SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA CHEMICKEJ A POTRAVINÁRSKEJ TECHNOLÓGIE

KATEDRA INFORMATIZÁCIE A RIADENIA PROCESOV

Bc. Peter Borkovič

Vytvorenie interface pre prenos konfiguračných údajov do systému Yokogawa

Diplomová práca

Vedúci diplomovej práce:
Doc. Dr. Ing. Miroslav Fikar

Bratislava, 2005

SLOVENSKÁ TECHNICKÁ UNIVERZITA

Fakulta chemickej a potravinárskej technológie

Radlinského 9, 812 37 Bratislava

Katedra: **informatizácie a riadenia procesov**

Číslo: 2/KIRP/2005

Vec: **Zadanie diplomovej práce**

Meno a priezvisko študenta: **Bc. Peter Borkovič**

Meno a priezvisko vedúceho diplomovej práce**: Doc. Dr. Ing. Miroslav Fikar**

Meno a priezvisko konzultanta diplomovej práce**: Ing. Lukáš Dermíšek**

Názov diplomovej práce:

Vytvorenie interface pre prenos konfiguračných údajov do systému Yokogawa

Vytvorenie databázy (v prostredí Excel, Access) a export týchto dát do riadiaceho systému Yokogawa. Štruktúra údajov bude upresnená firmou ProCS.

Termín odovzdania diplomovej práce: **21. mája 2005**

Diplomová práca sa odovzdáva v 3 exemplároch vedúcemu katedry.

Bratislava **1. februára 2005**

Doc. Dr. Ing. Miroslav Fikar
vedúci katedry

Prof. Ing. Dušan Bakoš, DrSc.
dekan

Poďakovanie:

Dovoľujem si touto cestou poďakovať Doc. Dr. Ing. Miroslavovi Fikarovi, vedúcemu diplomovej práce a Ing. Lukášovi Dermíškovi, konzultantovi diplomovej práce za všestrannú pomoc, cenné rady, ochotu a odborné vedenie. Zároveň ďakujem všetkým, ktorý mi akoukoľvek formou pomohli pri realizovaní tejto diplomovej práce.

Abstrakt

Diplomová práca sa zaoberá bližším priblížením riadiaceho systému Yokogawa CENTUM CS3000 a vytvorením programu na prenos konfiguračných údajov z databázových systémov do systému CENTUM. Konfiguračné údaje boli prenášané na nastavenie procesných kariet a vytvorenie grafických schém v Control Drawing Builderi.

Na vytvorenie programov bolo použité programovacie prostredie Visual Basic for Application.

Abstract

Diploma thesis deals with introducing the Yokogawa CENTUM CS3000 control system. It also deals with development of user firendly interface to transfer configuration data from database applications to system CENTUM. Configuration data were transfered to set up input-output module and to build up graphic schemes in Control Drawing Builder.

For programs developing was used programing environment Visual Basic for Application.

Obsah

Úvod

V súčasnosti s neustálym rozvojom informačných technológií je samozrejmé, že sa rozvíjajú aj riadiace systémy na nich založené. Vznikajú čoraz komplexnejšie riadiace systémy uspôsobené na plnenie rôznych užívateľských požiadaviek. Jeden z najznámejších takýchto riadiacich systémov je CENTUM CS3000 od firmy Yokogawa. Patrí medzi distribuované riadiace systémy, ktoré sa vyznačujú veľkou flexibilitou a preto môžu byť použité na riadenie rôznorodých procesov.

Pri riadení rozsiahlejších procesov treba definovať veľký počet parametrov, dodaných zákazníkom, na opis systému. Ide o jeden z najdôležitejších, ale hlavne najzdĺhavejších procesov pri nábehu riadenia.

Úlohou tejto práce bolo oboznámiť sa so systémom Yokogawa CENTUM CS3000 a vytvoriť program na automatický prenos konfiguračných údajov z databázy Microsoft Excel do tohto systému ako aj program pre prenos údajov opačnou cestou.

Diplomová práca sa skladá zo 4 kapitol. V prvej kapitole je opísaný riadiaci systém CENTUM CS3000 od firmy Yokogawa, jeho najdôležitejšie súčasti a postup pri vytváraní vstupno-výstupných modulov. Druhá kapitola opisuje použitý programovací jazyk Visual Basic. Tretia kapitola sa zaoberá možnou dátovou komunikáciou, ktorú používa systém CENTUM CS3000. V štvrtej kapitole je rozobraná samotná práca, popísané sú jednotlivé vytvorené programy, užívateľské rozhranie, postup vytvárania programov a vysvetlená podstata práce jednotlivých programov. V tejto kapitole sú taktiež zobrazené výsledky prenosu konfiguračných údajov.

1. Yokogawa CENTUM CS 3000

CENTUM CS 3000 je novým produktom distribuovaných riadiacich systémov firmy Yokogawa, ktorej história sa píše už od roku 1915. Používa sa na riadenie procesov väčšieho a stredného rozsahu. Systémy CS3000 ako ja staršia verzia CS1000 pracujú s najnovšími a vrcholovými technológiami a sú navrhnuté pre poskytovanie optimálnych riešení vďaka ich otvorenosti a rozsiahlosti. Pre svoju spoľahlivosť našli uplatnenie v mnohých prevádzkach po celom svete. CENTUM CS3000 pracuje v známom operačnom systéme MS Windows NT a ako prvý na svete používa aj operačné prostredie MS Windows XP.

Riadiace funkcie v FCS, zásahy a monitorovanie v HMI môžu byť testované v testovacom prostredí. To umožňuje vyladiť riadenie bez FCS hardvéru a testovať jednotlivé časti aplikácie, čo má pozitívny vplyv na dobu prípravy riadenia.

Návrh procesu sa robí grafickou cestou a navrhnutý systém možno testovať bez reálneho zariadenia (targetless testing). Pre každý typ procesu možno prispôsobovať hardware i software.

1.1. Hardvér systému

Napriek tomu, že sa v systémoch udomácnila otvorená architektúra, zároveň musí prevádzaný riadiaci hardware garantovať riadenie procesu v stabilných podmienkach a najdôležitejšie procesy s vysokou spoľahlivosťou.

1.1.1. Field Control Station (FCS)

FCS zdedila uznávanú spoľahlivosť celej CENTUM série. Ide o jednu z najspoľahlivejších FCS v priemysle. Odkedy riadiace výpočty zabezpečuje procesorová doska, je neodmysliteľná konfigurácia tejto karty so zdvojenou zálohou. Dve CPU jednotky sú umiestnené v pároch na každej procesorovej karte, takže môže vznikať len nepatrné množstvo výpočtových chýb. Tento typ konfigurácie sa nazýva "Pair & Spare". Všetky typy FCS (Štandardná, Rozšírená, Migračná a Vysoko decentralizovaná) sú dostupné pre decentralizovanú inštaláciu s minimálnymi priestorovými nárokmi (obr. 1.1).

Obr. 1.1 Zobrazenie niektorých typov FCS a Field network I/O modulu [5]

1.1.2. Procesné I/O moduly

V zostave CENTUMu sú dve série produktov I/O modulov, takže užívateľ si môže vybrať správny modul v súlade so zariadením a prístrojovou technikou. Prvý typ je Field network I/O (FIO). Je to kompaktný typ, zameraný ako priemyselný štandard novej generácie. Druhým typom je Remote I/O (RIO). Je to špecializovaný typ vyvinutý na prevzatí značného množstva osvedčených CENTUM skúseností s I/O.

1.1.3. Human Interface Station (HIS)

Je to stanica pre riadenie užívateľom - operátorská stanica. CENTUM CS3000 obsahuje tri typy HIS staníc (obr. 1.2).

Enclose Display Style Console HIS - zdedila klasický štýl CENTUM HIS staníc. Môžu byť umiestnené paralelne jedna s druhou.

Open Display Style Console HIS - nový typ HIS konzol, ktorý už na zobrazovanie používa LCD monitory.

Desktop HIS - HIS stanica, ktorá je vlastne všeobecné PC.

V súčasnosti HIS podporuje zapojenie s dvomi monitormi, takže môže byť súčasne zobrazených viacej okien, viacej procesných premenných a prípadné alarmy môžu byť rýchlejšie spozorované a spracované. CENTUM CS3000 vďaka použitiu Windows XP dosiahol výrazné zjednodušenia obsluhy. Hociktorý HIS monitor môže byť zobrazený aj mimo regulačnej miestnosti, takže užívateľ môže niektoré procesy ovplyvňovať aj so svojej kancelárie.

Obr. 1.2 Zobrazenie Enclosed a Open Display HIS stanice a operátorskej klávesnice

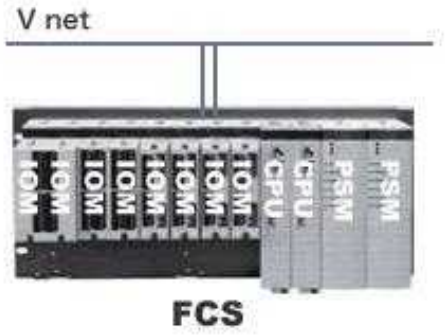
1.1.4. Operátorská klávesnica

Operátorská klávesnica je určená pre Desktop HIS. Je zabezpečená proti vode a prachu. Zachováva si všetky funkcie ako operátorská klávesnica zabudovaná v konzolových štýloch HIS.

1.2. Vytvorenie hardvérovej konfigurácie IOM (vstupno-výstupného modulu)

1.2.1. Vstupno-výstupný modul

Vstupno-výstupný modul je procesná karta zasunutá v slote Field Control Station (obr. 1.3). Slúži na zber dát, ich spracovanie a odovzdávanie dát. Je to jedna z najdôležitejších súčastí FCS.



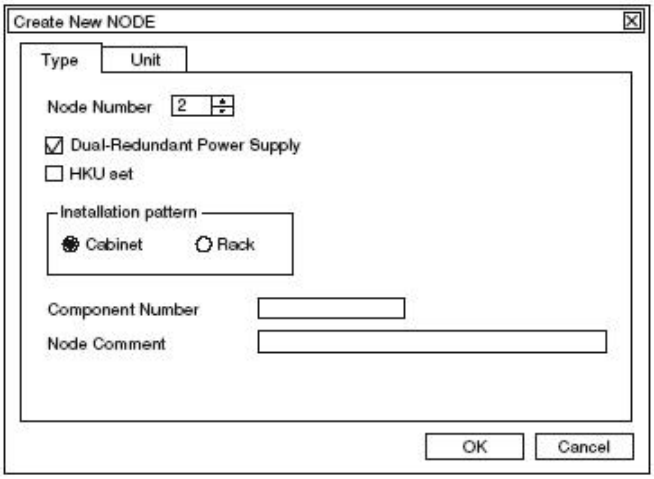
Obr. 1.3 Umiestnenie vstupno-výstupných modulov v FCS [6]

1.2.2. Vytvorenie IOM

Najprv sa definuje typ a umiestnenie vstupného a výstupného modulu inštalovaného v FCS. Typy I/O modulov inštalovaných v FCS sa menia s typom FCS a typom I/O zbernice. Keď sa ako typ stanice FCS použije KFCS alebo LFCS (štandardné typy FCS), pred vytvorením I/O modulu musí byť vytvorený uzol (NODE) pre inštaláciu I/O modulu. [6]

1.2.3. Vytvorenie nového uzla

Pri vytvorení novej FCS sa automaticky pridá jeden uzol do primárnej I/O zbernice. Potom môžu byť vytvorené ďalšie uzly, môžu byť mazané už existujúce uzly alebo menené ich vlastnosti. Pri vytváraní nového uzla sa zobrazí dialógové okno (obr 1.4), v ktorom sa nastavia požadované vlastnosti.



Obr. 1.4 Dialógové okno pri vytváraní uzlu [6]

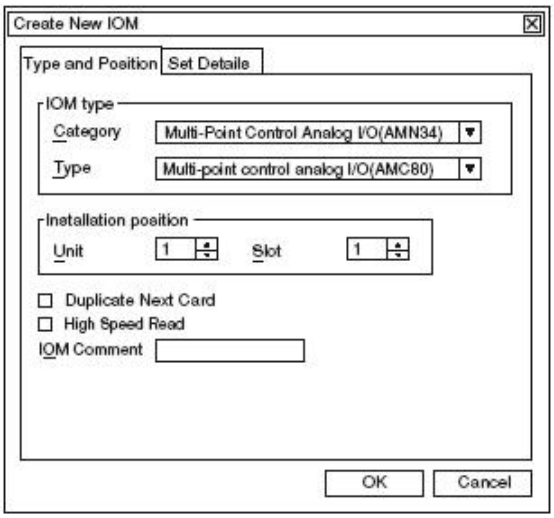
Popis jednotlivých častí [6]:

- **Node Number** je číslo pre identifikáciu uzla. Môže byť nastavené v rozmedzí 1-8. Prednastavená hodnota je 1
- **Component Number** je číslo priradené príslušnej priehradke na nainštalovanie FCS jednotky, ktoré sa používa na identifikáciu miesta na pripojenie kábla.
- **Node Comment** - komentár uzlu, ktorý môže obsahovať 24 znakov veľkosti 1 bajt alebo 12 znakov veľkosti 2 bajty.
- **Start mode** - položka sa nachádza v záložke Unit. Je to spôsob štartovania I/O jednotky. Treba nastaviť čas v rozmedzí 1 - 16 000 msec alebo vybrať "MAN" (manual). Prednastavená hodnota je 2000 msec.

1.2.4. Vytvorenie nového I/O modulu

I/O modul môže byť vytvorený pre FCS, ktorú vyberieme zo stromového adresára v System View. Vyberie sa adresár [FCS]-[IOM]-[NODExx] a potom sa z menu vyberie možnosť [Create New]-[IOM]. Otvorí sa dialógové okno pre vytvorenie modulu (obr.1.5). Okno obsahuje dve záložky - Type and Position a Set Details.

V týchto vyberieme typ IOM modulu, jeho kategóriu, pozíciu pre inštaláciu a zadáme komentár k modulu.



Obr. 1.5 Dialógové okno pri vytváraní nového I/O modulu [6]

1.3. Názvy tagov (Tag Names)

Funkčné bloky a riadiace elementy sú určené pomocou názvov tagov (Tag Names), kvôli ich jednoznačnej identifikácii. HIS používa názvy tagov na identifikovanie cieľov operácii a monitorovania. Názvy tagov sú taktiež používané v matematických výrazoch na reprezentáciu prepojenia funkčných blokov. Jeden tag reprezentuje práve jeden funkčný blok. Názov dvoch tagov nesmie byť rovnaký. Ak sa vyskytne identický názov tagov medzi integrovanými projektmi, project ID bude automaticky pridané k názvu tagu ako prípona. Project ID môže byť definované v Multiple projects connection builder pomocou dvoch alfanumerických znakov. Úprava tagu potom nastane podľa formátu [6]:

"Tag Name" + "@" + "Project ID"

Napríklad: **FIC100@P1**

Rozoznávame dva typy tagov:

- Užívateľom definované názvy tagov
- Systémové názvy tagov

1.3.1. Užívateľom definované názvy tagov

Sú to názvy tagov, ktoré môžu byť zadávané a menené užívateľom. Môže to byť reťazec 16 veľkých alfanumerických znakov, podtrhovníka a pomlčky, ktoré však nemôžu byť na začiatku tagu.

1.3.2. Systémové názvy tagov

Formát týchto tagov je nastavovaný systémom. Užívateľ môže zavolať nejaký HIS element použitím definovaných formátov. Systém pri vytváraní tagov používa presne stanovené pravidlá, ktoré sú zobrazené v tab. 1.1

Tab. 1.1 Tabuľka názvov jednotlivých typov systémových tagov [6]

Type	Format	Example
Common switch	%SWxxxxSddaa	%SW311S0102
Global switch	%GSyyyaaSddmm	%GS01201S0101
Annunciator Message	%ANxxxxSddaa01	%AN23S010301
Process I/O	%Z01usccSddaa (*1) %ZnnusccSddaa (*2)	%Z0111201S0101
Communication I/O	%WBxxxxbbSddaa	%WB101201S0102
Function Block	%BLxxxxSddaa01	%BL0125S010201

Jednotlivé znaky v tabuľke znamenajú:

SW,AN,Z,WB,BL - časť kódu indikujúci typ elementu
xxxx - číslo elementu
S- ukazovateľ, ktorý indikuje číslo domény a číslo stanice
dd - číslo domény (01 až 16)
aa - číslo stanice
yyy - číslo elementu (001 až 256)
mm - číslo stanice (01 až 64)
nn - číslo uzla (01 až 08)
u - číslo jednotky (1 až 5)
s - číslo slotu (1 až 4)
cc - číslo terminálu (01 až 32)
bb - počet bitov (01 až 16)

1.3.3. P&ID názvy tagov.

Sú špeciálne názvy tagov používané pri nákrese grafov prístrojovej techniky. V CS1000 sa používajú ako komentár k poznačeniu ich vzťahu k tagom v diagrame. P&ID názvy tagov nie sú zobrazené v okne grafickej schémy.

1.3.4. Užívateľom definované menovky

Sú to názvy, ktoré môžu byť definované pre analógový vstupno-výstupný proces. Menovky môžu byť použité pri definovaní I/O spojení, spojení funkčných blokov, sekvenčnej tabuľky a matematických výrazov. Môžu mať najviac 16 alfanumerických znakov a musia mať formát:

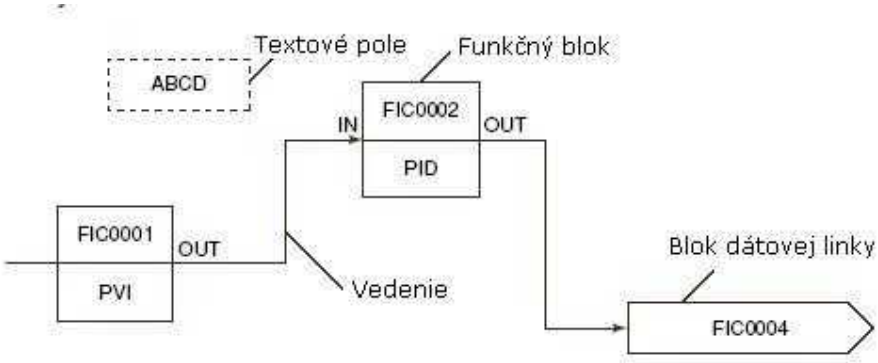
%%Mn...n kde: **%%** - zafixovanie do systému
M – alfanumerický znak veľkým písmenom
n...n - alfanum. znaky, najviac 13 znakov

1.3.5. Komentáre

Komentáre sú opisy pripojené k oknám, funkčným blokom, názvom tagov a súborom. Vďaka nim sa dá lepšie orientovať v projekte, ľahšie sa rozoznajú jednotlivé funkčné bloky a okná počas operácií a monitorovania. Správne definovaný komentár je zobrazený v riadku príslušného prvku v stĺpci Comment.

1.4. Control Drawing Builder

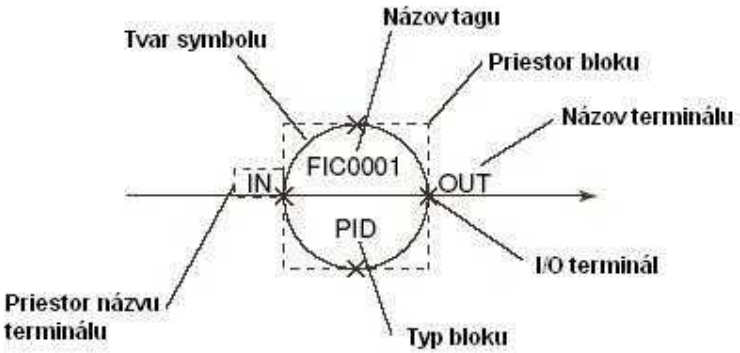
V CENTUM CS3000 sa návrh systému robí jednoduchšou grafickou cestou pomocou Control Drawing Builder. Používa sa pri konfigurácii základných riadiacich funkcií FCS. Pomocou neho sú operácie ako registrovanie funkčných blokov v súboroch a určovanie toku dát medzi funkčnými blokmi uskutočňované graficky kombinovaním objektov. Pri tomto vytváraní môže byť viacero schém otvorených súčasne, pričom možno porovnávať ich obsah, editovať a ukladať každú schému zvlášť. Rozoznávame 4 typy objektov: funkčné bloky, bloky pre linky ku dátam, vedenie a textové polia. (obr. 1.5)



Obr 1.5 Konfigurácia základných objektov Control Drawingu

Funkčný blok

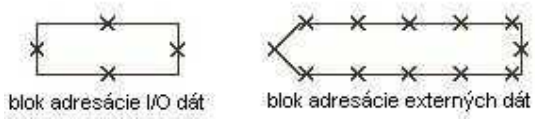
Objekt, ktorý reprezentuje funkčný blok, sa nazýva symbol funkčného bloku. Jeho štruktúra a parametre sú popísane na obr. 1.6.



Obr.1.6 Popis funkčného bloku v Control Drawingu

Blok dátovej adresácie

Blok dátovej adresácie sa používa v prípade, keď sa pripája funkčný blok k inému funkčnému bloku. Rozoznávame dva typy týchto blokov: blok vstupno-výstupných (I/O) dát a blok externých dát (obr. 1.7). Blok I/O dát je schopný pripojiť I/O proces alebo softvér. Blok externých dát je schopný pripojiť funkčný blok definovaný v inej schéme, alebo funkčný blok definovaný v inej riadiacej stanici.



Obr.1.7 Zobrazenie blokov dátovej adresácie v Control Drawingu

Vedenie

Reprezentuje tok dát medzi funkčnými blokmi alebo medzi funkčným blokom a blokom dátovej adresácie.

1.5. Operátorské rozhranie (HMI)

Operátorské rozhranie beží pod operačným systémom Windows NT. Vďaka tomu je toto prostredie jednoduché a prehľadné, využívajúce klasické prvky Windowsu ako sú ikony, okná. Pri spustení systému sa všetky okná orámujú zelenou farbou a v hornej časti obrazovky sa zobrazí ovládací panel (obr. 1.8), ktorý sa nedá prekryť žiadnym iným oknom. Panel obsahuje tlačidlá používané na volanie jednotlivých operátorských okien a prislúchajúcich funkcií. Taktiež zobrazuje systémové hlásenia a indikuje vznik prípadných alarmov.

Obr. 1.8 Panel operátorského rozhrania

Medzi základné operátorské okná patrí prehľad systémových alarmov, proces report informujúci o stave funkčných blokov alebo historical report archivujúci všetky operácie a stavy systému, ktoré sa vyskytnú počas prevádzky.

Pomocou grafického editoru, ktorý je ďalšou súčasťou systému CENTUM sa zakreslia jednotlivé súčasti prevádzky, aby bolo zobrazenie dejov čo najprehľadnejšie. Zakresleným objektom sa priradia prislúchajúce tagy a tým sa zabezpečí komunikácia

objektov s hodnotami v systéme. Pomocou náhľadového okna je možné zobraziť operačné a alarmové stavy veličín alebo jednotlivých komponentov schémy. Taktiež sa dá nastaviť zmena farby alebo blikanie niektorej hodnoty, keď presiahne určenú hranicu. Takáto zmena ihneď upozorní na výnimočný stav v systéme. V riadiacom okne sú zobrazené kontrolné panely jednotlivých funkčných blokov, z ktorých je možné určiť stav a hodnoty veličín a v prípade potreby meniť nastavenie parametrov. Ďalším operátorským oknom je okno na zobrazovanie trendov, v ktorom sa zobrazujú priebehy vybraných veličín. Aktualizácia okien je možná každú sekundu, takže operátor má výborný prehľad o stave systému.

Zobrazovanie okien je možné v dvoch režimoch. Vo Full screen mode aktuálne okno vypĺňa celú pracovnú plochu. V Multiple Windows mode sa okná zobrazujú ako v klasickom PC s operačným systémom Windows.

1.6. Využívanie externých súborov

CENTUM CS 3000 tak ako väčšina rozsiahlejších programov využíva export a import súborov (buider files) reprezentujúcich jednotlivé časti projektu (konfigurácie I/O modulov, schémy). Takto sa môžu prenášať informácie medzi jednotlivými projektmi. Systém vytvorí súbory s iným menom a príponou. Exportované súbory môžu byť editované i mimo pôvodného programu. Pri editácii však treba poznať pôvodnú štruktúru zápisu súboru. Ak sa pri nej spravia chyby, CENTUM nemusí editovaný súbor importovať.

Export súborov

Definovaný súbor systému CENTUM môže byť uložený do rôznych typov súborov. Táto operácia sa nazýva export súboru. Súbory môžu byť exportované ako CSV súbory alebo textové TXT súbory. Typ súborov však závisí na type CENTUM systému.

Typickým použitím funkcie export je uloženie chybnej časti projektu. Systém CENTUM pri ukladaní zmenenej časti najprv prevádza kompiláciu. Ak narazí na nejakú chybu, nedovolí túto časť projektu uložiť. Miesto toho ponúkne uloženie súboru vo formáte SVA. Toto uloženie je výhodné, ak editácia programovej časti neprebehne celá naraz a systém hlási nedokončený projekt ako chybu.

Import súborov

Obsah projektov a vytvorené systémové súbory v CS3000 môžu byť duplikované a použité v inom CS3000 projekte. Docieli sa to pomocou funkcie import. Importované môžu byť EDF, SVA, CSV a TXT súbory, taktiež konvertované súbory. Súbory, ktoré môžu byť importované sú súbory pochádzajúce z CS1000/CS3000 projektov alebo iných staníc, odkiaľ boli uložené pomocou funkcie export.

1.6.1. Súbory typu CSV (Comma-separated variable)

CSV súbory sú textovo editovateľné súbory produkované tabuľkovými procesormi a databázovými programami. Používa sa na výmenu informácií medzi rôznorodými aplikáciami. Tento formát, tak ako je použitý v Microsoft Excel, sa stal štandardom vo všetkých smeroch priemyslu, dokonca aj na ne-Microsoftových platformách.

CSV súbor je databáza exportovaná do formátu, kde každý záznam je zapísaný v jednom riadku a každé pole záznamu je oddelené pomocou čiarky. Tam kde je to potrebné (napr. výraz v poli obsahuje čiarku) bude pole navyše oddelené úvodzovkami.

Tento typ súborov je najčastejšie asociovaný so systémom Microsoft Excel a je to jedna zo štandardných ciest prenosu dát dovnútra a von z pracovného hárku. Keď chceme importovať CSV súbor do Excelu treba zvoliť položku import a špecifikovať comma delimited format ako importovaný formát.

Viacero databáz obsahuje nástroje na import a export CSV súborov. Niektoré systémy pri generovaní CSV súborov umiestňujú úvodzovky pred a za reťazec znakov, iné neumiestňujú. Neexistuje presne definovaný štandard [8].

Výhoda používania CSV formátu oproti formátu XML spočíva v jeho menších systémových nárokoch ako aj nárokoch na miesto na disku. Hlavne v prípadoch práce s veľkým objemom dát je výber CSV formátu lepšou alternatívou.

2. Visual Basic

Visual Basic a k nemu príbuzné programy sa v posledných rokoch stali hlavným programovacím prostriedkom na platforme produktov firmy Microsoft. Rôzne mutácie tohoto programovacieho jazyka sa používajú nielen pre programovanie samostatných aplikácii, ale aj sieťových aplikácii a tvorbu makier v balíku programov Microsoft Office (Visual Basic for Aplication). Tento programovací jazyk sa taktiež používa pri programovaní internetových aplikácii, spúšťaných na strane serveru i klienta.

2.1. Vývoj programovacieho jazyka Visual Basic

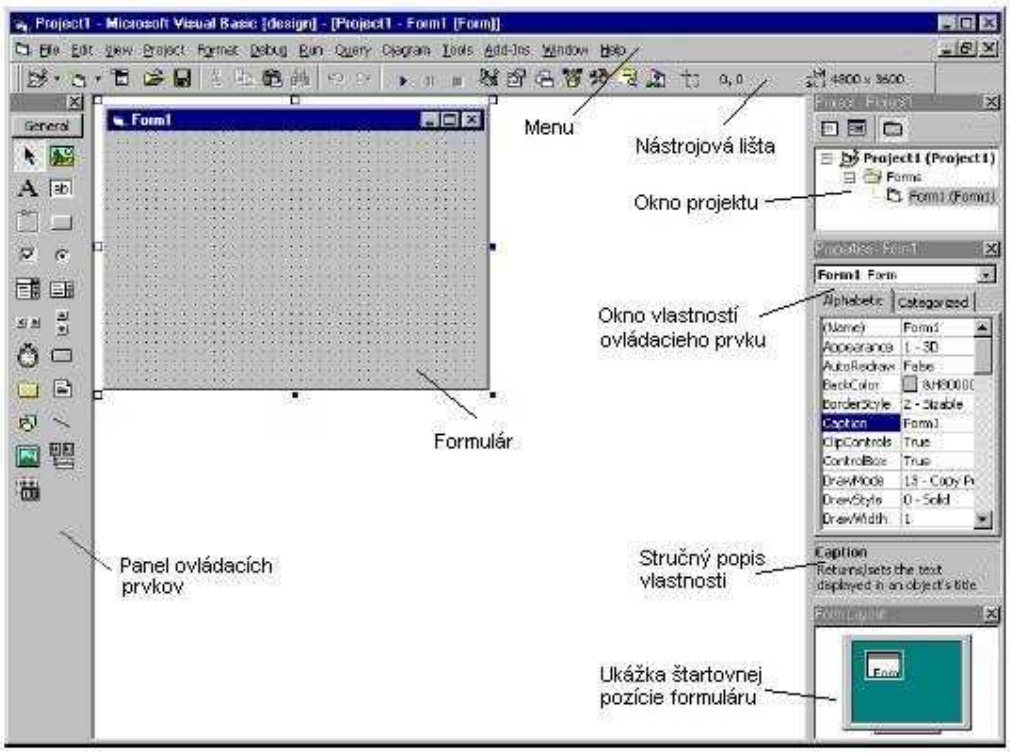
Visual Basic sa vyvinul z jazyka BASIC, známeho už dávno pred nástupom osobných počítačov. Za otca Visual Basicu sa dnes považuje Alan Cooper, ktorý vydal program pod menom Ruby. Koncepcia, ktorú navrhol, sa okrem užívateľov zapáčila aj Microsoftu a v roku 1991 splynula spolu s QuickBasicom do produktu menom Visual Basic. Už vtedy bolo jasné, že tento akt spôsobí doslova prevrat v celom vývoji programátorských nástrojov a ihneď sa stane veľmi populárnym. Ďalšia verzia preto na seba nedala dlho čakať. Okrem pár vylepšení bola citeľne rýchlejšia. O rok nato už na dvere klopala tretia verzia, šírená v štandardnej a profesionálnej edícii. Pracovať sa tu už dalo s databázami ako aj OLE (Object Linking and Embedding) prvkami. Najvýznamnejším krokom však v tomto období bolo integrovanie tzv. Visual Basic for Application do produktov rodiny Microsoft Office. Týmto spôsobom bolo a je dodnes možné písať aplikácie priamo pod MS Wordom či ďalšími produktmi rodiny kancelárskeho balíka od Microsoftu.

Verzia 4.0, vydaná už v roku 1995 samozrejme reagovala na nástup Windows 95. Ponúkala však stále možnosť kompilovať zdrojový kód pre 16-bitové aplikácie. Tu Microsoft urobil ešte ďalší krok ku integrácii – zahrnul syntax jazyka Visual Basic do tzv. VBScriptu, čím ho predurčil aj na využitie pri tvorbe web-stránok. Vo verzii 5.0, vydanej v roku 1997 už zmizla podpora pre 16-bitové systémy. Dnes sa táto verzia ešte stále vyskytuje. Najviac využívaná verzia VB je 6.0, ktorá na svet prišla v roku 1998. V súčasnosti sa už v rámci Visual Studio .NET vyskytuje najnovší Visual Basic .NET [4]

2.2. Užívateľské rozhranie

K obľúbenosti Visual Basicu prispela hlavne jednoduchá syntax, ľahká tvorba užívateľského rozhrania a veľká variabilita a flexibilita jazyka. Aplikácie vytvorené pod Visual Basicom sa graficky úplne zhodujú s prostredím operačného systému Windows. Tak sa každý užívateľ, ktorý pracuje vo Windows, veľmi rýchlo naučí pracovať s novým programom, ktorého ovládanie sa mu zdá intuitívne. Taktiež vlastné prostredie Visual Basicu má známu štruktúru Windowsu, takže nie je obtiažne s ním pracovať (obr. 1.9).

Hlavná výhoda oproti "klasickým" programovacím jazykom ako je C, C++, Pascal, ale aj Basic je, vo Visual Basicu sa programuje vizuálne. To znamená, že sa vytvárajú okná, kreslia sa na ne tlačidlá, menu, panely nástrojov a iné ovládače. Potom sa už len napíše kód, ktorý bude odpovedať napríklad na kliknutie tlačidla. Tento spôsob vytvárania je mimoriadne efektívny a rýchly [1]



Obr. 1.9 Užívateľské prostredie programu Visual Basic 6.0

2.3. Visual Basic for Application (VBA)

Visual Basic for Application je súčasťou Microsoft Visual Basicu, ktorý je zabudovaný do všetkých aplikácií programového balíku Microsoft Office, niektorých iných Microsoft aplikácií a čiastočne implementovaný v niektorých programoch ako AutoCAD a WordPerfect. Jeho výhodou je, že môže byť použitý na ovládanie takmer všetkých funkcií programu, v ktorom je zahrnutý ako aj na manipulovanie s časťami užívateľského rozhrania ako je menu, nástrojová lišta a na pracovanie s užívateľom vytvorenými formulármi a dialógovými oknami. Užívateľské formuláre môžu byť umiestnené priamo v pracovnom hárku ako jeho interaktívna súčasť.

Taktiež môže byť použitý na riadenie jednej aplikácie z druhej (napríklad vytvorenie správy vo Worde z Excelových údajov). Funkčne je bohatý a extrémne flexibilný, ale má niektoré dôležité obmedzenia.

2.3.1. Visual Basic for Application for Excel

Jedná sa o Visual Basic for Application zabudovaný v programe MS Excel. Dá sa sním pracovať veľmi jednoducho. Stačí spustiť nahrávanie makra, ktoré zaznamenáva celú činnosť, ktorú užívateľ prevádza až do skončenia nahrávania. Makro sa potom uloží pod nejakým menom a neskôr stačí toto makro zavolať a automaticky sa vykoná sled krokov, ktoré boli do makra zaznamenané. Má to veľkú výhodu ak je treba spracovávať dáta opakovane tým istým spôsobom.

Skúsenejší užívateľ môže z menu (alebo pomocou Alt + F11) zvoliť možnosť Visual Basic Editor a zobraziť zdrojový kód, ktorý vytvorilo makro. Poprípade môže vytvárať nové zdrojové kódy a využívať všetky možnosti Visual Basicu. Výhodou oproti samostatnému Visual Basicu je, že užívateľ môže priamo pracovať s konkrétnymi pracovnými hárkami, alebo konkrétnymi bunkami Excelu. Napríklad pomocou príkazu **Range(„A1:A100“).Select** systém vyberie stĺpec A s bunkami 1 až 100 a vyznačí ich funkciou Select. Funkcia **Cells(čísloStĺpca)(čísloRiadku) = „reťazec”** priradí konkrétnej bunke nejaký reťazec. Podobným spôsobom sa dajú sprístupniť rôzne údaje z tabuliek Excelu.[3]

2.4. Objektovo orientované programovanie

Programovanie vo Visual Basicu je počítané medzi objektovo orientované a udalosťami riadené techniky. Udalosťou sa obvykle rozumie nejaká akcia užívateľa alebo systému pri behu programu. Reakcia je potom naprogramovaná ako udalostná procedúra objektu

Programátor môže používať veľké množstvo preddefinovaných objektov ako sú formuláre, textové polia, príkazové tlačidlá, menu, popisky a veľké množstvo ďalších objektov. Súhrnne tieto objekty nazývame ovládacie prvky. Objekt je v terminológii jazyka Visual Basic spojenie údajov a programového kódu. Objekty väčšinou nemajú vizuálne rozhranie. Každý objekt má svoje vlastnosti, metódy a udalosti. Vlastnosti sú charakteristiky objektu. Metódy sú akcie, ktoré objekt môže vykonať.

Väčšina objektov má schopnosť rozpoznávať rôzne udalosti, pre ktoré môžeme napísať nejakú odpoveď. Každý objekt má vo Visual Basicu vlastnosť Name, ktorá ho jednoznačne identifikuje.

Každý ovládací prvok má definované vlastnosti, metódy a udalosti. Tieto vlastnosti udávajú výhľad a chovanie ovládacieho prvku v aplikácii [3].

3. Dátová komunikácia

3.1. User development Environment

Jedná sa o prostredie v ktorom užívateľ môže vyvíjať a spúšťať vlastne vytvorené programy pre získavanie a nastavovanie procesných dát. Prostredie zabezpečuje v systéme podporu pre vyvíjanie užívateľom špecifikovaných programov, takže užívateľ môže vytvárať vlastné programy na prístup k funkciám HIS. Na vývoj užívateľských programov môže byť použitý Microsoft Visual Basic 6.0 alebo Microsoft Visual Basic.NET. Keďže pomocou VB.NET sa nedá vytvoriť ActiveX riadenie, odporúča sa použiť Visual Basic 6.0. Pri vývoji užívateľských programov, programovanie bude oveľa efektívnejšie pri použití knižnice na prístup k dátam uloženým v HIS.

Pretože user development environment je funkcia zahrnutá v základných operačných a monitorovacích funkciách, nie je potrebné financovať ďalšie oddelené programy na spúšťanie užívateľského programu v HIS. Ak však užívateľský program bude pristupovať k procesným dátam cez OPC server alebo DDE server, je potrebné vlastniť "Exaopc OPC Interface Package (for HIS)" alebo "Open Data Interface Package for DDE"

3.1.1. Typy užívateľských programov

Užívateľské programy v HIS sú rozdelené do dvoch kategórií:

Spúšťateľné programy

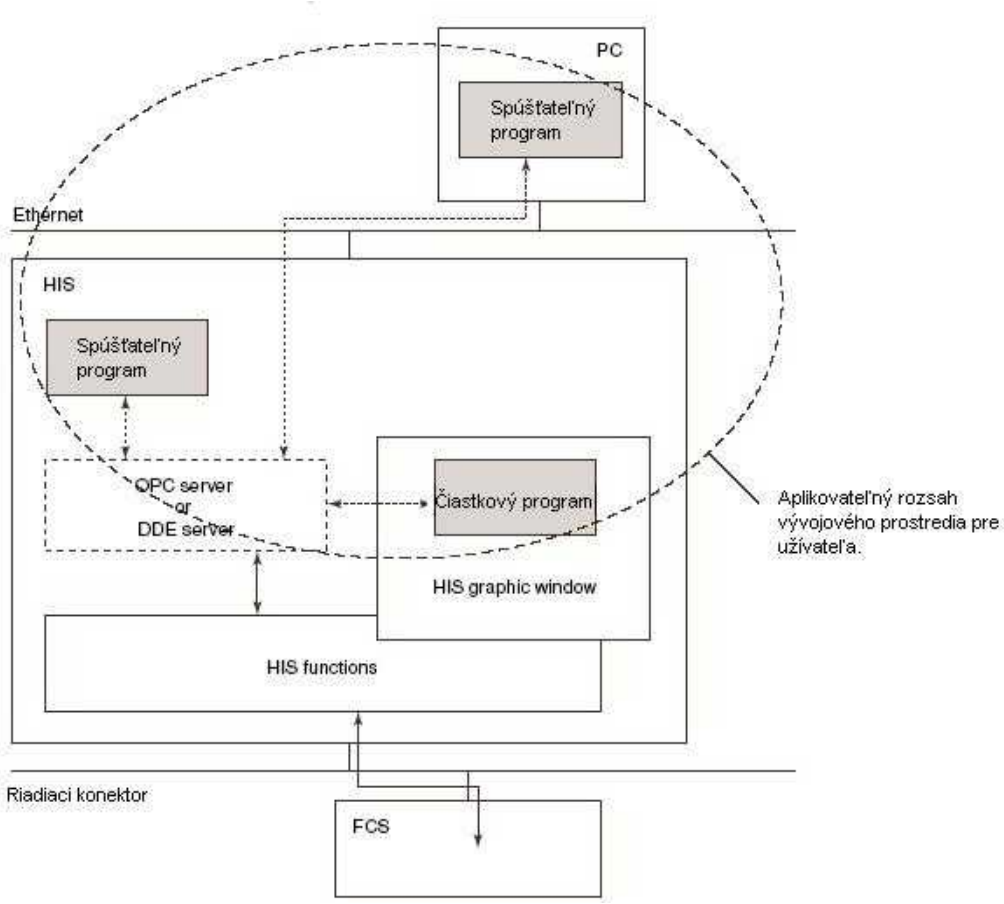
Spúšťateľné programy sú užívateľské programy ktoré bežia nezávisle na funkciách HIS. Čo primárne nastáva medzi týmito programami a HIS je výmena dát cez OPC server alebo DDE server. Taktiež v prípade spúšťateľných programov bežiacich v HIS platí, že nejaká udalosť, ktorá nastane v HIS, môže pôsobiť ako impulz na štart daného programu.

Spúšťateľný program je možné spustiť aj na iných PC, ktoré sú pripojené cez vnútornú sieť a môžu takto pristupovať k dátam. Prípona týchto programov je štandardne ".EXE"

Part Program(Čiastkové programy)

Čiastkové programy sú programy ktoré sú prepojené k HIS grafickému oknu a spustené. Tento užívateľský program je priradený pre každé grafické okno. Cesta, akou je program viazaný ku grafickému oknu je nasledujúca: Keď sa zobrazí grafické okno, automaticky sa spustí program. Keď je grafické okno zatvorené, program je taktiež ukončený. Takto vytvorené programy taktiež môžu byť spustené aj v inom softvéri podporujúcom ActiveX Control (MS Internet Explorer, MS Excel, MS Word a ďalšie) Prípona čiastkových programov je ".OCX"

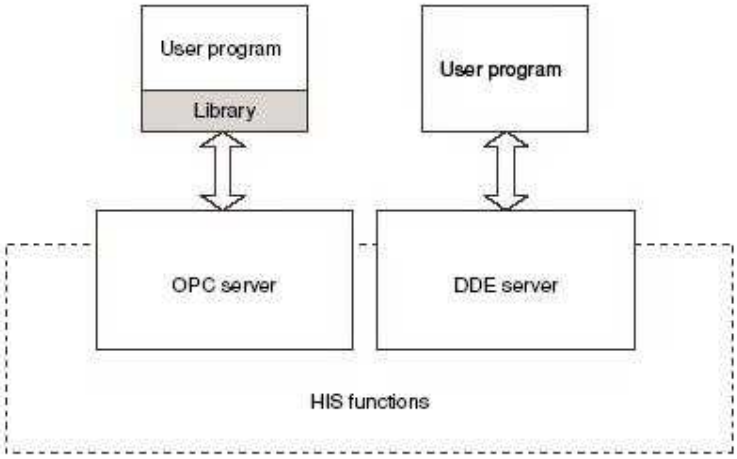
Umiestnenie týchto typov súborov v riadiacom systéme a ich vzájomná komunikácia je zobrazená na obr. 3.1.



Obr. 3.1 Pozícia jednotlivých typov užívateľských programov v riadiacom systéme [6]

3.2. Prístup k dátam

V user development environment existujú dva typy rozhrania dátovej komunikácie, OPC a DDE, pre prístup k procesným a iným dátam z užívateľského programu.



Obr. 3.2 Existujúce typy komunikačného rozhrania pre systém CENTUM [6]

Pre prístup dát cez OPC server môže byť použitá knižnica (obr. 3.2). Pri použití knižnice sa vývoj užívateľského programu stáva oveľa jednoduchším.

3.2.1. Prístup cez knižnicu

ActiveX Control je prístupný ako knižnica zabezpečujúca v užívateľskom programe vývojové prostredie. Knižnica uľahčuje čítanie a zápis procesných dát. ActiveX Control je všestranná softvérová časť, ktorá môže byť použitá v rôznych vývojárskych prostrediach. ActiveX Control pracuje cez OLE. Zmes aplikácií môže byť vyvinutá kombinovaním rôznych ActiveX Control alebo kombinovaním ActiveX Control s iným softvérom.

3.2.2. Prístup cez OPC

OPC je vlastne OLE (Object Linking and Embedding) pre riadenie procesov. OLE je jedna z rozšírene používaných metód pre výmenu dát pre MS Windows application.

OPC server bežiaci na HIS podporuje užívateľské programy na čítanie a zápis dát. OPC je rozhranie špeciálne navrhnuté pre komunikáciu medzi zariadeniami riadenia procesov. OPC je založené na COM(Component Object Model) technológii.

User development environment podporuje získavanie a nastavenie aktuálnej hodnoty procesnej veličiny pomocou OPC. Rozhranie na prístup k dátam je zabezpečené ako knižnica Visual Basicu. Použitím tejto knižnice, užívateľ môže pristupovať k dátam bez komplikovaných procedúr na komunikáciu s OPC serverom. Prístup je taktiež zabezpečený z akejkoľvek OPC-kompatibilnej aplikácii bežiacej pod systémom Windows a nie je potrebné vytvárať špeciálne aplikácie.

V OPC je aplikácia žiadajúca informácie označovaná ako OPC klient a aplikácia poskytujúca informácie je OPC server. OPC server operujúci na HIS zabezpečí prístup k rôznym typom dát v FCS a HIS pre OPC-kompatibilné aplikácie systému Windows. Navyše je tu aj možnosť zaznamenať alarmy, ktoré nastanú v FCS.

OPC interface pozostáva z dvoch typov serverov:

Data Acces (DA) server - Číta a zapisuje alebo nastavuje aktuálnu hodnotu procesnej veličiny použitím ID prvku ako identifikátora dát.

Alarms & Events (A&E) server - Ohlasuje alarmy a udalosti v procesnom poli, ktoré nastanú asynchrónne.

3.2.3. Prístup cez DDE

DDE (Dynamic Data Exchange) je všestranná metóda pre výmenu dát používana systémom Windows. Existuje mnoho aplikácii, ktoré spolupracujú s DDE. DDE server v HIS zabezpečuje čítanie a zápis procesných dát.

Funkcie, ktoré môžu byť použité v user development environment pomocou DDE komunikácie sú zobrazene v tabuľke 3.1.

Tab 3.1 Popis dát dostupných pre DDE komunikáciu a typ prístupu aký poskytujú [6]

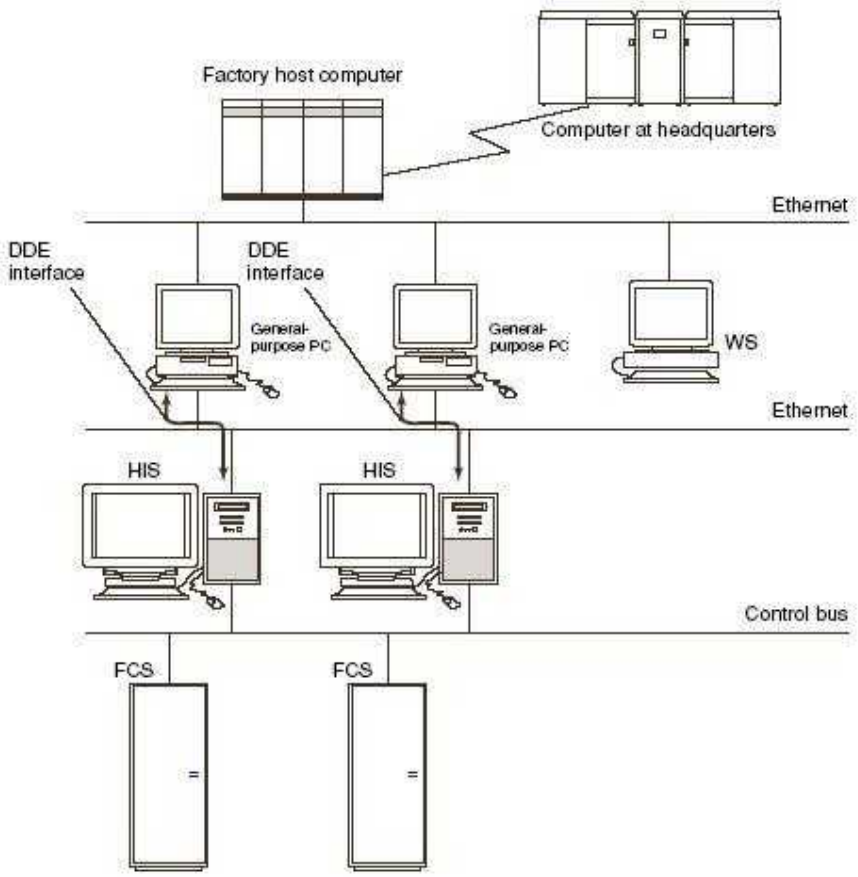
Classification	Data accessed	Function
Process data	Tag data (present value)	Read and write
	Historical trend	Read
	Historical message	Read
	Closing data	Read
Engineering data	Tag information	Read
	Tag list	Read
	List of data subject to trend acquisition	Read
	List of data subject to closing processing	Read

DDE rozhranie pre CENTUM je Yokogawou špecifikované rozhranie, ktoré používa DDE vo Windows prostredí. Toto rozhranie je použité na prístup k rôznym HIS a FCS dátam z DDE-kompatibilných aplikácií vo Windows cez DDE server spúšťaný v HIS.

DDE interface pozostáva zo serveru, ktorý poskytuje rôzne HIS a FCS dáta a z bežného rozhrania použitého na prístup k serveru. DDE rozhranie môže prevádzať operácie v HIS.

DDE je jedna zo štandardných komunikačných metód medzi aplikáciami v prostredí Windows. Je možné prenášať dáta medzi viacerými Windows aplikáciami používaním zdieľanej pamäte, a porovnávať dáta cez odkazy. Aplikácie, ktoré žiadajú dáta použitím DDE rozhrania sú označované ako DDE klienti, a tie ktoré poskytujú dáta sú DDE servre.

DDE rozhranie použité systémom CENTUM používa Net DDE funkcie, ktoré sú vyvinuté firmou Yokogawa Electric Corporation.



Obr. 3.3 Umiestnenie DDE serveru v riadiacom systéme [6]

3.2.4. Umiestnenie DDE serveru.

V DDE rozhraní poskytovanom od Yokogawy, DDE server beží na HIS a DDE klient beží na HIS alebo bežnom PC (obr. 3.3). Ako DDE klient môžu byť použité rôzne komerčné DDE-kompatibilné Windows aplikácie alebo aplikácie vytvorené vo Visual C++ alebo Visual Basicu. DDE klient môže pristupovať k procesným alebo staršie zapísaným dátam cez DDE server.

3.2.5. Výhody použitia DDE

Monitorovanie z inej ako riadiacej miestnosti.

Manažér alebo operátor môžu pristupovať k rôznym HIS a FCS dátam cez počítač, ktorý sa môže nachádzať aj mimo riadiacej miestnosti. To umožňuje monitorovať aktivity ďalej ako v riadiacej miestnosti.

Automatické spracovanie a upravovanie dát.

Použitím DDE klientského programu ako napríklad makro, rôzne dáta môžu byť sprístupnené sekvenčne a potom editovaný výstup v špecifický čas. Pomocou tejto funkcie môžu byť automaticky vytvorené denné a mesačné referácie.

Vytváranie tabuliek a grafov

Rôznorodé údaje prenesené DDE klientovi môžu byť transformované do rôznych formátov tabuliek a grafov. Táto možnosť uľahčuje grafickú interpretáciu dát. Napríklad z trendovej čiary sa dá získať tabuľka jednotlivých údajov.

Podpora analýzy dát

Analýza dát je prevádzaná jednoducho použitím vyhľadávacích a triediacich funkcií pracovného hárku. Napríklad zistenie frekvencie výskytu nejakého javu môže byť spracované vyhľadaním procesných správ a ich zoradením podľa ID čísla.

3.2.6. Typy prístupu k dátam cez DDE

DDE podporuje tri typy prístupu k dátam: Cold link, Warm link a Hot link. Yokogawa DDE rozhranie podporuje iba Cold link a Hot link.

Cold link

Dáta sú prenášané z DDE serveru iba ak DDE klient pošle požiadavku na prenos potrebných údajov.

Hot link

Dáta sú automaticky prenášané z DDE serveru DDE klientovi hneď ako sa zistí zmena linkovaných údajov. Server pristupuje k dátam v špecifikovaných časových intervaloch a porovnáva s hodnotami údajov prenesených v poslednom kroku. Dáta na porovnanie nie sú zobrazené v HIS ale sú držané vo vnútri FCS.

4. Vytvorenie interface pre prenos konfiguračných údajov

Základnou myšlienkou vytvorenia interface bolo uľahčenie zadávania konfiguračných údajov, pretože táto činnosť patrí medzi najzdĺhavejšie v celom procese návrhu riadenia. Konfiguračné údaje sa od zákazníka dodávajú v databázach, najčastejšie v xls formáte, teda v databáze programu MS Excel. Výhodou je prehľadnosť údajov aj fakt, že program MS Excel ovláda väčšina užívateľov.

Údaje sú v databáze identifikované pomocou názvu tagu (tag name) a systémovej adresy, podľa ktorej sa dá jednoznačne určiť, kam bude daný údaj v systéme umiestnený. Ďalej sú ku každému údaju pripojené ďalšie doplňujúce informácie. Niektoré treba do systému taktiež nastaviť (napr. dolné a horné ohraničenia, jednotky), iné slúžia len pre bližšiu špecifikáciu údaju (niektoré komentáre, popisy stavu).

Požiadavkou bolo preniesť tieto údaje z databázového tvaru do systému CENTUM CS3000, umiestniť ich na správne miesto spolu s niektorými ich doplňujúcimi údajmi a zabezpečiť funkčnosť systému a jednotlivých komunikačných väzieb.

Pri zmene údajov v systéme CENTUM by užívateľ musel zase manuálne prepisovať jednotlivé zmenené údaje v Exceli, aby boli totožné. Preto som vytvoril taktiež interface na prenos údajov spätne zo systému CENTUM do Excelu. V tretej časti je vytvorené rozhranie na vytvorenie špecifikovanej Excel tabuľky zo zakreslenej schémy v Control Drawingu.

V štvrtej časti bolo požadované preniesť konfiguračné údaje z Excelu do Control Drawingu, pričom by sa automaticky vytvorila požadovaná schéma aj s názvami a komunikáciami jednotlivých blokov.

4.1. Predpokladané riešenie.

V prvej fáze riešenia som sa zameral na vytvorenie komunikácie pomocou DDE rozhrania (kap. 3.2), keďže CENTUM CS3000 podporuje niektoré typy DDE rozhrania (kap. 3.2.6).

Vytvoril by sa užívateľský spúšťateľný súbor, ktorý by špecifikoval požadované údaje a vytvoril rozhranie na prenos údajov pomocou DDE. DDE server spustený v HIS by poskytoval požadované, užívateľom špecifikované údaje, ktoré by sa potom zapisovali do Excelu. V druhom prípade by sa vytvoril server poskytujúci informácie z Excelu pre CENTUM.

Komunikácia by prebiehala automaticky, takže každá zmena údaju v Exceli by sa okamžite preniesla aj do CENTUMu.

Po preštudovaní dostupnej literatúry [6] sa však táto možnosť javila ako nepravdepodobná. Všetky informácie a príkazy popísané v literatúre slúžia iba na získanie informácií o danom, už existujúcom systéme, o hodnotách jednotlivých veličín, a tagov. Len malé percento týchto údajov sa dá upravovať a nastavovať, väčšinou ide len o údaje na čítanie (tab 3.1) DDE rozhranie teda získa aktuálnu hodnotu nejakého tagu, ak však tag neexistuje, týmto spôsobom sa ani nevytvorí. Preto podľa mojich poznatkov nie je možné nakonfigurovať systém cez DDE rozhranie.

4.2. Použité riešenie

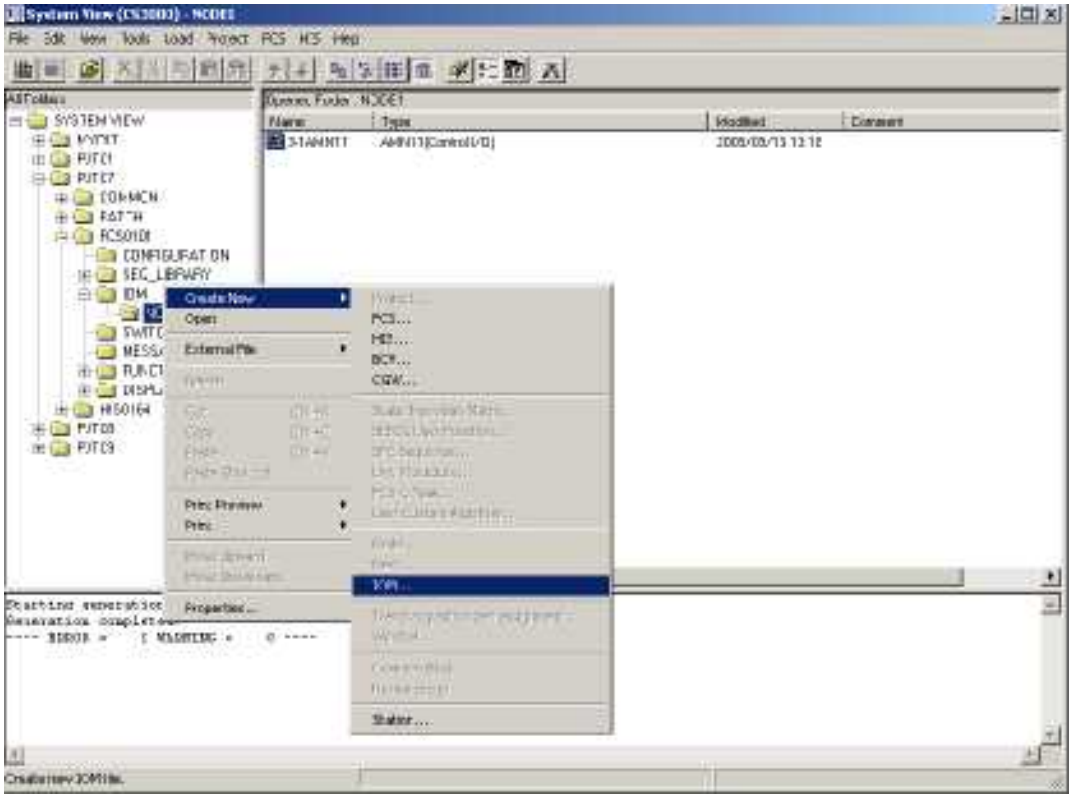
V prvom kroku bolo potrebné zistiť kam CENTUM ukladá konfiguračné údaje. Každá časť prostredia v systéme CENTUM (každá schéma, tabuľka) má priradený svoj vlastný súbor. Tento súbor je typu edf. Keďže tieto súbory boli kódované a nedali sa otvoriť žiadnym dostupným programom, bolo potrebné nájsť inú cestu prístupu k údajom. A tou bola funkcia export a import súboru (kap.1.6). Táto funkcia umožňovala vytvoriť ďalšie typy súborov, ktoré boli lepšie použiteľné. Najlepšie využitie mal súbor typu CSV (kap. 1.6.1), ktorý bol vlastne textový dokument s charakterom databázy. V tomto súbore bolo potrebné podrobne preskúmať štruktúru zápisu dát, zistiť aká zmena nastane v súbore po zmene v CENTUMe a takýmto spôsobom zistiť formu vytvárania tohto súboru a následne sa pokúsiť takýto súbor vytvoriť pomocou údajov zapísaných v Exceli.

4.3. Prenos údajov zo systému CENTUM CS3000 do MS Excel

4.3.1. Príprava systému CENTUM CS3000 na prenos údajov

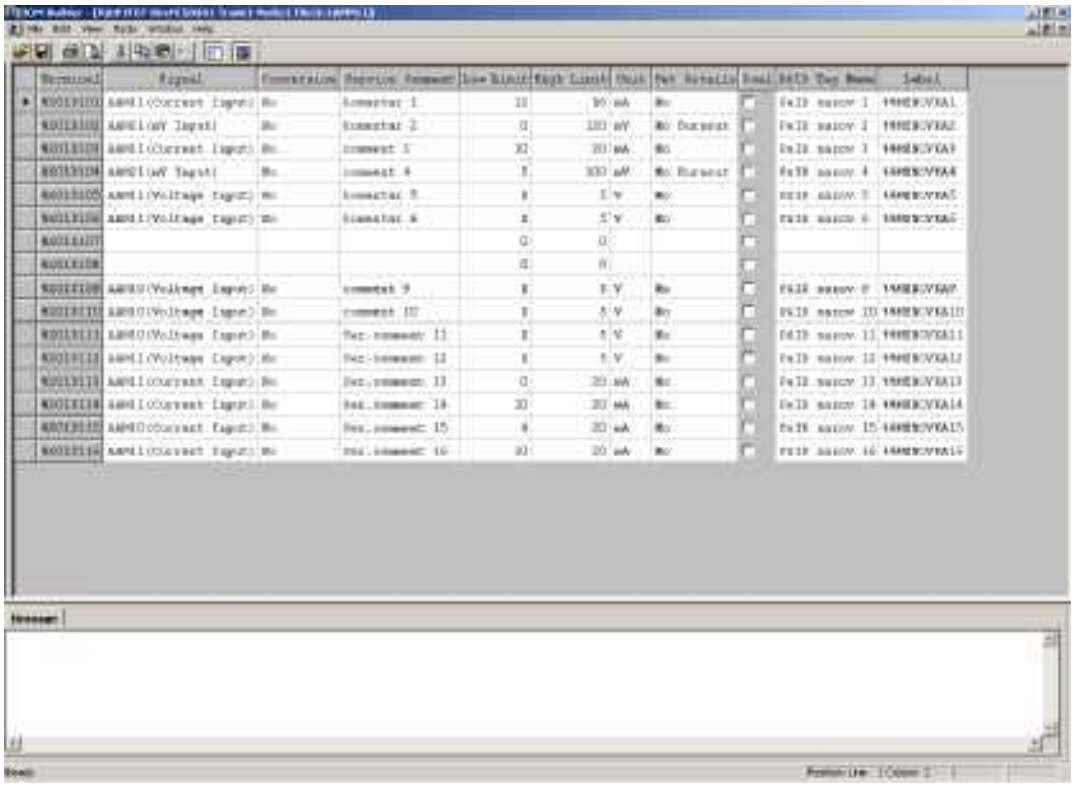
Údaje budú prenášané z tabuliek vytvorených v CENTUM. Každá táto tabuľka reprezentuje jednu procesnú kartu, vstupno-výstupný modul. Najprv teda bude vytvorená táto procesná karta.

V systém view nájdeme v stromovej štruktúre adresár [PJTxx] [FCSxxxx] [IOM] [NODExx] klikneme pravým tlačidlom myši a zvolíme Create new IOM (obr. 4.1)



Obr. 4.1 Vytvorenie nového vstupno-výstupného modulu.

Popis vytvárania vstupno-výstupných modulov a uzlov pre tieto moduly je podrobnejšie popísaný v kap. 1.2. Ak už karta, z ktorej údaje potrebujeme prečítať existuje, použije už túto vytvorenú. Keďže k dispozícii nebol žiadny reálny proces s vytvorenými procesnými kartami, kvôli demonštrácii som vytvoril novú kartu a vyplnil v nej jednotlivé údaje (obr. 4.2).



Obr. 4.2 Vytvorené procesná karta s vy plnenými konfiguračnými údajmi.

Vytvorená procesná karta obsahovala údaje:

- Terminal – je to systémová adresa, ktorú automaticky vytvára systém podľa umiestnenia karty v systéme, čísla FCS, v ktorej je procesná karta vytvorená, čísla jednotlivých tagov a čísla domény. Podrobnejší popis vytvárania týchto adries je v kap. 1.3.2
- Conversion – parameter udáva, ako sa má spracovať vstupný signál pre jednotlivé tagy. Môžeme vybrať jednotlivé druhy konverzie vstupného signálu ako sú definované zosilnenie, zoslabenie alebo vybrať možnosť bez konverzie.
- Service Comment – komentár, ktorý slúži iba na bližšiu identifikáciu systému. Umožňuje prehľadnejšie orientovanie v systéme, vysvetľuje na čo slúži daný tag.
- Low Limit – dolná hranica rozsahu operácií príslušného tagu.
- High Limit – najvyššia možná hodnota príslušného tagu.
- Unit – fyzikálna jednotka, s ktorou príslušný tag pracuje a v ktorej je udávaná jeho hodnota

- P&ID Tag name – sú vnútorne používané(nezobrazované) názvy tagov, bližšie popísané v kap. 1.3.3
- Label – menovka tagu, predstavujúca jeho adresu, vďaka ktorej je zabezpečená komunikácia medzi tagmi.

Po vyplnení údajov sa tieto uložia do systémového súboru typu edf, ktorý prislúcha danému zvolenému typu procesnej karty.

4.3.2. Export súboru do CSV formátu a jeho rozbor

Keďže, ako už bolo spomenuté, systémové súbory systému CENTUM typu edf sa nedali prečítať žiadnou dostupnou aplikáciu, bolo potrebné pomocou funkcie export (kap. 1.6) vytvoriť CSV súbor, s ktorým sa už dalo veľmi dobre pracovať. Jednotlivé údaje boli oddelené čiarkami a zoradené do stĺpcov ako v klasickej databáze. Súbor však mal vlastnosti obyčajného txt súboru (obr. 4.3).

```
EPRODUCTS,YOKOGAWA,CENTUM
[illegible]
EDATE,2005/04/14,13:56
WSHEET,1,
Terminal,Signal,Conversion,Service Comment,Low Limit,High Limit,No 1,Se  Details,Dual,PBID Tag Name,Label
%DM8101,1,1,Komentar 1,12,18,nN,1,Q,PBID nazov 1,%%Menovka1,DOPE Yes,SDOP No,
%DM8102,5,1,Komentar 2,4,120,nN,1,W,PBLU nazov 2,%%Menovka2,WOPE Yes,PBEW No,
%DM8103,1,1,Comment 3,-5,100,nN,1,W,PBIN nazov 3,%%Menovka3,INPT=Yes,SIMP=No,
%DM8104,5,1,Comment 4,5,100,nN,1,Q,PBID nazov 4,%%Menovka4,DOPE Yes,PBEW No,
%DM8105,2,1,Komentar 5,5,20,nN,1,W,PBLU nazov 5,%%Menovka5,WOPE Yes,PBEW No,
%DM8106,2,1,Komentar 6,2,2,E,1,1,PBIN nazov 6,%%Menovka6,
%DM8107,,,,,,,,Q,,,
%DM8108,,,,,,,,W,,,
%DM8109,4,,Comment 9,-5,100,E,,W,PBID nazov 9,%%Menovka9,
%DM8110,4,,Comment 10,4,100,C,,D,PBID nazov 10,%%Menovka10,
%DM8111,4,,Ser.Comment 11,25,25,nN,,W,PBLU nazov 11,%%Menovka11,
%DM8112,2,,Ser.Comment 12,18,18,mA,,W,PBIN nazov 12,%%Menovka12,
%DM8113,1,,Ser.Comment 13,0.0,nV,,Q,PBID nazov 13,%%Menovka13,
%DM8114,1,,Ser.Comment 14,10,25,mV,,W,PBLU nazov 14,%%Menovka14,
%DM8115,1,,Ser.Comment 15,18,24,mA,,W,PBIN nazov 15,%%Menovka15,
%DM8116,1,,Ser.Comment 16,16,20,mA,,Q,PBID nazov 16,%%Menovka16,
```

Obr. 4.3 Výpis štruktúry exportovaného CSV súboru

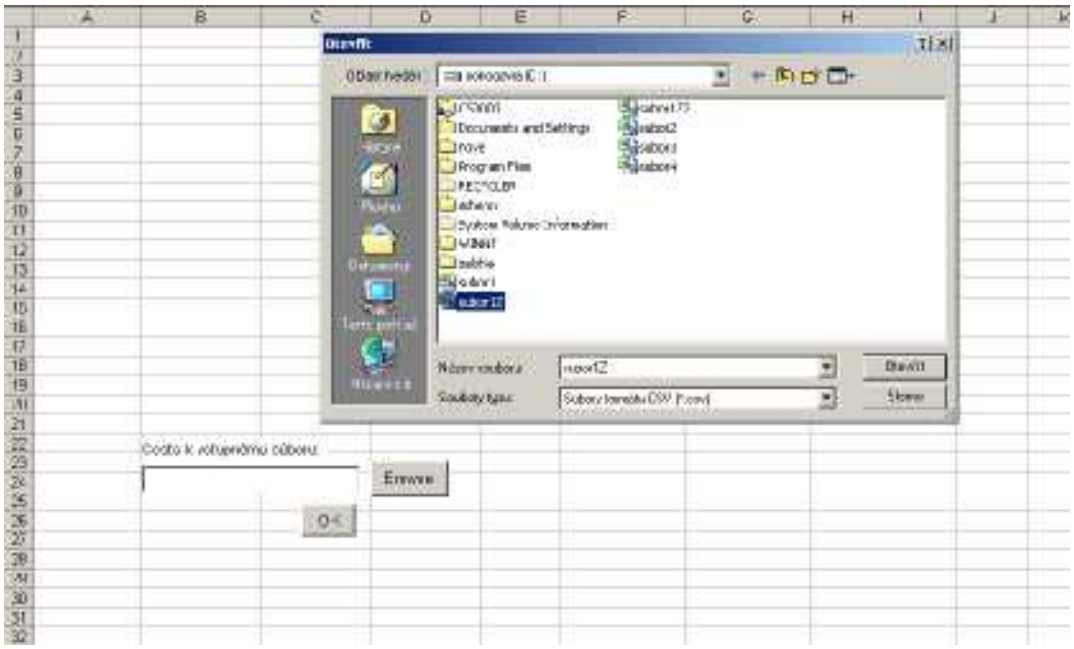
Zo štruktúry je zrejmé, že prvý riadok označuje pôvod exportovaného súboru, názov firmy a softvéru, ktorým bol súbor vytvorený. Ďalšie tri riadky súboru slúžia na bližšiu špecifikáciu procesnej karty, čas a dátum, kedy bola vytvorená a číslo pracovného hárku. V piatom riadku sú zoradené jednotlivé typy údajov v poradí tak, ako sú zobrazované v systéme CENTUM. Tvoria akúsi hlavičku tabuľky s konfiguračnými údajmi. Všetky ostatné údaje už tvoria vlastné konfiguračné informácie, ktoré potrebujeme čítať. Tieto sú zoradené do riadkov tak, že každému riadku prislúcha jeden tag a jeho doplňujúce údaje. Údaje sú usporiadané podľa piateho riadku a každý typ údajov je od iného oddelený čiarkou. Keď niektorý údaj nie je vyplnený, zapíšu sa len čiarky a medzi nimi nie je žiadny znak.

4.3.3. Tvorba programu pre prenos údajov

Pre vývoj programu pre prenos údajov som si zvolil programovací jazyk Visual Basic (kap. 2) konkrétne Visual Basic for Aplication (kap. 2.3) kvôli jeho priamej komunikácie s aplikáciami systému Windows, hlavne Excelu, do ktorého bolo žiadané zapísať konfiguračné údaje. Program som vytvoril ako súčasť pracovného hárku Excelu. V prvom kroku sa na hárok umiestnia jednotlivé ovládacie prvky vo vhodnom rozložení. Následne sa pre každý prvok vypracuje programový kód.

Popis užívateľského rozhrania

Rozhranie je vytvorené tak, že užívateľ v podstate pracuje iba so systémom MS Excel. Otvorí súbor xls, kde sa na pracovnom hárku nachádzajú jednotlivé ovládacie prvky. Ako prvé sa na hárku nachádza pole pre zadávanie textu s poznámkou: „Cesta k vstupnému súboru.“ Do tohto textového pola sa zadáva cesta a názov vstupného súboru, s ktorým sa bude pracovať, teda z ktorého sa budú čítať dáta. Pre zjednodušenie som vedľa textového poľa umiestnil tlačidlo browse, ktoré je nalinkované na Windows Dialog Box. Po definovaní jeho vlastností sa pomocou neho dá zobraziť adresárová štruktúra a požadovaný súbor jednoducho nájsť na disku prehľadávaním adresárov (obr. 4.4). Po vybratí požadovaného súboru sa jeho názov aj cesta k nemu automaticky vložia do textového poľa pre cestu k vstupnému súboru. Keď je zadaný vstupný súbor, stačí kliknúť na tlačidlo OK a údaje budú automaticky prenesené do tabuľky Excelu.



Obr. 4.4 Užívateľské rozhranie programu na prenos dát z CENTUMu do Excelu

Systém práce vytvoreného programu

Programy vo Visual Basicu sa tvoria ako sled udalostí spojených s jednotlivými ovládacími prvkami. Vytvorený program obsahoval dve takéto udalosti:

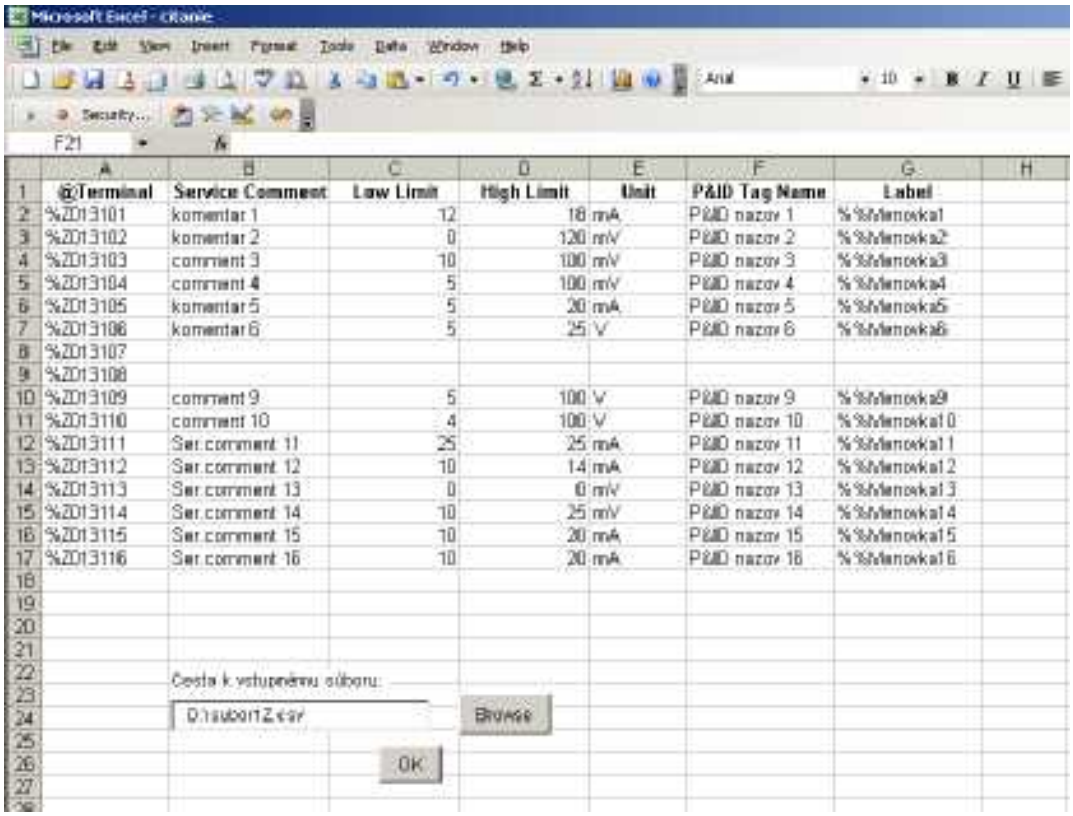
- otvorenie dialógového okna po kliknutí na tlačidlo **Browse**
- spustenie hlavného programu po kliknutí na tlačidlo **OK**

Program pracuje so súborom exportovaným systémom CENTUM. Adresu tohto súboru program prečíta z textového poľa z hodnoty Text (**TextBox.Text**), do ktorej bola zadaná užívateľom manuálne alebo vložená pomocou Dialog Boxu. Program špecifikovaný súbor otvorí iba na čítanie. Keďže som pracoval s CSV súborom, najvhodnejšie bolo použiť funkciu „**input #čísloSúboru, reťazec**“, ktorá číta dáta zo súboru otvoreného pod číslo „čísloSúboru“ a ukladá ich do premennej reťazec. Výhoda tejto funkcie oproti iným funkciám na čítanie bola, že funkcia **input** čítala všetky znaky, kým nenarazila na čiarku. Takto sa dali vyselektovať jednotlivé položky, ktoré boli práve čiarkou oddelené. Funkcia input je schopná načítať toľko reťazcov, koľko veličín uvedieme v jej zápise. Napr:

Input #1, reťazec1,reťazec2,... reťazecX

Získaný reťazec sa porovnával s vopred definovanými reťazcami, ktoré sú vlastne názvy najdôležitejších konfiguračných údajov. Porovnávanie bolo prevádzané pomocou funkcie **InStr(HľadanýReťazec,PrehľadávanýReťazec“).** Funkcia vracia integer hodnotu, ktorá je pozíciou hľadaného reťazca v prehľadávanom reťazci. Aby boli reťazce zhodné, musela táto funkcia vrátiť hodnotu 1. Ak nastala zhoda reťazcov, program si ukladal pozíciu daného názvu v riadku a súčasne číslo stĺpca, do ktorého bude ukladať daný typ údaju v Excelovskej tabuľke.

V ďalších riadkoch, kde už sú zapísané jednotlivé konfiguračné údaje, sa postupne vyberali do hodnoty „reťazec“ a ak bola pozícia prvku zhodná s pozíciou niektorého názvu údaju, dopísala sa do príslušného stĺpca a príslušného riadku. Adresácia údaju do príslušnej bunky bola zabezpečená príkazom **Cells(StĺpecHárku)(RiadokHárku) = „reťazec“.**



	A	B	C	D	E	F	G	H
1	@Terminal	Service Comment	Low Limit	High Limit	Unit	P&ID Tag Name	Label	
2	%ZD13101	komentar 1	12	18	mA	P&ID nazov 1	%%dlenovka1	
3	%ZD13102	komentar 2	0	120	mV	P&ID nazov 2	%%dlenovka2	
4	%ZD13103	comment 3	10	100	mV	P&ID nazov 3	%%dlenovka3	
5	%ZD13104	comment 4	5	100	mV	P&ID nazov 4	%%dlenovka4	
6	%ZD13105	komentar 5	5	20	mA	P&ID nazov 5	%%dlenovka5	
7	%ZD13106	komentar 6	5	25	V	P&ID nazov 6	%%dlenovka6	
8	%ZD13107							
9	%ZD13108							
10	%ZD13109	comment 9	5	100	V	P&ID nazov 9	%%dlenovka9	
11	%ZD13110	comment 10	4	100	V	P&ID nazov 10	%%dlenovka10	
12	%ZD13111	Ser.comment 11	25	25	mA	P&ID nazov 11	%%dlenovka11	
13	%ZD13112	Ser.comment 12	10	14	mA	P&ID nazov 12	%%dlenovka12	
14	%ZD13113	Ser.comment 13	0	0	mV	P&ID nazov 13	%%dlenovka13	
15	%ZD13114	Ser.comment 14	10	25	mV	P&ID nazov 14	%%dlenovka14	
16	%ZD13115	Ser.comment 15	10	20	mA	P&ID nazov 15	%%dlenovka15	
17	%ZD13116	Ser.comment 16	10	20	mA	P&ID nazov 16	%%dlenovka16	

Obr. 4.5 Vytvorená tabuľka konfiguračných údajov načítaná zo systému CENTUM

4.4. Prenos údajov z MS Excel do systému CENTUM CS3000

4.4.1. Príprava systému na prenos údajov.

Prenos údajov do systému bude spočívať vo vytvorení identického systémového súboru systému CENTUM, takého aký by bol v CENTUM zapísaný, ak by boli vyplnené všetky požadované konfiguračné údaje.

Podobne ako pri prenose údajov z CENTUMu do Excelu, potrebujeme poznať aké druhy vstupno-výstupných modulov budú v riadiacom procese použité. Tieto moduly sú špecifikované na začiatku každého súboru. Riešenie tohto problému mohlo prebiehať tak, že by sa pre každú kartu definovali potrebné údaje v programe a po testovaní a zistení o aký typ procesnej karty ide, by sa príslušné údaje zapísali na začiatok súboru, do ktorého by sa už iba preniesli konfiguračné údaje z Excelu. Nakoľko však typov procesných kariet je veľmi veľké množstvo, navyše každý typ je typický iným usporiadaním jednotlivých konfiguračných údajov, použijeme iný spôsob riešenia.

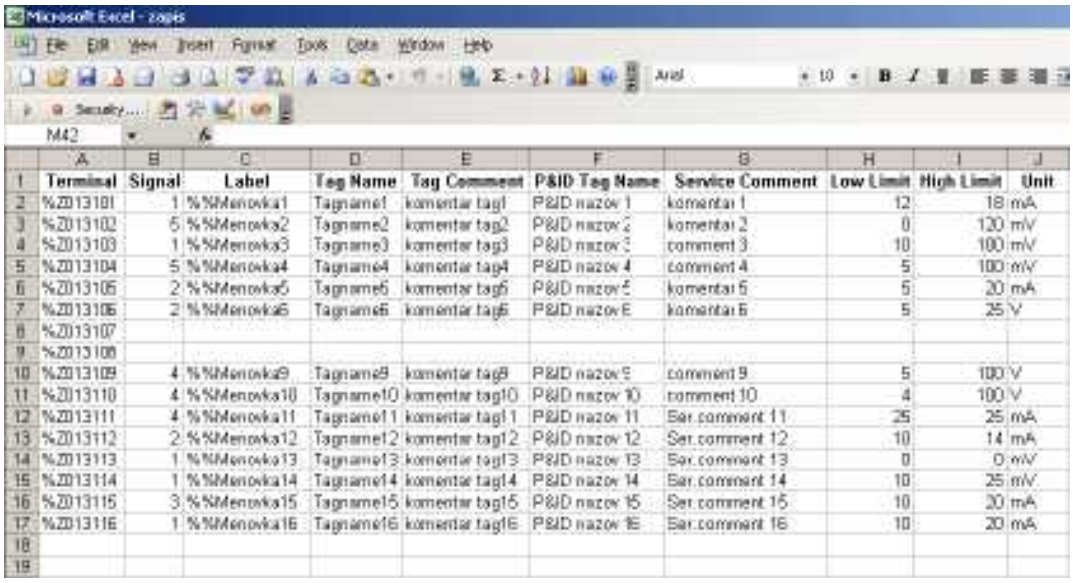
Tak ako v kap. 4.3.1 vytvoríme novú prázdnu procesnú kartu. Systém vytvorí jej konfiguračný súbor, do ktorého zapíše identifikačné údaje procesnej karty, kde je v systéme umiestnená, dátum vytvorenia a zoznam všetkých konfiguračných údajov, ktoré treba vyplniť. Opäť treba tento súbor exportovať do formátu CSV.

4.4.2. Tvorba programu pre prenos údajov

Podstatou riešenia bude vytvorenie súboru určenej procesnej karty vo formáte CSV pomocou funkcie export, vytvorenie programu, ktorý do tohto súboru zapíše požadované konfiguračné údaje (obr. 4.6) a následný import tohto súboru do systému CENTUM.

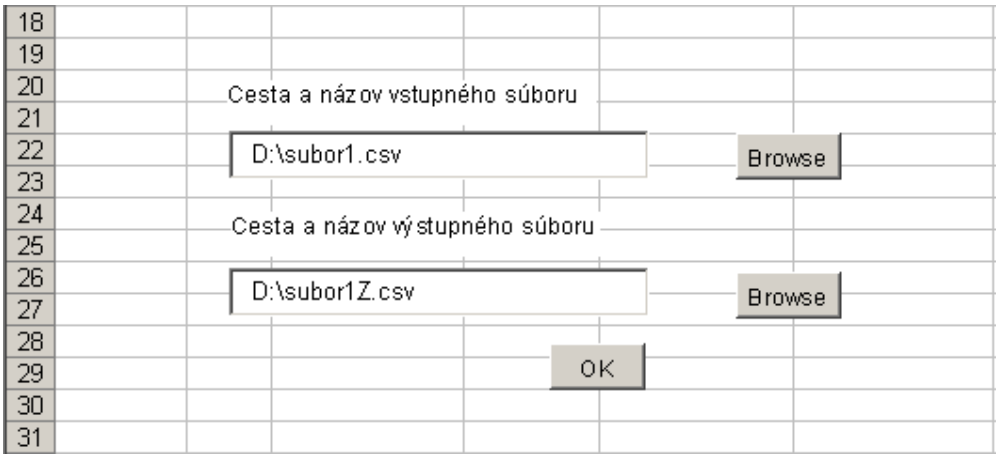
Popis užívateľského rozhrania

Užívateľ, tak ako v prvom prípade, pracuje s formulárom vytvoreným priamo na pracovnom hárku. Formulár obsahuje dve textové polia s popisom (obr. 4.7). Do nich sa zadáva adresa a názov vstupného súboru a výstupného súboru. Vstupný súbor bude súbor, ktorý bol exportovaný z CENTUM ako prázdna procesná karta. Výstupný súbor bude vytvorený a predstavuje súbor procesnej karty s už doplnenými údajmi. Názov vstupného a výstupného súboru sa musí líšiť. Zadávanie adresy súborov je opäť uľahčené vďaka tlačidlám Browse, ktoré otvoria Windows Dialog Box. Prenos údajov je zabezpečený pomocou tlačidla OK.



	A	B	C	D	E	F	G	H	I	J
1	**Terminal**	**Signal**	**Label**	**Tag Name**	**Tag Comment**	**P&ID Tag Name**	**Service Comment**	**Low Limit**	**High Limit**	**Unit**
2	%Z013101	1	%%Menovka1	Tagname1	komentar tag1	P&ID nazov 1	komentar 1	12	18	mA
3	%Z013102	5	%%Menovka2	Tagname2	komentar tag2	P&ID nazov 2	komentar 2	0	120	mV
4	%Z013103	1	%%Menovka3	Tagname3	komentar tag3	P&ID nazov 3	comment 3	10	100	mV
5	%Z013104	5	%%Menovka4	Tagname4	komentar tag4	P&ID nazov 4	comment 4	5	100	mV
6	%Z013105	2	%%Menovka5	Tagname5	komentar tag5	P&ID nazov 5	komentar 5	5	20	mA
7	%Z013106	2	%%Menovka6	Tagname6	komentar tag6	P&ID nazov 6	komentar 6	5	25	V
8	%Z013107									
9	%Z013108									
10	%Z013109	4	%%Menovka9	Tagname9	komentar tag9	P&ID nazov 9	comment 9	5	100	V
11	%Z013110	4	%%Menovka10	Tagname10	komentar tag10	P&ID nazov 10	comment 10	4	100	V
12	%Z013111	4	%%Menovka11	Tagname11	komentar tag11	P&ID nazov 11	Ser.comment 11	25	25	mA
13	%Z013112	2	%%Menovka12	Tagname12	komentar tag12	P&ID nazov 12	Ser.comment 12	10	14	mA
14	%Z013113	1	%%Menovka13	Tagname13	komentar tag13	P&ID nazov 13	Ser.comment 13	0	0	mV
15	%Z013114	1	%%Menovka14	Tagname14	komentar tag14	P&ID nazov 14	Ser.comment 14	10	25	mV
16	%Z013115	3	%%Menovka15	Tagname15	komentar tag15	P&ID nazov 15	Ser.comment 15	10	20	mA
17	%Z013116	1	%%Menovka16	Tagname16	komentar tag16	P&ID nazov 16	Ser.comment 16	10	20	mA
18										
19										

Obr. 4.6 Konfiguračné údaje v tabuľke Excelu určené na prenos do systému CENTUM CS3000



Obr. 4.7 Užívateľské rozhranie programu na prenos dát z Excelu do CENTUM

Systém práce vytvoreného programu

Hlavný program pracuje po kliknutí na tlačidlo OK. Prebehne celá procedúra a údaje sa zapíšu do výstupného súboru. V prvom prípade bol iba otvorený vstupný súbor a prečítané s neho údaje. V tomto prípade však treba do tohto súboru zapisovať. A nakoľko Visual Basic nezapisuje do súboru po jednom znaku resp. reťazci ale len po celých riadkoch, bolo potrebné nájsť iný spôsob zápisu.

Prácu som rozvrhol tak, aby sa so súbormi pracovalo čo najmenej. Keďže sa jedná o nie veľmi objemné súbory, celý súbor je pomocou procedúry otvorený len na čítanie a jeho obsah vložený ako text do premennej typu String. Z tejto premennej je potom pomocou parametra **ReadLine** prečítaný vždy jeden riadok do nejakého string reťazcu. Tento reťazec sa potom skúma a je rozdeľovaný na jednotlivé reťazce prislúchajúce jednej položke (reťazec medzi čiarkami).

S prvými piatimi riadkami nedochádza k žiadnej zmene, iba sa odpíšu do nového súboru, aby sme nezmenili základnú konfiguráciu procesnej karty. V piatom riadku sa pomocou cyklu, tak ako v prvom prípade testuje pozícia jednotlivých názvov konfiguračných údajov. Pozícia je uložená do jednotlivých Integer pozícii. V ďalšom cykle je porovnávaný údaj zapísaný v hlavičke Excel tabuľky s názvom údajov v systémovom súbore. Ak nastane zhoda údajov program si zapamätá číslo stĺpca, z ktorého bude zapisovať jednotlivé údaje. Po týchto počiatočných podmienkach už program pozná štruktúru zápisu v systémovom súbore aj štruktúru zápisu tabuľky v Exceli. Preto môže nastať samotný zápis súboru.

Keďže, ako bolo spomenuté, Visual Basic zapisuje údaje do súboru po celých riadkoch, najprv sa z požadovaných údajov poskladá celý riadok, ktorý sa potom vloží do súboru príkazom **Print #číslo Súboru, Reťazec.**

Riadok sa vytvára pripájaním jednotlivých údajov z tabuľky Excelu v zistenom poradí a pridaním čiarky za každý reťazec. Ak sa niektorý údaj v tabuľke nenachádza, alebo v nej tento typ údaju nie je vôbec zapísaný, vloží sa iba ďalšia čiarka. Týmto sa zabezpečí ukladanie údajov na správne miesto.

Ďalším problémom, ktorý bolo treba prekonať bol prípad, keď niektorý typ údaju obsahoval viacero položiek (napríklad pozícia spínača obsahovala až 4 hodnoty: „**ON,OFF,ON**“. Jednotlivé hodnoty údaju boli oddelené čiarkou, čo robilo komplikácie programu, ktorý podľa čiarky rozoznával pozíciu daných názvov údajov. Súbor CSV oddeľuje reťazce čiarkami a tie, ktoré obsahujú v sebe čiarku navyše v tesná do úvodzoviek (kap. 1.6.1), takže tento typ údajov sa dal jednoducho rozoznať.

Keďže užívateľ nepozoruje žiadnu zmenu v pracovnom hárku ako to bolo v prvom prípade, program o úspešnom prevedení dát informuje Message Boxom. Po vytvorení výstupného CSV súboru, treba tento importovať do systému CENTUM. V systéme

CENTUM sa otvorí tabuľka príslušnej procesnej karty, pre ktorú bol výstupný CSV súbor vytvorený. V menu sa zvolí funkcia import a vyberie sa súbor, ktorý bude importovaný. V tomto prípade ním je programom vytvorený CSV súbor. Kliknutím na OK sa údaje automaticky vložia do tabuľky procesnej karty.

Rozsah programu

Vytvorený program z tabuľky Excelu vyberá a vkladá iba tie údaje, ktoré sa nachádzajú aj v základnej konfigurácii I/O modulu. Program funguje univerzálne pre všetky typy vytvorených I/O modulov.

4.5. Prenos údajov z Control Drawing Buildera systému CENTUM CS3000 do MS Excel

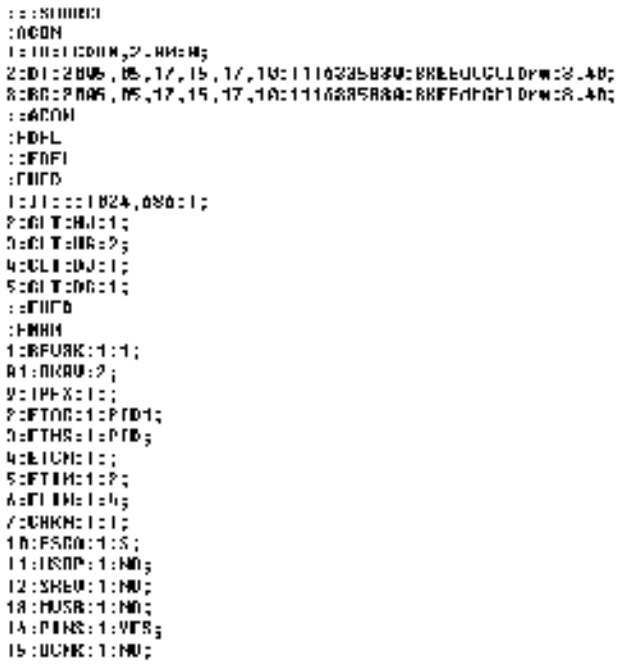
Požiadavka na prenos údajov boľa špecifikovaná ako vytvorenie programu pre čítanie dvoch funkčných blokov, ktoré sú používané v drvivej väčšine schém: PID blok a PVI blok. Na linkovanie vstupných a výstupných údajov bol použitý link **blok PIO**.

PID blok slúži prevádzanie väčšiny základných riadiacich funkcií na zabezpečenie proporcionálne-integračne-derivačného riadenia založeného na odchýlkovej veličine.

PVI blok je Input Indicator Blok slúžiaci na indikáciu vstupnej veličiny.

4.5.1. Príprava systému na prenos údajov

Control Drawing Builder sa používa pri konfigurácii základných riadiacich funkcií FCS a pri vytváraní riadiacich schém. Schémy sú vytvárané vkladaním potrebných blokov, vyplnením ich konfiguračnými údajmi a spájaním pomocou vedenia (kap. 1.4). Každej vytvorenej schéme je priradený systémový súbor edf. Štruktúra zápisu údajov (obr. 4.8) sa podstatne líši od zápisu systémových súborov procesných kariet. Preto aj vytvorený program bude mať inú štruktúru.



Obr. 4.8 Výpis časti štruktúry txt súboru exportovaného z Control Drawing Builder

V prvom kroku som vytvoril ukážkovú schému v Control Drawingu, z ktorej budú čítané údaje (obr. 4.9). Control Draving otvoríme vybratím adresáru [System View] [PJTxx] [FCSxxxx] [Function Block] zo stromovej štruktúry systém view.

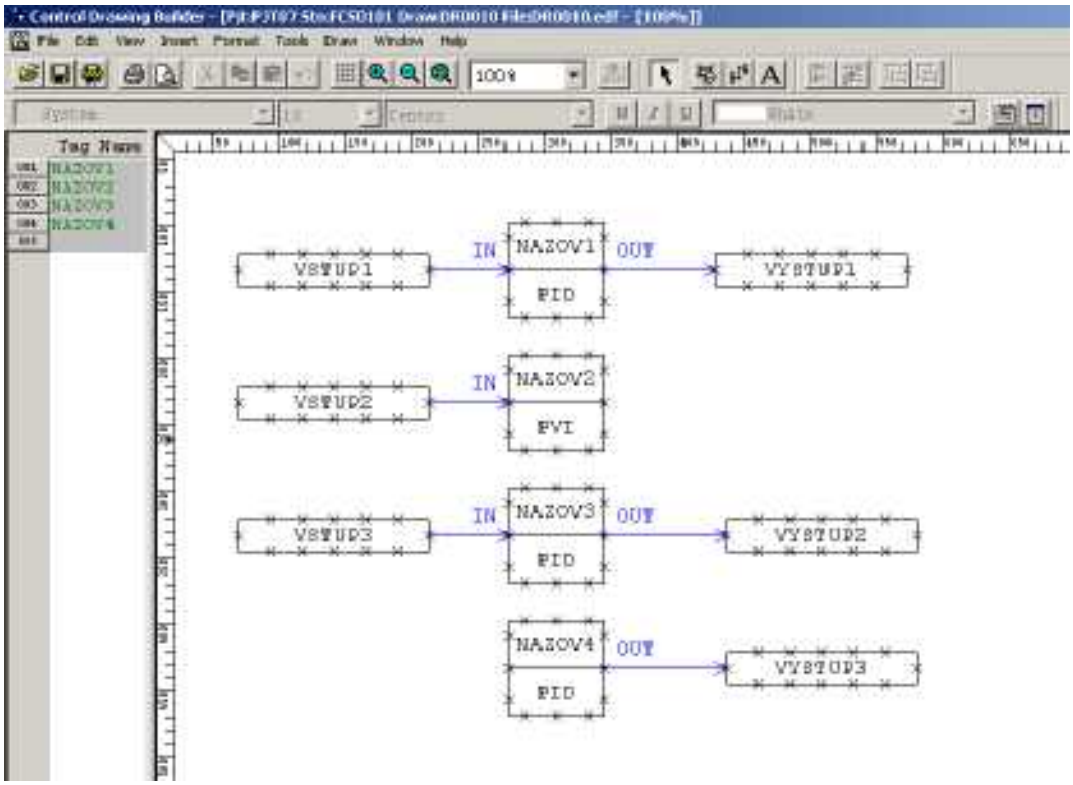
Pri uložení systém najprv vytvorenú schému zkompiluje a keď kompilácia prebehne bez chýb, uloží systémový edf súbor. Tento súbor je potrebné exportovať do vhodného súboru. Control Drawing Builder ponúka pre export formát txt.

4.5.2. Popis funkčnosti programu

Užívateľské rozhranie je totožné s programom na čítanie konfiguračných údajov procesných kariet (kap. 4.3.3). Užívateľ zadá názov vstupného súboru a kliknutím na tlačidlo OK sa spustí program.

Štruktúra exportovaného txt súboru bola tvorená jednotlivými konfiguračnými údajmi, pričom každému údaju bol vyhradený jeden riadok (obr. 4.8). Na začiatku sa nachádzajú údaje slúžiace na identifikáciu schémy, jej umiestnenie v systéme a ďalšie konfiguračné údaje. Ďalšie údaje sú už vymedzené pre jednotlivé bloky. Vždy keď je vytvorený nový blok, do systémového súboru je vložený kód špecifický pre tento blok.

V tomto kóde sa nachádzalo viacej údajov (pre PID blok viac ako 80) a bolo potrebné zistiť, ktoré položky kódu patria názvu bloku, jeho umiestneniu, komentáru, vstupného a výstupného konektora a ďaľších požadovaných údajov. Toto bolo docielené skúmaním zmien v exportovanom súbore po zmenách v schéme Control Drawing Builderi.



Obr. 4.9 Názorná schéma vytvorená v Control Drawing Builderi

Program otvorí vstupný súbor na čítanie a číta z neho jednotlivé riadky. Ak narazí na niektorý z požadovaných údajov, vyberie z riadku reťazec prislúchajúci názvu bloku alebo inému údaju. Tento uloží do príslušného stĺpca a riadku v tabuľke Excelu. Keď sa niektorý blok v schéme nenachádza (napr. PID blok má iba vstup a nemá definovaný výstup), nevloží sa do tabuľky žiadny reťazec. Taktiež musí systém rozoznávať, ku ktorému bloku patrí daný PIO blok a či je vstupom alebo výstupom tohoto funkčného bloku. Po spustení program vytvorí tabuľku s niektorými konfiguračnými údajmi (obr. 4.10). Zvolil som údaje: Názov bloku, jeho typ a komentár a či má blok definovaný vstup alebo výstup v podobe PIO bloku a ich názvy.

	A	B	C	D	E	F
1	**Názov tagu**	**Typ bloku**	**Komentár bloku**	**Input**	**Output**	
2	NAZOV1	PID	Komentar PID bloku1	VSTUP1	VYSTUP1	
3	NAZOV2	PVI	Komentar indikatoru 1	VSTUP2		
4	NAZOV3	PID	Komentar PID2	VSTUP3	VYSTUP2	
5	NAZOV4	PID	Komentar PID 3		VYSTUP3	
6						
7						

Obr. 4.10 Vytvorená tabuľka konfiguračných údajov načítaná z Control Drawing Buildera systému CENTUM CS3000

4.6. Prenos údajov z MS Excel do Control Drawing Buildera systému CENTUM CS3000

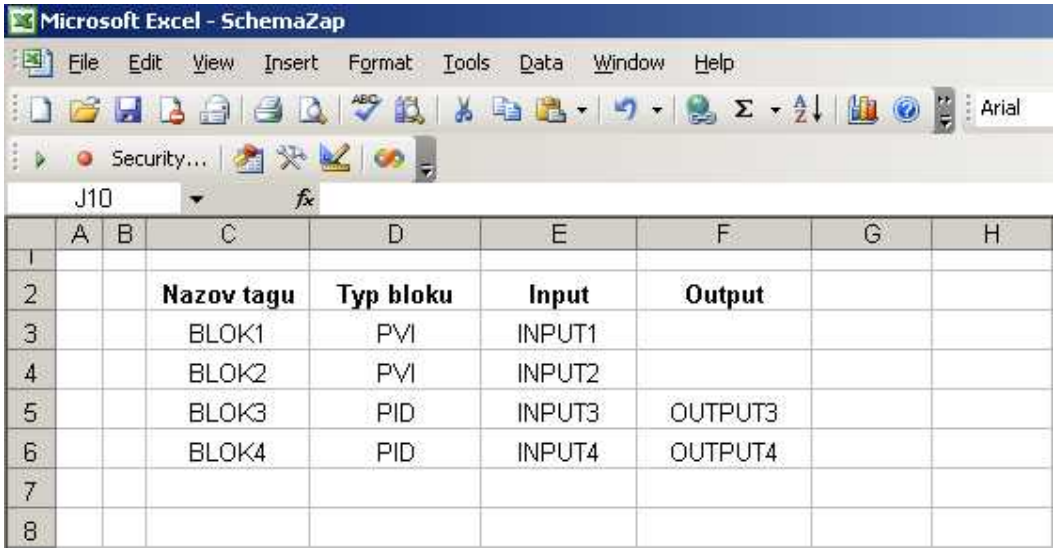
Prenos údajov na vytvorenie schémy z údajov Excelovskej databázy bude pracovať v opačnom smere ako v predchádzajúcom prípade. Bude potrebné vytvoriť program, ktorý do systémového súboru zapíše také údaje, ktoré by zapísal samotný systém pri vytváraní jednotlivých blokov.

4.6.1. Príprava systému na prenos údajov

Podobne ako pri zápise údajov procesnej karty, v systéme CENTUM exportujeme prázdne okno Control Draving Buildera v podobe txt súboru. Systém bude obsahovať konfiguračné údaje danej schémy, s ktorými sa nebude manipulovať.

4.6.2. Systém práce vytvoreného programu

Užívateľské rozhranie je rovnaké ako pri zápise údajov do procesných kariet. Užívateľ musí zadať vstupný súbor (exportovaný txt) a výstupný súbor (súbor, ktorý bude importovaný do Control Drawing Builderu). Kliknutím na tlačidlo OK sa spustí hlavný program.



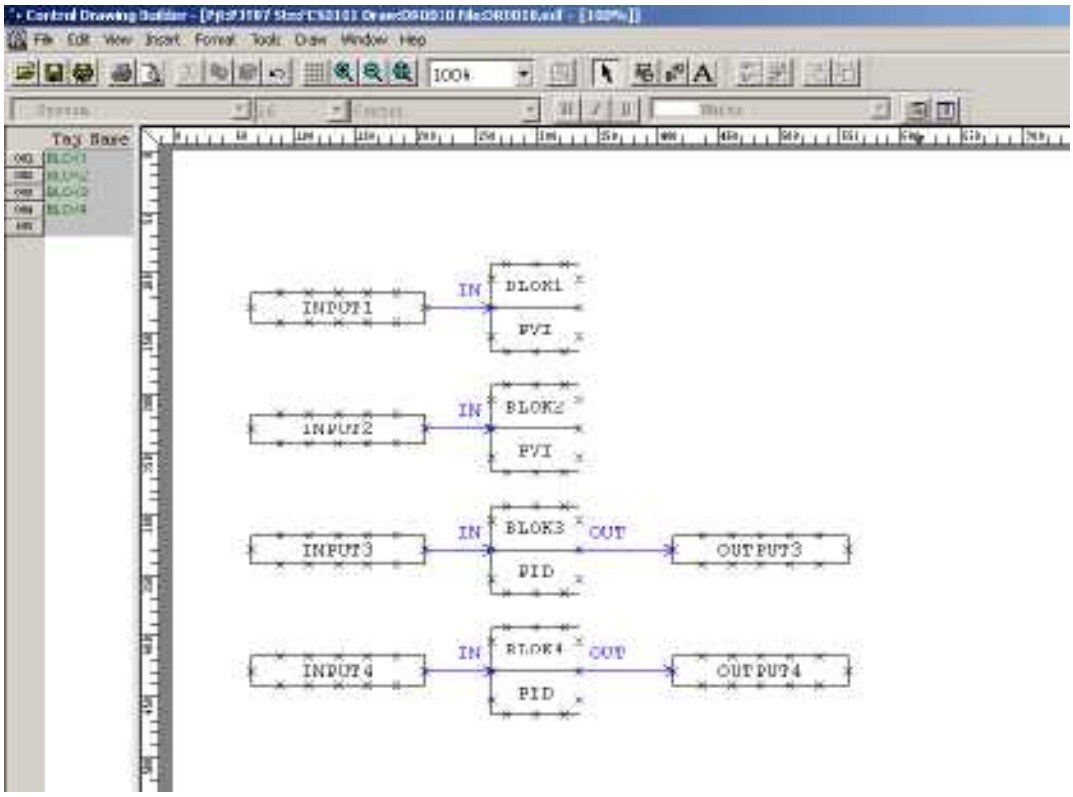
	A	B	C	D	E	F	G	H
2			**Nazov tagu**	**Typ bloku**	**Input**	**Output**		
3			BLOK1	PVI	INPUT1			
4			BLOK2	PVI	INPUT2			
5			BLOK3	PID	INPUT3	OUTPUT3		
6			BLOK4	PID	INPUT4	OUTPUT4		
7								
8								

Obr. 4.11 Konfiguračné údaje v Exceli prenášané do Control Drawing Buildera

Program otvorí vstupný súbor na čítanie a údaje zapisuje do súboru výstupného. Konfiguračné údaje schémy sa iba prepíšu z jedného súboru do druhého. Keďže zapisované údaje prislúchajúcemu jednému typu blokov sú rovnaké, menia sa len niektoré parametre, definoval som si všetky tieto údaje do jedného reťazca. V tabuľke Excelu (obr. 4.11) program zisťuje aký typ bloku sa aktuálne zapisuje a podľa toho zapisuje príslušný reťazec.

Do tohto reťazca sú predtým na určené miesta vložené údaje z iných stĺpcov v Exceli. Pozície týchto údajov sa zisťujú čítaním vstupného súboru a porovnávaním vopred definovaných názvov údajov.

Do súboru treba doplniť niektoré špeciálne reťazce, ktoré tvoria súčasť systémového súboru Control Drawing Builderu. (napr. reťazec „::SOURCE“ slúži na ukončenie kódu celého súboru). Taktiež program musí nájsť miesto, kam sa zapisuje údaj o pripojení jednotlivých blokov ako aj názov pripájacieho terminálu (kap. 1.4), aby sa po prenesení údajov vytvorené bloky správne poprepájali.



Obr. 4.12 Schéma systému vygenerovaná po importovaní vytvoreného txt súboru.

Po správnom prebehnutí programu sa vytvorí txt súbor, ktorého adresa bola vopred zadaná užívateľom. Tento txt súbor treba importovať v danom okne Control Drawing Buildera použitím funkcie import v menu súboru. Po importovaní sa automaticky vygeneruje schéma podľa údajov v konfiguračnej tabuľke v Exceli. Vygenerujú sa jednotlivé PID a PVI bloky a k nim prislúchajúce linkové PIO bloky (obr. 4.12).

Záver

Cieľom tejto diplomovej práce bolo navrhnúť spôsob prenosu konfiguračných údajov z databázy do systému CENTUM CS3000.

Úlohou práce bolo nájsť vhodné metódy zápisu a čítania systémových dát systému CENTUM. V práci boli vytvorené 2 programy na prenos konfiguračných údajov procesných kariet a 2 programy na prenos údajov do Control Drawing Buildera, kde sa z nich vygenerovala riadiaca schéma. Na vývoj programov bol použitý programovací jazyk Visual Basic, konkrétne prostredie Visual Basic for Application. Vytvorené programy boli univerzálne, dokázali spracovať a preniesť údaje pre všetky typy procesných kariet a grafických schém.

V budúcnosti by bolo vhodné rozšíriť vytvorené rozhranie aj o automatický import a export údajov do systému CENTUM, čím by sa väčšmi zjednodušila obsluha vytvorených programov.

Literatúra

[1] David Morkes, Učebnice Visual Basicu 6.0, Computer Press, Praha 2000

[2] http://www.pcserver.sk/programovanie.asp

[3] Visual Basic for Excel 97/2000, University of Bristol, 2004

[4] Pokorný, J., Úvod do Visual Basic .NET, UNIS Publishing, 2001

[5] CENTUM CS3000 hardware, http://www.yokogawa.com/dcs/products/cs3000/hardware/dcs-3k-0301en.htm

[6] CENTUM CS1000/CS3000 User's Manual, 10th Edition, 2003

[7] Schlutz, J., Riadiaci systém CENTUM CS3000, AT&P journal, 4/98 st.42

[8] CSV File Format, http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm

[9] Pokorný, J., Spolupráce aplikací MS Office, Kopp, České Budějovice 1997

Príloha A

Prehľad vstupno-výstupných modulov systému CENTUM CS3000 [6]

Category	Type (Model)	I/O Type	Terminal Number	Range	Raw Data
Analog Input	16-Channel Current Input (AAI141-S)	Current Input	1 to 16	4 to 20 mA	0 to 100%
	16-Channel Current Input; Isolated (AAI143-S)	Current Input	1 to 16	4 to 20 mA	0 to 100%
	8-Channel Current Input; Isolated (ASI133-S)	Current Input	1 to 8	4 to 20 mA	0 to 100%
	8-Channel Current Input; Isolated Channels (AAI135-S)	Current Input	1 to 8	4 to 20 mA	0 to 100%
	16-Channel Voltage Input (AAV141-S)	Voltage Input	1 to 16	1 to 5 V	0 to 100%
	16-Channel Voltage Input (-10 to 10V) (AAV142-S)	Voltage Input	1 to 16	Definable within -10 to 10 V	0 to 100%
	16-Channel Voltage Input; Isolated (AAV144-S)	Voltage Input	1 to 16	1 to 5 V (*1)	0 to 100%
	16-Channel Voltage Input (-10 to 10V); Isolated (AAV144-S)	Voltage Input	1 to 16	Definable within -10 to 10 V (*1)	0 to 100%
	16-Channel Thermocouple/mV Input; Isolated (AAT141-S)	Thermocouple Input	1 to 16	Rated range	Measured Temperature
		mV Input (%)		Definable within -100 to 150 mV	0 to 100%
		Thermocouple Input (V)		-20 to 80 mV	Engineering Unit (V)
	16-Channel Thermocouple/mV Input; Isolated (AST143-S)	Thermocouple Input	1 to 16	Rated range	Measured Temperature
		mV Input (%)		Definable within -100 to 150 mV	0 to 100%
		Thermocouple Input (V)		-50 to 75 mV	Engineering Unit (V)
Analog Input	15-Channel Thermocouple Input; Isolated Channels (MX Compatible) (AAT145-S)	Thermocouple Input	1 to 15 (*2)	Rated range	Measured Temperature
		Thermocouple Input (V)		-20 to 80 mV	Engineering Unit (V)
	12-Channel RTD Input; Isolated (AAR181-S)	RTD Input	1 to 12	Rated range	Measured Temperature
		RTD Input (ohm)		0 to 400 ohm	Engineering Unit (ohm)
	8-Channel RTD/Potentiometer Input; Isolated (ASR133-S)	RTD Input	1 to 8	Rated range	Measured Temperature
		Potentiometer Input		Definable within 0 to 10 Kohm	0 to 100%
		RTD Input (ohm)		Choose from 0 to 650, 0 to 1300, 0 to 2600, 0 to 5400	Engineering Unit (ohm)
	8-Channel Pulse Input (AAP135-S)	Pulse Input	1 to 8	Number of pulse 0 to 65535; Time stamp (1 ms)	Number of pulse (with time stamp)
	16-Channel Pulse Input (PMI Compatible) (AAP149-S) (*3)	Pulse Input	1 to 16	Number of pulse 0 to 65535 Time stamp (1 ms)	Number of pulse (with time stamp)

Category	Type (Model)	I/O Type	Terminal Number	Range	Raw Data
Analog Input and Output	8-Channel Current Input and 8-Channel Current Output (AAI841-S)	Current Input	1 to 8	4 to 20 mA	0 to 100%
		Current Output	9 to 16	4 to 20 mA	-
	8-Channel Voltage Input and 8-Channel Current Output (AAI841-S)	Voltage Input	1 to 8	1 to 5 V	0 to 100%
		Current Output	9 to 16	4 to 20 mA	-
	8-Channel Voltage Input and 8-Channel Current Output (MAC2 Terminal Arrangement) (AAI841-S)	Voltage Input	1,3,...,15 (Odd Number)	1 to 5 V	0 to 100%
		Current Output	2,4,...,16 (Even Number)	4 to 20 mA	-
	4-Channel Current Input and 4-Chnnel Current Output; Isolated Channels (AAI835-S)	Current Input	1 to 4	4 to 20 mA	0 to 100%
		Current Output	5 to 8	4 to 20 mA	-
Analog Output	16-Channel Voltage Output (-10 to 10 V) (AAV542-S)	Voltage Output	1 to 16	Definable within -10 to 10 V	-
	16-Channel Voltage Output (-10 to 10 V); Isolated (AAV544-S)	Voltage Output	1 to 16	Definable within -10 to 10 V	-
	8-Channel Current Output; Isolated (AAI533-S)	Current Output	1 to 8	4 to 20 mA	-
	16-Channel Current Output; Isolated (AAI543-S)	Current Output	1 to 16	4 to 20 mA	-
Analog Input and Output (HART Compatible)	16-Channel Current Input (HART) (AAI141-H)	Current Input	1 to 16	4 to 20 mA	0 to 100%
		HART Variable (*3)	1 to 32	-	Engineering Unit
	16-Channel Current Input; Isolated; HART (AAI143-H)	Current Input	1 to 16	4 to 20 mA	0 to 100%
		HART Variable (*3)	1 to 32	-	Engineering Unit
	8-Channel Current Input; Isolated; HART (AAI133-H)	Current Input	1 to 8	4 to 20 mA	0 to 100%
		HART Variable (*3)	1 to 32	-	Engineering Unit
	8-Channel Current Input; Isolated channels (HART) (AAI135-H)	Current Input	1 to 8	4 to 20 mA	0 to 100%
		HART Variable (*3)	1 to 32	-	Engineering Unit
	8-Channel Current Input, 8- Channel Current Output (HART) (AAI841-H)	Current Input	1 to 8	4 to 20 mA	0 to 100%
		Current Output	9 to 16	4 to 20 mA	-
		HART Variable (*3)	1 to 32	-	Engineering Unit
	4-Channel Current Input, 4- Channel Current Output (HART) (AAI835-H)	Current Input	1 to 4	4 to 20 mA	0 to 100%
		Current Output	5 to 8	4 to 20 mA	-
		HART Variable (*3)	1 to 32	-	Engineering Unit
	8-Channel Current Output; Isolated; HART (AAI533-H)	Current Output	1 to 8	4 to 20 mA	-
		HART Variable (*3)	1 to 32	-	Engineering Unit
	16-Channel Current Output; Isolated; HART (AAI543-H)	Current Input	1 to 16	4 to 20 mA	-
		HART Variable (*3)	1 to 32	-	Engineering Unit

Príloha B

Zdrojový kód programu na čítanie konfiguračných údajov zo systému CENTUM a zápis do Excelu (súbor citanie.xls)

```
Private Sub cmdOK_Click()
Dim retazec, label As String
Dim Pozicia, PoziciaLabel, PoziciaTag,
PoziciaSC, PoziciaLL, PoziciaHL, _
   PoziciaUnit, PoziciaTagName As Integer
Dim i, riadok, stlpec, StlpecTag, StlpecLabel,
StlpecSC, StlpecLL, StlpecHL, _
   StlpecUnit, StlpecTagName As Integer

retazec = txtCesta.Text
Open retazec For Input As #1
riadok = 1
stlpec = 1
i = 1
Do While Not EOF(1)
   Input #1, retazec
   If InStr(retazec, "@Terminal") = 1 Then
      PoziciaLabel = 0
      Cells(stlpec)(1) = retazec
      stlpec = stlpec + 1
      Do While InStr(retazec, "%Z") = 0
         Pozicia = Pozicia + 1
         If InStr(retazec, "Label") = 1 Then
            Cells(stlpec)(1) = retazec
            StlpecLabel = stlpec
            stlpec = stlpec + 1
            PoziciaLabel = Pozicia
         End If
         If InStr(retazec, "P&ID Tag Name") = 1
Then
            Cells(stlpec)(1) = retazec
            StlpecTag = stlpec
            stlpec = stlpec + 1
            PoziciaTag = Pozicia
         End If
         If InStr(retazec, "Service Comment") =
1 Then
            Cells(stlpec)(1) = retazec
            StlpecSC = stlpec
            stlpec = stlpec + 1
            PoziciaSC = Pozicia
         End If
         If InStr(retazec, "Low Limit") = 1 Then
            Cells(stlpec)(1) = retazec
            StlpecLL = stlpec
            stlpec = stlpec + 1
            PoziciaLL = Pozicia
         End If
         If InStr(retazec, "High Limit") = 1 Then
            Cells(stlpec)(1) = retazec
            StlpecHL = stlpec
            stlpec = stlpec + 1
            PoziciaHL = Pozicia
         End If
         If InStr(retazec, "Unit") = 1 Then
            Cells(stlpec)(1) = retazec
            StlpecUnit = stlpec
            stlpec = stlpec + 1
            PoziciaUnit = Pozicia
         End If
         If InStr(retazec, "Tag Name") = 1 Then
            Cells(stlpec)(1) = retazec
            StlpecTagName = stlpec
            stlpec = stlpec + 1
            PoziciaTagName = Pozicia
         End If
         Input #1, retazec
      Loop
   End If
   If InStr(retazec, "%Z") = 1 Then
      riadok = riadok + 1
      Cells(1)(riadok) = retazec
      i = 1
   End If
   If i = PoziciaLabel Then
      Cells(StlpecLabel)(riadok) = retazec
   End If
   If i = PoziciaTag Then
      Cells(StlpecTag)(riadok) = retazec
   End If
   If i = PoziciaSC Then
      Cells(StlpecSC)(riadok) = retazec
   End If
   If i = PoziciaLL Then
      Cells(StlpecLL)(riadok) = retazec
   End If
   If i = PoziciaHL Then
      Cells(StlpecHL)(riadok) = retazec
   End If
   If i = PoziciaUnit Then
      Cells(StlpecUnit)(riadok) = retazec
   End If
   If i = PoziciaTagName Then
      Cells(StlpecTagName)(riadok) = retazec
   End If
   i = i + 1
 Loop
Close #1
End Sub
```

Príloha C

Zdrojový kód programu na zápis konfiguračných údajov do systému CENTUM z tabuľky Excelu (súbor zapis.xls)

```
Private Sub cmdOK_Click()

Const ForReading = 1, ForWriting = 2
Dim f, fs, ts, riadok, a, aTemp, ZapisRiadok, retazec As String
Dim i, j, dlzka, pozicia, poradieLabel, poradieTag, poradieSignal, poradieComment, poradieLL,
poradieHL, poradieUnit As Integer
Dim stlpec, StlpecLabel, StlpecTag, StlpecSignal, StlpecComment, StlpecLL, StlpecHL, StlpecUnit,
RiadokBunka As Integer
Dim StlpecTerminal, poradieTerminal, StlpecTagName, StlpecTagComment, poradieTagName,
poradieTagComment As Integer

If TextBox1.Text = TextBox2.Text Then
   MsgBox "Nazov vstupneho a vystupneho suboru sa musi lisit."
   Exit Sub
End If

If TextBox1.Text = Empty Then
   MsgBox "Musite zadat nazov suboru"
End If
retazec = TextBox1.Text
Set fs = CreateObject("Scripting.FileSystemObject")
Set f = fs.GetFile(retazec)
Set ts = f.OpenAsTextStream(ForReading, 0) 'TristateUseDefault)
retazec = TextBox2.Text
Open retazec For Output As #1
i = 1
j = 1
Do While i < 6
   i = i + 1
   riadok = ts.ReadLine
   Print #1, riadok
   If InStr(riadok, "@Terminal") = 1 Then
      Do While InStr(riadok, ",") <> 0
        If InStr(riadok, "@Terminal") = 1 Then
          poradieTerminal = j
        End If
        pozicia = InStr(riadok, ",")
        dlzka = Len(riadok)
        a = Left(riadok, pozicia)
        riadok = Right(riadok, dlzka - pozicia)
        j = j + 1
        If InStr(riadok, "Label") = 1 Then
           poradieLabel = j
        End If
        If InStr(riadok, "P&ID Tag Name") = 1 Then
            poradieTag = j
        End If
        If InStr(riadok, "Tag Comment") = 1 Then
            poradieTagComment = j
        End If
        If InStr(riadok, "Tag Name") = 1 Then
```

```
            poradieTagName = j
          End If

          If InStr(riadok, "Service Comment") = 1 Then
            poradieComment = j
          End If
          If InStr(riadok, "Signal") = 1 Then
            poradieSignal = j
          End If
          If InStr(riadok, "Low Limit") = 1 Then
            poradieLL = j
          End If
          If InStr(riadok, "High Limit") = 1 Then
            poradieHL = j
          End If
          If InStr(riadok, "Unit") = 1 Then
            poradieUnit = j
          End If
        Loop
    End If
Loop
stlpec = 1
Do While Cells(stlpec)(1) <> Empty
    If Cells(stlpec)(1) = "Terminal" Then
       StlpecTerminal = stlpec
    End If
    If Cells(stlpec)(1) = "Label" Then
       StlpecLabel = stlpec
    End If
    If Cells(stlpec)(1) = "P&ID Tag Name" Then
      StlpecTag = stlpec
    End If
    If Cells(stlpec)(1) = "Tag Name" Then
       StlpecTagName = stlpec
    End If
    If Cells(stlpec)(1) = "Tag Comment" Then
       StlpecTagComment = stlpec
    End If
    If Cells(stlpec)(1) = "Signal" Then
       StlpecSignal = stlpec
    End If
    If Cells(stlpec)(1) = "Service Comment" Then
       StlpecComment = stlpec
    End If
    If Cells(stlpec)(1) = "Low Limit" Then
       StlpecLL = stlpec
    End If
    If Cells(stlpec)(1) = "High Limit" Then
       StlpecHL = stlpec
    End If
    If Cells(stlpec)(1) = "Unit" Then
       StlpecUnit = stlpec
    End If
    stlpec = stlpec + 1
Loop
riadok = 1
RiadokBunka = 2
Do While Cells(1)(RiadokBunka) <> Empty
    riadok = ts.ReadLine
    ZapisRiadok = Empty
```

```
    j = 0
    Do While InStr(riadok, ",") <> 0
        pozicia = InStr(riadok, ",")
        dlzka = Len(riadok)
        a = Left(riadok, pozicia)
        If InStr(a, Chr(34)) <> 0 Then "uvodzovky
          aTemp = a
          riadok = Right(riadok, dlzka - pozicia)
          pozicia = InStr(riadok, ",")
          dlzka = Len(riadok)
          a = Left(riadok, pozicia)
          riadok = Right(riadok, dlzka - pozicia)
          Do While InStr(a, Chr(34)) = 0
            pozicia = InStr(riadok, ",")
            dlzka = Len(riadok)
            aTemp = aTemp + a
            a = Left(riadok, pozicia)
            riadok = Right(riadok, dlzka - pozicia)
          Loop
        aTemp = aTemp + a
        a = aTemp
        End If
        riadok = Right(riadok, dlzka - pozicia)
        j = j + 1
        If j = poradieTerminal Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecTerminal)(RiadokBunka)) + ","
        ElseIf j = poradieLabel Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecLabel)(RiadokBunka)) + ","
        ElseIf j = poradieTag Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecTag)(RiadokBunka)) + ","
        ElseIf j = poradieSignal Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecSignal)(RiadokBunka)) + ","
        ElseIf j = poradieTagName Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecTagName)(RiadokBunka)) + ","
        ElseIf j = poradieTagComment Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecTagComment)(RiadokBunka)) + ","
        ElseIf j = poradieComment Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecComment)(RiadokBunka)) + ","
        ElseIf j = poradieLL Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecLL)(RiadokBunka)) + ","
        ElseIf j = poradieHL Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecHL)(RiadokBunka)) + ","
        ElseIf j = poradieUnit Then
           ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecUnit)(RiadokBunka)) + ","
        Else
           ZapisRiadok = ZapisRiadok + a
        End If
        If InStr(riadok, ",") = 0 Then
            If j < poradieLabel Then
               ZapisRiadok = ZapisRiadok + CStr(Cells(StlpecLabel)(RiadokBunka)) + ","
            End If
        End If
    Loop
    RiadokBunka = RiadokBunka + 1
    Print #1, ZapisRiadok
Loop
ts.Close
Close #1
MsgBox "Údaje boli úspešne zapísané."
End Sub
```

Príloha D

Zdrojový kód programu na čítanie konfiguračných údajov schémy v Control Dravingu systému CENTUM do tabuľky Excelu (súbor SchemaCit.xls)

```
Private Sub CommandButton1_Click()
retazec = txtCesta.Text
Dim riadok, blok As String
Dim velkost, pozicia, riad As Integer
Dim i, j, k As Integer

Open retazec For Input As #1
Line Input #1, riadok
Cells(1)(1) = "Nazov tagu"
Cells(2)(1) = "Typ bloku"
Cells(3)(1) = "Input"
Cells(4)(1) = "Output"
i = 1
j = 1
k = 1
Do While Not EOF(1)
   Line Input #1, riadok
   If InStr(riadok, "ETAG") <> 0 Then
      i = i + 1
      velkost = Len(riadok) - 1
      pozicia = InStr(riadok, "ETAG") + 6
      riadok = Left(riadok, velkost)
      Tag = Right(riadok, velkost - pozicia)
      Cells(1)(i) = Tag
   End If

   If InStr(riadok, "EINS") <> 0 Then
      j = j + 1
      velkost = Len(riadok) - 1
      pozicia = InStr(riadok, "EINS") + 6
      riadok = Left(riadok, velkost)
      Tag = Right(riadok, velkost - pozicia)
      Cells(2)(j) = Tag
   End If
   If InStr(riadok, "FREF") <> 0 Then
      Do While InStr(riadok, "::FREF") <> 1
         Line Input #1, riadok

         If InStr(riadok, "RTAG") <> 0 Then
            velkost = Len(riadok) - 1
            pozicia = InStr(riadok, "RTAG") + 6
            riadok = Left(riadok, velkost)
            Tag = Right(riadok, velkost - pozicia)
         End If
         If InStr(riadok, "RCNC") <> 0 Then
            If InStr(riadok, ".IN") <> 0 Then
               stlpec = 3
               velkost = Len(riadok) - 6
            Else
               stlpec = 4
               velkost = Len(riadok) - 7
```

```
            End If

            pozicia = InStr(riadok, "RTAG") + 10
            riadok = Left(riadok, velkost)
            blok = Right(riadok, velkost - pozicia)
         Do While Cells(1)(k) <> Empty
            k = k + 1
            If Cells(1)(k) = blok Then
               riad = k
            End If
         Loop
            Cells(stlpec)(riad) = Tag
            k = 1
         End If
      Loop
End If
Loop
Close #1
End Sub
```

Príloha E

Zdrojový kód programu na zápis konfiguračných údajov schémy v Control Dravingu systému CENTUM z tabuľky Excelu (súbor SchemaZap.xls)

Private Sub CommandButton1_Click()

```
Const ForReading = 1, ForWriting = 2
Dim fs, riadok, ZapisRiadok, retazec, PID, PIDTemp, PVI, PVITemp, PIDTag, PVITag, PIO, PIOTemp,
PIOTag As String
Dim i, j, k, Pozicia, OsY As Integer

retazec = "d:\pvi.txt"
Set fs = CreateObject("Scripting.FileSystemObject")
Set f = fs.GetFile(retazec)
Set ts = f.OpenAsTextStream(ForReading, TristateUseDefault)
retazec = "d:\schema1Zap.txt"
Open retazec For Output As #1
Do While InStr(riadok, "::FHED") = 0
riadok = ts.ReadLine
Print #1, riadok
Loop
j = 0
k = 0
i = 1
Do While Cells(3)(i + 2) <> Empty
   i = i + 1
   OsY = 200 + i * 60
   If Cells(4)(i + 1) = "PID" Then
    j = j + 1 'cislo bloku
    PIDTag = Cells(3)(i + 1)
PID = ":FNRM" + Chr(13) + "1:REV3K:1:1;" + Chr(13) + "68:BKRV:2;" + Chr(13) + "2:ETAG:1:" +
PIDTag + ";" + Chr(13) + "3:EINS:1:PID;" + Chr(13) + "4:ETCM:1:;" + Chr(13) + "5:ETIM:1:2;" +
Chr(13) + "6:ELIM:1:4;" + Chr(13) + "7:CHKN:1:" + CStr(j) + ";" + Chr(13) + "9:ESCA:1:S;" + Chr(13)
+ "10:MSBP:1:NO;" + Chr(13) + "11:SREV:1:NO;" + Chr(13) + "12:MVSR:1:NO;" + Chr(13) +
```

```
"13:PINS:1:YES;" + Chr(13) + "14:OCMK:1:NO;" + Chr(13) + "15:DVNO:1:AUTO;" + Chr(13) +
"16:MV01:1:REAL;" + Chr(13) + "17:CASM:1:AUTO;" + Chr(13) + "18:CMPM:1:NO;" + Chr(13) +
"19:ESCL:1:100.0:0.0;" + Chr(13) + "20:EINP:1:LINEAR;" + Chr(13) + "21:SQC!:1:0.5;" + Chr(13) +
"22:PIR!:1:1.00:AUTO;" + Chr(13) + "23:SSI!:1:1.000:0.000:106.25:-6.25;" + Chr(13) +
"24:EUNT:1:%;" + Chr(13) + "25:FLTR:1:AUTO;" + Chr(13) + "26:ESUM:1:NO:0%;" + Chr(13) +
"27:OVPV:1:NO;" + Chr(13) + "28:EALA:1:2;" + Chr(13) + "29:INOP:1:HL;" + Chr(13) +
"30:HHLL:1:HHLL;" + Chr(13) + "31:INHL:1:HL;" _
+ Chr(13) + "32:HHLH:1:2.0%;" + Chr(13) + "33:INVC:1:BOTH;" + Chr(13) + "34:IVHS:1:1:1:2.0%;"
+ Chr(13) + "35:DVTN:1:BOTH;" + Chr(13) + "36:DVFH:1:0.000:0:1.0%;" + Chr(13) +
"37:ALOP:1:YES;" + Chr(13) + "38:HLLT:1:HL;" + Chr(13) + "39:HLHT:1:2.0%;" + Chr(13) +
"40:ILCN:1:YES;" + Chr(13) + "41:PIDA:1:AUTO2;" + Chr(13) + "42:PIDD:1:REV;" + Chr(13) +
"43:CNTP:1:AUTO;" + Chr(13) + "44:NLGN:1:NO;" + Chr(13) + "45:NGGN:1:0;" + Chr(13) +
"46:NOCL:1:NO;" + Chr(13) + "47:NCHT:1:1.0%;" + Chr(13) + "48:IOCP:1:NO;" + Chr(13) +
"49:MTMC:1:NO:NO:YES;" + Chr(13) + "50:AUTF:1:NO;" + Chr(13) + "51:BKUP:1:MAN;" +
Chr(13) + "52:MV99:1:AUTO;" + Chr(13) + "53:AORA:1:100.0:0.0:%;" + Chr(13) +
"54:OVLM:1:100%:NO;" + Chr(13) + "55:CLPO:1:YES;" + Chr(13) + "56:CCOU:1:POS;" + Chr(13) +
"57:OUTP:1:LINEAR;" + Chr(13) + "58:AOF!:1:106.25:-17.19;" + Chr(13) +
"59:OTL!:1:1.000:0.000:106.25:-17.19;" + Chr(13) + "60:PWF!:1:0.00:0.00:0.00;" + Chr(13) +
"61:SSO!:1:1.000:0.000;" + Chr(13) + "62:SUOU:1:PV;" + Chr(13) + "63:SOAC:1:POSITION;" + _

Chr(13) + "65:HLPM:1:0000;" + Chr(13) + "66:UPPG:1:;" + Chr(13) + "67:UPPL:1:;" + Chr(13) +
"8:GBLK:200," + CStr(j * 100) + ":S1;" + Chr(13) + "::FNRM" + Chr(13)
      Do While InStr(PID, Chr(13)) <> 0              'zapis PID blok
         Pozicia = InStr(PID, Chr(13))
         dlzka = Len(PID)
         PIDTemp = Left(PID, Pozicia - 1)
         PID = Right(PID, dlzka - Pozicia)
         Print #1, PIDTemp
      Loop
   ElseIf Cells(4)(i + 1) = "PVI" Then
   j = j + 1
   PVITag = Cells(3)(i + 2)
PVI = ":FNRM" + Chr(13) + "1:REV3K:1:1;" + Chr(13) + "38:BKRV:2;" + Chr(13) + "2:ETAG:1:" +
PVITag + ";" + Chr(13) + "3:EINS:1:PVI;" + Chr(13) + "4:ETCM:1:comentar indikatoru;" + Chr(13) +
"5:ETIM:1:2;" + Chr(13) + "6:ELIM:1:4;" + Chr(13) + "7:CHKN:1:" + CStr(j) + ";" + Chr(13) +
"9:ESCA:1:S;" + Chr(13) + "10:SCCP:1:1:0;" + Chr(13) + "11:MSBP:1:NO;" + Chr(13) +
"12:SREV:1:NO;" + Chr(13) + "13:DVNO:1:AUTO;" + Chr(13) + "14:ESCL:1:100.0:0.0;" + Chr(13) +
"15:EINP:1:LINEAR;" + Chr(13) + "16:SQC!:1:0.5;" + Chr(13) + "17:PIR!:1:1.00:AUTO;" + Chr(13) +
"18:SSI!:1:1.000:0.000:106.25:-6.25;" + Chr(13) + "19:EUNT:1:%;" + Chr(13) + "20:FLTR:1:AUTO;" +
Chr(13) + "21:ESUM:1:NO:0%;" + Chr(13) + "22:OVPV:1:NO;" + Chr(13) + "23:EALA:1:2;" + Chr(13)
+ "24:INOP:1:HL;" + Chr(13) + "25:HHLL:1:HHLL;" + Chr(13) + "26:INHL:1:HL;" + Chr(13) +
"27:HHLH:1:2.0%;" + Chr(13) + "28:INVC:1:BOTH;" _
+ Chr(13) + "29:IVHS:1:1:1:2.0%;" + Chr(13) + "30:ILCN:1:YES;" + Chr(13) + "31:OUTL:1:LINEAR;"
+ Chr(13) + "32:SSL!:1:1.000:0.000;" + Chr(13) + "33:SUOP:1:PV;" + Chr(13) +
"34:SOAC:1:POSITION;" + Chr(13) + "35:HLPM:1:0000;" + Chr(13) + "36:UPPG:1:;" + Chr(13) +
"37:UPPL:1:;" + Chr(13) + "8:GBLK:200," + CStr(j * 100) + ":S1;" + Chr(13) + "::FNRM" + Chr(13)
      Do While InStr(PVI, Chr(13)) <> 0              'zapis PVI blok
         Pozicia = InStr(PVI, Chr(13))
         dlzka = Len(PVI)
         PVITemp = Left(PVI, Pozicia - 1)
         PVI = Right(PVI, dlzka - Pozicia)
         Print #1, PVITemp
      Loop
   End If
   If Cells(5)(i + 1) <> Empty Then
    k = k + 1
    PIOTag = Cells(5)(i + 1)
    PIO = ":FREF" + Chr(13) + "1:RTAG:1:" + PIOTag + ";" + Chr(13) + "2:RINS:1:PIO;" + Chr(13) +
"3:RCHK:1:@" + CStr(k) + ";" + Chr(13) + "4:GBLK:1," + CStr(j * 100 + 24) + ":S1;" + Chr(13) +
"::FREF" + Chr(13)
```

```
      Do While InStr(PIO, Chr(13)) <> 0            'zapis PIO blok
        Pozicia = InStr(PIO, Chr(13))
        dlzka = Len(PIO)
        PIOTemp = Left(PIO, Pozicia - 1)
        PIO = Right(PIO, dlzka - Pozicia)
        Print #1, PIOTemp
      Loop
    End If
   If Cells(6)(i + 1) <> Empty Then
     k = k + 1
     PIOTag = Cells(6)(i + 1)
     PIO = ":FREF" + Chr(13) + "1:RTAG:1:" + PIOTag + ";" + Chr(13) + "2:RINS:1:PIO;" + Chr(13) +
"3:RCHK:1:@" + CStr(k) + ";" + Chr(13) + "4:GBLK:350," + CStr(j * 100 + 24) + ":S1;" + Chr(13) +
"::FREF" + Chr(13)

      Do While InStr(PIO, Chr(13)) <> 0            'zapis PIO blok
        Pozicia = InStr(PIO, Chr(13))
        dlzka = Len(PIO)
        PIOTemp = Left(PIO, Pozicia - 1)
        PIO = Right(PIO, dlzka - Pozicia)
        Print #1, PIOTemp
      Loop
    End If
Loop
Print #1, "::::SOURCE"
Close #1
End Sub
```