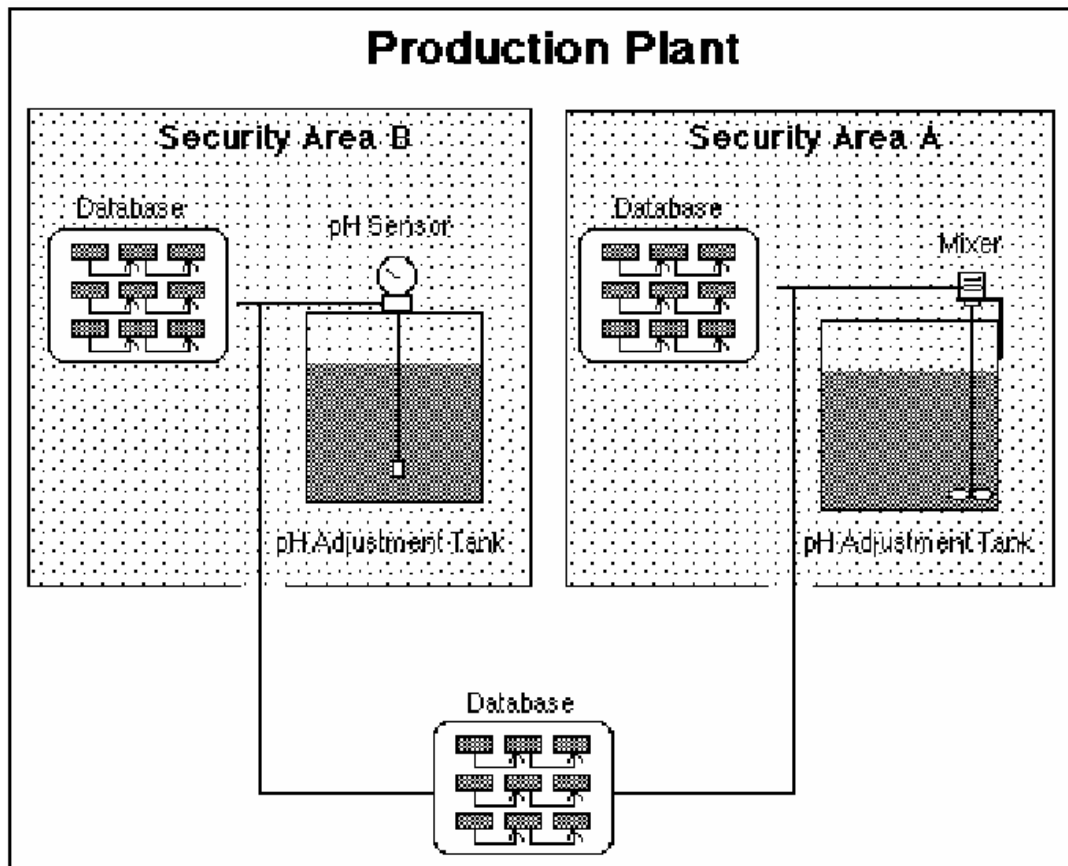
Da bi zaštitio blok od pisanja administrator sistema definira jednu ili više oblasti sigurnosti. Oblasti sigurnosti predstavljaju ili fizičku ili funkcionalnu podjelu postrojenja kojim sistem upravlja. Oblasti sigurnosti mogu biti dijelovi hardwarea postrojenja (kao što su napr. pumpe, motori, peći), ili pojedini energenti (gorivo, voda, para) ili funkcionalna podjela po zonama održavanja.

Jedanput kada su definisane, korisnik može doznačiti blok oblasti sigurnosti unoseći ime oblasti u polje : Security area, u okviru konfigurisanja bloka. Svaki dijalog boks za blok dozvoljava da imenujemo do tri oblasti sigurnosti.

Alternativno, možemo unjeti **ALL** ili **NONE** u polje : Security area. Unoseći **ALL** doznavačavamo blok u svaku oblast sigurnosti. Unoseći **NONE** ne doznavačavamo blok niti u jednu oblast sigurnosti.

Kada je blok doznačen oblasti sigurnosti, i sistem sigurnosti je omogućen, tada on postaje zaštićen od upisivanja (write protected). Blokovi koji nisu doznačeni niti jednoj oblasti sigurnosti nisu zaštićeni od upisivanja.

Naprimjer, predpostavimo da je administrator sistema definirao dvije oblasti sigurnosti A i B. Blokovi koji se koriste za upravljanje proizvodnom linijom na istočnoj strani fabrike su doznačeni u oblast sigurnosti A. Blokovi koji se koriste na zapadnoj strani fabrike su doznačeni u oblast sigurnosti B. Blokovi koji se koriste u obadvije proizvodne linije nisu doznačeni niti jednoj oblasti sigurnosti. Naredna slika ilustrira ovu konfiguraciju:



Primjer konfiguriranja po oblastima sigurnosti

Pored zaštite od upisivanja u blokove u oblastima sigurnosti A i B, konfiguracija sa prethodne slike omogućava svim Operatorima da mijenjaju samo one procesne vrijednosti u bazama podataka koje su rezidentne van oblasti sigurnosti A i B. Da bi se zaštitili ovi blokovi, treća oblast sigurnosti se može kreirati i ovi blokovi doznačeni u tu oblast. Alternativno, ovi blokovi mogli bi biti doznačeni bilo kojoj od postojeće dvije oblasti sigurnosti.

Da bi se promjenile procesne vrijednosti blokova u bilo kojoj oblasti sigurnosti, Operatoru je potrebno da im može pristupiti. Ako je autorizovan za oblast A on može mijenjati procesne vrijednosti blokova u oblasti A, ali ne i blokova u oblasti B. Alternativno, ako je oblast sigurnosti B doznačena korisničkom računu Senada, on može mijenjati samo blokove u toj oblasti.

Alarmi u FIX software-u

FIX software izdaje alarme i poruke (*alarms and messages*) da bi informirao Operatore o događajima u procesu. Alarm predstavlja prenos informacije koji zahtjeva neki odgovor od strane Operatora. Tipični alarm upozorava Operatora da se promjenila procesna vrijednost i da može da prouzrokuje probleme u daljem toku procesa.

Poruka je prenos informacije koja ne zahtjeva odziv Operatora. Tipična poruka je zapis događaja proizvedenih od strane blokova pri njihovom izvršenju, od Operatora ili SCADA čvorova.

Blokovi generišu poruke bazirane na događajima (event based).

Razvoj strategije alarmiranja

Da be se iskoristilo alarmiranje ili omogućilo slanje poruka o događajima, korisnik mora uspostaviti i ugraditi strategiju alarma i poruka. Kako se strategija razvija, potrebno je razmotriti blokove za koje želimo da omogućimo alarmiranje. Blokovi koji nadziru kritične vrijednosti u procesu zahtjevaju alarmiranje. Sa druge strane blokovi koji odgovaraju onima koji indiciraju statuse , ne zahtjevaju alarmiranje. Da be se omogućilo alarmiranje iz bloka, potrebno je selektirati ček boks: Enable alarming u dijalog boksu bloka.

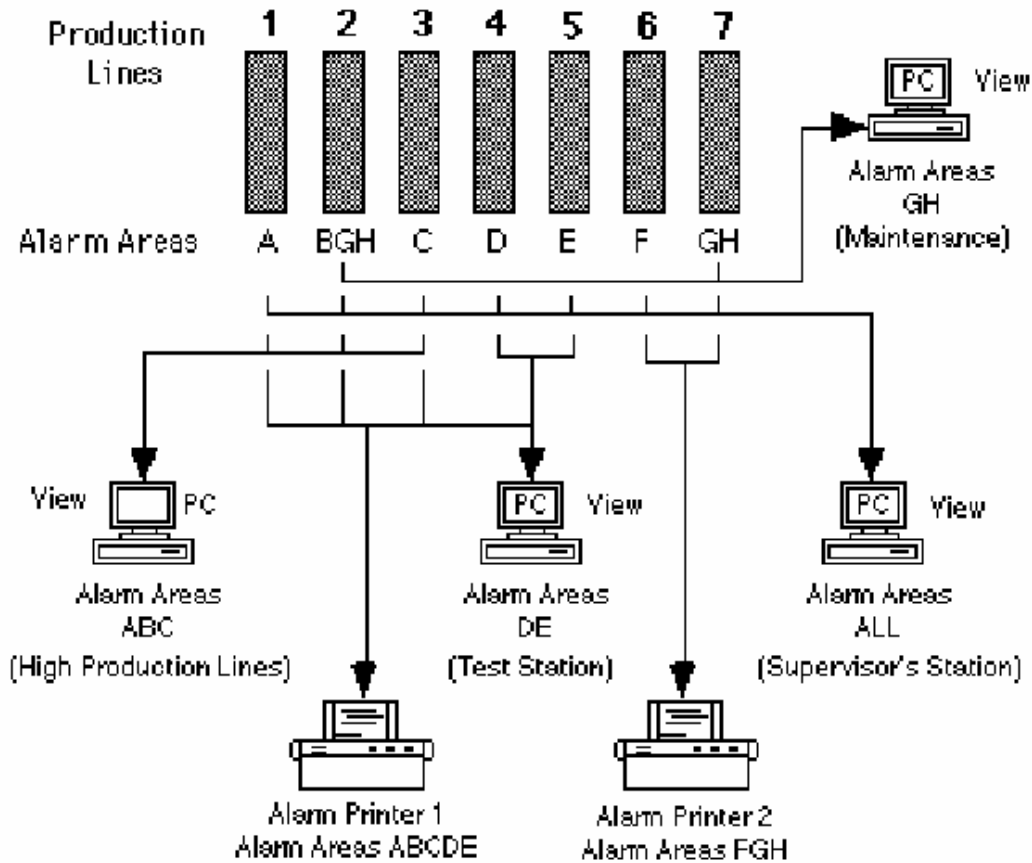
Alarmne oblasti

Takodjer, u okviru strategije alarma, potrebno je razmotriti i alarmne oblasti (*alarm areas*) koje će se doznačiti bloku. Alarmna oblast je logička podjela baze podataka koja korespondira sa fizičkom podjelom postrojenja kojim se nadzire i upravlja. Svaka alarmna oblast može imati i alarmne destinacije (alarm destination) doznačene oblastima. Destinacija alarma je printer, file ili čvor (node) koji zapisuje i pohranjuje ili prikazuje alarme i poruke.

FIX software obezbjedjuje 16 alarmnih oblasti A do P. Da bi se konfigurirale alarmne oblasti za blok, potrebno je unjeti slovo za alarmnu oblast u polje : **Alarm area** u dijalog boksu kod konfiguriranja bloka.

Alternativno, korisnik može unjeti ALL ili NONE u polje. Unoseći ALL on doznačava blok svim alarmnim oblastima, Ako unese NONE ne doznačava blok niti jednoj alarmnoj oblasti.

Na narednoj slici je pokazan primjer sa osam alarmnih oblasti :A do H. Alarmni printer 1 je doznačen za oblasti A do E. Ovo znači da su alarmi i poruka sa pet prvih proizvodnih linija ištampavani na alarmnom printeru 1.



Primjer alarmnih oblasti

Granice alarma

FIX software omogućava uspostavljanje blokova koji generiraju alarme. Analogni i digitalni blokovi podržavaju različite tipove alarma.

Alarmne granice za analogne blokove

Analogni blokovi dozvoljavaju korisniku da postavlja alarmne granice. Ove granice su specifične procesne vrijednosti ispod ili iznad kojih ne želimo da se procesna veličina nadje u toku odvijanja procesa.. Kada blok predje to graničnu alarmnu vrijednost , SAC će generirati alarm.

Naredna tabela pokazuje nekoliko tipova alarma koji se koriste sa analognim ulazima i blokovima analognih alarma:

Tabela: Neki od tipova analognih blokova alarma

Kada je alarmna granica ...	Granica alarma označava...
Visoka (high)	Visoka vrijednost u procesu. Vrijednost u bloku mora prevazići ovaj limit da bi se generirao alarm
Vrlo visoka (high high)	Kritično visoke procesne vrijednosti. Vrijednost bloka mora biti iznad ove vrijednosti da bi se generirao alarm.
Nizak (low)	Niska vrijednost procesne varijable. Vrijednost bloka treba pasti ispod ove vrijednosti da bi se generirao alarm
Vrlo nizak (low low)	Kritično niska vrijednost procesne varijable. Vrijednost bloka treba pasti ispod ove vrijednosti da bi se generirao alarm
Brzina promjene (rate of change)	Vrijednost procesne varijable se brzo mjenja. Blok treba da predje ovu granicu brzine promjene da bi se generirao alarm.
Devijacija	Odstupanje procesne varijable od nominalne vrijednosti.

Granice alarma su bazirane na EGU opsegu koji je specificiran za blok. Naprimjer, ako EGU opseg je između 0 i 100, najniža vrijednost koja se može postaviti za Low Low granicu alarma je 0. Slično, najveća vrijednost koja se može postaviti za High High alarmnu granicu je 100.

Alarmi za digitalne blokove

Pošto digitalni blokovi mogu imati samo dvije moguće vrijednosti (0 i 1), ovi blokovi koriste različite tipove alarma. Naredna tabela sumira ove tipove alarma.

Tabela : Opšti tipovi granica za digitalne alarmne blokove

Tip alarma	Generira alarm kada ...
Promjena sa normalno otvoreno	Vrijednost bloka se mjenja sa 1 na 0
Promjena sa normalno zatvoreno	Vrijednost bloka se mjenja sa 0 na 1
Promjena stanja	Vrijednost bloka se mjenja ili sa 0 na 1 ili sa 1 na 0

Alarmne granice za druge blokove

Prioriteti alarma

Pored definiranja granica alarma, korisnik može postaviti alarmne prioritete (*alarm priority*) svakog alarma i poruke koja se generira u bloku. Prioritet alarma je nivo opasnosti koji ide od niskog (low), do srednjeg (medium) i visokog (high). Sa linkom informacije o sistemu (system information link) , korisnik može postaviti prioritet alarma

za svaki čvor u View programu. Po defaultu, prioritet se postavlja kao nizak (low), međjutim, korisnik može promjeniti ovaj prioritet prema potrebi.

Postavljajući alarmne prioritete alarma, poruka baziranih na događajima (event based), i SCADA čvorovima, korisnik može skrinitati koji alarmi i poruke se prikazuju na čvorovima spojenim na SCADA čvor.

Ako je prioritet alarmnog bloka ..	Tada alarm ...
Veći ili jednak alarmnom prioritetu SCADA čvora	Se zapisuje od strane SCADA čvora i šalje na konfigurirane destinacije alarma, uključivo i udaljene čvorove.
Manji od alarmnog prioriteta SCADA čvora	Se prikazuje u linku ako je link setovan da se promjeni na bazi alarma. Alarm se ne šalje do udaljenih čvorova i ne pojavljuje se u linku sumarnih alarma.

Naprimjer:



Prioritet SCADA čvora je nizak. Zbog toga svi blokovi se šalju svim omogućenim alarmnim destinacijama spojenim na čvor.	Prioritet SCADA čvora je srednji (medium). Zbog toga , samo alarmi blokova sa srednjim i visokim prioritetom se šalju. Blokovi sa niskim prioritetom samo se prikazuju u linkovima.
---	--

Osnovi gradnje baze podataka

Graditelj baze podataka dozvoljava korisniku da kreira i manipulira procesnom bazom podataka za višestruke SCADA čvorove. Svaka baza se sastoji od blokova i lanaca koje korisnik kreira.

Spreadsheetovi

Graditelj baze podataka prikazuje informacije u spreadsheetu. Spreadsheet sadrži redove, kolone i ćelije. Red u spreadsheetu korespondira bloku u bazi podataka.

Kolona korespondira jednom polju bloka. Svaka kolona se identificira putem zaglavlja kolone. Korisnik može da kastomizira svoje prikaze editiranjem zaglavlja svake kolone. Naredna tabela daje instrukcije kako editirati ova zaglavlja kolona.

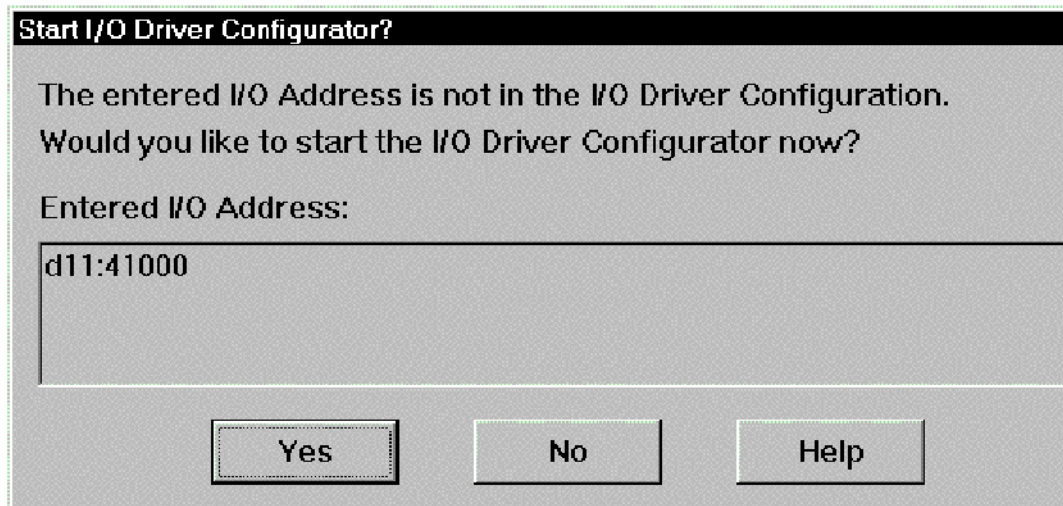
Čelija je presjek reda i kolone. Većina čelija sadrže podatke koje korisnik unese kroz dijalog boks pri konfigurisanju bloka, zatim podatke iz samog procesa, ili podatke koje unese korisnik kroz spreadsheet.

Startanje konfguracionog programa I/O drajvera

Graditelj baze podataka izlistava I/O drajvere koji su instalirani i konfigurirani u drajverskom meniju. Selektirajući jedan od ovih drajvera korisnik može startati konfiguracioni program I/O drajvera. Ovaj program omogućava dodavanje i modifikovanje kanala koji će biti izabirani i skanirani u okviru I/O uređjaja.

Opaska: SIM i DDE drajveri koji su uvijek raspoloživi u svakoj konfiguraciji FIX programa nemaju svoje konfiguracione programe.

Konfiguracioni program za I/O drajver starta automatski ako korisnik unese validnu I/O adresu koja nije definirana u DIT tabeli. Kada se to desi, pojaviće se dijalog boks starta I/O drajver konfiguratora kao što je to pokazano na narednoj slici:



Dijalog boks starta konfiguratora I/O drajvera

Korisnik unosi podatke u čeliju koristeći boks za editiranje teksta.

Manipiliranje sa bazom podataka

Graditelj baze podataka posjeduje mnoge komande koje dozvoljavaju korisniku da manipulira sa bazom. Naredna tabela izlistava komande i taskove koji su tim komandama omogućeni.

Komanda	Task
Verify	Ispituje bazu podataka na postojanje grešaka
Report	Štampa sadržaj spreadsheeta u file
Reload	Ponovno napuni bazu podataka iz fajla i radnu memoriju
Reload	Ponovno napuni bazu podataka iz fajla i radnu memoriju
Summary	Prikazuje veličinu, serijski broj, i I/O iznose u bazi podataka, također izlistava koliko blokova svakog tipa se koristi.
Save as	Duplicira bazu podataka
Export	Pohranjuje informaciju o blokovima u tekst file tako da može biti editirana i analizirana
Import	Importuje editirani tekst file u formatu baze podataka ili objedniji dvije baze

Verifikacija baze podataka

Često, nakon dodavanja blokova u bazu podataka, korisno je verifikovati da baza ne sadržava greške, prije nego što se postavi na online izvršenje. Ispitivanjem baze podataka, obezbjeđuje se da svaki blok se može procesirati i da baza podataka funkcioniše kao što je planirano.

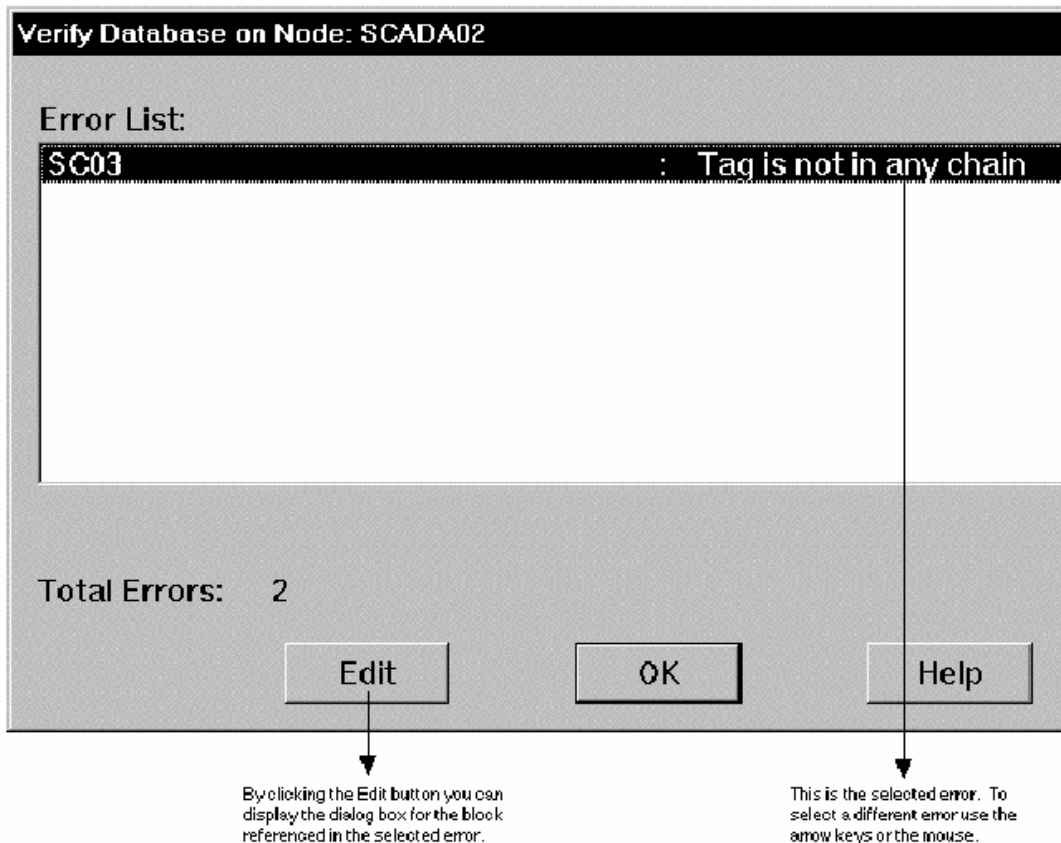
Možemo ispitati bazu podataka selektirajući **Verify** komandu iz Menija. Kada selektiramo ovu komandu, DBB obezbjeđuje da svaki blok:

- je samo u jednom lancu
- je povezan (linkovan) sa odgovarajućim blokom (naprimjer, Statistički kontrolni blok (statistical control block) može imati ispred sebe samo blok sa statističkim podacima (statistical data block)
- je korišten u korektnom kontekstu (naprimjer, kao samostalni (standalone), primarni ili sekundarni blok)
- ne referencira nepostojeće blokove

Ako DBB ne otkrije nikakvu grešku, tekst poruke u boks poruke će glasiti:

Database verification on node : [SCADA01] . No errors detected

Ako program nađe grešku, Verify dijalog boks će se pojaviti kao na slijedećoj slici:



Primjer dijalog boksa Verify sa izvješćem o grešci

Naredna tabela izlistava moguće verifikacione greške i kako postupati sa njima:

Poruka	Akcija ili značenje
<i>Tagname a, tagname b</i> . Tag is in more than one chain	Blok <i>tagname b</i> , ima više od jednog bloka koji mu prethodi (upstream) linkovan na njega. <i>Tagname a</i> identificira jedan od ovih blokova. Da bi se ovo korigiralo , treba otkloniti jedan ili više linkova ka <i>tagname b</i> .
<i>Tagname</i> :Tag is not in any chain	Može se imati sekundarni blok koji nije uključen niti u jedan lanac ili blok koji je ilegalno prvi u lancu. Blokovi koji su prihvatljivi kao samostalni blokovi ne generiraju ovu poruku. Da bi se korigirala ova situacija, potrebno je ili otkloniti sekundarni blok sa početka lanca ili dodati primarni blok na početak lanca
<i>Tagname</i> . : Block not found for NEXT	Blok, <i>tagname</i> ulančuje se na blok koji ne postoji. Da be se ovo korigiralo, modifikuj blok i unesi ime bloka koji ne egzistira u polju : Next block , ili kreiraj blok sa imenom specificiranim u polju Next Block .
<i>Tagname</i> : Chains to itself	Blok, <i>tagname</i> , sadrži svoje vlastito ime u polju Next Block . Da bi se ovo korigiralo, promjeniti ime bloka u polju Next Block , ili ostaviti polje praznim.
<i>Tagname</i> : is not defined	Blok, <i>tagname</i> , je referenciran od bloka u bazi podataka ali taj blok ne egzistira. Da bi se ovo korigiralo, ili kreirajte blok ili

	promjenite referencu na blok koji ne egzistira.
Fieldname. No such field in FDT	Polje bloka, fieldname, je referencirano od strane od bloka u bazi, ali ne egzistira. Da bi se ovo korigiralo, potrebno je promjeniti referencu na polje koje egzistira.
Exciding MAX chain size of 30	Baza podataka sadrži lanac sa više od 30 blokova. Redizajnirati ovaj lanac lomeći ga na dva manja lanca ili otklanjajući sve blokove koji nisu nužni.

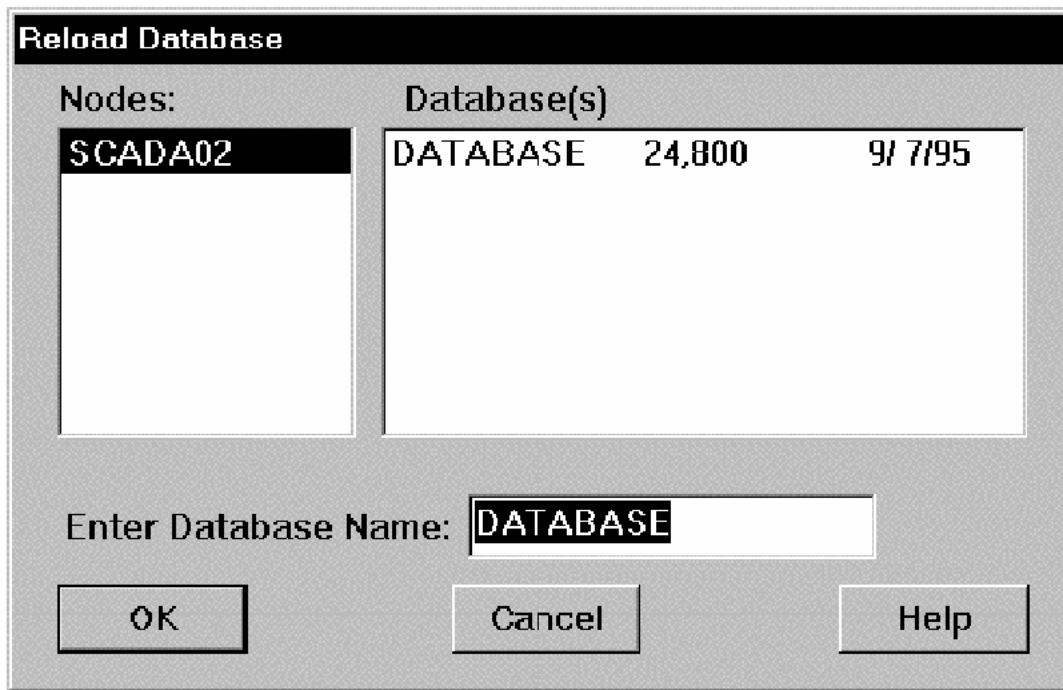
Ponovno punjenje (reloading) baze podataka

Mada korisnik može kreirati višestruke baze podataka za jedan SCADA čvor, DBB može samo da napuni i prikaže informacije o blokovima samo iz jedne baze istovremeno. Ako korisnik želi da napuni različitu bazu, može izabrati **Reload** iz Menija Baze podataka. Ova komanda takodjer dozvoljava korisniku da učini slijedeće:

- restaurira bazu podataka do posljednje spašene konfiguracije
- ponovno napuni u memoriju bazu nakon kompletiranja i pohranjivanja modifikacije
- Postavi na skaniranje konfigurirane lance da bi počelo procesiranje kada SAC starta.

Svaki SCADA čvor je konfiguriran da napuni bazu kada FIX program starta. Ako se napuni bilo koja druga baza osim startne baze, novo napunjena baza ostaje u memoriji sve dok se ne restarta FIX software. Jedanput restartovan, FIX software automatski napuni startnu bazu.

Dijalog boks za ponovno punjenje baze izgleda kao na slijedećoj slici:



Ako korisnik pokuša da napuni bazu kada nema autorizaciju za to, pojaviće se boks poruke sa slijedećim tekstom:

Unauthorized access attempted
(pokušao neautorizovan pristup)

Ponovno punjenje baze pomoću skript komandnog jezika (Command language Script)

Ako želimo da Operatori ponovno napune bazu iz View programa, možemo kreirati skriptu u komandnom jeziku (CLS) da se ponovno napuni baza koristeći slijedeću sintaksu:

```
RUNTASK DBBLOAD " -Nnodename -D database"
```

-N parametar na komandnoj liniji omogućava ponovno punjenje baze iz View programa, na udaljenom SCADA čvoru sa imenom *nodename*. Parametar na komandnoj liniji -D omogućava ponovno punjenje baze koja je različita od one koja se trenutno izvršava. Obadva parametra na komandnoj liniji su izborna i ako nisu unesena ponovno će se napuniti tekuća baza na lokalnom SCADA čvoru.

Prikazivanje sumarnih podataka o Bazi

Korisnik može biti zainteresiran da sazna koliko ima tipova blokova u bazi. Jedan od načina da dobije ovu informaciju je da pošalje upit (query) u bazu i broji redove u spreadsheetu. Brži način da sazna broj blokova koji se koristi je da izabere Summary komandu. Kada izabere ovu komandu, pojaviće se Database Summary dijalog boks, kao što je prikazano na slijedećoj slici. Ovaj dijalog boks izlistava blokove baze po tipu bloka. Prolazeći kroz listu, korisnik može vidjeti koliko se blokova koristi i koliko je dodjeljeno za svaki tip bloka.

Svaki SCADA čvor automatski unaprijed doznačava (pre-allocates) specifični broj blokova za svaki tip bloka da bi učinio efikasnim korištenje memorije i poboljšao performansu softwera sistema. Kada SCADA čvor iskoristi sve pre alocirane blokove , onda dodjeljuje novi set blokova.

Pod Windows operativnim sistemom SCADA čvor alocira blokova datog tipa onoliko koliko može stati u 4 KB memorije. Pošto blokovi variraju u veličini u skladu sa njihovim tipom, tačan broj blokova alociran varira sa svakim tipom bloka. Naprimjer, SCADA čvor alocira 21 blok analognih ulaza (AI) u jednom lotu, dok istovremeno doznačava samo 4 programska bloka.

Pored broja blokova koji su alocirani za korištenje i koji se stvarno koriste, Summary komanda takodjer prikazuje veličinu, serijski broj lokalne i udaljene baze i broj ulaza i izlaza u bazi (I/O count). Veličina indicira broj bajta koji su u stvarnosti korišteni od strane baze.

Serijski broj (SN) je jedinstveni kod koji kreira DBB svaki put kada se doda ili izbriše blok iz baze. Korisnik može koristiti ovaj serijski broj da odredi da li je bilo ko drugi unesio neke promjene u bazi nakon što je on posljednji put pohranio bazu.

Broj I/O (ulaza/izlaza) pokazuje broj alociranih blokova koji koriste ulazne ili izlazne kanale. Najčešće je ovaj broj ograničen licencom koja je kupljena za FIX paket. Kada vrijednost I/O broja je dostigla broj kupljenih licencom I/O kanala njihov broj nije moguće dalje povećavati bez proširenja licence.

Database Summary for Node: SCADA02

Database: DATABASE SN: 620215124
 Size: 467 I/O Count: 1

Type	Used	Allocated
AA - Analog Alarm	0	0
AI - Analog Input	1	10
AO - Analog Output	0	0
AR - Analog Register	0	10
BB - On-Off Control	0	0
BL - Boolean	0	0
CA - Calculation	0	10
DA - Digital Alarm	0	0
DC - Device Control	0	0
DI - Digital Input	0	0
DO - Digital Output	0	0
DR - Digital Register	0	0
DT - Dead Time	0	0

OK Help

Sumarni dijalog boks za bazu.

Optimizacija baze podataka

Nakon kreiranja i korištenja baze, korisnik može željeti da poboljša njenu performansu putem finog podešenja jednog ili više lanaca. Sa ovim se podešava vrijeme skaniranja ili mjenja faziranja da bi se poboljšala efikasnost baze.

Kao što je već ranije rečeno, vrijeme skaniranja bloka određuje kako često Scan, Alarm and Control (SAC) program (skaniraj, alarmiraj i upravljaj), će procesirati instrukcije u bloku i prenjeti tekuće vrijednosti na slijedeći blok u lancu. Blokovi koji su vezani za primarni blok se također procesiraju u vremenu primarnog bloka. Faziranje je vremensko pomjeranje (kašnjenje . staggering), kada će blokovi biti skanirani, da bi se efikasnije koristilo CPU vrijeme.

Slijedeće preporuke trebaju biti poštovane :

- ako je moguće , unesi **E** u polje vremena skaniranja da blok(lanac) bude precesiran sa vremenom izuzeća (exception based).

- Ako lanac ne može biti procesiran na bazi izuzeća, unjeti vrijeme skaniranja i fazirati ga koristeći slijedeći format :

time unit; phase

- Za vremenski bazirane lance, doznačite im najveće dopustivo vrijeme. Naprimjer ako lanac treba skaniranje od 2 minute, unesite dvije minute a ne 30 sekundi ili 1 minutu.
- Za blokove koji imaju ista vremena skaniranja , fazirajte ove blokove što je moguće ravnomjernije. Naprimjer, ako je potrebno skanirati 100 blokova svako 5 minuta, unesite faziranje od 1 minute za prvih 20 blokova, fazu od 2 minute za slijedećih 20 , i tako dalje.

Promjena redoslijeda skaniranja blokova

Moguće je takodjer fino podesiti bazu podataka promjenom redoslijeda skaniranja blokova. Da bi se razumjelo kako ovo radi, potrebno je razumjeti kako SAC skanira blokove baze. SAC skanira svaki blok baze u specifičnom redoslijedu. Ovaj redoslijed je poznat kao redoslijed izvršenja (*order of solve*). Pravila koja upravljaju redoslijedom izvršenja su slijedeća:

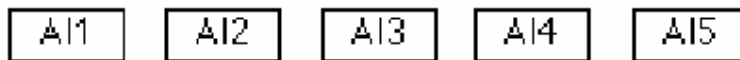
1. SAC procesira svaki blok koji izbacuje vrijednost na I/O uređaj (koristeći vrijednost hladnog starta (*cold start value*), bez obzira od njegovog redoslijeda u bazi podataka. Alarmiranje se ne izvršava kada se vrijednost pošalje na destinaciju.
2. Kada skanira primarni blok, svi blokovi u njegovom lancu se procesiraju prije nego SAC skanira slijedeći primarni blok.
3. Predpostavljajući da su svi blokovi inicijalno stavljeni na skaniranje i da koriste isto vrijeme skaniranja i fazu, primarni blokovi se procesiraju u slijedećem redoslijedu:
 - analogni ulazni blokovi
 - analogni izlazni blokovi
 - digitalni ulazni blokovi
 - digitalni izlazni blokovi
 - ramp blokovi
 - višestepeni blokovi digitalnog ulaza
 - blokovi statističkih podataka
 - Bulovi blokovi
 - Blokovi kontrole uređaja
 - blokovi analognih alarma
 - blokovi digitalnih alarma
 - Pareto blokovi
 - tekst blokovi
 - programski blokovi

Opaska: Blokovi analognih i digitalnih registara ne zahtjevaju SAC procesiranje. Oni se procesiraju samo onda kada slika koja sadrži link na neki od ovih blokova je prikazana u View programu.

4. Svi blokovi istog tipa sa istim vremenom skaniranja se skaniraju u specifičnom redosljedu. Ovaj redosljed zavisi od toga koliko je blokova dodato, izbrisano i poslije toga ponovno dodato. Ako nije dodat niti jedan blok, SAC će skanirati blokove u redosljedu dodavanja. Međutim, ako su blokovi bili izbrisani, i novi blokovi dodati, tada SAC skanira blokove u redosljedu u kojem se pojavljuju u bazi podataka. Slijedeći primjer ovo ilustrira:

Primjer

Predpostavimo da korisnik kreira 5 blokova analognih ulaza , pokazanih na slijedećoj slici. SAC skanira ove blokove u redosljedu u kojem su dodati



Blokovi doznačeni sekvenci skaniranja

Sada predpostavimo da korisnik izbriše treći analogni ulazni blok jer nije više potreban. Kao što se vidi na slijedećoj slici, ovo će ostaviti prazan prostor u bazi podataka.



Blok izbrisan iz sekvence skaniranja

SAC će sada skanirati blokove u slijedećem redosljedu:

- AI1
- AI2
- AI4
- AI5

Predpostavimo sada da korisnik želi dodati dva nova bloka AI6 i AI7 , kao na slijedećoj slici:



Blokovi dodati u sekvencu skaniranja

Prvi dodati blok će popuniti mjesto gdje je bio izbrisan blok AI3. Drugi novi blok je dodat nakon AI5. SAC će sada skanirati blokove u slijedećem redosljedu:

- AI1
- AI2
- AI6
- AI4
- AI5
- AI7

Graditelj baze izvozi primarne blokove u redosljed u koji je bio ranije dat. Ovo će rezultirati u tome da će se svi blokovi analognog ulaza prvo pojaviti u izvoznom fajlu, zatim poslije njih će biti svi blokovi analognih izlaza, itd.

Redosljed izvršenja koji koristi SAC da bi procesirao tipove blokova u bazi podataka se može mijenjati za blokove na taj način što se mogu dodati ili skinuti sa skaniranja.

Način na koji se blokovi postavljaju na skaniranje može također promijeniti redosljed izvršenja. Naprimjer, posmatrajmo blokove AI1 i AI2. Ako kriramo AI2 poslije AI1 ali ne želimo da AI2 bude prvi skaniran, potrebno je obezbjediti da je AI2 stavljen na skaniranje kod starta a AI1 je stavljen na skaniranje od strane drugog bloka (naprimjer, Programskog bloka).

Alternativno, možemo doznačiti AI2 sekundno ili subsekundno vrijeme skaniranja, a doznačiti AI1 vrijeme skaniranja od 1 minute ili duže. Imajmo na umu da vremenska jedinica vremena skaniranja (sekunda ili minuta napr.), definira i njihov prioritet u skaniranju, kao što je to vidljivo iz slijedeće tabele:

Jedinica vremena skaniranja	Proritet
Subsekunda	1
Sekunda	2
Minuta	3
Sat	4

Primjer

Kod starta svake minute, SAC skanira prvo subsekundne blokove, nakon toga blokove sa sekundnim vremenom skaniranja, i nakon toga sa minutnim vremenima.

Ako obadva bloka zahjevaju isto vrijeme skaniranja, potrebno je obezbjediti da se AI2 pojavi prvi u bazi, kao što je to pokazano u daljem tekstu, tj.

Originalna sekvenca AI1, AI2, AI3, AI4
Modificirana sekvenca AI2, AI1, AI4, AI3

Da bi se promijenio redosljed skaniranja , potrebno je izvesti (eksportovati) bazu i editirati ASCII fajl. Provjeriti da su sve informacije za blok kojeg želimo promijeniti su pomjerene na željenu lokaciju. Na primjer, da bi se skanirao blok AI2 prvi, sve informacije o tom bloku se moraju pojaviti u ASCII fajlu prije bloka AI1.

Kada završimo editiranje fajla, treba uvesti fajl u praznu bazu podataka.

Konfiguriranje alarma

Drugi način optimiziranja performanse je da se omoguće alarmi samo na onim blokovima koji ih trebaju. Sa onemogućavanjem alarma na blokovima koji ne nadgledaju direktno proces, mi elimineramo potrebu da SAC alarmne granice ovih blokova i time poboljša performansu sistema.

Slično, nije potrebno doznačavati blok svakoj alarmnoj oblasti ukoliko to nije potrebno. Ukoliko blok doznačimo u manje alarmnih oblasti, biće potrebno manje vremena SACu da provjeri i generira alarme.

Gradnja velikih baza podataka

FIX software može podržavati i baze podataka veće od 1MB. Ako planiramo izgraditi veću od ovoga bazu podataka, treba voditi računa o slijedećem:

- Veličina baze koja se može konstruirati je ograničena količinom ekspanzione memorije koja nam stoji na raspolaganju u računaru.
- Velika baza podataka koja implementira dobro vrijeme skaniranja i šemu faziranja će obezbjediti bolju performansu sistema nego velika baza (sa hiljadama tačaka skaniranja) koja ima blokove koji se svi istovremeno skaniraju.
- FIX software posjeduje niz karakteristika koji mogu pomoći pri procesiranju velikih baza podataka, kao:
 - Procesiranje bazirano na izuzeću (trigerovano promjenom na I/O adresi)
 - Jednostruko procesiranje lanca
 - Duga vremena skaniranja (do 24 sata sa konfiguracijama faziranja koja uključuju : HRS:HRS, HRS:MIN, MIN:MIN, MIN:SEC, itd.)
 - Subsekundno procesiranje

Opaska: Ponovno punjenje baze (reload), Serch/Replace, i Generate komande, kada se izdaju u slučaju velikih baza podataka , mogu prouzročiti da lokalni SCADA čvor izgubi sesiju (komunikaciju sa udaljenim čvorom), ili prouzrokuje kašnjenje SAC-a u izvršenju (SAC overruns). Kada se završi izdata komanda DBB (graditelja baze), rad SCADA će se vratiti u normalni režim.

Mnemonic (skraćeno označavanje blokova koji se mogu konfigurirati u bazi je dato u slijedećoj tabeli :

Tip bloka	Mnemonic
Analogni alarm	AA
Analogni ulaz	AI
Analogni izlaz	AO
Analogni registar	AR
Bulov blok	BL
Računarski blok	CA
Mrtvo vrijeme (zona neosjetljivosti)	DT

Kontrola uređjaja	DC
Digitalni alarm	DA
Digitalni ulaz	DI
Digitalni izlaz	DO
Digitalni registar	DR
Akcija po događaju	EV
Prošireno trendiranje	ETR
Izbacivanje (fanout)	FN
Histogram (historijski zapis)	HS
Kolo prednjačenja/kašnjenja	LL
Višestepeni digitalni ulaz	MDI
On-off upravljanje	BB
Pareto	PA
PID regulator	PID
Programski blok	PG
Rampa (nagibna funkcija)	RM
Odnos (ratio), pomak (bias)	RB
Selekcija signala	SS
SQL podatci	SQD
SQL trigger	SQT
Statističko upravljanje	SC
Statistički podatci	SD
Tekst	TX
Tajmer (vremenski član)	TM
Totalizator	TT
Trendiranje	TR