14th ICCRTS: “C2 and Agility”

Supporting C2 with a Service Oriented Framework for Opportunistic Sensors and Sensor Networks

Preferred topic: no 9
Alternative topic: no 10 or no 1

Marianela García Lozano, Pontus Hörling,
Farshad Moradi, and Edward Tjörnhammar
FOI, Swedish Defence Research Agency
SE-164 90 Stockholm, Sweden

Point of contact: Marianela García Lozano
Phone: +46-8-555 031 33
E-mail: garcia@foi.se

14th ICCRTS: "C2 and Agility"

Supporting C2 with a Service Oriented Framework for Opportunistic Sensors and Sensor Networks

***Abstract*—Getting information at the right time and to the right person is of great importance when managing military operations. Sensors play a vital role in gathering and providing data, but current sensors often require complicated configuration and set-up procedures before being operational. This is time consuming and requires specialized knowledge of the operator.**

The operator's work load could be greatly simplified if the sensors and the network instead had capabilities for: agile dynamic composition and automatic configuration, provided sensor ad-hoc connectivity, published sensor information as services and allowed the flexibility of choosing and combining those services to meet the end user's needs. Instead of focusing on single data streams end users could be given a variety of sensor services for their specific situation. The users would not need to care about the underlying infrastructure.

We consider Command and Control work and propose a framework that combines the ad-hoc properties of opportunistic sensors and sensor networks with the transparency and generality of the service oriented architectures, thus providing more agile C2 systems. We describe the framework requirements and how they were obtained. We also describe the prototype architecture and implementation.

I. Introduction

The service concept has been in focus within information technology for a long time, and during several years also within the C2 community. Ideas from Service Oriented Architectures (SOA) are regarded as one of the corner stones within Network Enabling Capability (NEC), and a way to migrate from more tightly integrated stove-pipe systems to loosely coupled networking C2 components, which is one important goal in the design efforts for modern C2 systems. Plug-and-play capability of interoperable components, as well as redundancy, is an important aspect to make the whole architecture more robust against temporary and permanent failures of parts of it.

Another important issue for C2 systems is the interest in commercially available software as well as hardware components rather than special military designed soft- or hardware in small series, which strongly increase the prices. The disadvantage of not getting the optimum tailored design is outweighed by the advantages of much lower prices, cheaper support, and the continuous civilian sector testing of products.

If C2 systems are not continuously fed with relevant information, they loose their meaning. Much of the important information originates in sensors and / or human observers that monitor the situation (area, process, etc.) where there are resources to command and control in order to obtain an advantageous course of action. Sensor information can be obtained from a variety of sources ranging from the main surveillance radar in airport C2 systems to heterogeneous, dynamic and maybe mobile sensors and sensor networks. In the airport case the coupling to the radar is often tightly bound, the placement of the radar is optimized concerning its functionality, and it is carefully calibrated. Such a radar is an expensive sensor intended to be the main information source (together with transponder systems on the aircrafts or similar) to shape the air picture.

In this paper we are mainly concerned with the latter case; many small, cheap, heterogeneous, static or mobile sensors with loosely coupled and often unreliable accessibility which could be, for instance, suitable in chaotic situations during war fighting or disaster relief actions. Situations where there will be limited time to place sensors at optimal locations; sometimes they could be dropped down from aircrafts, using artillery or robots with sensors as payload, deployed by soldiers on foot or from vehicles. The kind of situations where it is not possible, or there will not be enough time, to manually calibrate the sensors or to set up their communication with C2 nodes or other sensors. In this kind of situations they have to do much of this (calibration, configuration, communication) themselves in an opportunistic manner. In this paper we are also concerned with the operator's point of view; when an information need arises the operator may not know which sensor provides what information only that they are interested in the information.

The project TOppS (Swedish acronym for Service-based Opportunistic Sensor Networks) studies mainly two aspects relevant for such situations, both resulting from the mere necessity when quickly coping with large amounts of sensors of the last kind:

- What software solutions are needed for something similar to a plug-and-play architecture for such sensors?
- How should the information from such sensors be made available as shared information resources to users often having different needs?

A third aspect that also can be added is:

- How should these sensors be monitored and managed,

or how could they monitor themselves and keep the C2 system updated about it?

In this paper, we first briefly address the challenges for C2 systems that could make benefits from this type of solution. We then describe the TOppS vision. The second section is a survey of related research fields and trends from which some ideas and results could be used and combined to achieve this vision. The third section describes scenarios where the ideas could be applicable, together with the results obtained from an end user workshop we conducted with people from different fields within the military, police, customs and rescue services. The fourth section elaborates on architectural challenges and the framework design. In section five, we describe our plans for testing our ideas and framework through a "game". Finally, we sum up, present our conclusions and ideas on future work.

A. C2 Challenges

Historically, war fighting C2 was divided into three stratified levels: strategic, operational, and tactical. However, in today's war fighting environment, these demarcation lines are no longer distinct. Today's and tomorrow's C2 systems intended for missions in abruptly appearing and quickly changing situations need agile systems for situation monitoring to keep users updated. Three important characteristics for such C2 system are: survivability, rapid development and evaluation, and interoperability [1]. Survivability means that C2 systems should be able to survive various attacks including physical attacks as well as electronic attacks, and operate in spite of failures of some of the participating systems. Rapid development and evaluation requires that the C2 system must have been designed and developed in a manner that can accommodate agile integration (without any interruption) of new systems into existing frameworks. Interoperability refers to the fact that the C2 system must inter-operate with other existing systems including weapon systems, communications, sensor systems etc., in all from technical to semantical context; C2 systems are strongly dependent on the notion of information sharing. The information fed into these systems will come from human observers or sensors, which have to be chosen and deployed during the mission. Compare a disaster relief scenario or a military mission in a region with hardly any infrastructure at all, or an urban scenario where IT systems, telephone or electricity are not available or have been destroyed. If no or only little time has been available on beforehand to plan the mission, it is important that the technical sensors, as well as the communication infrastructure, can be set up on the fly at deployment if it has to be done manually.

The best would be for sensors to automatically: i) set up their own situation-dependent behaviour concerning: calibration, choice of sensing modes, ii) establish connections with the C2 system, iii) indicate what information that could reasonably be offered, iv) make decisions on data fusion, inter sensor cueing and communication, and v) offer remote management functionality.

In practice, something intermediate will be the most realistic case, but the less personnel resources and the more sensor resources are available, the more of the set-up has to be done automatically.

Either way, sensor behaviour as mentioned would lead to a highly adaptive information supply to the C2 systems, thus making them more robust and agile. This requires, however, a new way of viewing sensor information and a looser coupling between sensors and C2 systems, a functionality that is typical for different layers and services in a Service Oriented Architecture (SOA). Presenting sensors as services is a plausible step towards the enablement of dynamic interaction between sensors and C2 systems.

On the user side, we have the human operators responsible for the C2 systems. They have different roles and information needs on different resolution levels, information age, duty cycles etc. To achieve agile C2 systems this must be matched to what can be offered from the sensors. Often several roles can share sensor resources. A way to match what is offered with what is required is to package what can be offered as different types of services with searchable descriptions and possible compositions.

B. TOppS vision

Changing the view from the designer to the operators, sensors should be used in an opportunistic fashion. Depending on the user's purpose and needs, the system should be able to identify and utilize the sensors that are currently available and able to fulfill the intended task. For example, a user would like to know the amount of traffic along a certain traffic route. In this case the user connects to a TOppS-enabled web site that supply live video from cameras placed along the road combined with satellite data currently pointed at the area of interest together with auxiliary information on e.g. traffic flow history and forecasts.

Service orientation aims at separating tasks, by breaking up a computer program into distinct modules with minimum overlap in functionality. This is a design paradigm often used in Service Oriented Architectures, with which it is possible to access independent services without any knowledge of their underlying structure (platform). The major benefits of using a SOA is that it gives transparency of sensor location, architecture, communication and implementation.

The TOppS vision, as addressed by the three bullets in the introduction, is to make greater use of existing and new sensor networks by enabling them to provide sensor ad-hoc connectivity, dynamic composition and automatic configuration capabilities, publishing sensor information as services and allowing the user the flexibility of choosing and combining these services to meet the user's needs. This should be possible for the user without having any detailed knowledge about the underlying physical system (the sensors, the network and the communication), see Figure 1. The difference from traditional systems is the dynamic nature. The services are not built requiring specific sensors but rather on the notion that a service demand should be met. Given this vision we would be providing input for a more flexible, adaptive and agile C2-system.

II. Related Research

In this section we present some of the technology areas and works that have been studied, considered and used within the

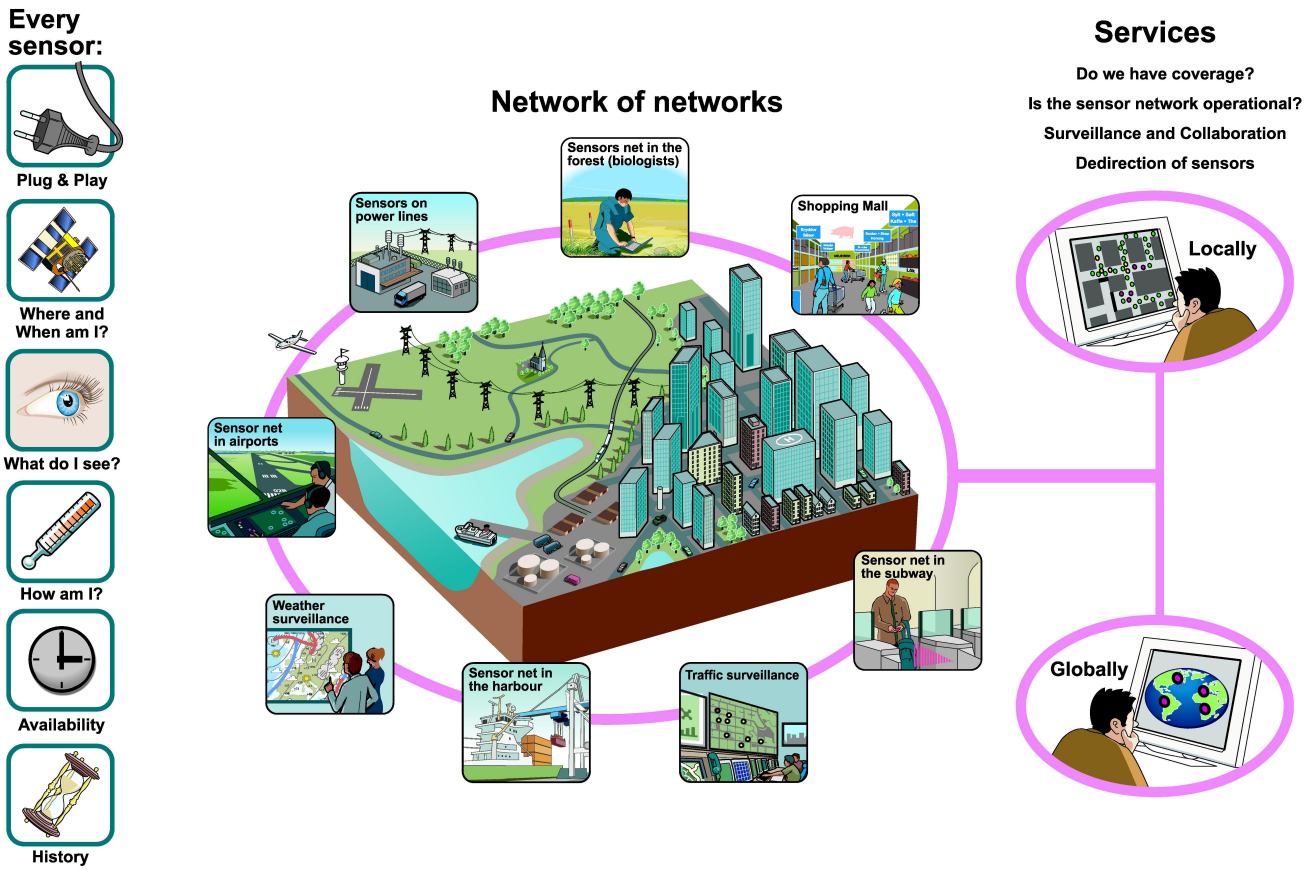

Fig. 1. The TOppS vision: To use available sensors and sensor networks to provide the Commander with sought after information in a transparent way. Achieved by building a framework that combines the ad-hoc properties of opportunistic sensors and sensor networks with the transparency and generality of the service oriented architecture.

context of this project.

A. *Opportunistic sensors and networks - oppnets*

Any system which relies on high risk near realtime and/or realtime data needs to be built, or based upon, systems which follow sound principles, e.g. system which are self-organizing and perhaps even self-healing. C2 systems are, in partial, used as a view and somewhat decoupled from the underlying sensors that feed information to these systems. It is preferable for these underlying systems of sensors to be able to automatically set up their own situation-dependent behaviour concerning calibration, choice of sensing modes, data fusion, inter sensor cueing and communication. Furthermore, they should provide information about how connections should be established with the C2 system and what information that could reasonably be offered, as well as how the sensors could be remotely managed. Hence, on the sensor side one of the most prominent feature is the ”plug-and-play” ability. This may be achieved through opportunistic sensors and networks (oppnets), see [2] and [3]. The common features of opportunistic sensors are that they are, by definition, dynamically configured and connected to available networks automatically when there is an opportunity. The mainstream ideas for these sensors are that the sensor layer is hidden as much as possible from the user, who is presented with a standardized view of the measured environment via a middleware layer. Challenges here are how to cope with sensor data heterogeneity, and how to present similar information to the user even if the actual sensor set-up is different from time to time.

In dynamic environments there must be a quick opportunistic responsiveness to the ad-hoc sensors, combined with effective information processing. Sensors are expected to connect and disconnect to the network, due to leaving the area, or changing their status. The system must be able to handle these changes by adapting parameter settings, signal processing and fusion algorithms. This service must be incorporated in the control loop.

Security and privacy are two important challenges that opportunistic sensor networks have to face. There are reasons to believe that privacy aspects may be more important to a pervasive network of sensors than a network based on other concepts. Security, on the other hand seems to have a wide

range of feasible solution strategies through cryptography, as proposed by [3].

Another important aspect is sensor information modelling and how to present and interpret fused information depending on the situation at hand. There are several types of architectures for fusion processing. The most generally used is amongst the so-called distributed fusion architecture, which offers the greatest benefits. However, it is also the most difficult architecture to design because of the lack of clear domains concerning different responsibilities in the system.

B. NECC

NECC (Net-Enabled Command Capability) is the US DoD's principal command and control capability that will be accessible in a net-centric environment. It focuses on providing the commander with the data and information needed to make timely, effective and informed decisions [4]. NECC draws from the C2 community to evolve current and provide new C2 capabilities into a fully integrated, interoperable, collaborative Joint solution. War fighters can rapidly adapt to changing mission needs by defining and tailoring their information environment and drawing on capabilities that enable the efficient, timely and effective command of forces and control of engagements.

The NECC program will deliver continuous C2 enhancements to the war fighter. It is founded on a single, net-centric, service oriented architecture and will provide the decision support infrastructure that will enable the war fighter to access, display, and understand the information necessary to make efficient, timely, and effective decisions. The program will be responsive to the war fighter through tightly coupled capability needs, development, test, and user engagement processes.

C. OGC, SWE

Within the Open Geospacial Consortium (OGC) there is a program aimed at Sensor Web Enablement (SWE). The program has resulted in a number of interesting tests, components, models, xml encodings and pending standards, which are quite relevant to the work done within the TOppS project. It currently consists of seven main parts:

- The Observations & Measurements (O&M) part consists of general models and XML encodings for sensor observations and measurements. It provides standard constructs for accessing and exchanging sensor observation results.
- The Sensor Alert Service (SAS) is a service by which a client can register for and receive sensor alert messages. It supports both pre-defined and custom alerts and covers the process of alert publication, subscription, and notification.
- Sensor Model Language (SensorML) encompasses general models and an XML scheme for describing sensors and sensor data as processes. Its information models enable the discovery and tasking of any web-resident sensor and the exploitation of sensor observations.
- The Sensor Planning Service (SPS) is a service by which a client can determine collection feasibility for a desired set of collection requests for one or more sensors/platforms, or a client may submit collection requests directly to these sensors/platforms.
- The Transducer Markup Language (TML) provides general descriptions of transducers (both receivers and transmitters), their data, how that data is generated, the phenomenon being measured by or produced by transducers, transporting the data, and any support data (meta data) necessary for later processing and understanding of the transducer data.
- The Web Notification Service (WNS) is a service by which a client may conduct asynchronous dialogues (message interchanges) with one or more other services. This service is useful when many collaborating services are required to satisfy a client request, and/or when significant delays are involved in satisfying the request.

D. Service Oriented Architecture - SOA

A major feature of TOppS is that the sensors and sensor data streams are presented as services instead of hard wired connections and feeds. In order to facilitate this we aim at using the SOA design paradigm, with which it is possible to access independent services without any knowledge of their underlying structure (platform). SOA has been identified as one of the key enablers for achieving network centric C2 capabilities, and is the architecture used in DOD's NECC and GIG (Global Information Grid) [4]. Service orientation aims at separating tasks, by breaking up a computer program into distinct modules with minimum overlap in functionality [5], providing a design framework for rapid and low-cost system development and total system quality improvement. SOA uses the Web services standards and technologies. Web services are platform and language independent and composable software components, which are designed to provide interoperability between diverse applications [6], [7]. Hence, enabling users to access business functionalities and support heterogeneous enterprise application integration. The major benefits of using SOA and Web services is that it can give transparency of sensor location, architecture, communication and implementation. It also provides modularity and scalability.

E. Pervasive Computing

Pervasive computing is an emerging research area that has received a lot of attention during the past decades. Pervasive or ubiquitous computing implies a new model for human-computer interaction, in which computing devices are integrated in our daily lives. In this paradigm we may use many devices simultaneously during our daily activities, sometimes without even knowing that. Sensors, sensor networks, wireless technologies and service-oriented infrastructures are all important concepts in this paradigm. The TOppS vision is based on the notion of pervasive computing and faces similar challenges. Therefore it has been essential for us to study this field and follow the developments.

Two special cases of pervasive computing are Pervasive Games (PG) and Pervasive Healthcare (PHC). Pervasive Games is a new generation of interactive games/distributed

real-time applications that combine computer games with the real world. Doing so, PG extends the playing board with environments in the physical world and provides location-based games that surround the players. Players in these types of games can move through e.g. city streets with mobile devices (PDAs, mobile phones, etc.) and interact with the game and be part of it. Sensors gather information about e.g. position of the players as they move. This information is then utilized to create a gaming experience which adapts to where the players are, what they do, and even how they feel. This way the players can experience a game that is interwoven with the real world and is potentially available anywhere and at anytime. One of the research activities within the field of pervasive gaming is conducted by the IPerG project, which is financed by the European Community and is led by the Swedish Institute of Computer Science (SICS). IPerG conducts research both within the technical and design aspects of PG (http://iperg.sics.se). The goal of the project is to develop infrastructure, tools, and methods for PG in order to amongst others, facilitate rapid and cost efficient development of PGs, grasp the needs of potential users and understand the social effects of PG.

Pervasive Healthcare [8] aims at addressing many of the existing and emerging challenges within the current healthcare systems, such as poor coverage of healthcare services in many parts of the world, increasing cost of those services, and increased level of stress within healthcare systems. Pervasive Healthcare envisions quality healthcare to anyone at anytime and anywhere by using emerging technologies such as wireless communication (as mentioned above), in order to overcome time and location constraints. As in the case of PG there are many open issues and challenges facing the vision of PHC, these include: lack of comprehensive coverage of wireless and mobile networks, reliability of wireless infrastructure, general limitations of hand held devices, medical usability of sensors and mobile devices, interference with other medical devices, privacy and security, and many management issues in pervasive healthcare.

F. Service-oriented Device Architecture (SODA)

Pervasive computing requires an infrastructure that enables connection between different devices and integrating them into a distributed computing system. One such infrastructure is SODA (Service-oriented Device Architecture), which is an extension of the SOA (Service-oriented Architecture) paradigm. SODA is based on OSGi[1] and takes this vision one step further by also providing means for devices (such as sensors and actuators) to be presented as services [9]. In SODA devices are connected to the architecture through "device adapters", which on one side communicate with devices and on the other present an abstract service model of the device. Doing so, they present device data as SOA services over a network. The principles and vision of SODA are quite similar to those of TOppS, however there is not enough detailed information to compare the two approaches. Furthermore, the TOppS concept is more comprehensive in a sense that it covers the step after provision of services, which includes presentation of those services and communication with the users. Another potential difference (it is not certain because of the lack of in-depth information about SODA) is the plug&play feature of sensors within the TOppS concept that is missing in SODA.

[1]http://www.osgi.org/Main/HomePage

III. Workshop and Scenarios

Since none of the current related research and project ideas fully fulfilled our requirements and vision, and we also wished to put TOppS into a broader context, a number of scenarios and activities were studied. To receive the end user community's point of view and receive feedback on the TOppS vision and scenarios we held a Workshop. In the workshop both military and civilian personnel operating at tactical and operational levels were present. The participants were from government authorities (Police, Customs), industry, educational institutions and the Armed Forces.

After the workshop discussions with the participants it became clear that security of stationary installations (e.g. a harbour or an airport), personal health and status monitoring (e.g. military personnel, fire fighters or even medical patients) combined with monitoring of surroundings, were areas where the concept of opportunistic networks and sensor based services will play an important role.

In the following sections we give a description of one of the aforementioned scenarios and the workshop results. This scenario is also the one that got the most attention at the Workshop and also the scenario that has been the basis for the planned tests, see section V. To read more on the workshop, scenarios and results see [10].

A. Personal Monitoring

The personal monitoring scenario includes two main topics, namely: human carried sensors for monitoring of the external environment and sensors for monitoring of health status of the carrier.

The former is associated with sensors that potentially are used by soldiers, policemen and rescue personnel. In military vocabulary the abbreviation ISTAR is often used, covering diverse activities such as Intelligence, Surveillance, Target Acquisition and Reconnaissance, ranging from large air- and space-borne surveillance systems to small simple ground based sensors. Here we focus on ISTAR sensors that are employed for monitoring of the nearby surroundings but, in the case of soldiers, also for Identification of Friend or Foe (IFF).

The latter type is used for internal monitoring of physiological parameters of the human carrier. Relevant applications are health monitoring of soldiers, policemen and rescue personnel during missions, see Figure 2 and 3. A peripheral application, still interesting from a C2 perspective, are devices that monitor the functionality and status of other on-body sensors and carried equipment.

The technique of on-body sensors require that sensors, cables, antenna, batteries must have minimal weight and low energy consumption. Further, the system has to be robust, especially in mobile applications. The harsh environment of soldiers and rescue personnel puts extra demands on reliability.

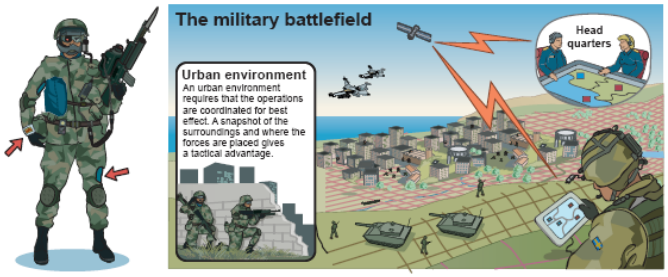

Fig. 2. On-body sensors are used to monitor the external environment and to monitor the health status of the carrier. Both the head quarters and the rescue center have use for similar sensor services relaying data on the mission bound soldiers and rescue personnel.

Fig. 3. In this scenario on-body sensors are primarily monitoring the vital status of fire fighters and rescue personnel in dangerous environments. Sensors can also be used to monitor the location of the individual fire fighters as well as external environment, e.g. ambient temperature, oxygen content, chemical 'sniffers' for explosive gases, etc.

It is necessary to design the systems to allow for on-site configuration of the sensors, in order to optimize for the specific up-coming mission. This feature includes fast changes to the sensor set-up. To make the sensors versatile they have to be easily connectible and automatically configurable into a local network that connects to an external node by wire or wireless. Here, a highly practical solution is offered by the "Plug-and-play" technique where the sensors appear as ready for use short after connection. Sensor heterogeneity offer the possibility to include a broader range of information. The advantage is that a more comprehensive view of the surroundings or the monitored person himself is possible to generate. However, the information will also become heterogeneous, which increases the complexity of information strategies, communication, storage and visualization. An important functionality is the development of services that "decides" on the type of processing and aggregation that can be offered with the currently connected sensors. In the case of mobile carriers, the situation with sensors appearing and disappearing in the network due to limited communication ranges could pose a problem. Services that allow for fast registration and re-registration (after a communication failure) of sensors would simplify the task of keeping the network up and running. Services for exchanging and correlating information that has been buffered during periods of no connectivity could be included here as well. It has to be underlined that irrespectively of gateway connectivity some services have to be accessible locally by the carrier, i.e. the system has to allow for both online and offline use.

For on-body "ISTAR"-sensors, the collected environmental information could be processed and presented to the carrier. Alternatively, sent to a local center or to a central C2 site, where the information is either used directly or queued. Sensors that are carried by the soldiers and deployed (and maybe later re-collected) at special locations, such as a cross-road, can be included in the concept as well, even though there is no direct connection or physical contact between the deploying unit and the sensors. Connection could be established intermittently between deploying carrier and sensors or the information could be transfered by a spontaneously passing relay unit. Depending on the type of on-body sensors, different kinds of health monitoring services could be produced. A refinement of information could be performed locally by producing alerts, e.g. when a physiological threshold is reached. The processed information can be made available as a subscribed service that a commanding officer or a medical doctor can use for evaluating the status of the carrier. The information can also be presented to the carrier in order to alert for his deteriorating status. For example, in physiological demanding events dehydration are commonly occurring and hence avoided by alerting. Moreover, the aggregation of health status of a rescue or combat unit could be offered as a service and used for determination of the endurance status. It is important to develop simulations services for the two categories of sensors, e.g. health and environmental monitoring. For the "ISTAR" sensors, a mission could be simulated beforehand in order to match the sensors to the mission. This in turn requires a capacity to accurately simulate the environment.

B. Workshop Results

As a result of the workshop important features of a typical TOppS-system were identified. It was concluded that the user community identified modularity of both the software and the hardware as key features of agile C2 systems. Further, due to continuity and cost reasons, introductions of new types of service-based systems have to incorporate legacy sensors as well as new ones. Additional requirements included usability and reliability aspects, automatic sensor recognition and identification. The system should facilitate dynamic Command and Control, semi-automatic functions and robust decision support. Thereby decreasing the manpower needed for operating and handling the systems.

To sum up the results:

- modular system required (both software and hardware)
- possibility to incorporate legacy sensors as well as new ones
- ease of use
- plug and play (fast automatic sensor configuration and identification)
- reliability
- light weight sensors
- low energy consumption
- better data control i.e. meta data needed

IV. TOppS Architecture

In this section we describe the challenges related to the architecture, the framework design and implementation details.

A. Challenges

In addition to the results (end user requirements) obtained from the workshop there are further challenges to be considered in the design of the architecture.

- When bandwidth varies strongly, and sometimes is absent, the sensors could enter a mode where situations they sense are cached, together with related sensor and situation related meta data. When bandwidth conditions improve, sensors reconnect and send their cached data. A special case could be sensors that upload data and download sensor management data only at predefined times, another is a purely opportunistic sensor which activates itself whenever another mobile connection node is present. Needless to say any increased autonomy and caching comes at the price of larger batteries, memory, and processing cost at the sensor side.
- A still common way to regard sensors is deploy-calibrate-connect-use, where ”use” includes the expectation of having constant availability to sensor data, often in a more or less streamed format, sometimes emulated by the IP protocol. But with sensor connectivity coming and going in an indeterminate fashion, it could be better to have a middle layer of information processing, where the sensor data presently available (or available from historic records from dropped-out sensors) is fused or assembled, and the situation picture that can be built from it is presented as a service with a certain Quality of Service (QoS). The user should be allowed to focus on his work process and not repeatedly check every sensor needed to give him a good situation picture. The service layer or an agent keeping track of available sensors, dropped-out sensors and reconnecting sensors as well as what situation picture that can reasonably be offered from present and historic sensor data could simplify his work considerably. Specifically we require a layer which when a sensor joins and leaves in an ad-hoc fashion reconfigures any information exchange. A new or reconnected sensor might add additional value like fusion or inter-sensor queuing capability besides merely increasing the size of the surveillance space. Different combinations of sensor types give different outcomes in what services can be offered on higher levels.
- Information Fusion (IF) as a scientific field seeks to cover the whole chain from information collector (such as a sensor) to decision taker, with all control feedbacks of the fusion process therein. It was originally formulated as a paradigm within military situational picture information processing, but the idea is as well applied to most general information control-chains of civil applications. To put it in its most general terms; by using smart automatic algorithms for data collection, information collation, alignment and association, uncertainty management, redundancy reduction and information presentation, IF aims at extracting the most important features in an often overwhelmingly large information flow and present it to an operator in a user-friendly way. Challenges here are how to cope with sensor data heterogeneity, and how to present similar information to the user even if the actual sensor set-up is different from time to time. As is discussed in this article, service layers or well designed middleware are ideas that are suggested solutions to obtain sensor data independence [2].
- Sensors are expected to connect and disconnect to the network, due to leaving the area, or changing their status. The system must be able to handle these changes by adapting parameter settings, signal processing and fusion algorithms. This functionality must be incorporated in the control loop. The sensor combination that happen to be at hand offer information which, when combined, can be used to derive fused information on different levels of quality.
- The types of abilities to expect from a certain combination or distribution of sensors should be made clear to specialized services from some knowledge base (KB), or using resource description ontologies like in the Semantic Grid [11], [12]. Data from homogeneous sensors are easiest to fuse due to similar and aligned data types. Heterogeneous sensors might result in spin-off effects when data from them are combined in an intelligent way, which could be reflected in this KB. This might be added abilities for detection and identification of targets, or suppression of false alarms. By levering from services at a process level a user could be able to fine-tune a combination of these services as well as to and add own knowledge to how this is done. Users should be presented with a judgement over the current sensor's information quality. This QoS judgement is multi layered and must be done from information on the sensors own status, the quality of the communication, and the (automatically and maybe manually fine-tuned) chosen fusion or information integration algorithm. No doubt such an ability would be an essential part in an information fusion system that is to work under rapidly changing situations. By integrating IF-approaches into a set of adapted services, the tools needed to face the challenges described above would be tangible, and result in a user friendly TOppS system.
- When sensor availability changes sporadically in an opportunistic network, it is not effective to use pre-defined static or ”hardwired” algorithms for fusing the information at the higher system levels. In [2], it is pointed out that sensor heterogeneity will lead to difficulties when trying to fuse data in a consistent way, due to the differences in handling measurement uncertainties. Sensor self-identification and self-registration will simplify the development of fusion algorithms. A specific middleware model is proposed, which should handle necessary tasks, such as sensor discovery and tackling of sensor heterogeneity. Other important aspects is sensor information modeling and how to present and interpret fused information depending on the situation at hand. It is necessary for the sensors to always provide information

on their inherent specifications and properties to the network since they most probably are not known by the network *a priori*. This is also the case with time-varying and environmental dependencies that might influence the sensor functionality as well as the interpretability and ”fuseability” of its output.

- There are several types of architectures for fusion processing, The most generally used is amongst the so-called distributed fusion architecture, which offers the greatest benefits. Here, there is no well defined central node that is responsible for the fusion; rather the fusion can take place at any node in in the network that happen to fit best. However, it is also the most difficult architecture to design because of the lack of clear domains concerning different responsibilities in the system, as well as the ’data incest” or ”rumour” problem: Do I receive information, fully or partly, that I have myself transmitted into the system before? These are issues that can be more easily handled in hierarchical architectures. It is still to be figured out which fusion architecture to be the optimal for opportunistic networks.

B. Design

Given the related trends, workshop results with end user requirements and other challenges for creating adaptive, flexible and agile C2 systems we designed the TOppS framework. An overview of the TOppS framework design can be seen in the Figure 4. The modules of the framework are:

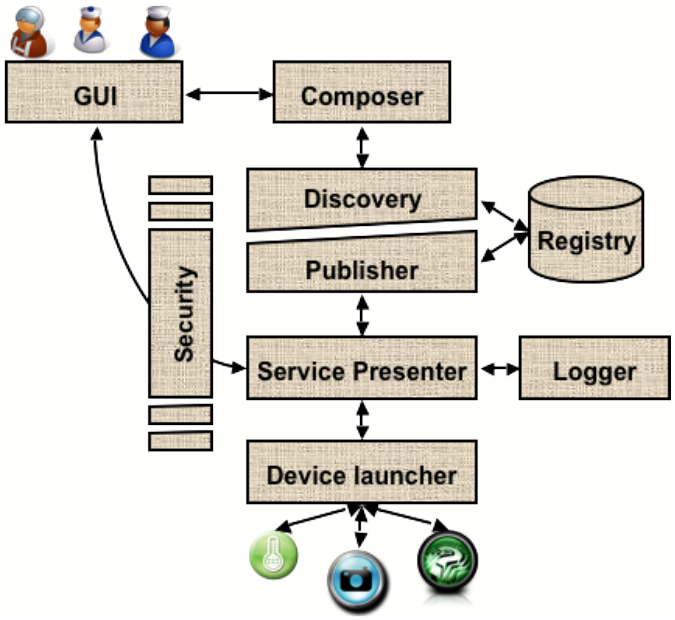

Fig. 4. Overview of the TOppS framework. Supporting C2 systems with sensor services.

Device Launcher The Device Launcher is the sensor and hardware closest module of the framework. It adapts a sensor to the framework and enables us to communicate with it. It can be specialized according to the communication mode of a sensor i.e. to Bluetooth, USB, Ethernet, etc.

Service Presenter The Service Presenter presents the sensor as a service and describes how we communicate with it. Access points for data and instruction streams requires that we also can describe them (data and instructions) in a satisfactory way i.e. allowing us to find the service that provides the sought after information. The Presenter provides an is-alive function allowing us to verify whether a service is up and running before we provide it to the user.

Logger If required or appropriate a Logger may be used to provide a (temporary) storage of sensor data. It can be implemented as a cyclic buffer. The sensor presenter handles this module and provides also an interface to it allowing a user to retrieve ”older” information

Broker (Publisher, Discovery) The Broker provides a matching between user needs and available services. It provides a common interface towards other components consisting of two modules:

- The publisher registers new services with the Registry
- The Service Discovery makes a lookup in the Registry for required services

Registry The Registry is the module that keeps track of published services and the one who performs garbage collection. It might also be a collection of several registries providing one interface towards the framework. The registry also keeps track of which sensors are present more than once, so that the user gain notice when using different views of the same data.

Composer The Composer translates the user requests into service types and service properties, it queries the Service Discovery module and composes a new service if necessary.

Security The Security module is used to authenticate and authorize users and services. It has three main points of interaction: between the GUI and the composer, the service presenter and the Broker when publishing and discovering services. The system makes use of RSA[2] and for **all** service layer activity it is specified that:

- Identities of data producers and data consumers and their public keys are stored by the registry, private keys are never stored by the registry
- Interactions between data producers and consumers are always encrypted between **all** endpoints
- Persistent data is stored in encrypted form by the data producer
- Access rights for any user(data consumer) of the framework are established either for a specific sensor or for a group of sensors(i.e. an organization). All data consumer must be given rights explicitly. It is then assumed that any sensor the user has access to, internal or external, are indistinguishable from his own
- Sensor data is always owned by the producer of the data, i.e. the sensor’s organization, and is assume to operate after best intentions

C. Implementation

In order to increase the modularity of the code and its potential for future reuse we choose to use an OSGi con-

[2]RSA is an algorithm for public key cryptography

forming container for both of the GUI and the service bus. Different containers have been evaluated but we decided to use Knopflerfish[3] since it seems to be the most mature.

Services are presented through the use of WSDL[4] and invocation performed via SOAP[5] message exchange.

The implementation remains a work in progress.

V. Test and Evaluation

For the testing and feasibility evaluation of the framework and TOppS vision we decided to use a scenario that would focus on personal monitoring. Partly the decision was based on the workshop results. For the personal monitoring we designed a "capture the flag" game.

The purpose of the game is to test:

- the framework
- discovery of services
- composition of services
- sensor information requests
- plug and play functionality of sensors

The aim of the game is, as the name suggests, to compete in groups in order to achieve a goal (capture the flag). The players of the game consist of groups with three soldiers and one commander in each. The setting for the game is an indoor area. The task is to visit a number of rooms, perform tasks and gather clues.

Each soldier has a limited number of "health points" which are used to perform tasks. When leaving the room the room becomes "radioactive". If two soldiers from the same group enter a room at the same time the room becomes "mined". To disarm a mine two soldiers must again enter the room.

A soldier may be wounded by mines and radioactivity. To heal up a soldier must visit a "healer" and remain there for some length of time.

When conflicts occur (i.e. two soldiers from different groups wish to enter the same room at the same time) it is the one who first manages to authenticate themselves to the system that wins.

The commander of each group has a C2 GUI that allows him/her to keep track of his/her soldiers' status, the game board and services. The kind of services a commander will be able to request will for example be, the soldiers' health status, their locations, alarm services that alert when someone enters a room, notifications when a soldier from their own group authenticates themselves to the system, the currently radioactive rooms, etc.

The game is won by the group that first achieves to gather clues to where the flag is located.

A. Results

The execution of the game is scheduled for the summer of 2009. Until then the framework prototype will be finished and sensors (cameras, motion detectors, thermometers, location sensors, etc. will be rigged. Together with this a C2 GUI allowing for (sensor) service requests will be connected to the framework. A new end user workshop is also planned. This time the focus of the workshop will be more on C2 aspects.

[3]http://www.knopflerfish.org/

[4]http://www.w3.org/TR/wsdl

[5]http://en.wikipedia.org/wiki/Simple_Object_Access_Protocol

VI. Summary and Conclusion

Given the large number of sensors available (small, cheap, heterogeneous, static or mobile sensors) and the amount of the information provided by these sensors it is of value to regard and utilize them not merely as a large set of unrelated data providers, but rather as providers of information. This information could be on different levels of quality depending on how the sensors might be located, how their information could be fused, and the quality of their ad-hoc communication network. On the C2 system users side, different people have different needs and expectations on this information. However, they are neither interested nor should be burdened by the underlying details regarding, types of sensors, data they provide, how to manage them, etc. One way to hide all these details for the user, is by using a Service Oriented Architecture (SOA), which is the core technology used in our framework.Our aim has been to build functionality to turn sensors into shared information resources rather than data injectors into tightly integrated C2 or ISTAR systems. However, this could still be allowed in many legacy systems where the operator might need fast low level sensor data, but also wrapping them as application services offered to the network, the sensor output can be made available as shared resources to many more users.

In order to design our architecture and identify end-user needs a workshop was conducted from which requirement specifications and useful scenarios were merged. The requirements were synthesized in a modular architecture with emphasis on separating sensor management functionalities from C2 application through utilisation of SOA. To test and evaluate our system a personal monitoring scenario using a "capture the flag" game was designed. The purpose of the game is to test the framework, discovery of services, composition of services, sensor information requests, and plug and play functionality of sensors. Although the test-bed is not ready our preliminary results indicate our approach and framework is feasible and addresses some of the challenges the C2 and sensor communities are facing such as survivability and rapid development of C2 systems, as well as data heterogeneity and dynamic configuration of sensors. The next step is to complete the test-bed and run a series of experiments to evaluate the approach and the framework in a realistic and dynamic environment with many heterogeneous sensors.

A. Future work

Plans for future work include, as previously indicated, the testing of the framework prototype together with a C2 GUI, and an end user (C2) workshop allowing for user tests to be held in the middle of 2009.

Bibliography

References

[1] R. A. Paul and W. T. Tsai, "Service-oriented architecture for command and control systems with dynamic reconfiguration," in *CCRTS 2004*, June 2004.

[2] S. Challa, T. Gulrez, Z. Chaczko, and T. Paranesha, "Opportunistic information fusion: a new paradigm for next generation networked sensing systems," in *Proc. 8th International Conference on Information Fusion*, vol. 1, July 2005.

[3] L. Lilien, Z. H. Kamal, and A. Gupta, "Opportunistic networks: Research challenges in specializing the p2p paradigm," in *Proc. 3rd International Workshop on P2P Data Management, Security and Trust (PDMST'06)*, September 2006, pp. 722–726.

[4] US Defence Information Systems Agency (DISA) description of NECC, last visited 19 jan 2009. [Online]. Available: http://www.disa.mil/news/pressresources/factsheets/necc.html

[5] Wikipedia about Service-Oriented Architecture, last visited 19 jan 2009. [Online]. Available: http://en.wikipedia.org/wiki/Service-oriented_architecture

[6] Wikipedia about Web Services, last visited 19 jan 2009. [Online]. Available: http://en.wikipedia.org/wiki/Web_service

[7] W3C Web Services Architecture ,last visited 19 jan 2009. [Online]. Available: http://www.w3.org/TR/ws-arch

[8] U. Varshney, "Pervasive healthcare and wireless health monitoring," *Mobile Networks and Applications*, vol. 12, no. 2-3, pp. 113–127, March 2007.

[9] S. de Deugd, R. Carroll, K. Kelly, B. Millett, and J. Ricker, "SODA: Service oriented device architecture," *IEEE Pervasive Computing*, vol. 5, no. 3, 2006.

[10] T. Fristedt, M. G. Lozano, P. Hörling, F. Moradi, j. Pelo, and P. Sigray, "Service oriented opportunistic sensor networks (topps) - feasibility study year 2007," FOI, User report FOI-R–2348–SE, December 2007.

[11] Wikipedia about the Semantic Grid, last visited 19 jan 2009. [Online]. Available: http://en.wikipedia.org/wiki/Semantic_Grid

[12] Semantic Grid Community Portal, last visited 19 jan 2009. [Online]. Available: http://www.semanticgrid.org