FOI
TOTALFÖRSVARETS
FORSKNINGSINSTITUT

FOI-R--0237-SE
November 2001
ISSN 1650-1942
Teknisk rapport

Dan Strömberg

Integration och Styrning av Sensorer i Nätverk

Ledningssystemteknik
581 11 Linköping

TOTALFÖRSVARETS FORSKNINGSINSTITUT
Ledningssystemteknik
581 11 Linköping

FOI-R--0237-SE
November 2001
ISSN 1650-1942
Teknisk rapport

Dan Strömberg

Integration och Styrning av Sensorer i Nätverk

Utgivare	Rapportnummer, ISRN	Klassificering
Totalförsvarets Forskningsinstitut - FOI	FOI-R--0237-SE	Teknisk rapport
Ledningssystemteknik	**Forskningsområde**	
581 11 Linköping	8. Människan i totalförsvaret	
	Månad, år	**Projektnummer**
	Oktober 2001	E7022
	Verksamhetsgren	
	5. Uppdragsfinansierad verksamhet	
	Delområde	
	81 MSI med fysiologi	

Författare/redaktör	Projektledare
Dan Strömberg	Författaren
	Godkänd av
	Johan Mårtensson
	Uppdragsgivare/kundbeteckning
	Försvarsmakten
	Tekniskt och/eller vetenskapligt ansvarig
	Lennart Nyström

Rapportens titel

Integration och Styrning av Sensorer i Nätverk

Sammanfattning (högst 200 ord)

På moderna stridsplattformar finns många sensorer men mycket litet av dataintegration. Det skapar problem för plattformsoperatörer som måste ha hög situationsmedvetenhet, eftersom tid måste ägnas åt att manövrera och tolka sensordata. Ett annat problem är att den flexibilitet som finns i moderna sensorsystem kräver lämpligt operatörsstöd för snabb och flexibel sensorstyrning. Kunnande börjar nu etableras om hur plattformsburna sensorsystem under operatörens kontroll kan arbeta tidseffektivt, det vill säga självständigt kan anpassas efter de dynamiska behov som finns av situationsmedvetenhet och sensorstyrning.

Ett verktyg har utvecklats på FOI för att studera dessa problem. Den centrala principen har varit att utgå från beslutscirkeln (eng: OODA-loopen). Verktyget kan simultant bearbeta ett (godtyckligt) antal operatörsstyrda eller maskingenererade OODA-loopar. Det interagerar med alla operatörer och sensorer ombord på plattformen, och utgör en datafusionsnod. Den kan också samverka med andra plattformar av samma eller annan typ (fartyg, flygplan, markfordon). Datafusion, situationsbedömning, hotanalys och sensorstyrning utförs under bearbetningen av de individuella OODA-looparna. I ett par applikationsexempel visas hur noden kan användas inte bara för att styra plattformens egna sensorer utan även sensorer på andra plattformar. Avslutningsvis visas hur noden kan användas för att bygga upp operatörstjänster vilka kan utnyttjas av användare som inte måste känna till vilka sensorer som egentligen finns att tillgå.

Nyckelord

Beslutscirkel, OODAloop, Datafusionsnod, Multisensorintegration, Datafusion, Sensorstyrning, Sensornät

Övriga bibliografiska uppgifter	Språk Svenska
ISSN 1650-1942	**Antal sidor:** 26 s.
Distribution enligt missiv	**Pris: Enligt prislista**
	Sekretess

Issuing organization	Report number, ISRN	Report type
FOI – Swedish Defence Research Agency C2W Tech SE-581 11 Linköping	FOI-R--0237-SE	Technical report
	Research area code 8. Human Systems	
	Month year November 2001	**Project no.** E7022
	Customers code 5. Commissioned research	
	Sub area code 81 Human Factors and Physiology	

Author/s (editor/s)	Project manager
Dan Strömberg	The author
	Approved by Johan Mårtensson
	Sponsoring agency Swedish Armed Forces
	Scientifically and technically responsible Lennart Nyström

Report title (In translation)

Integration and Management of Sensors in Networks

Abstract (not more than 200 words)

Modern combat platforms have many sensors but little integration of data from them. This causes problems for operators in situation awareness tasks, since they must take time for maneuvering and fusion of the sensors. Another problem is that the high degree of flexibility in modern sensor systems requires high level operator support for fast sensor control. Knowledge is now being established about how to use platform-based sensors in an time-effective way, that is how to effectively use them when many sub systems and operators simultaneously call for them.

A tool has been developed to study these problems. The main principle has been to exploit the OODA loop concept (Observe/Orient/Decide/ Act). The tool can simultaneously process a number of OODA loops, generated autonomously or by operators. It interacts with operators and sensor systems on board the platform, and is a data fusion node. It may also interact with other platforms of the same or different type (ship, plane, ground vehicle). Sensor integration, situation analyses, threat assessment and sensor control are performed during the OODA loop processing. Two examples show that it can interact with other platforms for efficient resource exploitation. Finally, it is shown how the node may be used in a network where users may ask for services that use sensors whose existence the users may not be aware of.

Keywords

OODA loop, Multi sensor integration, Data fusion, Sensor management, Sensor net

Further bibliographic information	Language Swedish
ISSN 1650-1942	**Pages** 26 p.
	Price acc. to pricelist
	Security classification

Innehållsförteckning

Arbetet har utförts inom ramen för projekten "Datafusion för Plattformsoperatörer" och "Digitala Gruppantenner" särskilt arbetspaket A och E.

Inledning

Datafusion på maskinell nivå mellan sensorer på dagens operativa plattformar är närmast obefintlig. Detsamma gäller datafusion mellan sensorer på olika plattformar. I nuvarande plattformssystem är det istället normalt att individuella sensorer och andra subsystem är monterade helt separerade från varandra, med skilda "trådar" fram till operatörens (operatörernas) bildskärmar och manöverorgan. Orsaken till denna separering mellan de skilda delsystemen är den höga komplexitet som vidlåter var och en av dem, vilket dock i ett senare läge försvårar eller omöjliggör en god systemintegration mellan dem. Den ansats till datafusion på plattformsnivå som vi här presenterar har vänt upp och ner på detta förhållande: Om en datafusionsnod sätts i centrum av plattformens ledningssystem, på så sätt att alla delsystem och operatörer lätt kan anslutas, då blir integrationen mellan dem möjlig och enkel att realisera. Alla operatörer får tillgång till resurser från alla sensorer, alla resurser kan utnyttjas optimalt och sensorsamverkan blir en naturlig process. Också samverkan mellan sensorsystem på olika plattformar blir möjlig, och operatörer på varje plattform ges tillgång till de resurser som behövs för deras uppgifter, på egen och andras plattformar. Detta synsätt är revolutionärt och utgör ett helt nytt tänkande på ledningssystemnivån. Här visas hur en sådan datafusionsnod kan konstrueras och implementeras.

Detta kapitel ägnas i fortsättningen åt introduktion av viktiga begrepp som behandlas i rapporten.

Sensorsamverkan

Förutom datafusion finns i noden också sensorstyrning. Dessa två funktioner, datafusion och sensorstyrning, är grundläggande för sensorsamverkan. Med detta begrepp avses alla typer av funktioner där sensorer på enstaka plattformar eller på grupper av plattformar samverkar: (a) observation av en omgivning med flera sensorer (på ett flertal plattformar) och åtföljande datasammanläggning, (b) aktiv styrning av vissa sensorer i avsikt att förbättra omgivningsanalysen och (c) allokering av sensorer på olika funktioner. Ett sätt att hantera denna komplexitet är att utgå från beslutsfattarens perspektiv, och dela in fusionsarbetet i ett antal beslutsloopar. En beslutsloop har två faser, en för situationsanalys och en för handling.

Datafusionsnod

Rapportens huvuddel beskriver ett egenutvecklat verktyg - en datafusionsnod - samt några experiment som använder detta verktyg. Verktyget består av en scenariogenerator och en simulator. I scenariogeneratorn kan ett antal mål och ett antal plattformar med sensorer skapas. Olika typer av sensorer kan modelleras, bland annat flerfunktionssensor såsom digital radar. På varje plattform finns en datafusionsnod, vars viktigaste del är en schemaläggare som styr arbetet i plattformens sensorer på alla nivåer, från operatörsnivå till millisekundnivå. För att hantera flerfunktionella sensorer har en variant av OODA-loopen, 'uppdrag', införts. Varje uppdrag definieras såsom en sensorberoende funktion som realiseras genom ett antal intermittenta sensoraktioner. Varje schemaläggare kan hantera flera uppdrag simultant.

Verktyget

Avsikten med utvecklandet av datafusionsnoden är att få fram ett enhetligt verktyg för utveckling och utvärdering av algoritmer och arkitekturer för datafusion och sensorstyrning. Vägledande vid valet av grundläggande designprinciper har varit att bevara största möjliga flexibilitet och kraftfullhet i dessa avseenden. Den forskningsprodukt som nu har utvecklats uppfyller dessa kriterier. Vissa tecken tyder på att den kan bli direkt användbar i en stridsplattform. Det aktuella utvecklingsläget är att grundstrukturer och funktioner för datafusion och sensorstyrning på låg nivå är utvecklade och implementerade, medan högnivåfunktioner såsom situationsanalys och hotpresentation ännu inte infogats. Om datafusionsnoden installerades i en plattform, skulle datafusion och sensorstyrning bli centrala komponenter i dess ledningssystem.

Reaktiv sensorsamverkan

Sensorsamverkan kan vara ickereaktiv eller reaktiv. Med ickereaktiv systemuppbyggnad avses integration av data som i tidigare skeden insamlats av olika sensorer. Med reaktiv sensorsamverkan menar vi en systemuppbyggnad i realtid som fortlöpande anpassas efter rådande förhållanden beträffande det observerade rummet, de använda sensorernas belastning, de sensorbärande

plattformarnas aktuella och planerade läge samt de plattformsbaserade operatörernas övergripande sensorstyrningsstrategi. Verktyget har konstruerats för att kunna modellera reaktiv sensorsamverkan.

OODA-loopen

En grundläggande princip vid utvecklingen av datafusionsnoden har varit att sätta beslutsloopen (OODA-loopen) i centrum. Detta har åstadkommits genom att bygga upp en särskild sorts objekt (agenter). Varje OODA-loop representeras i datafusionsnoden som ett eget objekt till vilket data och kunskap som associeras med loopen kan knytas. Fördelarna är flera. För det första har man sedan länge modellerat beslutsfattande i ledningssystem såsom hanteringen av ett antal samtidiga OODA-loopar. Nästling och konflikthantering av OODA-loopar har studerats. För det andra finns det inom den näraliggande teorin för situationsmedvetenhet (situation awareness) en kunskap om iterativa processer för situationsuppfattning, som lätt kan avbildas i dator. En liknande teoribildning finns också inom datafusionsteorin. För det tredje kan den kunskap om hur omgivningen representeras som finns inom beteendevetenskapen utnyttjas för att representera motsvarande information i programvara.

Datafusionsnoden har blivit en OODA-maskin som kan bli navet i ledningssystemet hos en generisk sensorbärande stridsplattform. Till denna maskin kan alla sensorer och andra omvärldsinriktade subsystem i plattformen anslutas, liksom alla de operatörsinteraktioner som behövs i plattformen.

Idén att använda OODA-loopen som den grundläggande principen vid datafusion och sensorstyrning innebär ett nytänkande som kräver en teoriutveckling. Som ett första steg mot en sådan teori presenteras en taxonomi för en plattformsgenerell datafusionsnod. Kraven på en teori är stora; bland annat ska den kunna modellera flerfunktionssensorer och dynamiska OODA-loopar. Andra faktorer som bör kunna hanteras är resurseffektivitet, delat beslutsfattande (många operatörer) och kvantitativ prestandaanalys.

Applikationsexperiment

Rapporten beskriver inledningsvis krav på plattformsbaserad datafusion och sensorstyrning. Två experiment redovisas också, där datafusionsnoden har använts i plattformsnätverk. I dessa används två sensorer på olika plattformar för att observera ett antal mål. För att bestämma vilken sensor som ska användas vid varje mättillfälle görs i experiment 1 en uppskattning av vilken kvalitet som förväntas vid mätning från varje sensor. I det andra exemplet tas dessutom hänsyn till vilken last och operatörspolicy som tillämpas i de sensorerna.

Plattform

Rapporten fokuserar på datafusion inom och mellan plattformar. Med plattform avses en rörlig farkost eller fast installation som bär minst en sensor. Detta innebär att även markbaserade radarstationer sitter på 'plattformar'. Operatörerna kan sitta ombord på plattformarna eller på annan godtycklig plats.

Nätbaserat försvar

Försvarsmakten strävar efter ett nätbaserat försvar. Avslutningsvis diskuteras dels en nätarkitektur för försvarets alla sensorer inklusive nödvändiga stödkomponenter, dels en princip för hur resursallokeringen i nätet kan utformas. Datafusionsnoden kan användas som komponent i denna arkitektur.

Samverkan mellan sensorer i nätverk kan också beskrivas utifrån ett operatörsperspektiv: Varje operatör bör ha tillgång till tjänster som stödjer hans uppgifter, och detta utan att veta vilka resurser som finns och var de finns. Detta kan organiseras som tjänster av Internet-liknande typ. Systemet skall ge stöd för detta, och stödet kan vara distribuerat i noder för datainsamling, datafusion, sensorstyrning och resursallokering. Dessa frågor behandlas i de avslutande kapitlen om nätverk.

Krav på fusion och sensorstyrning i plattformar

Nedanstående kravlista avgränsas till tekniska frågor, men utgångspunkten finns i FM kravställning för RMA, LedsystT och nätbaserat försvar.

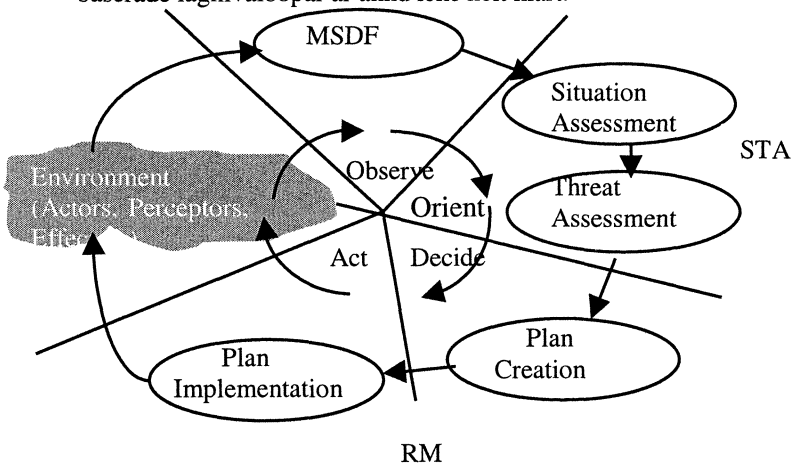
Eftersom datafusion och sensorstyrning i sig innebär att man tar ett samlat grepp på flera sensorer och operatörsstationer i en plattform, är det naturligt att dessa funktioner utförs i en central plats i plattformen. Vi kallar denna plats för datafusionsnod.

Datafusionsnod

En nod för datafusion och sensorstyrning på plattformsnivå skall kunna hantera OODA-loopar, flerfunktionella sensorer, operatörsinteraktion, flersensorsystem (flera sensorer på samma plattform), flera plattformar, nättjänster, dynamisk resursallokering samt distribuerad programvara. Var och en av dessa begrepp presenteras nedan.

OODA-loop

Datafusionsnoden skall baseras på OODA-loopen, som visas i figur 1. OODA-loopen kan användas i flera nivåer, vilket illustreras i figur 2. Datafusionsnoden skall också kunna hantera OODA-loopar på en lägre nivå där operatören inte har tid att medverka eftersom besluten fattas på millisekundnivå, se närmare förklaring under rubriken flerfunktionssensorer. Hur operatören skall hantera dessa sensor-baserade lågnivåloopar är ännu icke helt klart.

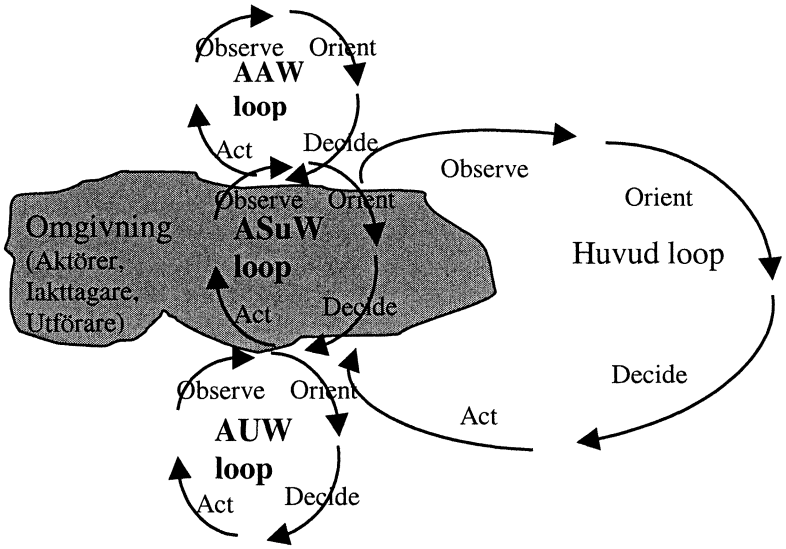


Figur 1. Figur hämtad från [RB]. OODA-loopen

Vanligen utförs flera OODA-loopar inuti varandra. Ett exempel visas i figur 2, som visar hur ett fartygs kommandocentral utför en loop som innefattar tre mindre loopar för respektive luftbevakning, ytbevakning och undervattensbevakning. Dessa tre områden kan in sin tur sönderdelas i ett större antal mindre loopar för bevakning av enskilda sökområden, mål etcetera. På lägre nivåer finns därför ett antal mycket snabba och självständiga OODA-loopsliknande funktioner. Datafusionsnoden skall kunna hantera OODA-loopar på olika nivåer.

Flerfunktionalitet

Funktionsbegreppet i denna term är analogt med begreppen uppdrag, uppgift, OODA-loop och beslutsloop. Vi använder i fortsättningen benämningen *uppdrag*. Exempel på typer/moder av *uppdrag* är: följning, sökning, målfångning, ECM/telekrig, styrning, kommunikation. Ett *uppdrag* är en instans/förekomst av en sådan mod. Eftersom olika sensorer skall kunna användas för ett *uppdrag*, bör det definieras på plattformsnivå snarare än på sensornivå. *Uppdraget* ensamt känner till vad det går ut på, det vill säga taktisk mod, målsättning, styrning, operatörsinteraktion etc. En naturlig representationsform för *uppdrag* i modern programvara är objekt, i den betydelsen detta begrepp har i



Figur 2. OODA-loopar för luft-, yt- respektive undervattensbevakning som nästlas i en huvudloop.

objekt-orienterad programmering. Varje sensor kör vanligtvis många *uppdrag* simultant, med tidsdelning (time-sharing) och/eller rumsdelning (såsom i flerkanalig digital radar), och varje sensor behöver därför ett schema som i varje tidsläge bestämmer vad (vilket *uppdrag*) som ska göras. Tidsdelningen innebär att varje uppdrag sönderdelas i många små sensoraktioner, mellan vilka uppdraget ligger i vila. Sensorerna, å andra sidan, utför de schemalagda sensoraktionerna så att varje sensoraktion utförs på sin bestämda tid och enligt den specifikation som getts i schemaelementet. Det innebär att sensorn kan byta uppgift mycket frekvent. Det innebär också att sensorn måste känna till eller kunna efterfråga 'bör'-värden för alla nödvändiga parametrar i den typ/mod som *uppdrag* tillhör.

Vilken sensor som ska användas i nästkommande sensoraktion bestäms av en plattformsbaserad schemaläggare som "förhandlar" med berörda sensorer vid planeringen av kommande sensoraktioner - varje *uppdrag* kan använda flera olika sensorer, samtidigt eller i sekvens. Eventuella resurskonflikter mellan olika *uppdrag* löses i samband med denna schemaläggning. Schemaläggaren arbetar sekventiellt, men hinner normalt med flera tiotals schemaläggningsoperationer per sekund, eftersom en typisk sensoraktion, till exempel en målföljeuppdatering, tar sensorn i anspråk under storleksordningen några få tiotals millisekunder. Schemaläggaren är uppdelad i en plattformsnivå och en sensornivå. Först görs på plattformsnivån ett sensorval. Sedan genereras en beställning till vald sensor, och denna schemalägger och parameteriserar aktionen i ett eget schema.

Operatörsinteraktion

På eller till varje plattform finns en eller flera operatörer. Dessa har att styra sin plattforms sensorsystem på en högre nivå än vad schemaläggaren gör. Några av de operationer som operatören har att utföra, förutom att följa och styra den direkta informationspresentationen, är att (1) generera nya uppdrag, (2) omprioritera mål, (3) välja direktiv för vilka åtgärder som ska vidtas vid överlastsituationer i sensorerna, (4) bestämma regler för hur olika typer av situationer skall hanteras, och (5) välja policy för energiemitteringen (främst från radarstrålningen). Punkterna 3-5 samlas vanligen ihop i sensorstyrningspolicyn, som var och en är en uppsättning regler och principer för plattformens sensoranvändning. Operatören har flera sådana policyn att välja mellan, och han kan byta sensorstyrpolicy när han så önskar och på taktiska grunder. På så sätt motsvaras sensorpolicyn av dagens modbegrepp. För presentationen av information från situation- och hotanalysen krävs dessutom en uppsättning presentationssätt som på lämpligt sätt kan väljas av operatören. Ett sådant presentationssätt är den temporala hotdisplayen [LSPAA].
Antalet operatörer per plattform skall kunna variera mellan noll och flera. Användare som ej är placerade i någon plattform skall kunna utnyttja systemets resurser genom att beställa tjänster av olika slag, se nedan under Tjänstearkitektur.

Flersensorsystem

På varje plattform kan finnas flera sensorsystem, till exempel radar (som kan vara digital), IRST-sensor, radarvarnare och radarsignalspaningsutrustning. På plattformsnivå, och i vissa fall på

sensornivå, finns kinematisk och attributbaserad målinformation i form av målspår med tillhörande attribut ([BP], kap 8). Fusionen av sådan målinformation kan realiseras med två metoder, distribuerad respektive central målföljning. Målföljeprocedurerna utgörs av IMM-filter ([BP], kap 4)], som är multipla Kalmanfilter i vilka olika målmodeller samtidigt underhålls. För fusion av lokala målspår används kovariansintersektion [NJU], som är en fusionsmetod på målspårsnivå. Målspårsfusion framtvingar hanteringen av ett antal metodproblem, för bland annat dataassociation ([BP], kap 6-7), begränsad korrelation [Han], återmatning från centrala till lokala målspår och adaptiv samplingshastighet [Kar]. Implementering av dessa problem beskrivs i [SHL].

Flerplattformsapplikationer

Varje *uppdrag* hanteras normalt på en plattform, varvid en plattformsbaserad schemaläggare kommunicerar med plattformens individuella sensorer och operatörer vid schemaläggning och sensoraktion. I flerplattformssystem finns dock behov av att kunna anlita sensorer på annan plattform, eller att samköra uppdrag som finns i olika plattformar. För detta behövs en uppdragshantering också på nätnivå, vilket kan realiseras av en stödjande funktion på varje plattform, en Net Task Manager (NTM). En NTM har bland annat förmåga att välja plattform och sensor för planerad sensoraktion utifrån ett antal faktorer. Dessa är huvudsakligen förväntad förbättrad målspårskvalitet, aktuell och förväntad lastkapacitet hos användbara sensorer, operatörens sensorpolicy samt uppdragets prioritet.

Tjänstearkitektur

I ett tjänstebaserad arkitektur kan ett *uppdrag* initieras av en godtycklig operatör på en godtycklig plattform eller plats, åtminstone i princip. I verkligheten beror ju detta bland annat på vilka rättigheter som getts åt olika operatörsroller. För tjänster som kräver användning av sensorer på andra plattformar än operatörens egen, kan en tjänst behöva initieras för att utföra detta uppdrag. En sådan tjänst kräver att flera funktioner utförs som finns i olika (geografiskt distribuerade) noder. Exempelvis kan en söktjänst kräva medverkan av funktioner för resursallokering, datafusion, sensorstyrning och ett antal sensorer. Dessa funktioner måste kommunicera med varandra på något sätt, till exempel via någon metod för programdistribution.

Programdistribution

En datafusionsnod skall vara komponent i ett nätverk där andra noder kan bestå av till exempel operatörsarbetsplatser, sensorer och resursallokerare. Datafusionsnoden själv antas vara installerad i en plattform. För varje instansiering av en tjänst sammansätts ett nät av noder med de funktioner som behövs i denna tjänst. Varaktigheten av ett sådant löst kopplat nät är relativt kort; det existerar bara under den tid tjänsten existerar. Enstaka noder i detta nätverk kan också vara temporära och existera under en del av den tid som tjänsten existerar; detta gäller om till exempel en viss sensor är tillgänglig endast under en del av det tidsintervall tjänsten existerar.

I ett längre driftsperspektiv måste varje nod i nätverket vara utbytbar, gamla noder ska kunna tas bort och nya läggas till. Detta gäller exempelvis om en viss sensor byts ut i en plattform, eller om en ny operatörsarbetsplats installeras. För att detta ska kunna fungera krävs en öppen systemarkitektur och hantering av flertrådiga processer.

Simuleringskrav

Datafusion och sensorstyrning skall kunna hantera scenarior med flera mål. Den simuleringsutrustning som behövs för att köra datafusionsnoden måste dessutom kunna hantera ett godtyckligt antal mål, plattformar, sensorer, uppdrag, observationer och operatörer.

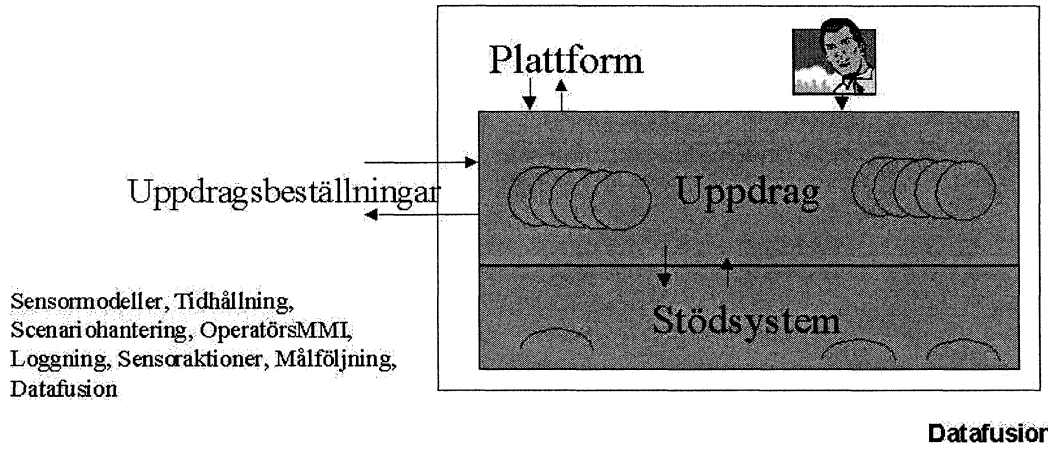
Konstruktionsmetodik

För att kunna realisera en programkonstruktion som klarar ovanstående omfattande kravlista krävs metodik för hantering av komplexa system. För programvarukonstruktionen och underhållsarbetet krävs en hållbar målsättning, en enhetlig programmeringsmetodik, en objektparadigm, en hierarkisk algoritmansats, datastrukturer samt ett antal regler för konstruktion, verktygsanvändning, dokumentation, språkval och plattformsval. I objektparadigmen beskrivs objektansats och struktur. Med begreppet hierarkisk algoritmansats avses en hierarki av nivåer med en väl vald huvudalgoritm på högsta nivån. Datastrukturerna bör väljas med omsorg och vara få till antalet.

Datafusionsnoden

Följande texter och bilder beskriver på ett översiktligt sätt den implementerade programvaran. Den tekniskt intresserade hänvisas till [Str1].

Datafusionsnoden är en 'maskin' för generering och körning av OODA-loopar (beslutsloopar), här kallade *uppdrag*. Den är anpassad till moderna sensorer såsom ESA radar, IRST, med flera. Varje *uppdrag* har en mod/typ till exempel någon typ av multimålföljning, enkelmålföljning, ECM, kommunikation, vapenstyrning. Varje *uppdrag* kan använda en eller flera sensorer, och anlitar då dessa under korta tidsintervall avskilda med längre tidsintervall i viloläge. Varje enskild sensoraktion schemaläggs i sensorn och beräknas pågå under ett tidsintervall som kan variera mellan några millisekunder och någon sekund. Efter varje sensoraktion utförs en efterbearbetning som kan omfatta signalbearbetning, observationshantering, dataassociation, målspårsuppdatering, fusion av målspårsestimat, attributfusion, operatörsinteraktion, uppdragsbearbetning, sensorval och aktionsschemaläggning. I denna efterbearbetning ingår val av sensor för nästkommande sensormätning liksom schemaläggning i vald sensor. Datafusionsnoden är därför också en 'maskin' för generering och körning av sensoraktioner. Dessa tolkas av varje sensor i kronologisk ordning, enligt sensorns individuella schema, och nya aktioner schemaläggs allteftersom de gamla exekveras. *Uppdragen* hanteras på plattformsnivå. Varje *uppdrag* kan i termer av agentteknik betraktas som en agent som beställer tjänster hos sensorerna som alltså är producerande agenter. Operatören/Operatörerna övervakar presentationen av insamlade och fusionerade data, samt styr på en hög nivå användningen av plattformens sensorsystem, i form av en sensorpolicy. Hastigheten hos schemalagda och exekverade högnivåsensoraktioner hos en plattform kan vara 1-50 Hz, men sensoraktionerna tillhöriga ett visst uppdrag exekveras mera sällan, normalt i hastigheten 0.1-1 Hz. Likheten med ett operativsystem gör att datafusionsnoden kan sägas vara ett Sensor Management & Data Fusion Operating System, ett SMDFOS , se figur 3.

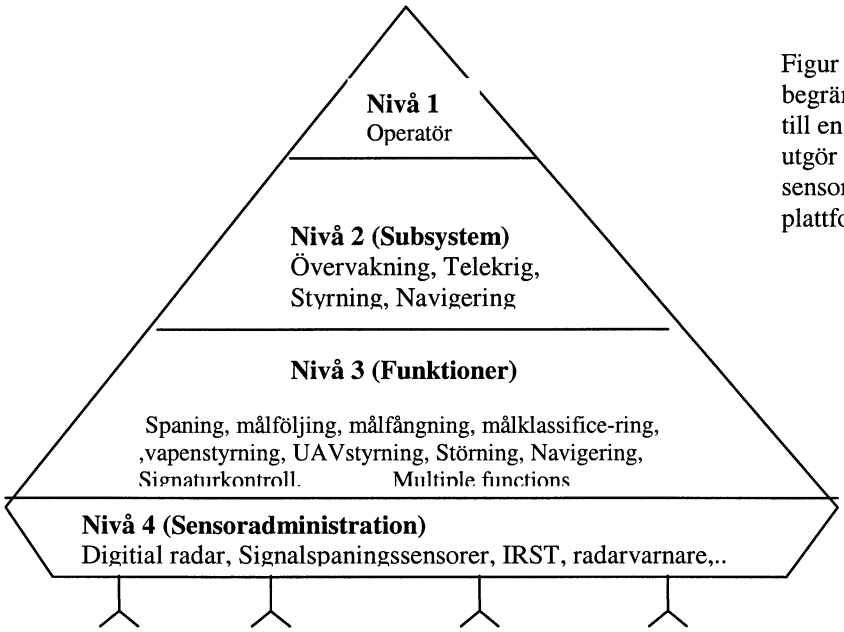


Figur 3. Sensor Management and Data Fusions Operating System, SMDFOS

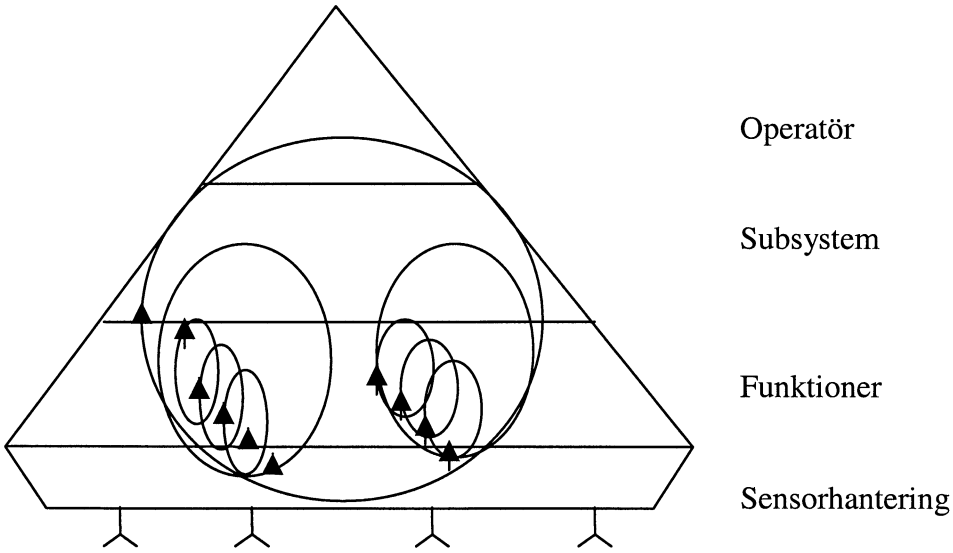
Uppdragen kan sägas vara ett slags agenter som beställer 'tjänster' av sensorerna. Dessa producerar data (observationer med mera) som levereras tillbaka till uppdragen som konsumerar dessa data. Båda typerna av agenter (producenter och konsumenter) lever på en 'marknad' där tillgång och efterfrågan på resurser och kapacitet spelar en avgörande roll. Detta spel mellan konsumenter och producenter har i andra sammanhang modellerats med multi-agent-teknik [WEI].

Uppdragen realiseras såsom objekt av olika klasser av sensorberoende uppgifter ombord på plattformen. Dessa klasser kan i sin tur indelas i olika makroklasser som kan associeras till olika delsystem ombord på plattform. Hur denna klasshierarki ser ut illustreras i figur 4a, där också ett antal sensorer är antydda underst i figuren.

De uppdrag på plattformen som kräver direkt sensorbearbetning konkurrerar om de tillgängliga sensorerna. Eftersom de tillgängliga sensorerna är färre än uppdragen, se figur 4b, krävs någon form av konflikthantering. Regler och principer krävs för att fördela resurserna på ett optimalt sätt, och de behövs också för att specificera hur olika typer av uppdrag ska skötas, vilka sensorer som ska användas, vad som ska hända vid sensoröverlast med mera. Operatören har ett antal strategier eller sensorstyrningspolicyn att välja mellan för denna hantering, och han/hon kan i varje ögonblick kunna byta policy.



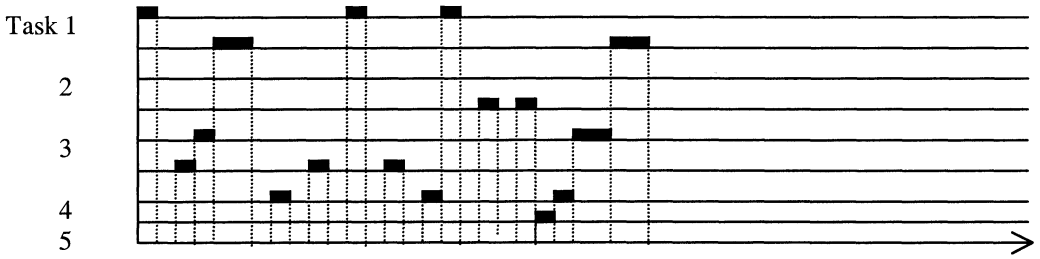
Figur 4a. Multipla funktioner på nivå 3 anlitar ett begränsat antal sensorer på nivå 4 vilket ger upphov till en resursbegränsning. Funktionerna på nivå 3 utgör nedersta nivån i en hierarki av sensorfunktioner, där en indelningsgrund är plattformens subsystem.



Figur 4b. Nästlade OODA-loopar/funktioner i flera nivåer. Konflikter uppstår vid resursbegränsning

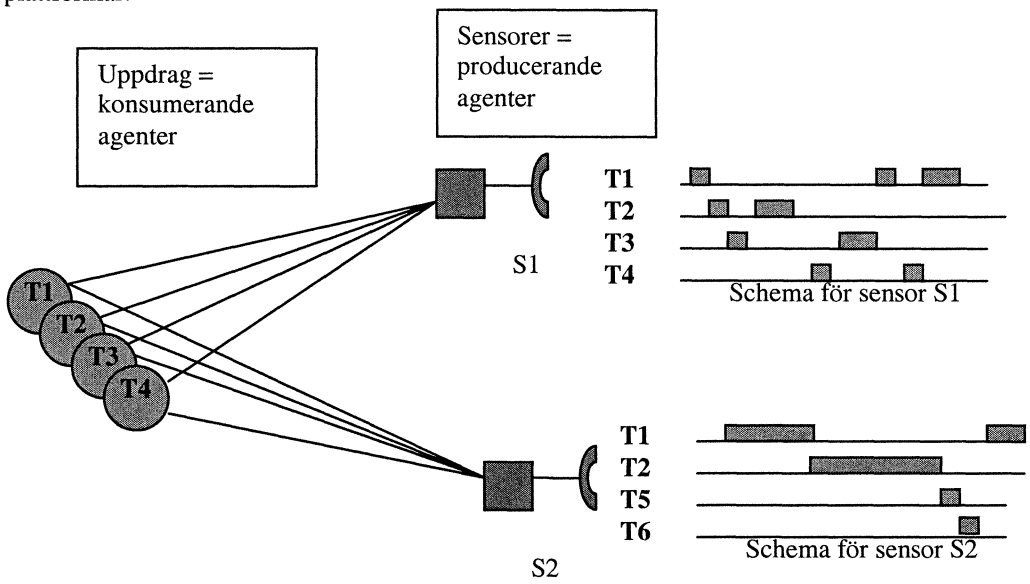
SMDFOS har för närvarande laboratoriestatus.

Varje uppdrag bearbetas i en serie korta processer, och mellan varje process ligger det i vila. Processerna schemaläggs vanligen med ett steg i taget. Varje process inleds med en sensorbearbetning som följs av operationer för signalbehandling (förenklas), målföljning, datafusion, situationsanalys, presentation och interaktion med operatör, schemaläggning samt eventuellt andra specialfunktioner. Varje sensor har ett eget schema för planerade och schemalagda aktioner, där varje schemaelement tillhör ett visst uppdrag. En sensorns schema kan se ut som figur 5.



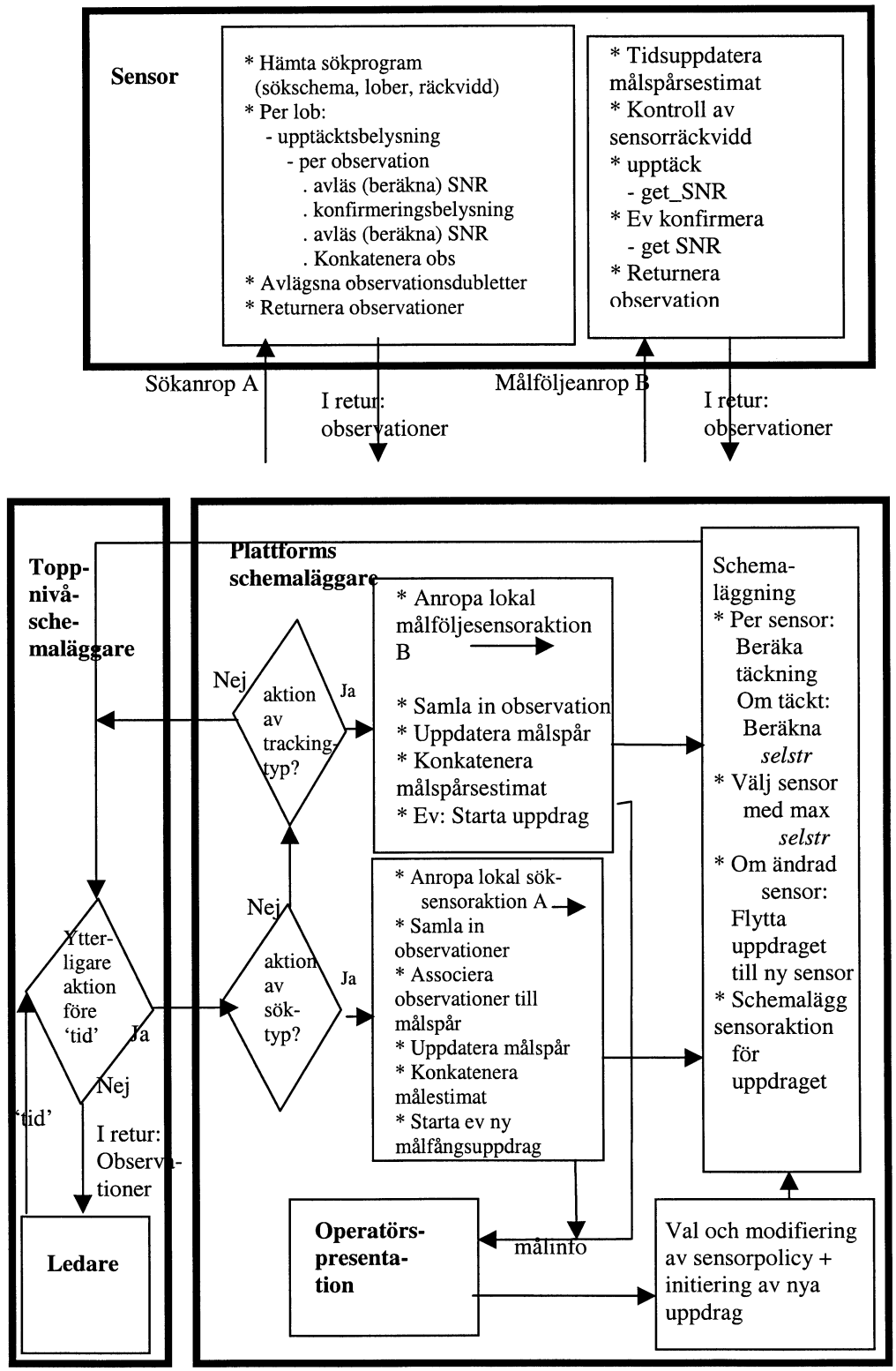
Figur 5. Tentativt schema för en flerfunktionssensor. Uppdragen kan vara av olika typ (mod).

Figur 6 visar en översiktlig bild över datafusionsnoden. Operatör eller annan beställare genererar OODA-loopar och uppdrag. Dessa kan vara av flera olika slag till exempel övervakning, kommunikation eller vapenstyrning. Varje uppdrag är i normalfallet oberoende av andra uppdrag, vilket innebär att den självständigt söker uppfylla sina mål. Uppdragsbeställaren kan vara ombord på plattformen eller utanför denna. Antalet plattformar, mål, uppdrag och sensorer är obegränsat. Varje uppdrag kan involvera flera sensorer på operatörens egen plattform och eventuellt uppdrag på andra plattformar.



Figur 6. Uppdragen och OODA-looparna representeras som agenter som konsumerar information vilken producerats av sensorerna. Dessa är producerande agenter. Varje uppdrag kan beställa information från olika sensorer, och varje sensor utför aktioner som beställts av olika uppdrag.

Figur 7 nedan visar ett blockschema över hanteringen av uppdrag i en enstaka datafusionsnod. Det grafiska användarinterfacet (GIU) anropar schemaläggaren en gång per tidsenhet (för närvarande: sekund). Schemaläggaren undersöker om det globala schemat innehåller någon kvarvarande sensoraktivitet som skall startas denna sekund. För varje sådant schemaelement startas den utpekade sensoraktiviteten som alltså tillhör visst uppdrag. För vissa typer av aktiviteter, bland annat sökning och målföljning, kan utdata förväntas från sensorn. Dessa omhändertas i en rad av efterföljande processer, bland annat associering med till uppdraget kopplade målspår, uppdatering av till observationer kopplade målspår, klassificeringsarbete av dessa målspår, generering av uppdrag för initiering av nya målspår, val av sensor för nästa sensoraktivitet, schemaläggning av denna samt returnering. Alla målspårsestimat som tillhör uppdrag till vilka en sensoraktion utförts returneras till GUI. För anslutning av ett antal operatörsarbetsplatser till en plattforms schemaläggare används teknik för programdistribution (CORBA) och flertrådighet.



Figur 7. Blockschema över datafusionsnoden. Det finns flera sensorer och flera operatörsplatser per plattform, och det finns flera plattformar. Variabeln *selstr* som används vid schemaläggningen förklaras i nästa avsnitt.

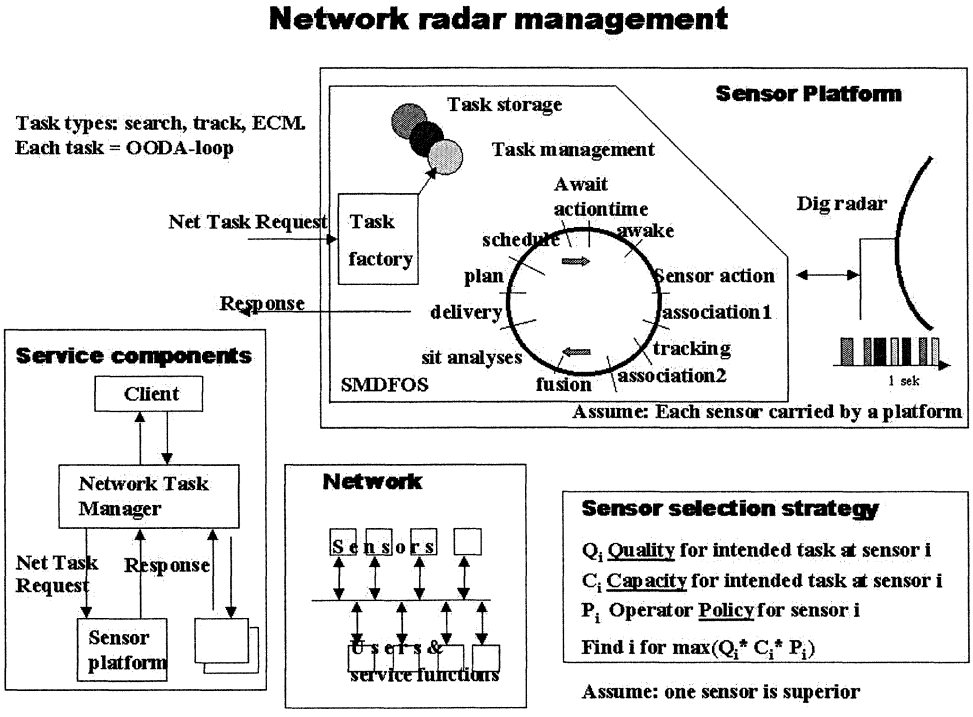
.

Användning av datafusionsnoden i plattformsnätverk

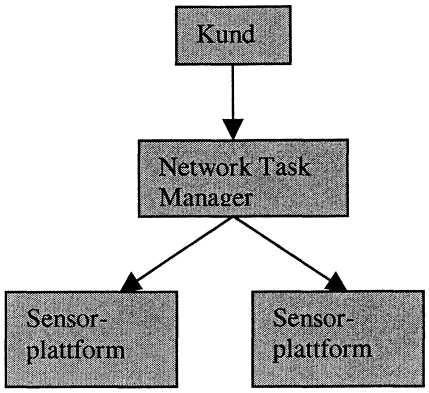
Datafusionsnoden kan utgöra datafusionscentral i varje plattform, och flera datafusionsnoder kan sammankopplas i nätverk. Hur denna inkoppling ser ut, och hur datafusionsnoden fungerar i ett sådant nätverk presenteras i detta kapitel. Figur 8 är en sammanfattning. Figuren visar att noden kan hantera ett flertal uppdrag i en viss plattform. Varje uppdrag fungerar som en automatisk OODA-loop, och har därför getts en cirkulär form. Ett task kan genereras av operatör eller automatik på plattformen, eller på beställning av någon aktör utanför plattformen - denne blir då 'klient' till detta uppdrag. För klienter utanför plattformen finns särskilda administrativa funktioner, Network Task Managers NTM, som administrerar beställningar i nätet. Hur beställningar hanteras av NTM beror på vilken systemarkitektur som tillämpas; figuren visar en algoritm för selektering av lämplig plattform för visst uppdrag. Denna algoritm selekterar viss sensor genom att välja den sensor som ger maximalt värde på en produkt av faktorer för mätkvalitet, lastkapacitet och sensortillgänglighet. Denna algoritm har använts i experiment 2, se nedan.

Funktionaliteten i en NTM beror på vilken typ av systemarkitektur [Str2] som används. För en förhandlingsbaserad systemarkitektur behövs en arbetsuppdelning mellan noderna som beskrivs i figur 9a och 9c. Figur 9c visar hur de olika processtegen hos en task kan utföras i datafusionsnoden respektive sensorn. På motsvarande sätt har arbetsfördelningen mellan datafusionsnod och sensorer på samma plattform visualiserats i figur 9b.

Varje uppdrag hör hemma i viss plattform, men det kan beställa uppdrag och sensoraktioner i annan plattform. Samverkan mellan sensorstyrning på nätverks-, plattforms- och sensornivå illustreras i figurerna 8 och 9. Speciellt visas i figur 9c hur ett uppdrag på nätnivå leder till uppdragsbeställning i en eller flera plattformar som i sin tur genererar sensoraktioner i enstaka sensor.

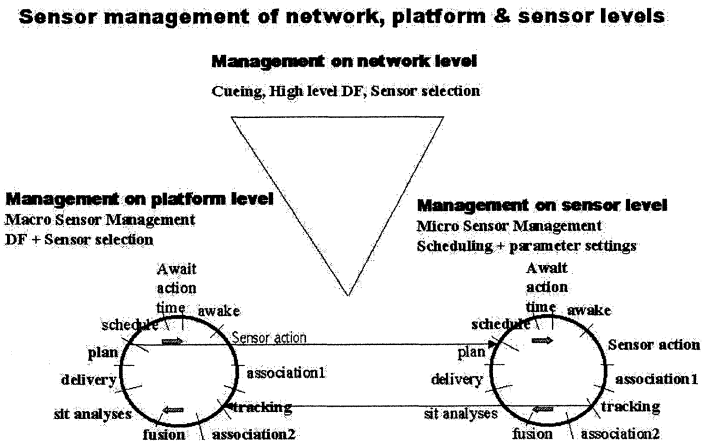


Figur 8. Figur hämtad från [Str3]. Övre delen visar en sensorplattform med en datafusionsnod och en sensor. Till noden hör ett antal uppdrag. Undre vänstra bilden visar hur en 'kund' i nätverket via en nätverkshanterare anlitar en datafusionsnod i en sensorplattform. Mittersta undre bilden visar att nätet kan innehålla flera sensorer och plattformar. Högra undre bilden visar algoritmen för sensorval, vars indata utgörs av förväntad kvalitetsfaktor, lastkapacitet och operatörens sensorpolicy.

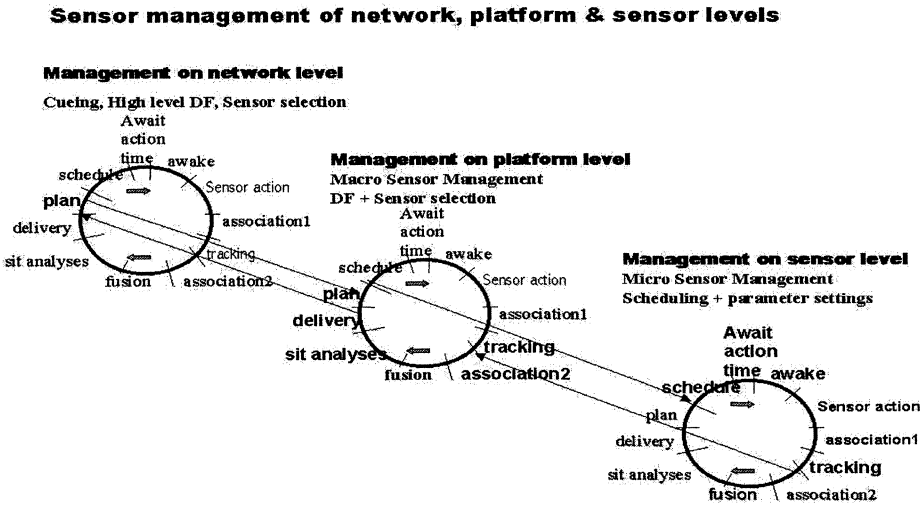


1. Beställning från kund.
2. Välj sensor med störst selekteringsstyrka, se figur 8.
3. Generera uppdragsbeställning hos plattformen till vald sensor.
4. Vald plattform genererar uppdrag och startar sensoraktioner.
5. Valda plattformen levererar till NTM men kan avsluta uppdraget när som helst.
6. NTM kontrollerar att leveranser inkommer. Om leveranser upphör väljs ny sensor med hjälp av selekteringsalgoritmen.
7. En alternativ plattform får beställning och startar dataleveranser.

Figur 9a. Ett sätt att hantera uppdrag som samverkar med sensorer i flera plattformar är att inrätta en Network Task Manager i någon plattform. Denna har då till uppgift att beställa uppdrag i andra plattformar samt kontrollera att dessa fungerar enligt plan.



Figur 9b. Figur hämtad från [Str3]. Ett uppdrag beställer aktioner i sensorerna. Dessa aktioner kan omfatta sensorarbete med också lokal målföljning etcetera. I vissa fall kan uppdraget delegera hela uppdragshanteringen till en viss sensor.



Figur 9c. Figur hämtad från [Str3]. Figuren illustrerar hur ett nätomfattande uppdrag i en NTM (Network Task Manager) kan beställa uppdrag i enstaka plattform som sedan beställer aktioner hos enstaka sensornivå.

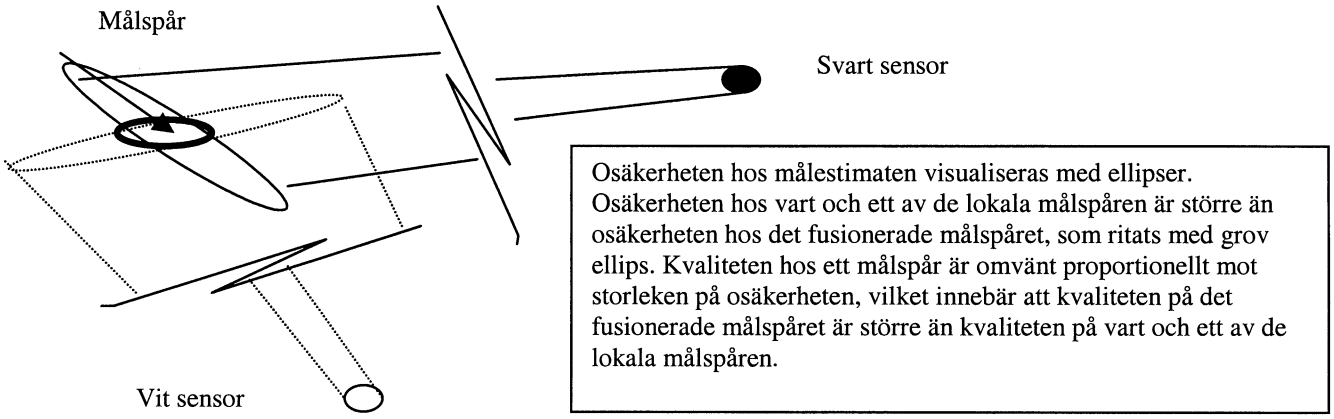
Experiment 1

Bakgrund

Vid målföljning upprätthålls ett estimat av det följda målets position och beteende. Vid varje mätning erhålls ett kvalitetsmått på sensorobservationen, och vid en uppdatering erhålls ett kvalitetsmått på den kinematiska målrepresentationen (detta uttryck används i stället för målspårsestimat eller track). Dessa kan beräknas även för simulerad mätning. Även fusionerade målrepresentationer kan tillskrivas kvalitetsmått. Den överlägset bästa algoritmen för målspårsfusion är kovariansintersektion [NJU]. Det finns flera kvalitetsmått, baserade på olika parametrar i målrepresentationen (såsom avstånd, avståndshastighet, bäring) liksom för hela kovariansmatrisen. Figur 10 illustrerar den högre kvaliteten på en fusionerad målrepresentation jämfört med kvaliteten på lokala målrepresentationer.

Mål

Styrning för maximal kvalitetsförbättring.



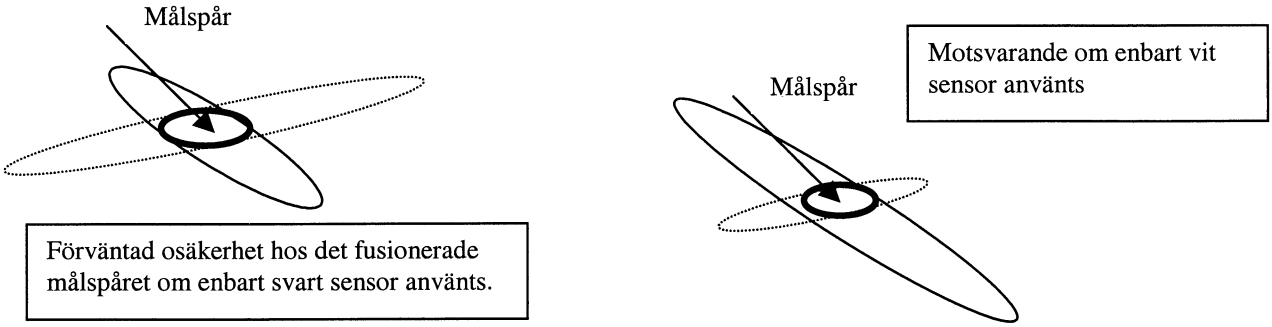
Figur 10. Osäkerhetsellipser hos observationer kan förbättras genom fusion.

Metod

Eftersom målestimatets kvalitet är en funktion av flera samverkande faktorer (bland annat målbeteende, tidsintervall mellan mätningar, typ och tidsintervall för målspårsfusion och målspårsåterkoppling), kan det vara svårt att analytiskt bestämma vilken kvalitetsförbättring som erhålls med mätning i ena eller andra sensorn. Däremot kan mätuppdatering från en simulerad mätning i den ena sensorn och en tidsuppdatering i övriga sensorer användas som grund för att bestämma den förväntade kvalitetsförbättringen hos det fusionerade estimatet sedan det beräknats enligt ovan. Om detta görs för alla sensorerna kan den sensor identifieras som ger det största kvalitetsförbättringen vid nästa mättillfälle. Figur 11 illustrerar kvaliteten hos det fusionerade målspåret då uppdatering har gjorts med den ena respektive andra sensorn.

Detta resonemang kan utvidgas till att gälla ett godtyckligt antal sensorer. Notation och algoritm för motsvarande sensorhantering följer då enligt nedan.

Lokalt målspår M*i* underhålls av sensor S*i*.
M*i* har kvaliteten Q*i*.
Simulerad mätning + tidsuppdaterat målspår M*ie(t)* har kvaliteten Q*ie*.
Enbart tidsuppdaterat målspår *Miu(t)* har kvaliteten Q*iu*.
Det fusionerade målspåret *Cie* har kvaliteten Q*ci*, betraktat från S*i*.
Det fusionerade målspåret är en funktion av mätuppdatering från Si och tidsuppdatering från övriga sensorer, dvs
C*ie(t)* = f(M*ie(t)*,M*ju(t)*), all *j*#*i*.
C*ie(t)* har kvaliteten Q*cie(t)*.
Välj sensor i, där i = arg $\max_i$(Qc*ie*-Qc*i*).



Figur 11. Kvaliteten hos det fusionerade målspåret beror på vilken sensor som används. Se också figur 10.

Försöksuppsättning

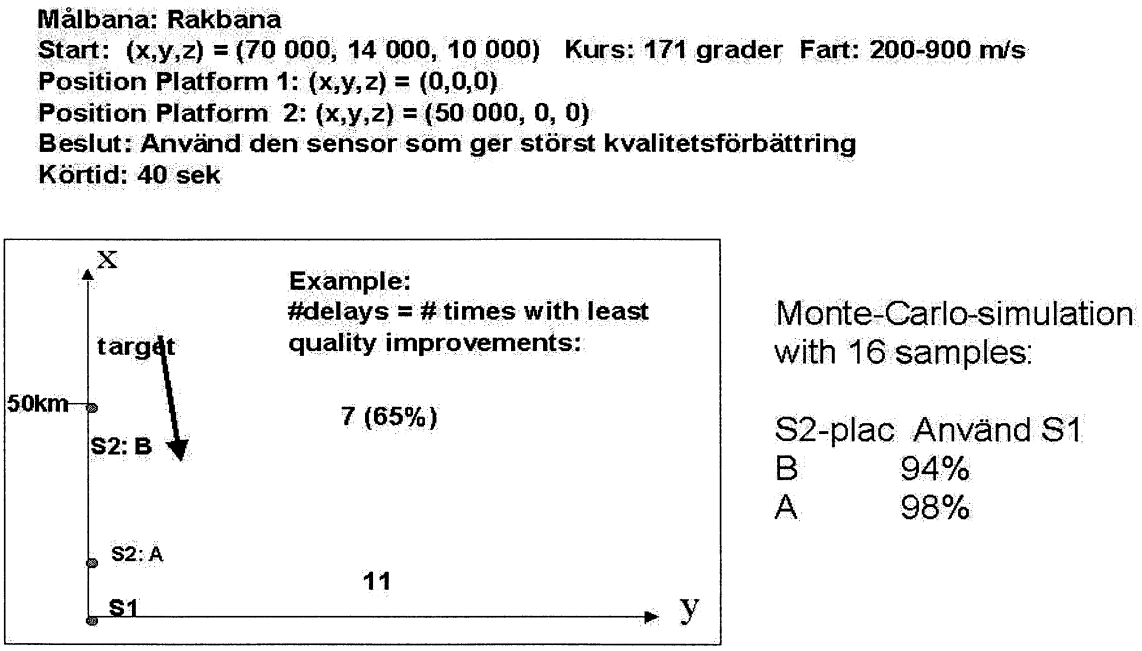
Om flera sensorer täcker ett mål kan det vara önskvärt att enbart använda enbart den som vid nästa mättillfälle förväntas ge störst ökning av målspårskvaliteten hos det fusionerade målspåret. I experimentet demonstreras detta förhållande genom att vid varje mättillfälle välja den av två sensorer som förväntas ge störst kvalitetsförbättring hos ett fusionerat spår, varvid den sensor som inte valts vilar. Tidsintervallet mellan konsekutiva mätningar är konstant och lika med en sekund.

Mätningar

Algoritmen tillämpades i ett scenario som beskrivs av figur 11 nedan som också visar resultatet. Det bör observeras att den fjärmaste sensorn enligt algoritmen används mindre ju närmare målet den andra sensorn står, vilket verkar rimligt. Antalet gånger som den närmaste sensorn förväntas ge störst kvalitetsförbättring ökar alltså ju närmare den står målet.

Resultat

Resultatet i figur 12 visar dock att inga drastiska skillnader uppträder om man flyttar sensorn närmare målet. En förklaring kan vara att den kvalitetsförbättring som den mest overksamma sensorn förväntas ge ökar ju längre tid som den står overksam.



Figur 12. Figur hämtad från [Str3]. Då sensor S2 placeras långt från målet får den 2% fler mätningar än S1. Då den placeras närmare målet får den 6% fler mätningar än S1.

Experiment 2

Bakgrund

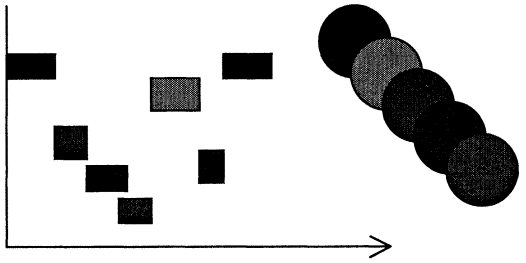
I detta experiment antas varje sensoruppdrag kunna bli utfört av en eller flera plattformar, vardera med en sensor. Sensorerna utför lokal målföljning mot samma mål. Vid varje tidpunkt används enbart en sensor, nämligen den som är minst belastad. Det så uppdaterade målestimatet överförs därefter till den andra sensorn. Experimentet illustrerar dessutom hur denna sensorselekteringsalgoritm kan ta hänsyn också till kvalitet (jämför Experiment 1) och operatörernas sensorpolicy.

Mål

Lastbalansering.

Metod

En förhandlingsbaserad systemarkitektkur (se Str2) har använts, se figur 9a och 9c. Algoritmen för uppdragshantering och sensorselektering framgår av figur 13.



Algoritm för beräkning av selekteringsstyrka:

Välj sensor i, där i=arg max(Q_i*C_i*P_i),
Q_i = *förväntad kvalitet hos uppdateringen*
C_i = *lastkapacitet hos sensor i,*
P_i = *sensorstyrpolicy för sensor i hos plattformen*

Algoritm för uppdragshantering:

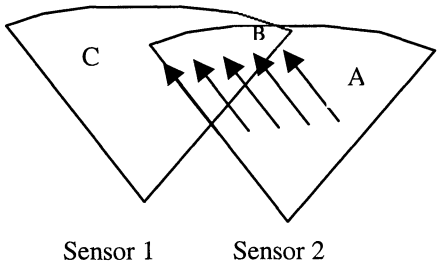
Utför repetetivt:
- Välj den sensoraktion som utpekas av det första elementet i det globala schemat.
- Utför sensoraktionen i utpekad sensor vid den tidpunkt som anges i schemaelementet.
- Utför tillhörande efter-bearbetningar (målföljning, fusion, sitiuationsanalys). Dessa specificeras i mod/typ för det uppdrag som "äger" sensoraktionen.
- Välj sensor för nästa sensoraktion*.
- Schemalägg aktionen i schemat för vald sensor samt det globala schemat

Figur 13. Algoritmer för uppdragshantering och sensorval.

Försöksuppsättning

Två plattformar bär var sin digitala radarapparat, med räckvidd enligt figur 14. Sensorernas räckvidder överlappar delvis varandra. Antalet mål är 20, och varje mål motsvarar noll eller ett målföljningsuppdrag hos varje sensor. Sensorerna arbete påverkas av fyra faktorer, nämligen sensorbelastning, förväntad förbättring av målspårskvalitet, operatörens sensorstyrningspolicy och plattformens prioritering av målet. Enbart en sensor används vid varje tidpunkt för uppdatering, och valet av sensor påverkas av en algoritm som beräknar den så kallad sensorstyrkan för varje sensor, samt väljer den sensor som ger högst sensorstyrka.

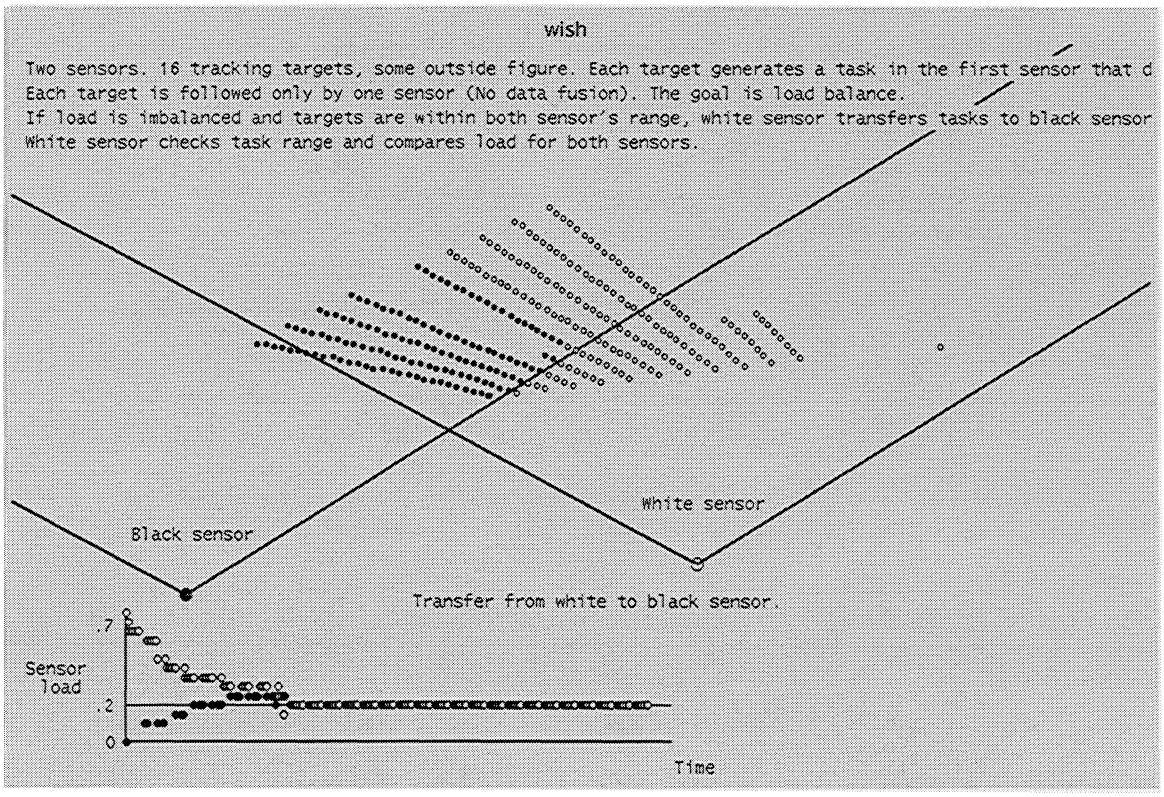
Figur 14 nedan visar hur scenariot ser ut. Målen rör sig så att de flyger från området A in i området B. Från början finns inga mål i området B eller C. Måltätheten är alltså större till höger än till vänster i scenariot.



Figur 14. Två sensorer enligt figur. Ett antal mål som flyger enligt pilarna. Från början täcks målen enbart av sensor 2, tills de flyger in i täckningsområdet för sensor 1. Från början har sensor 2 alla målföljeuppdrag, sedan överförs de efterhand till sensor 1. Lasten ökar hos sensor 1 och minskar hos sensor 2.

Mätningar och Resultat

Figur 15 nedan är en visualisering av sensorval och mätningar. Varje observation är svart eller vit beroende på om svart eller vit sensor använts för mätningen. Figuren visar att svart sensor från början är obelastad, samt att den övertar uppdrag från vit sensor i den takt motsvarande mål flyger in i täckningsområdet för svart sensor. När lasten så småningom (efter cirka fem uppdragsövertaganden) är lika hos svart och vit sensor upphör flyttningen av uppdrag från vit till svart sensor. I experimentet har antagandet gjorts att kvalitet, sensorpolicy och prioritering är lika hos de båda sensorerna för varje mål.



Figur 15. Täckningssektorerna för varje sensor är 120 grader. Målet rör sig snett uppåt vänster. Antalet mål är 20, varav en del ligger på ett avstånd som ej täcks i figuren. Målföljningen övertas av vänster (=svart) sensor allt eftersom de kommer in i dess täckningsområde, men detta övertagande upphör då lasten (se nedre delen av figuren) i de båda sensorerna blivit lika stor vilket gäller för mål nummer sju räknat nedifrån. Det femte, elfte och tolfte målet tappas i målföljealgoritmen på grund av en brist i associationsprocessen.

Taxonomi

Här definieras en taxonomi för datafusionsnod och scenarioomgivning. Detta görs i form av nedanstående syntax med kommentarer. Det mest centrala begreppet i taxonomin är uppgiftsbegreppet. Till detta kan en mängd andra begrepp associeras.

Nedan beskrivs den viktigaste kunskapsstrukturen i datafusionsnoden i en BNF-liknande syntax [AU]. Startsymbol (toppsymbol) är '**globalscheduler**'. Varje symbol utom *terminalerna* definieras i en klausul. Varje symbol med sufffixet 's' är en mängd. Symboler med suffix 'x' är objekt. Stjärnan '*' markerar mängd av objekt, och plustecknet '+' markerar noll eller ett element. I varje klausul finns en vänstersymbol och ett antal högersymboler; vilka kan vara objektreferenser. Kommentarer följer efter klausulerna. (Alla mängder i programmet implementeras såsom atomära listor.) Scenarioomgivningen beskrivs i klausul 1. Övriga klausuler definierar datafusionsnoden.

Klausuler

1. (**globalscheduler** platforms targets)
2. (platforms platformx')
3. (targets targetx*)
4. (platformx tasks scheduler sensors operators targetx)
5. (tasks taskx*)
6. (scheduler schedelx*)
7. (sensors sensorx*)
8. (operators operatorx')
9. (targetx *position OP*)
10. (taskx mod tracks+ spacevol observations associations *OWN*)
11. (schedelx *StartTime StopTime* taskx sensorx schedelx)
12. (sensorx load schedelx* *coverage OP*)
13. (operatorx *SensorPolicy GraphicalUserInterface)*
14. (mod *EnumMode*)
15. (tracks trackx*)
16. (spacevol lobex*)
17. (trackx localtrackx *Q* fusedtrackx+ *fusedQ*+ othersensorstrack+ observationx+)
18. (localtrackx *immtrack*)
19. (fusedtrackx *immtrac*k)
20. (othersensorstrack trackx*)
21. (observations observationx*)
22. (associations associationx*)
23. (associationx trackx observationx)
24. (observationx *MeasuredData*)

25. (lobex *direction width*)

26. (load *CurrentLoad PredictedLoad*)

Kommentarer till klausulerna

1. Toppklausul.
2,3,5,7,8,15,21,22. Varje mängd består av element som definieras.
6. Varje mängd av schemaelement utgör ett schema.
10. Tasks/Uppdrag finns av olika typ (mod). Symbolen *OWN* representerar variabler om uppdraget själv, t ex termineringsvillkor, födelsetid, presentationsdisplay och källa.
9,12: *OP* = Other Parameters, ej specificerade. Beror på sensortyp.
11. Schemaelement; skapas för att veta vid vilken tidpunkt en schemaaktion skall utföras.
11. Den andra symbolen 'schedelx' refererar till ett element i schemat hos en lokal sensor alternativt till ett element i det globala schemat, för element i globala respektive lokala scheman.
11. Varje schemaelement existerar endast under en enda viloperiod hos ett uppdrag.
14. Hittills har två typer av uppdrag definierats, målföljning och TrackWhileScan.
12. Hittills har enbart sensormodell för ESA-radar definierats.
17. *Q* och *fusedQ* är variabler för något kvalitetsmått hos det lokala resp fusionerade målspåret (och kan uttryckas t ex med hjälp av uncertainty determinant).
17. Element med suffix '+' finns endast om uppdraget fusionerar målspåret.
18,19. *immtrack* är målspårsestimat av IMM-typ.
17. othersensorstrack är en lista av tracks där varje sensor bidrar med ett track.
21-22. Dessa mängder är temporära och existerar endast under ett begränsat tidsintervall inom varje arbetsperiod (se nedan); de bildas under sensoraktionen och används under målföljeprocessen. Finns endast i vissa typer av uppdrag.
25. Avser mottagande lob.
1-26. Klausulerna kan betraktas som en grammatik för ett språk, men beskriver också alla data - eller den datauppsättning - som finns i noden vid varje godtycklig tidpunkt. Språket utgörs alltså av mängden möjliga datauppsättningar i programmet. Grammatiken är den mest kompakta beskrivning som kan konstrueras för de data som finns i noden. En parser kan genereras ur grammatiken, men det befintliga programmet utgör också en parser.

Huvudalgoritm

Programmets huvudfunktion är att processa uppdrag. Ett uppdrag beställer sensorfunktioner och kan ingå i en tjänst. Exempel på uppdrag är sökning efter flygfarkoster i en avgränsad luftvolym, målfångning, följning av enstaka flygplan, störning av en annans radar, kommunikation med annan farkost, telekrigsåtgärd och UAV-styrning. Ett uppdrag har en begränsad livstid, ofta några minuter. Uppdraget kan genereras av operatören, farkosten, annat uppdrag, eller aktör utanför plattformen. Det finns normalt flera tiotals uppdrag samtidigt i varje plattform. Varje uppdrag kan - men måste inte - ha en defaultsensor knuten till sig; den aktuella sensoranknytningen specificeras i schemaelementet. Ett uppdrag befinner sig alltid antingen i vilotillstånd V eller arbete A, och har övergångarna A -> V och V -> A.

1. V -> A. Övergång Vila till Arbete görs när en av uppdraget tidigare schemalagd och planerad sensorhändelse har uppnått starttid.

2. A -> V. Övergång Arbete till Vila görs när uppdraget har utfört de processer som förorsakats av sensoraktionen. Normalt utgörs den sista arbetsprocessen (före övergång till vila) av planering och schemaläggning av en ny sensoraktion. Det reella tidsintervall som uppdraget tillbringar i Arbete har storleksordningen 0.1 sekund.

Andra viktiga processer är scenariogenerering, dataassociation, målföljning, målspårsfusion, sensorval, situationsbedömning och hotvärdering.

Högst ett uppdrag exekverar vid varje tidpunkt i en sensor.

Arbetsprocesserna hos uppdraget bestäms av den mod (=typ) som uppdraget tillhör.
Varje sensor har i varje godtycklig tidpunkt ett schema av planerade aktioner.

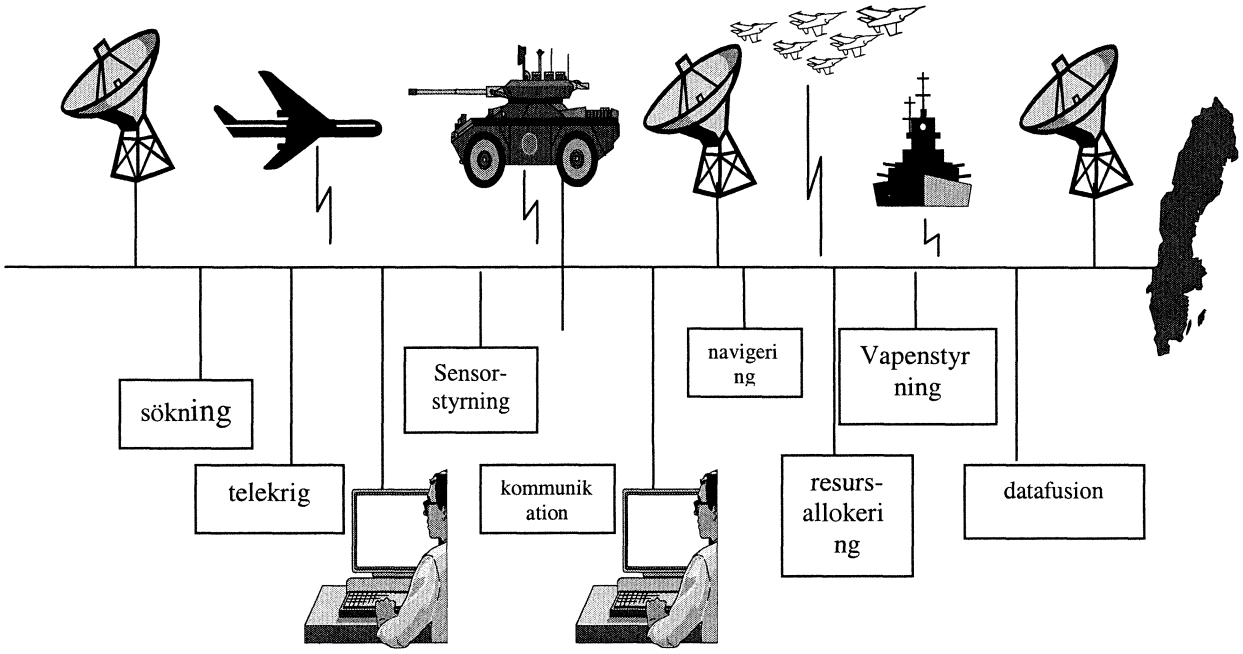
Exempel på uppdrag som genererar andra uppdrag är (detta är ej implementerat)

- ett sökuppdrag som vid upptäckt genererar ett målfångningsuppdrag
- ett målfångningsuppdrag som vid målbekräftelse genererar ett gruppföljningsuppdrag
- ett gruppföljningsuppdrag som genererar ett högupplösningsuppdrag
- ett högupplösningsuppdrag som vid måldelning genererar ett antal målföljeuppdrager
- ett målföljeuppdrag som på vissa villkor genererar ett klassificeringsuppdrag
- en operatör som generar ett sökuppdrag
- en operatör som generar ett störningsuppdrag
- en plattform som automatiskt genererar ett uppdrag för korträckviddig hotsökning
- ett (godtyckligt) uppdrag som efter mottagning av starka signaler genererar ett telekrigsuppdrag
- en insatsförberedelse som genererar ett insatsuppdrag
- ett insatsuppdrag som genererar ett vapenförberedelseuppdrag
- ett insatsuppdrag som genererar ett UAV/vapenstyrningsuppdrag
- ett insatsuppdrag som genererar ett efterbedömningsuppdrag

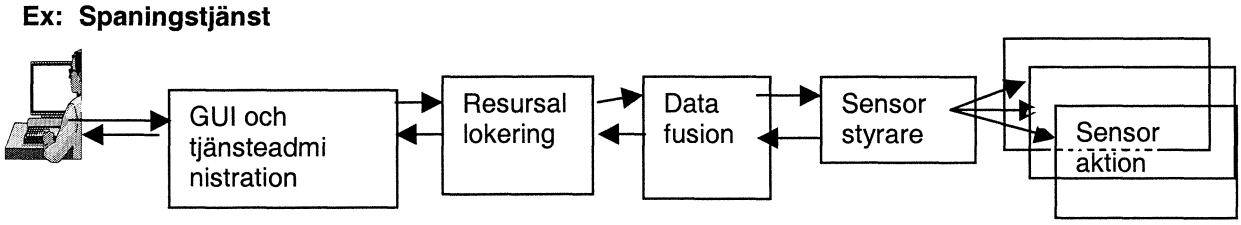
Varje OODA-loop representeras lämpligen av ett uppdrag.

Systemarkitekturer för nät och nättjänster

I det nya nätbaserade försvaret skall alla sensorer - i princip - vara tillgängliga för alla operatörer, vilket innebär att alla funktioner för ledning, informationsinsamling och verkan skall ingå i ett gemensamt ledningssystem. En praktisk operatörsaspekt är att i detta ledningssystem ge operatörerna tillgång till olika tjänster, och att varje typ av operatör har till sitt förfogande ett antal rollbaserade tjänster; detta gäller oavsett var eller i vilken plattform operatören befinner sig. Exempel på en tjänst av informationsinsamlingstyp är spaning i visst mark-, luft eller havsområde. Tjänster av denna typ kräver att för ändamålet användbara sensorer utnyttjas, oavsett om de finns på fartyg, mark eller flygplan. Varje sådan tjänst kräver då att ett antal funktioner tillhandahålls, till exempel resursallokering, datafusion, sensorstyrning och sensormätning. Varje sådan funktion kan realiseras i en komponent som tjänsten kan koppla upp sig mot, och komponenterna kan distribueras i olika plattformar, på geografiskt olika platser. Då en tjänst beställs av en operatör instansieras den. Detta innebär att till exempel datafusion vid varje tidpunkt är en funktion i ett stort antal samtidigt existerande och simultana tjänsteinstanser. Varje tjänsteinstans existerar exakt så länge som operatören håller igång den. Figur 16 visar en schematisk bild över ett nät där olika tjänster kan utföras. Figur 17 visar hur en tjänst, ett sökuppdrag, kan distribueras i ett sådant nät, med ett objektorienterat programmeringssystem.



Figur 16. Löst kopplade komponenter i nätverk



Figur 17. Programvarukomponenter som distribuerats på olika nätnoder. GUI = Graphical User Interface.

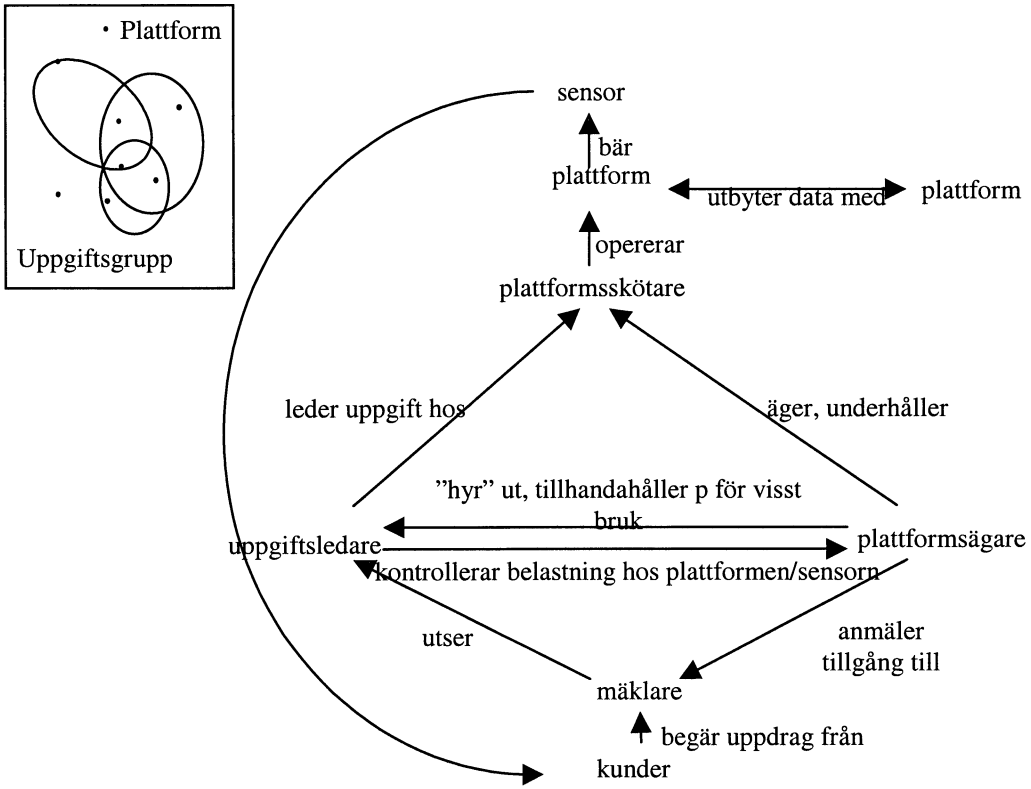
Tjänsterna utför en dynamisk komponenthantering, vilket innebär att till exempel plattformsburna sensorer kan utnyttjas av tjänsten utan att deras existens eller geografiska läge är känt av operatören. Ett sätt att konstruera sådana tjänster är att använda någon COTS-baserad metod för programdistribuering. En sådan metod är baserad på arkitekturen CORBA, och denna metod implementeras för närvarande såsom kommunikationsgränssnitt för samverkande datafusionsnoder. CORBA tycks passa väl in på de krav om interoperabilitet, funktionalitet, skalbarhet och flexibilitet som ställts i RMA-arbetet [RMA].

Datafusionsnoden har en given plats i detta sammanhang. Det är den enhet inom varje plattform som fusionerar data och styr sensorerna. Alla sensorer och operatörsplatser är kopplade till datafusionsnoden. Hanteringen av inkommande beställningar från andra plattformar liksom hanteringen av beställningar till andra plattformar handhas i datafusionsnoden.

Resursallokering

Detta avsnitt beskriver en modell för dynamisk resursallokering av sensorer och plattformar i nätverk. Utgångspunkten är att ett flertal användare, oberoende och ovetande av varandra, kan vilja använda tillgängliga sensorer för spaning eller andra uppdrag. En enskild användare kan dock aldrig veta vilka sensorer som finns eller kan göras tillgängliga för uppdraget, krävs en organisation som hjälper användaren att anskaffa användbara sensorer och plattformar. Här presenterar en organisation som bygger på begreppet mäklare. Den visualiseras i figur 18 och sammanfattas i nedanstående punkter. En grundidé är att varje resurs "ägs" av en ägare som kan hyra ut delar av den till olika användare.

· En elektriskt styrd radar samt eventuellt andra sensorer i varje plattform.
· Varje plattform kan vara luftburen, markfast, sjöburen eller satellitburen.
· Många samtidiga uppgifter (sensoruppgifter, spaningsuppgifter).
· Varje sensor utför simultant flera uppgifter (intermittent).
· Varje uppgift utförs av en grupp plattformar (uppgiftsgrupp).
· Plattformarna i en uppgiftsgrupp har protokoll för ömsesidigt datautbyte och utför kontinuerligt datautbyte.
· Datafusion görs inom varje grupp, hos noll, en eller flera/alla plattformar.
· Varje plattform (med sensorer) "ägs" av en plattformsägare (jfr "linjen").
· Varje plattform kan samtidigt ingå i en eller flera uppgiftsgrupper, och simultant utföra uppgifter i flera olika uppgiftsgrupper.
· Uppgifterna är dynamiska, dvs uppgifter kan bildas och leva kortare eller längre tid.
· Uppgiftsgruppernas sammansättning är dynamisk, dvs deltagarnas medverkan kan börja och sluta under uppgiftens genomförande.
· Varje uppgift leds av en uppgiftsledare (befälhavare).
· Uppgiftsresultat lämnas kontinuerligt till uppgiftsledaren/kunder av uppgiftsdeltagarna.
· Kunder/beställare kan begära informationsinsamling och data från en mäklare.
· Mäklingen utförs då en ny uppgift skall skapas, grupp bildas, plattformar allokeras. Mäklaren utser uppgiftsledare.
· Varje plattform ägs av en ägare.
· Ägaren och mäklaren har övergripande information om plattformens och plattformsensorernas aktuella status.
· Ägaren (motsvarande) styr plattformen (uppdrag, vägval, underhåll mm).
· Larm utlöses till alla plattformar då någon upptäcker ett hot.
· Vissa uppgifter utförs automatiskt (t ex self-protect mod).
· Operatörsstöd hos uppgiftsledare, mäklare och ägare.



Figur 18. Princip för resursallokering som separerar plattformsägarna från uppdragsledarna.

Avslutning

Rapporten beskriver en datafusionsnod för i första hand luft- och sjömål. Den grundläggande principen för konstruktionsarbetet har varit att utgå från beslutscirkeln eller OODA-loopen. Det finns flera fördelar att vinna på denna princip, bland annat :

- det är lätt att i datorn modellera varje för beslutsfattaren sensorberoende beslutscirkel,
- 'funktionerna' moderna flerfunktionssensorer kan med fördel realiseras som OODA-loopar,
- både operatörsassocierade och automatiska OODA-loopar kan nästlas i flera nivåer,
- multipla OODA-loopar som konkurrerar om samma uppsättning sensorer kan modelleras som agenter med hjälp av metoder inom multiagenttekniken och operativsystemtekniken.

Rapporten beskriver hur den implementerade datafusionsnoden klarar vissa av de krav som kan ställas. Detta gäller i första hand fusion av data från sensorer på samma plattform och från sensorer på flera plattformar, samt styrning (val och schemaläggning) av sensor att använda i ett nätverk av sensorer. Dessutom visas hur datafusionsnoden kan användas i en systemarkitektur med många operatörer, sensorer och plattformar. Avslutningsvis presenteras en metod för resursallokering.

Förkortningar

AAW	Anti Air Warfare
ASuW	Anti Surface Warfare
AUW	Anti Underwater Warfare
BNF	Backus Normal Form
CORBA	Common Object Request Broker Architecture
COTS	Commercials Of The Shelf

DF	Data Fusion
ECM	Electronic Counter Measures
ESA	Electronic Scanned Antennas
GUI	Graphical User Interface
IMM	Interactive Multiple Models
IRST	InfraRed Search & Track
LedsystT	FMV-projekt
NTM	Network Task Manager
OODA	Observe Orient Decide Act
RM	Resource Management
RMA	Revolution in Military Affairs
SA	Situation Awareness
SIS	SIgnalSpaning
SMDFOS	Sensor Management & Data Fusion Operating System
SNR	Signal Noise Ratio
UAV	Unmanned Air Vehicle

Referenser

[AU] Aho, Ullman, Principles of Compiler Design, AddisonWesley, 1979

[BP] Blackman & Popoli: Design and Analyses of Modern tracking Systems, Artech House 1999.

[Han] Hanebeck, New Results for Stochastic prediction and Filtering with Unknown Correlations, MFI'01, Baden-Baden, Tyskland, Augusti 2001.

[Kar] Karlsson: Decentralized tracking with feedback adaptive sample rate, FOA-R--00-01539-706--SE, Mars 2000.

[LSPAA] Linde, Strömberg, Pettersson, Alfredson, Andersson: Utvärdering av en temporal hotindikator, FOA-R--00-01781-706--SE. December 2000.

[NJU] Nicholson, Julier, Uhlmann: DDF: An Evaluation of Covarince Intersection, Fusion2001, Montreal Canada, Augusti 2001.

[RB] Roy, Bossé: Conflict Management in the Shipboard Integration of Multiple Sensors, Fusion '98, Las Vegas, USA, Juli 1998.

[RMA] FOA Memo 00-4717/L, 2000.

[SHL] Strömberg, Hörling, Lantz: En Plattformsbaserad Datafusionsnod - Delrapport 1, FOA-R--00-01716--SE, December 2000.

[Str1] Strömberg: Dokumentation över Datafusionsnoden, PM, Diarienummer 01-3084, FOI, September 2001.

[Str2] Strömberg: Värdering av systemarkitekturer, FOI Memo. 01-3253, Oktober 2001.

[Str3] Strömberg: Integrering av sensorer i nätverk, FOI Demo, 01-3133, September 2001.

[Wei] Weiss (ed): Multiagent Systems - A Modern Approach to Distributed Artificial Intelligence, The MIT Press 1999.