PWR_RTT_EDIT

Användarhandledning

Revision:	96 04 15	Claes Sjöfors
Version:	V2.7-2	

Inledning

Pwr_rtt_edit är en editor för att bygga bilder för underhåll och enklare operatörskommunikation i proview. Pwr_rtt_edit omfattar ett verktyg för att bygga upp menyer av den typ som används i objektshierarkierna i rtt, samt en editor för att bygga bilder som presenterar värden på attribut i realtids-databasen.

Pwr_rtt_edit startas med symbolen pwr_rtt_edit med med ett lämpligt programnamn som argument (t ex nodnamn). Det rtt program som skapas kommer att namnges rs_rtt_'programnamn'.

Pwr_rtt_edit kräver att två logiska namn finns definierade: pwrp_rtt och pwrp_rttbld. Dessa ska peka på filkataloger i projekt-trädet. pwrp_rtt kommera att innehålla källkoden för menyer och bilder, och pwrp_rttbld diverse filer för att bygga rtt. Standard är filkatalogerna <.rtt> resp <.rttbld> under pwrp_src, vilket definieras med kommandona, på VMS

```
$ define/job pwrp_rtt            pwrp_groot:<common.src.rtt>
$ define/job pwrp_rttbld         pwrp_groot:<common.src.rttbld>
```

och på LYNX

```
export pwrp_rtt=$pwrp_groot/common/src/rtt
export pwrp_rttbld=$pwrp_groot/common/src/rttbld
```

pwr_rtt_edit kan köras på VAX_VMS, AXP_VMS och X86_LYNX.

Arbeta med rtt editorn

Här följer ett introduktion för att kunna sätta igång och jobba med rtt-editorn.

Starta editorn

Man startar editorn från konstrukörsmenyn eller med kommandot

```
$ pwr_rtt_edit
```

Editorn frågar efter 'Program' som är ett namn som kommer att ingå i alla filer som genereras. rtt-programmet som genereras med program-namnet 'VHXN2R' kommer att namnges pwrp_exe:rs_rtt_vhxn2r.exe. Startar man editorn från DCL kan program-namnet skickas med som argument.

```
$ pwr_rtt_edit vhxn2r
```

Meny editorn

När program-namnet är inmatat kommer man in i meny-editorn. Man kan här skapa ett meny-träd med meny-entryn för bilder, undermenyer mm.

Menyeditorn visar meny-entryn i den form de kommer att få i det genererade programmet. Enda skillnaden är att meny-entryts typ skrivs inom parentes efter menytexten. Översta menyn kallas huvudmenyn. Det innehåller en huvudrubrik, ett menyentry, och ett (ännu ej synbart) titel prefix i övre vänstra hörnet.

Genom att trycka på ctrl/b får man fram en prompt och kan skriva kommandon. Med 'pil upp' kan man återanvända och modifiera gamla kommandon.

Huvudrubrik

Huvudrubriken skrivs default som "RTT IN 'program'".
Man kan modifiera den med kommandot 'modify /maintitle". Vill man ha den på svenska kan man t ex skriva

```
rtt_edit> mod/maintitle="Rtt i VHXN2R"
```

Om titeln innehåller blanktecken eller små bokstäver ska den omges av citations-tecken.

Titel prefix

Titel prefix är en text som skrivs ut i över västra hörnet på samtliga sidor i rtt (utom editerade bildsidor). Prefixet bör innehålla system och nod-namn så att man enkelt kan se vilken nod man är inne i. Editerar man ett program som ska kunna exekveras i olika noder och tom olika system, kan man inte hårdkoda ett nod- eller system-namn. Det finns då möjlighet att använda rtt-symboler i prefixet. Nedan används symbolerna RTT_NODE och RTT_SYS som innehåller aktuellt nod- resp systemnamn.

Prefixet ändras med kommandot

```
rtt_edit> mod/titleprefix="RTT-#'RTT_NODE#'-#'RTT_SYS#'
```

#-tecket gör att inte rtt-symbolerna konverteras av editorn vid inmatningen (symbolerna finns definierade även i editorn eftersom denna bygger på rtt). Titleprefix kommer att innehålla texten "RTT-'RTT_NODE'-'RTT_SYS' som vid runtime kommer att konverteras till t ex "RTT-VHXOP4-VHXN2R".

Setup

Ett enklare sätt att lägga in huvudtitel och title prefix är att gå in i setup med kommandot

```
rtt_edit> setup
```

Välj ut ‘Main title’ eller ‘Title prefix’ och mata in mha PF3. Här behöver man inte använda # tecknet som i exemplet ovan.
Passa även på att lägga in de plattformar som rtt-programmet ska byggas för. I fältet ‘Operating system’ lägger man in koden för plattformen (eller summan av koderna om man vill bygga för flera plattformar).

Skapa menyentryn

När men editerar ett program första gången får man ett dummy -menyentry med namnet "'progamnamn' MENU", t ex "VHXN2R MENU". Man skapar nya menyentryn med funktionen 'create'.

```
rtt_edit> create/menu UNDERHÅLL
```

Nya menyentryn skapas under det menyentry som är utvalt. Vill man skapa ett menyentry överst måste man skapa det nya entryt under det översta, och sedan flytta ner det översta ett snäpp.

Skapa en undermeny

Man kan bygga menyträd genom att skapa undermenyer med kommandot 'create/child'. Undermenyer kan skapas till menyentryn av typen 'menu'. Välj ut ett entry av typen 'menu' och skriv kommandot

```
rtt_edit> create/menu/child "entry namn"
```

Genom att trycka på return öppnar man undermenyn, som nu innehåller ett menyentry. Flera menyentryn i undermenyn skapas pss som i huvudmenyn. Med PF4 återgår man till föregående meny.

Typer av menyentryn

Det finns ett antal olika typer av menyentryn.

menu

menu innehåller en undermeny skapas med kommandot

```
rtt_edit> create/menu "entryname"
```

picture

picture innehåller en editerad bild

```
rtt_edit> create/picture "picturename"
```

permpicture

Permpicture är liksom picure en editerad bild. Skillnaden är att en permpicture behåller sina prenumerationer när man går ur bilden, och har dem tillgängliga nästa gång man tar upp bilden. En picture tar bort prenumerationerna och måste knyta upp dem igen varje gång bilden tas upp.
En bild som används ofta bör göras som en permpicture.

```
rtt_edit> create/permpicture 'picturename'
```

function

function är en editerad bild med användaskriven kod bakom. I create-kommandot anges ett funktions-namn på en c-function som läggs in av konstruktören i filen pwrp_rtt:ra_rtt_'program'.c. c-funktionen anropas när bilden initieras, avslutas, när data i bilden ändras, samt cykliskt för uppdatering av bilden.

```
rtt_edit> create/picture/function=VHXN2R_ZON1 "Zon 1"
```

Imatad function för ett menyentry visas med

rtt_edit> show function

exit

exit är ett menyentry som avslutar exekveringen av rtt.

```
rtt_edit> create/exit "AVSLUTA"
```

objecthierarchy

objecthiearchy visar proview-databasen.

```
rtt_edit> create/objectherarchy "OBJEKT HIERARKI"
```

command

command utför ett rtt-kommando som anges vid skapandet av entryt. Följande menyentry kommer att visa larmlistan.

```
rtt_edit> create/command="alarm show" "Larmlista"
```

Man kan även sätta attribut i databasen från ett menyentry

```
rtt_edit>crea/command="set param/name=vhxn2r-zon1-
starta_ugn.ActualValue/value=1" "Starta ugn"
```

Imatat kommando för ett menyentry visas med

```
rtt_edit> show command
```

commandhold

En variant på ‘command’ som ska användas för funktionstangenter om man vill behålla den aktuella bilden. Om man har ett ‘command’ entry definierat som en funktionstangent kommer man att lämna den aktuella bilden innan kommandot utförs (med undantag för kommandot ‘set’). Med ‘commandhold’ behåller man aktuell bild och bilden kommer inte heller att ritas om när kommandot är utför.

keys

keys innehåller en undermeny med menyentryn som kan startas från funktions-tangenterna på ett rtt-tangenbord.

Ordningen av menyalternativen bestämmer vilken funktionstangen den kopplas till. Det översta menyentryt kopplas till F1, det näst översta till F2 osv. Keys-entryt måste ligga i huvudmenyn.

```
rtt_edit>create/key "Funktionstangenter"
```

vmscommand

vmscommand utför ett VMS-kommando. Detta kan användas för service-menyer i VMS-miljö. Ett godtyckligt VMS-kommando kan anges, lämpligen en kommandofil med parametrar.

```
rtt_edit>create/vmscommand="@pwrp_exe:logg_list 1" "Lista loggning"
```

Inmatat vms -kommandot för ett menyentry visas med kommandot

```
rtt_edit> show vms
```

vmshold

vmshold är samma som vmscommand med den skillnaden att den aktuella rtt menyn inte ritas om när VMS-kommando har exekverats. En utskrift som genereras av en kommandofil som anropas kommer att hamna på nedersta raden i rtt-menyn. Man kan pss skriva ut felmeddelanden eller information i kommandofilen.

```
rtt_edit>create/vmshold="@pwrp_exe:logg_list 2" "Lista mera loggning"
```

vmsconfirm

vmsconfirm exekverar ett vms -kommando, men kräver först en bekräftelse från användaren. Användaren bekräftar genom att trycka på PF1 och avbryter genom att trycka på PF4.

```
rtt_edit> create/vmsconfirm="@pwrp_exe:logg_list 3" "Lista ännu mer..."
```

vmsnowait

vmsnowait är ytterligare en variant av vms -kommando, där man inte väntar på att kommandot ska exekvera färdigt. Kommandot kan fortsätta och leva sitt eget liv i en subprocess, medan rtt-programmet forsätter att köra. Man kan t ex skapa en decterm och exekvera ett program där.

```
rtt_edit> create/nowait="@pwrp_exe:logg_list 4" "Lista i ett separat
fönster"
```

objpicture

objpicture är objektsbilder, dvs bilder som presenterar ett objekt av en viss klass. Objektsbilder finns för klasserna PID och Av. Förutom objektsnamnet anges i 'create'-kommandot vilken objektsbild som ska visas (RTTSYS_OBJECT_'klass').

```
rtt_edit> create/objpicture=rttsys_object_pid/name=vhxn2r-zon1-tempreg
"Temperaturregulator zon 1"
```

Man kan visa den inmatade objektsbilden för ett menyentry med kommandot

```
rtt_edit> show objpicture
```

Objektsnamnet visas med

```
rtt_edit> show name
```

syspicture

syspicture är bilder som visar information om ett proviewsystem, t ex vilka noder man har kontakt med (SHOW NODES).

```
rtt_edit> create/syspicture=rttsys_show_nodes "SHOW NODES"
```

Man kan visa den inmatade systembilden för ett menyentry med kommandot

```
rtt_edit> show syspicture
```

Ta bort menyentryn

Ett menyentry tas bort genom att välja ut det och trycka på delete (radera). Innehåller menyentryt en undermeny kommer även denna att tas bort. Den senaste delete-operationen kan återskapas med 'undo delete' kommandot.

```
rtt_edit>undo delete
```

Flyttning av menyentryn

Delete - 'undo delete' funktionerna kan användas för att flytta menyentry eftersom 'undo delete' återskapar det senast borttagna menyentryn nedanför det utvalda entryt.

Ändra ett menyentry

Man kan ändra på texten i ett menyentry med modify funktionen. Välj ut menyentryt och ange den nya texten.

```
rtt_edit> modify "Nytt namn"
```

Typen på ett menyentry kan ej ändras. Man kan inte heller ändra på kommandon och funktioner. Skapa nya menyentryn i dessa fall.

Flyttning av meny-träd mellan program

Ett meny-träd kan skrivas på en text -fil och läsas in igen med kommandona

```
rtt_edit> write menu "filnamn"
rtt_edit> include menu "filnamn"
```

Inläsning av standardmenyer

Standardversionen av rtt innehåller att antal menyer med systembilder som även bör finnas i projekt-specifika rtt-program. Genom att includera filen ssab_inc:rtt_menu_template.dtt_m includerar men dessa menyer.

```
rtt_edit> include menu ssab_inc:rtt_menu_template
```

Editera bilder

Öppnar men ett menyentry av typen picture, permpicture eller function, kommer man in i bildeditorn. Man kan här skriva texter, rita linjegrafik och skapa uppdaterings-fält. Editeringsarean är 80 tecken bred och 22 tecken hög.
Man förflyttar sig med piltangenterna till den plats i bilden man vill editera.
Liksom i meny-editorn ger man kommandon med ctrl/b.

Inverse mode

Med kommandot 'set inverse' kan man skriva tecken i inverse mode, dvs vita tecken på svart botten. Blanktecken blir svarta rutor som kan använda som grafik. Både text och linjegrafik kan skrivas i inverse mode.

```
rtt_edit> set inverse
```

Med 'set noinverse' lämnar men inverse mode.

```
rtt_edit> set noinverse
```

Linjegrafik

Med 'set line' kan man rita linjegrafik. Se figur vilka tecken som är användbara.

```
rtt_edit> set line
```

Återgå till text med kommandot

```
rtt_edit> set ascii
```

Skapa uppdateringsfält

Ett uppdateringsfält kopplas till ett attribut i Proview-databasen. Editerar man en funktionsbild (menyentry av typen function) kan man även koppla fältet till en variabel i en intern databas.

Ett fält skapas genom att man placerar cursorn där fältet ska ligga och matar in kommandot

```
rtt_edit> create %
```

Fältet markeras med 'U' följt av ett nummer.

Editera uppdateringsfält

Man kan editera data för fältet genom placera cursorn på fältets första position och trycka på PF1. Välj ut det fält-attribut som ska ändras och mata in ett nytt värde med 'modify'.

Analoga attribut

Antag att att Av-objekt ska visas i fältet. I dataentryt 'Parameter' ska namnet på Av-objektet med attribut (ActualValue) anges. Välj ut 'Parameter:' med piltangenterna och skriv kommandot

```
rtt_edit> mod "VHX-ZON1-Temperatur.ActualValue"
```

Ett enklare sätt är att ta upp navigatorn och kopiera in namnet därifrån. Ange i 'Characters:' antal tecken i fältet, och i 'Decimals:' antal decimaler.

Exempel

Number	1
Text	%
Type	UPDATE
Parameter	vhx -Zon1-Temperatur.ActualValue
Text/Dualpar	
Privileges	NO
Ouputflags	
Characters	6
Decimals	2
Maxlimit	0.00000
Minlimit	0.00000
Database	GDH
Declaration	

Stapeldiagram

Ett analogs attribut kan presenteras i form av en liggande stapel, genom att 'Ouputflags' sätts till "BAR". Lägg in min och max-värdet för stapeln i 'MinLimit' och 'MaxLimit', och stapelns längd i 'Characters'

Exempel
En 20 tecken lång stapel, 0-500 C

Number	1
Text	%
Type	UPDATE
Parameter	vhx -Zon1-Temperatur.ActualValue
Text/Dualpar	
Privileges	NO
Ouputflags	BAR
Characters	20
Decimals	0
Maxlimit	0.00000
Minlimit	500.00000
Database	GDH
Declaration	

Digitala attribut

Antag att att Dv-objekt ska visas i fältet. I dataentryt 'Parameter:' ska namnet på Dv-objektet med attrubut (ActualValue) anges. Välj ut 'Parameter:' med piltangenterna och kopiera in namnet på Dv'n

```
rtt_edit> mod "VHX-ZON1-UgnenStartad.AcutualValue"
```

Exempel

Number	1
Text	%
Type	UPDATE
Parameter	vhx -Zon1- DriftUgn ActualValue
Text/Dualpar	
Privileges	NO

Ouputflags	
Characters	1
Decimals	1
Maxlimit	0.00000
Minlimit	0.00000
Database	GDH
Declaration	

Text

'Ouputflags:' anger hur fältet ska presenteras i bilden. Anges ingenting kommer värdet att presenteras som en siffra, dvs 0 eller 1 för digitala attribut. Genom att ändra på 'Ouputflags' kan man få utskriften i form av en text ("On/Off","True/False","Auto/Man").

Man kan även bestämma texten själv genom att ange "TEXT" i 'Outputflags' och ange texten i 'Text/Dualpar:'. Ett '/'-tecken avgränsar vad som kommer att skrivas ut vid 1 resp 0.

Följande innebär att "Till" skrivs ut om Dv'n är 1, "Från" om Dv'n är 0.

```
rtt_edit> mod "Till/Från"
```

Utropstecken markerar att texten ska skrivas i inverterad mode. Nedan kommer "Till" att skrivas inverterad, men inte "Från"

```
rtt_edit> mod "!Till/Från"
```

Med pefixet _L_ markeras att texten ska skrivas ut som linjegrafik. I texten skrivs ascii-tecken som motsvarar linjegrafik teckenen. T ex

```
rtt_edit> mod "_L_qqqq/_L_xxxx"
```

kommer att skrivas ut som ”`----`” vid 1 och ”`||||`” vid 0.

Ange även i 'Characters:' storleken på fältet (dvs den längsta av de båda texterna).

Blinkande text får man genom att ange ”FLASHTEXT” i ‘Outputflags’.

Exempel

Number	1
Text	%
Type	UPDATE
Parameter	vhx -Zon1-UgnDrift.ActualValue
Text/Dualpar	!Zon1 är i drift/Zon1 är stoppad
Privileges	NO
Ouputflags	TEXT
Characters	15
Decimals	0
Maxlimit	0.00000
Minlimit	0.00000
Database	GDH
Declaration	

Ändringsbara fält

I dataentryt 'Text:' kan man ange en text som kommer att skrivas ut framför fältet. Denna text används när man

vill ändra värde på attributet i fältet, genom att texten inverteras när fältet är utvalt. Med piltangenterna kommer man att kunna flytta sig mellan olika fält, och ordningen mellan fälten bestäms av innehållet i 'Number:'. Genom att trycka på 'pil ned' eller 'pil höger' kommer fältet med närmast högre nummer att väljas ut. Med 'pil upp' eller 'pil vänster' kommer fältet med närmast lägre nummer att väljas. Om två fält har samma nummer avgör slumpen vilket som kommer först.

Har man valt ut önskat fält kommer man att kunna ändra på värdet genom att trycka på PF3 och mata in nytt värde. I 'Privileges' anges vem som får ändra i fältet, och i 'Maxlimit' och 'Minlimit' max och min-värden för inmatning.

Exempel

Number	1
Text	Börvärde temperatur
Type	UPDATE
Parameter	vhx -Zon1- TempBörvärde.ActualValue
Text/Dualpar	
Privileges	OP
Ouputflags	
Characters	6
Decimals	2
Maxlimit	500.00000
Minlimit	0.00000
Database	GDH
Declaration	

Ej ändningbara fält

Om ett fält inte ska kunna ändras ska 'Text:' anges till "%" och 'Privileges:' ska vara "NO".

Inverterade fält

Genom att lägga in ett utropstecken i 'Text/Dualpar' skrivs fältet i inverse mode.

Number	1
Text	%
Type	UPDATE
Parameter	vhx-Zon1-Fläkt-Varvtal.ActualValue
Text/Dualpar	!
Privileges	NO
Ouputflags	
Characters	5
Decimals	2
Maxlimit	0.00000
Minlimit	0.00000
Database	GDH
Declaration	

Tryckknappar

Man kan ändra värdet på ett digitalt genom att välja ut det, och trycka på PF1 eller PF2. De möjligheter som finns är

- Sätta värdet till 1 med PF1 (SET)
- Återställa värdet med PF1 (RESET)
- Sätta värdet med PF1 och återställa med PF2 (SET_RESET)
- Toggla värdet med PF1(TOGGLE)

Ofta är värdet av attributet ointressant. Man anger att det inte ska visas genom att sätta 'Outputflags' till "NO"

Exempel

Dv'n motorstart sätts med PF1

Number	1
Text	Starta motor
Type	SET
Parameter	vhx-Zon1-MotorStart.ActualValue
Text/Dualpar	
Privileges	OP
Ouputflags	NO
Characters	0
Decimals	0
Maxlimit	0.00000
Minlimit	0.00000
Database	GDH
Declaration	

Objid

Attribut av typen objid, t ex plåt-objekt i celler, visas genom att objid'n konverteras till objektsnamn. Om man sätter 'Decimals' till 1 kommer enbart sista segmentet av namnet att visas.

Exempel

Number	1
Text	%
Type	UPDATE
Parameter	vhx-Plåtföjning-w-Cell.Data1_ObjId
Text/Dualpar	
Privileges	NO
Ouputflags	
Characters	10
Decimals	1
Maxlimit	0.00000
Minlimit	0.00000
Database	GDH
Declaration	

Tid

Attribut av typen pwr_tTime, dvs UTC-tid, konverteras till en sträng med formatet ”DD-MMM-ÅÅÅÅ TT:MM:SS.HH”, t ex 30-MAR-1999 12:35:10.40. Om man sätter 'Decimals' till 1 visas inte datum utan enbart klockslag.

Öppna bild på tryckknapp

Man kan lägga in en tryckknapp som utför ett valfritt rtt-kommando PF1 aktiveras. Genom att använda rtt-kommandot 'show menu' kan man öppna andra bilder från tryckknappen.

Lägg in "COMMAND" i 'Type' och aktuellt rtt-kommandot i 'Text/Dualpar'.

Om uttrycket som ska matas in 'Text/Dualpar' innehåller blanktecken ska det omgärdas med dubbelfnuttar. Vill man att dubbelfnuttarna ska stanna kvar i den inmatade pga att rtt-kommandot i sig kräver dubbelfnuttar, skriver man \" iställlet för " (se exemplet nedan)..

Exempel

Number	1
Text	Visa pumpar
Type	COMMAND
Parameter	
Text/Dualpar	show menu "underhåll-pumpar-översikt pumpar"
Privileges	OP
Ouputflags	NO
Characters	0
Decimals	0
Maxlimit	0.00000
Minlimit	0.00000
Database	USER
Declaration	

Öppnar bilden underhåll-pumpar-översikt pumpar. För att kunna få texten i 'Text/Dualpar' har följande kommandorad matas in:

```
rtt_edit> mod "show menu \"underhåll-pumpar-översikt pumpar\""
```

Ändring av fält utan att öppna det

Man behöver inte öppna ett fält för att ändra i det. Genom att placera cursorn på fältets första position kan man med 'modify' ändra data i fältet. Detta är användbart när man kopplar databasattribute till fältet eller om man ska sätta samma data i fler fält.

```
rtt_edit> mod/par="vhx-Zon1-Temperatur.ActualValue"
rtt_edit> mod/char=7
```

Ta bort ett fält

Ett fält tas bort genom att cursorn placeras på fältet och kommandot 'delete item' ges.

```
rtt_edit> delete item
```

Cut-Copy-Paste

Ett område i bildeditorn kan väljas ut genom placera cursorn i ena hörnet, trycka på PF2, och förflytta sig till andra hörnet. Det utvalda området markeras genom att det inverteras.

Man kan kopiera ett utvalt områdets till paste-bufferten genom att trycka på PF3 (eller ge kommandot 'copy'). 'cut' tar dessutom bort grafik och fält i det utvalda området. Innehållet i paste-bufferten kopieras till editeringsarean med 'paste' eller ctrl/f. Cursorns position bestämmer var den nya arean placeras. Kopiering kan även ske mellan olika bilder i samma program (för att kopiera mellan program måste man skriva ut innehållet på fil, se nedan).

```
rtt_edit> cut
```

```
rtt_edit> paste
```

Återställning av ett utvalt område sker med 'unselect' kommandot

```
rtt_edit> unselect
```

Editera fält i text-editor

Om man har flera likartade fält i en bild kan man skriva ut innehållet i samtliga fält i bilden i en textfil, modifiera textfilen, och läsa in fälten igen. Kommandot

```
rtt_edit> write items items.txt
```

skriver ut fälten på textfilen item.txt. Filen kan läsas in i edt eller tpu. För att ladda in fälten i bilden, rensar man först bort de gamla fälten.

```
rtt_edit> clear items
```

och läser in filen

```
rtt_edit> include items items.txt
```

Kopiera bilder mellan program

För att kopiera en bild från ett program till en annan, tar man upp bilden i bildeditorn, och skriver ut den med kommandot

```
rtt_edit> write picture bild.tmp
```

Bilden lagras på filen bild.tmp Filen kan läsas in i en tom bildeditorn med

```
rtt_edit> inlcude picture bild.tmp
```

Leta efter attribut

Ibland vill man veta i vilket fält, en visst attribut påverkas. Med kommandot

```
rtt_edit> write items/all items.tmp
```

skrivs samtliga fält i samtliga bilder ut på filen items.tmp Genom att leta i filen hittar man enkelt fältet för aktuellt attribut.

Hjälptexter

Med kommandot

```
rtt_edit> edit help
```

kommer man genom edt in i en textfil där man kan lägga in hjälptexter för bilder och menyer. Hjälptexten består av två delar, dels en text som visas upp genom att trycka på Hjälp-tangenten eller ctrl/h, dels en text-rad som skrivs längst ner på en meny-sida. Kopplingen mellan en hjälp-text och bilden/menyn utgörs av tileln (texten i meny-entryt).

En hjälptext för undermeny "Underhåll" kan se ut så här

```
RTT_HELP_SUBJ("UNDERHÅLL")
RTT_HELP_INFO("\
 Välj ut önskat ämne med piltangenterna och tryck på RETURN")
RTT_HELP_TEXT( "\
          Diverse funktioner för underhåll av systemet.\n\n\
VENTILER       Visar ventiler i anläggningen.\n\
MOTORER        Visar motorer i anläggningen.\n\
TEMPERATURER   Visar termoelement i ugnen.\n\
")
```

För bilder används ej info-texten.

Ett exempel på hjälptexten till bilden "Motorer"

```
RTT_HELP_SUBJ("MOTORER")
RTT_HELP_INFO("")
RTT_HELP_TEXT( "\
Bilden motorer hanterar motorerna.\n\n\
För varje motor visas huvudkontaktor och larmstatus.\n\
Starta och stoppa motorerna med PF1.\n\
 ")
```

Filen är en c include-fil och syntaxen följer stränghantering i c. \n betyder ny rad., och \ i slutet på en rad markerar att texten fortsätter på nästa rad. Hjälpfilen är gemensam för samtliga menyer och bilder i programmet.

Spara

Kommandot 'save' sparar menyer och bilder. Gör man save i meny-editorn sparas menyerna, gör man save i bild-editorn sparas den aktuella bilden.

```
rtt_edit> save
```

Skapa rtt-program

När alla menyer och bilder är klara skapar man exe-filen med kommandot

```
rtt_edit> link
```

Exe-filer genereras för både VMS och ELN (på Alpha enbart för vms). Vill man skapa filer enbart för VMS eller ELN finns kommandona

```
rtt_edit> link vms
rtt_edit> link eln
```

Köra programmet

Har man anget programnamn "VHXN2R" genereras exe-filerna pwrp_exe:rs_rtt_vhxn2r.exe (VMS) resp pwrp_root:<vax_eln.exe>rs_rtt_vhxn2r.exe_eln (ELN)

På VMS kör programmet från DCL med

```
$ run pwrp_exe:rs_rtt_vhxn2r.exe
```

på ELN från ECL med

```
ECL> ex/w rs_rtt_vhxn2r
```

Om man har startat näthanteraren kan man köra igång programmet direkt från rtt-editorn

```
rtt_edit> rtt
```

Avsluta

Avsluta editeringen med 'quit'

```
rtt_edit> quit
```

Editera på olika plattformar

Man kan editera rtt-bilder både på VAX och på Alpha. Från VAX kan man generera rtt-program för VAXELN och VAX VMS, från Alpha genererar man program för Alpha.

Om man har editerat bilder eller menyer på en Alpha och vill generera program för VAX, måste man gå in på en VAX och starta pwr_rtt_edit. Därefter måste nya och ändrade bilder sparas innan rtt-programmet kan genererars med link-kommandot. Är man osäker på vilka bilder man har ändrat i kan man spara samtliga med ‘save/all’ kommandot

```
rtt_edit> save/all
rtt_edit> link
```

Motsvarande gäller om man har editerat på VAX och vill generera bilder för Alpha. Logga in på en Alpha och spara samtliga nya och ändrade bilder.

Menyer

Menyer används i rtt för att navigera till olika bilder, för att komma in i objektshierarkin och för att avsluta sessionen. Ett meny träd byggs upp i pwr_rtt_edit med hjälp av kommandot create. När ett menyträd är skapat används detta för att navigera och öppna bilder och undermenyer i pwr_rtt_edit på samma sätt som i det slutgiltiga rtt programmet.

Meny mod

I meny mod skapar man menyer och navigerar i de skapade menyerna. Följande tangenter är definierade i en meny bild.

Tangent	Funktion
PILTANGENTER	Navigera och välj ut ett meny entry.
RETURN	Öppna en under meny eller bild.
PF4 eller CTRL/R	Gå tillbaka till föregående meny.
NEXT PAGE eller CTRL/F	Gå till nästa sida i menyn.
PREV PAGE eller CTRL/D	Gå till föregående sida i menyn.
CTRL/Z	Tillbaka till rotmenyn
CTRL/W	Rita om bilden.
HELP	Hjälp.
DELETE	Ta bort ett meny entry.
DO eller CTRL/B	Kommando mod.

.

Skapa menyer och menyentryn

Nya menyentryn skapas med 'create' kommandot.
Det finns ett antal olika typer av meny entryn:

menu	entry till en under meny.
picture	entry till en bild.
objecthierarchy	entry till rtt's objekts hierarki.
exit	entry för att avsluta.
command	entry för att exekvera ett rtt kommando.
vms	entry för att exekvera ett vms kommando
permpicture	entry till en bild som lagrar sina prenumerationer. En bild som använd ofta bör skapas som en permpicture.
keys	entry till en meny under vilken funktionertill snabbstarts tangenterna skapas.
syspicture	entry till en systembild.
objpicture	entry till en objektsbild.

Ett menyentry skapas genom att man väljer ut det menyentry som ligger ovanför den plats man vill skjuta in den nya entryt, och ger kommandot create med lämplig kvalifierare:

`rtt_edit> create/menu 'title'`	skapa entry för en submeny.
`rtt_edit> create/picture 'title'`	skapa entry för en bild.
`rtt_edit> create/objecthierarchy 'title'`	skapa entry för objekts hierarkin.
`rtt_edit> create/exit 'title'`	skapa entry för att avsluta.
`rtt_edit> create/command="kommando" 'title'`	skapa entry för att exekvera ett rtt kommando.
`rtt_edit> create/cmdhold="kommando" 'title'`	skapa entry för att exekvera ett rtt kommando från funktionstangent utan att lämna aktuell bild.
`rtt_edit> create/vms="kommando" 'title'`	skapa entry för att exekvera ett vms kommando.
`rtt_edit> create/permpicture 'title'`	skapa entry för en permanent bild.
`rtt_edit> create/keys 'title'`	skapa entry för snabbstarts funktioner.
`rtt_edit> create/syspict='syspicture' 'title'`	skapa entry för en systembild.
`rtt_edit> create/objpict='objpicture'/name='objektsnamn' 'title'`	skapa entry för en objektsblid.

'title' är den text som kommer att skrivas i menyentryt. Om man vill ha blanktecken i texten ska '#' matas in istället för space (eller omgärdas av ""). För att kunna öppna ett menu entry måste det finnas minst ett entry i undermenyn. Detta skapas genom att välja ut förälder entryt och lägga /child till ovanstående kommandon. När detta är gjort kan man öppna entryt med return och fortsätta och skapa entryn i undermenyn.

`rtt_edit> create/menu/child 'title'`	skapa entry för en submeny.
`rtt_edit> create/picture/child 'title'`	skapa entry för en bild.
`rtt_edit> create/objecthierarchy/child 'title'`	skapa entry för objekts hierarkin.
`rtt_edit> create/exit/child 'title'`	skapa entry för att avsluta.
`rtt_edit> create/command="kommando"/child 'title'`	skapa entry för att exek vera ett rtt-kommando.
`rtt_edit> create/vms="kommando"/child 'title'`	skapa entry för att exek vera ett vms-kommando.

En meny titel kan ändras med modify kommandot

```
rtt_edit> modify "text"
```

Kommandot i en kommandentry visas med 'show command'.

Ta bort menyer

Ett menyalternativ tas bort med Delete (radera) tangenten. Om menyalternativet innehåller en undermeny, eller ett meny-träd, kommer även dessa att tas bort.

Senaste borttagningen kan återskapas med kommandot 'undo delete' om man skulle ångra sig.

Flytta menyer

Ett menyalternativ kan flyttas mha delete - undo delete funktionerna. Senaste delete operationen lagras i en buffer. Vid kommandot 'undo delete' läggs innehållet i bufferten in under utvalt menyentry.

Editering av bilder

Ett entry till en bild skapas med

```
rtt_edit> create/picture 'title'
```

Genom att välja ut entryt och trycka på return kommer man
in i editerings mod och kan editera bilden.

Editerings mod

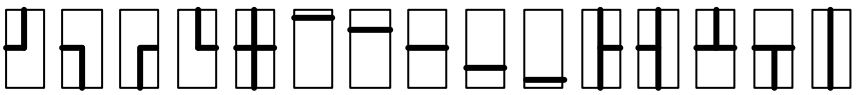
I editerings mod är följande tangenter definierade.

Tangent	Funktion
PILTANGENTER	Navigera i arbetsarean.
PF1	Öppna ett uppdaterings fält.
PF2	Select.
PF3	Kopierar utvald area till paste bufferten.
PF4 eller CTRL/ R	Gå tillbaka till menyn.
CTRL/Z	Gå tillbaka till rotmenyn.
CTRL/W	Rita om bilden.
CTRL/N	Visa collection picture.
CTRL/V	Koppla ett uppdaterings fält till utvalt objekt i collection picture.
CTRL/F	Paste.
HELP	Hjälp.
DO ellr CTRL/B	Kommando mod.

I editerings mod skapas en arbetsarea på 22X80 tecken att lägga in tecken och uppdateringsfält på. De två nedersta raderna är reserverade för kommandon och meddelanden.

Teckenset ändras med kommandot set:

```
rtt_edit> set ascii
rtt_edit> set line
rtt_edit> set inverse
rtt_edit> set noinverse
```



Teckenuppsättning vid line

Uppdateringsfält skapas med kommandot create:

```
rtt_edit> create 'text'
```

'text' är den text som kommer att inverteras när man går
runt med pilarna i den slutgiltiga bilden.

Koppling till databasen

Ett uppdateringsfälten kopplas enklast till ett rtdbobjekt på följande sätt:

- Ta upp navigatören, välj 'CopyMode' normal+Attribut, och välj ut önskat objekt.
- Öppna uppdateringsfältet i rtt_edit, välj ut 'Parameter'.
- Skriv vid kommando prompten

```
rtt_edit> mod "
```

- Kopiera objektnamnet från navigatören genom att klicka med MB2 i rtt_edit-fönstret.

```
rtt_edit> mod "VHX-Zon1-ZonTempertur1.ActualValue
```

- Mata in ett avslutande "-tecken och tryck på return.

```
rtt_edit> mod "VHX-Zon1-ZonTemperatur1.ActualValue"
```

Dubbelfnuttarna kan utelämnas, men objektsnamnet kommer då att lagras med stora bokstäver. Detta har ingen betydelse för funktionen, men blir med svårläst. Om fönsterhantering inte finns tillgänglig, men väl näthanteraren kan följande (något föråldrade) alternativa användas för att koppla ett fält:

1 Starta rtt men kommandot

```
rtt_edit> rtt
```

2 Vandra runt i rtt, och samla ihop de objekt och parametrar som ska visas i bilden med collectfunktionen. När man tar upp collection picture finns dessa objekt samlade där.
3 Välj ut ett objekt i collection picture.
4 Gå tillbaka till bilden och placera cursorn på ett uppdaterings fält.
5 Genom att trycka på CTRL/V kopplas fältet till objektet.

Pss kan man koppla dualparametern för ett fält med kommandot

```
rtt_edit> dualconnect
```

Editera ett uppdaterings fält

Genom att placera cursorn på första tecknet i ett uppdaterings fält kan man öppna detta (PF1) och ange data för fältet. Följande tangenter är definierade i fältdata bilden.

Tangenter	Funktion
PILTANGENTER	Välj ut en ett element i fältet.
PF4 eller CTRL/R	Gå tillbaka till menyn.
HELP	Hjälp.
DO eller CTRL/B	Kommando mod.

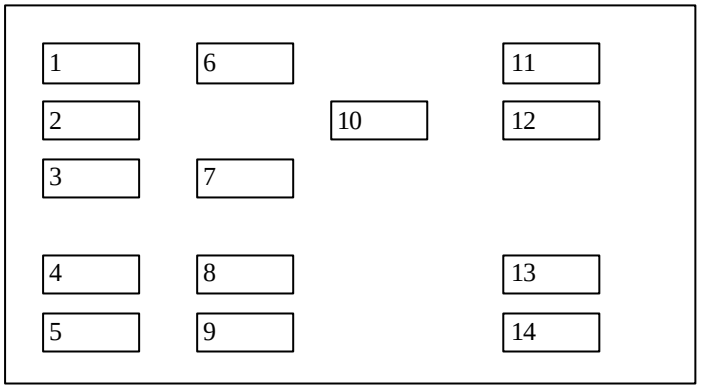
Data ändras med modify commandot. Välj ut önskat element och mata in ett nytt värdet med

```
rtt_edit> modify 'value'
```

Data för ett fält

Number

Number anger den ordning i vilken fälten kommer att bli utvalda i bilden när ett fält väljs ut med piltangenterna pil-upp och pil-ner. Pil-vänster och pil-höger väljer närmaste fält till vänster resp. till höger på samma rad, oberoende av 'Number'. Fälten bör numreras kolumnvis med lägsta nummer uppe till vänster och hösta nere till höger, se figur nedan.



Numrering av fält

Text

Texten som markerar att fältet är utvalt. Skrivs med inverterat när fältet är utvalt. Om texten sätts till '%' kommer ingen text att skrivas ut.

Type

Typ av funktion för att förändra en parameter i databasen.

Type	Beskrivning
UPDATE	Värdet på parametern ändras med change value funktionen (PF4).
SET	Parametern sätts genom att trycka på PF1 (endast boolean).
RESET	Parametern återställs med PF1 (endast boolean).
SET_RESET	Parametern sätts med PF1 och återställs med PF2 (endast boolean).
TOGGLE	Parametern togglas med PF1 (endast boolean).
DUAL_SET	En parameter visas i bilden och en annan parameter (dualparametern) påverkas av set fuktionen, dvs dualparametern sätts med PF1.
DUAL_RESET	En parametern visas i bilden och en annan (dualparametern) återställs med PF1
DUAL_SET_RESET	En parameter visas i bilen och en annan (dualparametern)sätts med PF1 och återställs med PF2.
DUAL_TOGGLE	En parameter visas i bilden och en annan (dualparametern)togglas med PF1.
SET_DUALSET	En parameter sätts med PF1 (och visas eventuellt) och en annan (dualparametern) sätts med PF2.
TOGGLE_DUALTOGGLE	En parameter togglas med PF1 (och visas eventuellt) och en annan (dualparametern) togglas med PF2.
COMMAND	Ett rtt-kommand angivet i Text/Dualpar utförs med PF1.

Parameter

Namn på objekt och parameter som ska visas i fältet.

Text/Dualparameter

Vid outputflags = TEXT läggs texten in här. Texterna för för true resp false separeras men ett '/'.
Om texten ska inverteras läggs ett utropstecken som första tecken i texten.
Om texten ska skrivas som linjegrafik läggs prefixet _L_ till texten. Texten anges med de ascii-tecken som motvarar linjegrafik-tecken.
I Characters läggs antalet tecken i den längsta av texterna ut.

Ex "Motorn är startad/!Motorn är stoppad"

När variablen är true kommer texten "Motorn är startad" att läggas ut i bilden, när variabeln är false läggs texten "Motorn är stoppad" ut inverterad.

Vid type = COMMAND läggs rtt-kommandot in i Text/Dualparameter.

Vid dualfunktion läggs namn på objekt och parameter som ska påverkas vid dualfunktionen in här (type = DUAL_SET, DUAL_RESET, DUAL_SET_RESET eller DUAL_TOGGLE).

Privileges

Privilegier som krävs för att ändra data i fältet.

Privilegier	Beskrivning
OP	Samtliga har behörighet att ändra ävärdet
EL	Användare som loggat in som el eller sys har behörighet
PROC	Användare som loggat in som proc eller sys har behörighet
SYS	Användare som loggat in som sys har behörighet
NOOP	Alla utom op
NO	Ingen får ändra värdet

Outputflags

Typ av presentation av en boolean eller en float.

Outputflags	Beskrivning
ONOFF	boolean, vid true visas ON, vid false OFF.
AUTOMAN	boolean, vid true visas AUTO, vid false MAN.
OPENCLOSED	boolean, vid true visas OPEN, vid false CLOSED.
TRUEFALSE	boolean, vid true visas TRUE, vid false FALSE
NO	boolean, värdet visas ej (endast texten).
BAR	float, värdet visas i form av en liggande stapel.
TEXT	boolean, vid true skivs en textsträng ut, vid false en annan. Textsträngarna anges i Text/Dualparameter fältet.
FLASHTEXT	boolean. Blinkande text, i övrigt samma funktion som ‘TEXT’ ovan.

Characters

Storlek på fältet. Antal tecken.

Decimals

Antal decimaler i ett flyttal.

Om objektet är ett gdh-objekt av typen Objid kan man markera att endast sista namnledet ska skrivas ut genom att sätta Decimals till 1.

Om objektet är av typen pwr_tTime eller TIME kan man markera att enbart klockslag, ej datum, ska skrivas ut genom att sätta Decimals till 1.

Maxlimit

Maxbegräsning vid inmatning med change value funktionen i rtt.

Minlimit

Minbegräsning vid inmatning med change value funktionen i rtt.

Database

Databas som för parametern eller variablen.

Database	Beskrivning
GDH	parametern finns i gdh.
RTT	parametern skapas lokalt i rtt och hanteras av användaren i funktionen till en funktions bild. Endast tillåtet i funktions bilder.
USER	pekaren till parametern tilldelas av användaren i funktionen till en funktions bild. Endast tillåtet i funktionsbilder.

Declaration

Deklaration av en lokal variabel. Behöver endast anges om elementet database är RTT.

Declaration	Beskrivning
BOOLEAN	Boolean (unsigned char).
INT	Int (long).
FLOAT	Float.
CHAR	Char.
STRING	En sträng med 80 tecken.
STRING4	En sträng med 4 tecken.
STRING10	En sträng med 10 tecken.
STRING20	En sträng med 20 tecken.
STRING40	En sträng med 40 tecken.

Stapeldiagram

En float kan presenteras som en liggande stapel. Fältet ska ha följande uppsättning

Text	%
Type	UPDATE
Privileges	NO
Ouputflags	BAR
Characters	Stapelns läng vid maxvärdet.
Maxlimit	Minvärde för stapeln.
Minlimit	Maxvärde för stapeln
Database	GDH

Fält som visar tiden

Med kommandot

```
rtt_edit> create /time
rtt_edit> create /fulltime
```

skapas fält som visar upp tiden med eller utan datum (/fulltime resp /time).

Fält som visar larm

Man kan skapa fält som visar de fem senaste rådande larmen.
Fälten skapas som ett vanliga fält med create och ska ha följande uppsättning.

Text	%
Type	UPDATE
Parameter	RTT_ALARMTEXT1 (visar det senaste larmet) RTT_ALARMTEXT2 (visar det näst senaste larmet) ... RTT_ALARMTEXT5 (visare det femte senaste larmet).
Privileges	NO
Characters	Ange maximal storlek som ska skrivas ut i bilden.
Database	RTT
Declaration	STRING

Bilder med bakomliggande kod

Bilder med användarskriven c-kod kan skapas med kommandot

```
rtt_edit> create /picture/function='function name'
```

'function name' är en funktion som kommer att anropas

- när bilden tas upp
- när man gör exit ur bilden
- varje scan
- när man trycker på ‘Nästa sida’ och ‘Föregående sida’.
- när värdet på någon parameter i bilden ändras av operatören.

När man första gången gör save på en "funktions bild" skapas en fil som användaren själv editerar och underhåller. Filen heter ra_rtt_'program namn'.c och kan nås med edit kommandot

```
rtt_edit> edit
```

I den här filen editeras in en funktion för varje funktions bild som har skapats I olika entryn i funktionen lägger man in kod som ska exekveras för att initiera eller avsluta bilen, eller när någon parameter i bilden ändrats av operatören. När filen är editerad ska den kompileras. Detta enklast med kommandot

```
rtt_edit> compile
```

eller om enbart vms resp eln version önskas

```
rtt_edit> compile vms
rtt_edit> compile eln
```

Lokala variabler

Lokala variabler kan visas i en funktions bild genom att man skapar ett uppdaterings fält i bilden, lägger in ett lämpligt variabelnamn i parameter elementet, samt anger "RTT" som database och c-typen på variabeln (BOOLEAN, CHAR, INT, FLOAT eller STRING) i declaration elementet. De lokala variablerna är direkt åtkomliga i funktions filen (ra_rtt_'program namn'.c) genom att de automatiskt deklareras i en includefil som funktionsfilen inkluderar.

Funktions fil

I funktionsfilen ska deklareras en funktion med samma namn som angavs när bilden som ska associeras med funktionen skapades (versaler). Funktionen ska vara av typen int och deklareras på följande sätt

```
int 'funktions namn' (          menu_ctx   ctx,
```

```
                                int         event,
                                char        *parameter_ptr)
```

ctx är en identitet för bilden som normalt ej behöver användas.
event anger vilken typ av händelse som gör att funktionen anropas och kan anta följande värden

Event	**Beskrivning**
RTT _APPL_INIT	anrop när bilden tas upp.
RTT_APPL_EXIT	anrop när man går ur bilden.
RTT_APPL_UPDATE	cykliskt anrop för att uppdatera paraparameterar i rtt databasen.
RTT_APPL_VALUECHANGED	anrop när ett värde ändras av operatören.
RTT_APPL_NEXTPAGE	anrop när tangenten 'nästa sida' aktiverats.
RTT_APPL_PREVPAGE	anrop när tangenten 'föregende sida' aktiverats.

parameter_ptr är pekare på det värde som ändrats vid RTT_APPL_VALUECHANGED.

Se appendix A för ett exempel på en funktions fil.

Funktionstangenter (snabbstart)

I rtt finns möjlighet att starta upp till femton menyalternativ på snabbstarts tangenter. Alla olika typer av menualternativ är möjliga att lägga som snabbstart, dvs man kan starta upp bilder, systembilder, menyer, utföra rtt- eller vmskommandon.

Snabbstartsmeny

Snabbstartsfunktionerna ska ligga under ett speciellt menyalternativ. Detta menyalternativ måste ligga i den översta nivån i menyhierarkin. Menyalternativet skapas genom att man placerar sig på lämpligt ställe i den översta menynivån och ger kommandot

```
rtt_edit> create /keys 'title'
```

De menyalternativ som skapas under detta menyalternativ kommer att aktiveras vid tryck på snabbstartstangenterna. Kopplingen till respektive tangen bestämms av ordningen av menyalternativen. Första menyalternativet akiveras av den första tangenten etc.

Från ett vanligt tangenbort aktiveras snabbstartsfuntionerna med tangentsekvenserna ESC*A, ESC*B,...,ESC*O.

Hjälptexter

Till varje meny och bild kan en hjälptext anges somvisas när hjälptangenten aktiveras.
dtt_appl_'programnamn'_m.rhlp

Första gången man gör save skapas en mall för denna fil.

Det finns ett antal makron för att ange hjälptexten.

- RTT_HELP_START Start av hjälptext definitionen
- RTT_HELP_SUBJ("'titel'") Bild eller meny som ska kopplas till texten. 'titel' är titel på menyn eller bilden.
- RTT_HELP_INFO("Välj ut önskat ämne med piltangenterna och tryck på RETURN") Infomations rad i en meny Informations raden skrivs ut på rad 23 i meny bilden.
- RTT_HELP_TEXT("Pwr_rtt visar info om databasen...") Hälptext
- RTT_HELP_END Slut på hjälptext definitionen.

Ny rad i en sträng i textfilen erhålls genom ett raden avslutas med ett '\' (observera att det inte får finnas något tecken till höger om \, inte heller space eller tab). Ny rad i den slutliga hjälptexten erhålls med teckenföljden '\n\'. Exempel:

```
RTT_HELP_TEXT("\
Inledning med en tom rad efter\n\n\
hjälptext\n\
mera hjälptext")
```

Systembilder

Det finns ett antal systembilder i rtt som kan inkluderas genom med följande kommandon:

Standard-layouten för systembilderna finns i filen ssab_inc:rtt_menu_template.dtt_m. Välj ut det menyentry som ska ligga ovanför menyträdet med systembilderna, och mata in kommandot

```
rtt_edit> include menu ssab_inc:rtt_menu_template
```

Vill man ha en egen layout kan man skapa den på detta sätt.
Skapa ett meny alternativ på översta nivån med

```
rtt_edit>create SYSTEM
```

Skapa första systembilden under SYSTEM

```
rtt_edit> create/child/syspict=RTTSYS_SHOW_SYS "SHOW SYSTEM"
```

Skapa övriga systembilder genom att gå ner under SYSTEM

```
rtt_edit> create NETHANDLER
rtt_edit> create/syspict=RTTSYS_ERROR "SHOW ERROR"
rtt_edit> create/syspict=RTTSYS_PLCPGM PLCPGM
rtt_edit> create/syspict=RTTSYS_GRAFCET GRAFCET
rtt_edit> create/syspict=RTTSYS_DEVICE DEVICE
rtt_edit> create/syspict=RTTSYS_WATCHDOG WATCHDOG
rtt_edit> create/syspict=RTTSYS_PID PID
rtt_edit> create/syspict=RTTSYS_LOGGING LOGGING
```

Välj ut NETHANLDLER och skapa första systembilden under NETHANDLER

```
rtt_edit> create/child/syspict=RTTSYS_SHOW_NODES "SHOW NODES"
```

Skapa övriga bilder under NETHANDLER genom att gå ner under NETHANDLER

```
rtt_edit> create/syspict=RTTSYS_SHOW_SUBCLI "SHOW SUBSCRIPTION CLIENT"
rtt_edit> create/syspict=RTTSYS_SHOW_SUBSRV "SHOW SUBSCRIPTION SERVER"
```

I standard rtt finns även menyalternativet STORE. Skapa detta med följande kommandon.

```
rtt_edit> create STORE/command=”show file”
```

Objektsbilder

För vissa klasser finns speciella bilder som visar data för objekt som tillhör klassen. Bilderna kan läggs som menyentryn och för varje entry anges vilken klassbild som ska användas och vilket objekt den ska kopplas till. Skapa menyentryt med kommandot

```
rtt_edit> create /objpict='klassbild' /name='objektsnamn' 'titel'
```

Exempel

```
rtt_edit> create/objpict=RTTSYS_OBJECT_PID /name=vhx-z1-tempregl-w-pid0
"TEMPERATUR REGULATOR"
```

Följande objektsbilder finns:

klass	klassbild
PID	RTTSYS_OBJECT_PID
Av	RTTSYS_OBJECT_AV

Objektsbild PID

Bilden visar ärvärde, börvärde och utsignal i stapel och siffer-form. Vidare visas mode pid-algoritm, inverse, bias. Det finns möjlighet att ändra mod, samt mata in värden på börvärde (i automod), utsignal (i manuell mod), Gain, IntTime, DerTime mm. Området som visas i stapeldiagrammet kan anges genom att lägga in värden på SetShowMin, SetShowMax, OutShowMin och OutShowMax.

Objektsbilden kräver att modeobjektet är angivet i PID objektet.

Rtt utan gdh

Man kan skapa rtt menyer och bilder som inte har någonting med proview och gdh att göra. Men kan t ex göra en meny som startar upp vms kommandofiler i olka meny entryn (av typ vms). Normalt kommer dock rtt att försöka koppla sig till proview's näthanterare, men även om den inte lyckas med detta fortsätter den glad i hågen. Kommandon som kräver näthanterare ger felmeddelande. Biler editerade med pwr_rtt_edit som länkar sig mot rtdb bör ej finnas, eftersom rtt kommer att avsluta exekveringen närman försöker ta upp en sådan bild.

Vill man skapa en meny som absolut inte kopplar sig mot proview's näthanterare, t ex en meny som startar resp stoppar näthanteraren gör man på följande sätt.

Skapa en meny men diverse menyentryn av typ vms, och bygg rttprogrammet. Starta därefter det skapade rttprogrammet med username som inleds med 'NONETH_' t ex 'NONETH_SKIFTEL'. Skapa en symbol som gör det här, t ex

```
$ menu == "$pwrp_rtt:rs_rtt_xxx NONETH_SKIFTEL"
```

Kommandon

Kommando prompten nås men 'Utför' eller CTRL/B.

Bild editor

clear picture

Rensar bilden på allt.

```
rtt_edit> clear picture
```

clear items

Rensar bilden på samtliga updaterings-fält.

```
rtt_edit> clear items
```

connect

Koppla ihop ett objekt i rtdb med ett uppdaterings fält.
Placerar utvalt objekt i collection picture i parameter elementet för aktuellt uppdaterings fält (även CTRL/V).

```
rtt_edit> connect
```

copy

Kopiera utvald area till paste bufferten (även PF3).

```
rtt_edit> copy
```

create

Skapa uppdaterings fält i en bild.
Med /time visas tiden upp i fältet (hh:mm:ss), med /fulltime visas även datum (dd-mmm-yyyy hh:mm:ss).

```
rtt_edit> create 'text'
rtt_edit> create /time 'text'
rtt_edit> create /fulltime 'text'
```

cut

Ta bort utvald area och kopiera den till paste bufferten.

```
rtt_edit> cut
```

delete

Ta bort ett uppdaterings fält (delete item) eller
utvald area i en bild.

```
rtt_edit> delete
rtt_edit> delete item
```

dualconnect

Koppla ihop ett objekt i rtdb med ett uppdaterings fält.
Placerar utvalt objekt collection picture i dualparameter elementet för aktuellt uppdaterings fält (även CTRL/V).

```
rtt_edit> dualconnect
```

include items

Läs in en textfil med uppdateringsfält skapad med 'write items' kommanot.

```
rtt_edit> include items 'filnamn'
```

include picture

Läs in en bild i aktuell bild buffert.

```
rtt_edit> include picture 'filnamn'
```

modify

Ändra data för ett uppdaterings fält.
Data för ett uppdaterings fält kan ändras direkt i bildeditorn utan att öppna fältet.

```
rtt_edit> modify /number=
rtt_edit> modify /text=
rtt_edit> modify /type=
rtt_edit> modify /parameter=
rtt_edit> modify /dualparameter=
rtt_edit> modify /privileges=
rtt_edit> modify /outputflags=
```

paste

Kopierar pastebufferten till aktuell bild (även CTRL/F).

```
rtt_edit> paste
```

save

Spara en bild. Bilden sparas för de plattformar som finns angivet i ‘Operating system’ i setup. Om man vill spara för en specifik platform kan denna skickas med som argument.

```
rtt_edit> save
rtt_edit> save axp_vms
```

select

Välj ut en area (även PF2).
Utvald area markeras genom att ritas inverterad.

```
rtt_edit> select
```

set

Välj teckenset (ascii eller line) samt inverterad eller
ej inverterade tecken.

```
rtt_edit> set line
rtt_edit> set ascii
rtt_edit> set inverse
rtt_edit> set noinverse
```

unselect

Återställ select.

```
rtt_edit> unselect
```

write items

Skriv ut uppdateringsfält i en bild på en textfil.
Textfilen kan läsas in igen med 'include items' kommandot

```
rtt_edit> write items 'filnamn'
```

write picture

Skriv bilden.
Men write kommandot kan bilden sparas i en specificerad fil och kopieras eller användas som mall för en ny bild med include kommandot.

```
rtt_edit> write picture 'filnamn'
```

modify

Välj ut önskad data element och ändra värde med

```
rtt_edit> modify 'data'
```

Meny editor

create

Skapa nya menyentryn.
Kvalifierarna /menu, /picture, /exit och /objecthierarchy skapar ett entry av respektive typ.
Entry't läggs in under utvalt entry.
Med kvalifieraren /child skapas ett entry i en undermeny till ett entry av typen meny.

```
rtt_edit> create /menu 'title'
rtt_edit> create /picture 'title'
rtt_edit> create /picture/function="funktions namn" 'title'
rtt_edit> create /exit
rtt_edit> create /objecthierarchy
rtt_edit> create /command="kommando" 'title'
rtt_edit> create /cmdhold="kommando" 'title'
rtt_edit> create /menu /child 'title'
rtt_edit> create /picture /child 'title'
rtt_edit> create /exit /child
rtt_edit> create /objecthierarchy /child
rtt_edit> create /command="kommando"/child 'title'
```

Exempel

```
rtt_edit> create/comma="debug children/name=vkv-vkvau1-rack1-kort3" "kort
nr 3"
rtt_edit> create/picture this#is#a#picture
```

include menu

Läser in ett menyträd från en textfil. skapad men 'write menu' kommandot. Menyn placeras nedanför det utvalda menyalternativet. rtt_edit> include menu 'filnamn'

modify

Ändra texten på ett meny entry eller på titeln i huvudmenyn.
'mainmenu' är titeln på rot-menyn.
'titleprefix' är en text som står till vänster om titeln på varje meny-sida, och skrivs i inverterad mod. I 'titleprefix' kan man lägga in den nod eller den identitet man kör som, genom att använde rtt symboler. Rtt's standardversion innehåller strängen "RTT-'RTT_NODE'-'RTT_SYS'". För att kunna mata in symbolnamnen måste '-tecknen bytas ut mod #' eftersom symbolerna annars konverteras till deras aktuella värde vid inmatningen. Kommandot för inmatning av standardversiones prefix blir då

```
rtt_edit> modify /titleprefix=RTT-#'RTT_NODE#'-#'RTT_SYS#'
```

Om blank-tecken ingår i texten, omges den med citationstecken vid inmatningen.

```
rtt_edit> modify 'text'
rtt_edit> modify /maintitle='title'
rtt_edit> modify /titleprefix=
```

show

Visar
- inmatat kommando för ett kommando meny entry (rtt eller vms kommando),
- inmatad funktion för funktionsbild entry, systembilds entry eller objektbilds entry.
- inmatat objekt för objeksbildsentry.

```
rtt_edit> show command
rtt_edit> show function
rtt_edit> show syspicture
rtt_edit> show vms
rtt_edit> show objpicture
rtt_edit> show name
```

undo delete

Återskapar senast borttagna menyentry.
Menyentryt återskapas under utvalt menyentry. Undo delete kan även användas för att flytta menyentryn-

```
rtt_edit> undo delete
```

write menu

Skriver ett menyträd på textfil. Filen kan kan läsas in med 'include menu' kommandot. Med /all kvalifieraren skrivs samtliga menyalternativ i aktuell meny inlklusive

underligganden träd till dessa. Utan /all skrivs endast utvalt menyalternativ och trädet under detta i filen.

```
rtt_edit> write menu /all 'filnamn'
rtt_edit> write menu 'filnamn'
```

Gemensamma

compile

Kompilerar funktions filen och lägger in den i bibliotek.
Man kan välja att endast skapa en vms eller eln version.

```
rtt_edit> compile
rtt_edit> compile vms
rtt_edit> compile eln
```

edit

Editering av funktionsfilen. Startar edt med funktions filen som parameter.

```
rtt_edit> edit
```

exit

Spara menyerna och aktuell bild och avslutar exekveringen.

export gdhreflist

Listar gdhreferenser i samtliga bilder på specificerad fil.

```
rtt_edit> export gdhreflist 'filnamn'
```

export externref

Skapar en pwr_plc kommandofil som skapar ExternRef objekt i utvecklingsdatabasen för gdhrefereinser i samtliga bilder. Detta gör att gdhreferenserna kommer att visas i proviews korsreferenslistor.

```
rtt_edit> export externref 'filnamn'
```

link

Kompilerar och länkar programmet. Länkningen sker för den eller de plattformar som finns angivet i 'Operating system' i setup. Om det inte finns någon plattform angiven i 'Operating system' länkar man för den plattform som man exekverar på.
Man kan välja att länka för en viss plattform genom att ange plattformen som argument. Plattform kan vara vax_eln, vax_vms, axp_vms, x86_lynx.

```
rtt_edit> link
rtt_edit> link axp_vms
rtt_edit> link x86_lynx
```

quit

Avslutar exekveringen utan att spara.

save

Spara menyträdet och/eller aktuell bild.
Om savekommandot get i en bild sparas bilden och menyträdet, om kommandot ges i en meny sparas menyträdet.

```
rtt_edit> save
```

setup

Visar setup menyn.

Attribut	**Beskrivning**
Program name	Progamnamn. Vid byte av namn måste samtliga bilder sparas.
Main title	Titel i huvudbilden.
Title prefix	Text som visas i övre vänstra hörnet i samtliga menyer.
Operating system	Plattform för vilken rtt-programmet ska byggas. Kod för plattformen ska anges. Flera plattformar kan anges genom att summera koderna. Koderna är följande vax_eln 1, vax_vms 2, axp_vms 4, ppc_lynx 8, x86_lynx 16
Default directory	Default filkatalog för editering-sessionen.
Verify	Om kommandofiler i editerings-sessionen ska köras med verify eller ej.
Source direktory	Filkatalog för källkod (pwrp_rtt).
Build direktory	Filkatalog för div byggfiler (pwrp_rttbld).

show collection

Visar collection picture.

```
rtt_edit> show collection
```

Appendix A

Exempel på en funktion till en funktions bild

I följande exempel på en funktion kopplad till en funktions bild har de lokala variblerna HASTIGHET (FLOAT), START (BOOLEAN) och START_TEXT (STRING) skapats mha uppdaterings fält i bilden. Bilden är skapad med kommandot

```
rtt_edit> create /picture/function=START_FUNKTIONEN "STARTA MOTOR"
```

```
/************************************************************
*
* Name:         START_FUNKTIONEN
*
* Type          int
*
* Type     Parameter       IOGF  Description
* menu_ctx ctx             I     context of the picture.
* int      event           I     type of event.
* char     *parameter_ptr  I     pointer to the parameter which value
*                                has been changed.
*
* Description:
*       Starta motorn...
*
************************************************************/

int START_FUNKTIONEN (          menu_ctx             ctx,
                                int                  event,
                                char                 *parameter_ptr)
{
  /************************************************************
  *     Update rtt database
  ************************************************************/
  if ( event == RTT_APPL_UPDATE)
  {
        return RTT__SUCCESS;
  }
  /************************************************************
```

```
  *     Initialization of the picture
  ********************************************************/
  if ( event == RTT_APPL_INIT)
  {
        HASTIGHET = 44.;
        START = 0;
        strcpy( &TEXT, "Motorn klar att startas startas");
  }
  /********************************************************
  *       Exit of the picture
  ********************************************************/
  else if ( event == RTT_APPL_EXIT)
  {

  }
  /********************************************************
  *       The value of a parameter is changed.
  ********************************************************/
  else if ( event == RTT_APPL_VALUECHANGED)
  {
        if ( parameter_ptr == &START)
        {
           if ( START == 1)
             strcpy( &TEXT, "Motorn är startad");
           else
             strcpy( &TEXT, "Motorn är stoppad");
        }
  }
  return RTT__SUCCESS;
}
```