An Ontology-Centric Approach to Sensor-Mission Assignment

Mario Gomez[†], Alun Preece[‡], Matthew P. Johnson[+], Geeth de Mel[†], Wamberto Vasconcelos[†], Christopher Gibson[◇], Amotz Bar-Noy[+], Konrad Borowiecki[‡], Thomas La Porta[*], Diego Pizzocaro[‡], Hosam Rowaihy[*], Gavin Pearson[§], and Tien Pham[¶]

[†]University of Aberdeen, UK; [‡]Cardiff University, UK
[+]City University of New York, USA; [◇]IBM, UK; [*]Pennsylvania State University, USA
[§]Defence Science and Technology Laboratory, UK; [¶]Army Research Laboratory, USA

Abstract. Sensor-mission assignment involves the allocation of sensor and other information-providing resources to missions in order to cover the information needs of the individual tasks in each mission. This is an important problem in the intelligence, surveillance, and reconnaissance (ISR) domain, where sensors are typically over-subscribed, and task requirements change dynamically. This paper approaches the sensor-mission assignment problem from a Semantic Web perspective: the core of the approach is a set of ontologies describing mission tasks, sensors, and deployment platforms. Semantic reasoning is used to recommend collections of types of sensors and platforms that are known to be "fit-for-purpose" for a particular task, during the mission planning process. These recommended solutions are used to constrain a search for available instances of sensors and platforms that can be allocated at mission execution-time to the relevant tasks. An interface to the physical sensor environment allows the instances to be configured to operate as a coherent whole and deliver the necessary data to users. Feedback loops exist throughout, allowing re-planning of the sensor-task fitness, reallocation of instances, and reconfiguration of the sensor network.

1 Introduction

Sensor-mission assignment involves the allocation of sensors and other information-providing resources to missions in order to cover the information needs of the individual tasks in each mission. This is an important problem in the intelligence, surveillance, and reconnaissance (ISR) domain, for a variety of reasons.[1] Firstly, the informational demands placed on available sensors and other ISR resources typically exceeds their supply in terms of inventory: commanders tend to want more than the available assets can provide, so careful allocation and resource sharing is usually necessary. Secondly, the deployment environment is chaotic and subject to frequent changes: mission plans must evolve to cope with unforeseen events, leading to changes in the tasks required and hence the ISR needs. Any solution to the sensor-mission assignment problem must therefore support a high degree of agility in terms of identifying alternative resources and re-assigning or re-purposing resources to tasks.

This paper approaches the sensor-mission assignment problem from a Semantic Web perspective: the core of the approach is a set of ontologies describing mission

[1] For background on this problem from a military perspective, see for example http://www.dtic.mil/doctrine/jel/new_pubs/jp2_01print.pdf, pages III–10–11.

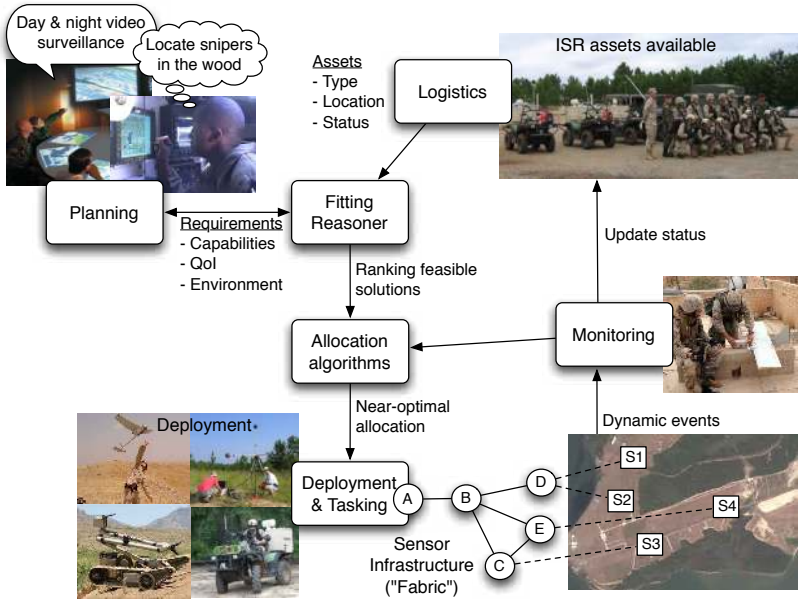

Fig. 1: Overall picture of the sensor-task assignment process

tasks, sensors, and deployment platforms. Figure 1 shows how the various aspects of our approach fit together. Working from the mission planners' ISR requirements (what capabilities are needed, in what environment, and with what quality-of-information), semantic reasoning is used to recommend collections of types of sensors and platforms that are known to be "fit-for-purpose" for a particular task. The reasoner is able to take account of logistical information concerning the potential availability of assets (including their location, types, and operational status). Then, these recommended solutions are used to constrain a search for available instances of sensors and platforms that can be allocated at mission execution-time to the relevant tasks. This stage takes account of data on the operational status of the various assets in the field (for example, if they degrade in some way, perhaps due to damage). An interface to the physical sensor environment then allows the instances to be configured to operate as a coherent whole and deliver the necessary data to users. Feedback loops exist throughout, allowing re-planning of the sensor-task fitness, re-allocation of instances, and re-configuration of the operational sensor network.

Our overall goals are to provide three elements of an integrated solution to the sensor-mission assignment problem:

- A framework that offers a "top-to-bottom" solution to the problem of deploying sensors to meet the information needs of tasks in a mission context. At the core is a set of modular ontologies covering task requirements, sensor capabilities, and a structured framework to associate tasks with sensors.
- A combination of reasoning at mission-planning time, and optimisation algorithms at mission execution-time: the reasoner recommends a collection of sensor types

which cover the needs of the mission, while the optimisation algorithms select the best collections of instances of those sensor types.
– Support to dynamically configure a deployment of selected sensor instances by means of a sensor infrastructure (the Sensor Fabric), which handles subscription to physical-world sensors, routing, information fusion, and delivery of information to users.

In this paper, we focus mainly on the first of these elements, the ontology-centric framework. Although the second and third elements are not the primary focus of this paper, we will describe the relationship between the ontology-centric framework and the other elements, and summarise the status of our overall integrated approach.

Figure 2 provides an overview of our conceptual architecture. At its core is a set of interlinked *ISR ontologies*, together with a reasoner that performs *sensor-task fitting*. The interlinked ontologies define various dimensions of the ISR domain, including the information requirements of mission tasks, and descriptions of various aspects of sensors and sensor platforms. These build on existing ontologies, schemas, and catalogues of tasks, sensors, etc where appropriate, including attributes exposed by the *Sensor Fabric*, which provides an interface to the deployed sensor environment, allowing sensor selection, data routing, fusion and filtering. The *Missions and Means Framework (MMF) ontology* supports the sensor-task fitting reasoner by articulating the relationships between task requirements and sensor/platform capabilities. The reasoner is able to recommend collections of sensor types to cover mission requirements: this is done at mission-planning time, and allows for user intervention in the selection process. The selected collections of sensor types are made available at mission execution-time to the *sensor-task allocation algorithms*, which attempt to assign sets of sensor instances in a near-optimal manner. The *delivery* layer is tightly coupled with the fabric, and allows users to subscribe dynamically to the sensor instances assigned by the sensor-task allocation algorithms.

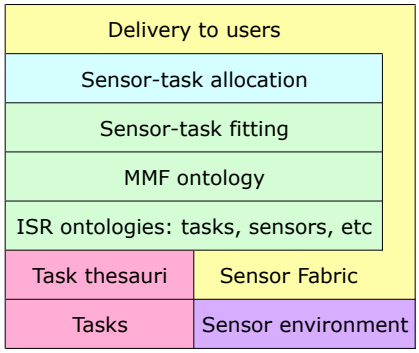

Fig. 2: Conceptual architecture

The paper is organised as follows: Section 2 describes the various ontologies we have developed to model the relationships between tasks and assets in the ISR domain. Section 3 describes the reasoning procedure for sensor-task fitting. Section 4 describes the sensor-task allocation framework, and explains how it complements the work done by the reasoner. Section 5 examines the status of our implementation based on the Sensor Fabric. Finally, Section 6 summarises and concludes the paper.

2 Ontologies for assessing the fitness-for-purpose of sensors to tasks

We advocate the use of semantic matchmaking [1] to address the assessment of the fitness-for-purpose of alternative means to accomplish a given ISR task, which in turn will support the effective allocation of assets to multiple competing tasks. This approach relies on the use of ontologies as an expressive and logically-sound way to represent knowledge and reason with it. More specifically, our approach uses ontologies in the following activities:

– specifying the requirements of a task, in terms of the ISR capabilities needed;
– specifying the capabilities provided by ISR assets (sensors and sensor platforms);
– comparing the specification of a task against the specification of available assets to assess their fitness-for-purpose.

2.1 Missions and Means Framework Ontology

Although it is possible to imagine a single all-encompassing "ISR ontology", we adhere to the Semantic Web vision of multiple interlinking ontologies covering different aspects of the domain: sensors, platforms, tasks, etc. This allows us to build on substantial pre-existing work but, as we will see, leave us with the problem of relating sensors/platforms to tasks. To fill this gap, we provide an ontology based on the Missions and Means Framework (MMF) [2], which is essentially a collection of concepts and properties to reason about the capabilities required to accomplish a mission it (e.g. mission, task, capability, asset, etc.). MMF was developed by the US Army Research Laboratory to provide a model for explicitly specifying a military mission and quantitatively evaluating the utility of alternative means to accomplish it. Ours is the first attempt to define an ontology based on the framework; using MMF allows us to benefit from its familiarity to users. The way MMF describes the linking between missions and means — shown in Figure 3 — naturally fits the notion of matchmaking: on the one hand, we have missions breaking down into operations, and operations into tasks, where each task may require different capabilities to be accomplished; on the other hand, we have the capabilities provided by assets as a result of aggregating the capabilities of its constituent systems and subsystems.

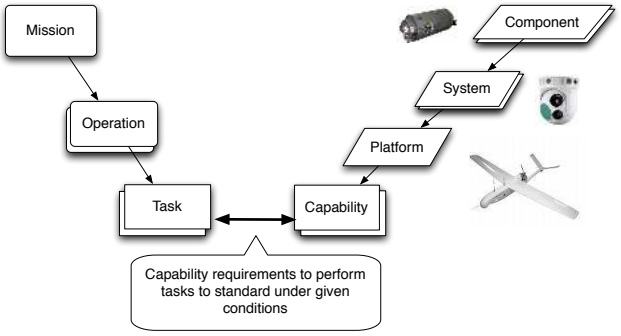

Fig. 3: Mission and Means Framework

Figure 4 sketches the main concepts of our MMF ontology. On the left hand side, we have the concepts related to the mission: a mission comprises several operations to be carried out, and each operation breaks down into a number of tasks that must be accomplished. On the right hand side we have concepts related to means: a sensor is a system that can be attached to a platform; inversely, a platform can mount one or more systems; both platforms and systems are assets; assets provide capabili-

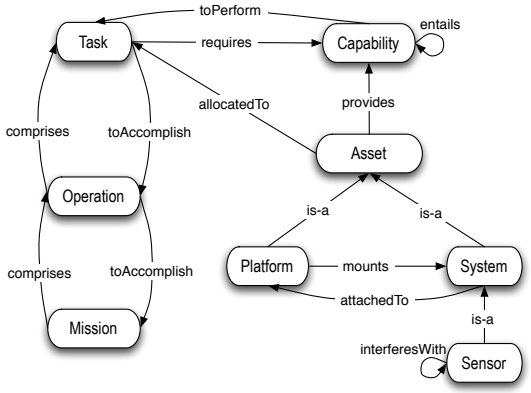

Fig. 4: Main concepts and relations in the MMF ontology

ties; a capability can entail a number of more elementary capabilities and is required to perform certain type of tasks; and inversely, a task is enabled by a number of capabilities; some sensors can interfere with other sensors, so they cannot be used simultaneously; and finally, at some point, assets will be allocated to specific tasks that require the capabilities provided by them.

Note that all concepts shown in Figure 4 are general MMF concepts with the exception of *sensor*, which we have introduced (as a refinement of the MMF core concept *system*) in order to link the MMF ontology with the ISR domain specific ontologies. This is because, while the MMF ontology describes the main concepts used in our matchmaking framework — and is generic to military and military-style missions and means in the widest sense — in order to describe specific instances of those concepts we need domain-specific vocabulary discussed in the next section.

2.2 ISR Ontology

There is already a sizeable amount of work done in providing descriptive schemas and ontologies for sensors, sensor platforms, and their properties, such as SensorML [3], OntoSensor [4], CIMA [5], and the MMI Platforms ontology [6] among others. There are also several well-known structured descriptions of tasks in the military missions context, most notably the US Universal Joint Task List (UJTL)[2], the CALL thesaurus[3], the US JC3IEDM model and the UK JETL/METL task lists. These existing representations provide partial coverage of what we need to model but, as they originate in either the sensor/platform or task spaces, they lack knowledge of how capabilities provided by the types of sensors/platforms may satisfy the capabilities required by tasks. Consequently, our approach has been to reuse existing concept sets, and to extend these with representations of capabilities.

During the knowledge analysis and acquisition stage we found a number of issues that we have taken into account in our approach to the representation of the ISR domain, including the following:

- The absence of standardised taxonomies, and the existence of alternative, sometimes inconsistent, classifications for the same concepts.
- Existing attempts to conceptualize the domain are based on different dimensions, and more usually, several dimensions are mixed. For example, UAV (Unmanned Air Vehicle) classifications tend to mix dimensions such as size or weight (e.g. MicroUAV vs MiniUAV), performance (e.g. Medium Altitude vs High Altitude), task type (Maritime Reconnaissance, Wide Area Surveillance, etc.), or ad-hoc features such as their landing and take off capabilities.
- There are fuzzy concepts that are difficult to classify as a single category. For example, LIDAR (LIght Detection and Ranging) is a type of sensor that has properties of both optical sensors and radars.
- Concepts that are supposed to refer to the same aspect of the domain are described at different abstraction levels. Closely related to this issue is the tension between considering a concept as primitive, or as a composition of more basic elements; for example, a reconnaissance capability might be seen as implying a combination of mobility and sensing capabilities.

[2] See http://orlando.drc.com/semanticweb/daml/ontology/condition/ujtl/condition-ont for an existing UJTL-based ontology.

[3] http://call.army.mil/thesaurus

In order to deal with the challenges introduced above, we propose a compositional and multidimensional approach to conceptualize the ISR domain. Such an approach is well suited to Description Logics (DL) [7] languages such as OWL DL. One of the most powerful features of DLs is their ability to define classes in terms of sufficient and necessary conditions. New concepts can be defined by specifying property restrictions and relations on existing concepts. As an example, consider the following DL definitions for some concepts relating to aerial sensor platforms:

- Aircraft $\equiv$ (Platform $\sqcap \exists$ hasRealm.Atmosphere)
- UnmannedVehicle $\equiv$ (Platform $\sqcap \exists$ hasQuality.Without-crew-mobility)
- UAV $\equiv$ (Aircraft $\sqcap$ UnmannedVehicle)
- CombatUAV $\equiv$ (UAV $\sqcap \exists$ providesCapability.Firepower)
- MALE $\equiv$ (UAV $\sqcap \exists$ providesCapability.MediumAltitude $\sqcap \exists$ providesCapability.LongEndurance)
- EnduranceUAV $\sqsubseteq \neg$ (SmallUAV $\sqcup$ TacticalUAV).

By using OWL DL, we can apply off-the-self reasoners to infer new classifications based on concept definitions, which is very appropriate because different classifications are useful for different purposes. For example at some point one might be interested in selecting a platform in terms of the type of tasks that it can perform (reconnaissance, surveillance, battle damage assessment, etc), but in other circumstances one might be interested in selecting a platform according to its takeoff and landing capabilities (catapult, runaway, VTOL, etc). Some of these dimensions are:

- For platforms: mobility, realm, performance (range, endurance, altitude, speed, etc.), application or mission type (surveillance, reconnaissance, target acquisition, etc.), firepower, landing and takeoff, communications, vulnerability and survivability, availability.
- For sensors: phenomena detected (type and spectrum), performance (resolution, sample rate, etc), weather/terrain/contamination influence, vulnerability, interferences with other sensors.

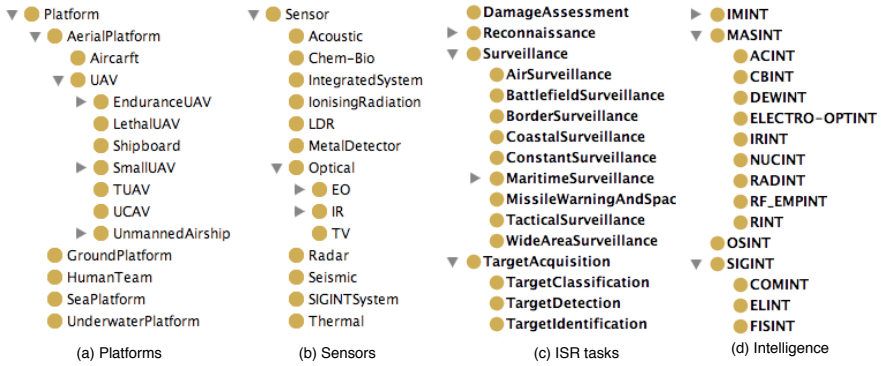

Fig. 5: Sample of concept taxonomies relevant to the ISR domain

Figure 5 shows a sample of some of the taxonomies we have developed for the ISR ontology. Platforms (a) and sensors (b) are used to characterize ISR assets; the former draws on concepts from OntoSensor and CIMA, the latter from the MMI Platforms

ontology (as referenced at the start of this section). Note that a member may belong to multiple classes in different branches of the tree, as far as it complies with the class definition; for example, PredatorB is asserted to be a subclass of MALE (Medium Altitude Long Endurance) UAV, but it is also classified as a subclass of CombatUAV because it is a UAV that provides Firepower capability, as stated in the definition of CombatUAV above. ISR tasks (c) represent the main requirements used to select a type of platform, while intelligence disciplines (d) are used to select sensor types supporting the production of specific types of intelligence (e.g. optical sensors support the production of IMINT (Imagery Intelligence)). Both (c) and (d) are drawn mainly from the ISR-related sections of the CALL thesaurus, though (c) also draws upon UJTL.

We use different properties to relate sensors and platforms to the capabilities they provide. There is a main object property called providesCapability, that takes instances of Asset as its domain, and has instances of Capability as its range. In addition we have properties that are subtypes of the providesCapability property, such as hasRange, hasCoverage, etc. Some of these properties are used to represent capabilities that are represented as partition values, like the UAVRange class, which is partitioned into three classes: CloseRange, ShortRange, and LongRange. We have included also a number of data-type properties that are essentially used to characterised numeric attributes of an asset, such as for example the ceiling, endurance, mission radius and speed of an aircraft.

Figure 6 shows class examples of both a sensor and a platform, as shown in the Protégé Ontology Editor[4]. On the right hand-side of the figure we see the description of the FLIR (forward looking infrared) sensor class, which is a subclass of IR (infrared). FLIR sensors are able to detect thermal energy, which gives them the ability to operate night and day, have foliage penetration (FOPEN) capability, provide high-quality identification of targets, fair resolution and good coverage. On the left-hand side we see the description of the PredatorB platform class, which is a subclass of MALE UAV. The PredatorB can carry several types of sensors, including optical sensors (FLIR), synthetic aperture radar (SAR), laser designator and range finder (LDRF), and Signals Intelligence (SIGINT) sensors. Some capabilities are inherited from superclasses (e.g. MediumAltitude is inherited from MALE), while others are specifically asserted (e.g. FirePower).

3 Semantic matchmaking of sensors and tasks

The use of semantically-rich specifications enable the use of specific forms of matchmaking that are not available when using a syntactic approach, such as the use of subsumption (e.g. [1, 8]) and logical satisfaction (e.g. [9]). Figure 7 shows very basic examples of the subsumption-based matching relations we have considered, using examples from the ISR domain. Q denotes a query which specifies some intelligence requirements to be met, and $S1$ through $S5$ denote the specification of ISR assets (sensors and sensor platforms) to be matched against Q.

In particular, our example query poses two requirements to be met: provide infrared (IR) vision, and be able to carry out a night reconnaissance task. From left to right we see first the specification of the query Q, then we see the specification of several assets with different types of matching. Following [1], we list them below in decreasing order of matching strength:

[4] http://protege.stanford.edu/

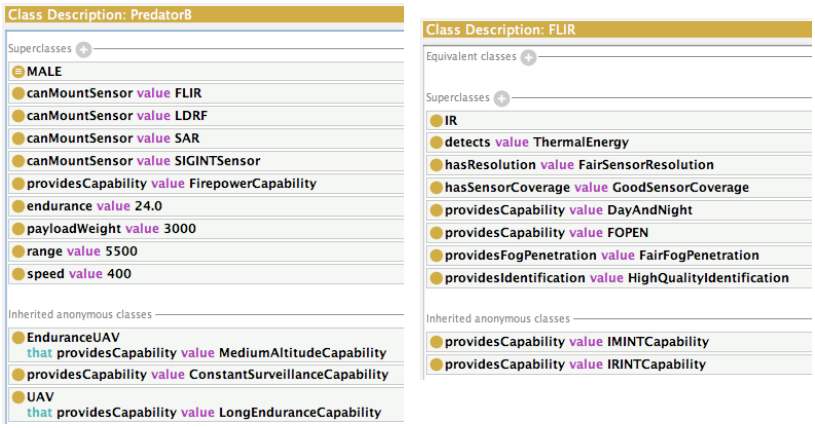

Fig. 6: Examples of ISR classes: PredatorB platform and FLIR sensor

1. *Exact*: holds when the query Q is equivalent to the specification S: $S \equiv Q$. In the example, $S1$ describes an asset that provides IR vision and is designed to perform night reconnaissance tasks, just as stated in Q, so $S1$ and Q are equivalent.
2. *Plugin*: holds when S is subsumed by Q: $S \sqsubseteq Q$. In the example, the asset described by $S2$ refers to a cooled FLIR, which is a specific type of IR camera, and provides night reconnaissance, so $S2$ is subsumed by Q. (Note that exact matches are a special case of plugin, where $S \equiv Q$.)
3. *Subsumes*: holds when Q is subsumed by S: $Q \sqsubseteq S$. In the example, $S3$ refers to an asset providing night vision capability, which is a more general concept than IR vision, and it provides also night reconnaissance, so Q is subsumed by $S3$.
4. *Overlaps*: holds when the conjunction of Q and S is not empty: $Q \sqcap S \neq \bot$, in other words, $Q \sqcap S$ is satisfiable. In our example, $S4$ describes an asset that provides night reconnaissance as required by Q, but the first requirement is not satisfied, since it carries a type of radar (SAR) instead of an IR camera, and these two concepts are disjoint.
5. *Disjoint(S4, Q)*: holds when the conjunction of Q and S is empty: $Q \sqcap S = \bot$, in other words, $Q \sqcap S$ is unsatisfiable. In the example, $S5$ describes an asset that provides TV video and is suited to perform day reconnaissance tasks; radar imagery is disjoint with IR vision, day reconnaissance is disjoint with night reconnaissance, so there is no intersection between Q and $S5$.

A matchmaking application is not entirely characterised by the basic semantic relations that can be established among concepts. An important issue of a matchmaking application is the distinction between the attribute-level and the component-level: a component may be described by different attributes, and so different matching schemas could be applied to each attribute depending on the particular meaning or role it plays within the component.

In our application, we have identified two main classes of components to be matched against the ISR requirements of a task, each one characterised by different attributes that

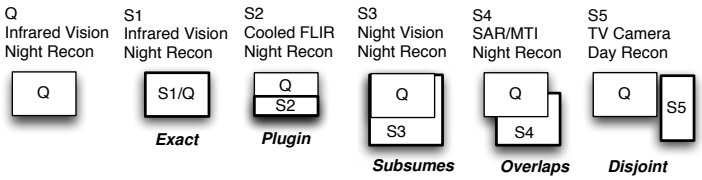

Fig. 7: Basic matching relationships

deserve a separate treatment. Note that the kind of capability requirements relevant to selecting a specific kind of sensor are quite different from the requirements that are relevant to select a platform. For example, in order to assess the utility of different sensors it is very important to consider the kind of intelligence to be produced (IMINT, ACINT, SIGINT, etc.), since each type of sensor provide information that supports a different kind of intelligence (e.g. infrared cameras support IMINT, while acoustic sensors support ACINT). In addition, to select a specific platform (e.g. UAV) for a reconnaissance mission there are other factors to consider, such as the range to the targets of interest, the presence or absence of enemy anti-air assets, and so on. Moreover, UAVs are limited in the weight and type of sensors they can carry, and the performance of some sensors may be influenced by conditions that depend on the platform they are attached to, such as the altitude. Therefore, one cannot select sensors and platforms independently; instead, the interaction between these kinds of components must also be taken into account.

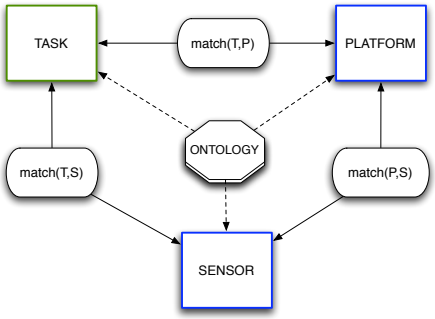

Fig. 8: Abstract matching architecture

To address the issues introduced above, we propose an abstract architecture shown in Figure 8. This architecture has three main components:

- *Task*: defines the goals to be achieved and the capabilities required to accomplish those goals. In addition, a task may have environmental conditions (weather, terrain, enemy, etc) attached that are expected to impact the performance of a task.
- *Sensor*: these are the assets that collect the information required to satisfy the intelligence requirements of a mission. However, sensors do not operate as independent entities, as they have to be attached to systems that provide them with energy, protection, mobility, etc.
- *Platform*: these are the systems to which sensors are attached so as to get energy, protection, mobility, communication, etc. Platforms include both static and mobile systems operating on land, sea and air.

The three components involved and the dependencies between them result in the following three matching relations:

- *Task-Sensor matching*: a sensor class S matches a task T, $match(T, S)$, if S provides the information-collecting capabilities satisfying T's ISR requirements.

- *Task-Platform matching*: a platform class P matches a task T, $match(T, P)$, if P provides the kind of ISR-supporting capabilities (mobility, survivability, communication) required to perform T.
- *Platform-Sensor matching*: a sensor S matches a platform P, $match(P, S)$, if S can be carried by and is compatible with the characteristics of P.

In order to satisfy the ISR requirements of a task one needs to select (at least) both a platform and a combination of sensors such that our three matching relations are simultaneously satisfied.

In some situations, all requirements can be satisfied by a single platform mounting one or multiple sensors: given a task T specifying a set of requirements $\mathcal{R}^T = \{R_1, ..., R_n\}$, a single solution for T is a **platform configuration** $\Pi = \langle P, \mathcal{S} \rangle$, where P is a type of platform, and $\mathcal{S} = \{S_1, ..., S_m\}$ is a set of sensor types that can be mounted in P simultaneously (there are no interferences among them). A platform configuration is a **valid solution** if the combined capabilities of P and $\mathcal{S}$ satisfy $\mathcal{R}^T$, where satisfaction is computed as a *plugin* (or *exact*) match, that is:

$$\langle P, \mathcal{S} \rangle \in \mathcal{V}(T) \Longleftrightarrow \forall R_i \in \mathcal{R}^T : (P \sqsubseteq \exists C.R_i) \sqcup (S_i \in \mathcal{S} \sqsubseteq \exists C.R_i)$$

where $\mathcal{V}(T) = \{\Pi_1, ..., \Pi_n\}$ is the set of valid solutions for task T, and C denotes the object property providesCapability. Any subproperty of providesCapability is thus applicable in the former definition.

For example, given a set of requirements $\mathcal{R}^T$ = {MaritimeSurveillance, IMINT, DayAndNight} a valid solution will be ⟨FireScout, {IRCamera}⟩, since FireScout is a class of UAV that provides the capability MaritimeSurveillance (exact match), and IRCamera is a class of sensor that supports the production of IRINT, which is subsumed by IMINT (plugin match), and provides DayAndNight operation capability.

In many cases, a task involving several requirements can not be achieved by a single platform, but it can be achieved by a combination of different platforms. In general, a combination of different assets will increase the utility of the information obtained from a single asset, or be the same. To address the general case, we enumerate all valid multiple-platform multiple-sensor solutions: each solution may comprise multiple platforms, and each platform may mount several sensors. A solution is valid if the combined capabilities of all the assets (platforms and sensors) included in the solution together satisfy the requirements. We model this as an instance of the set cover problem, where sets comprise the combined capabilities provided by a single platform configuration $\langle P, \mathcal{S} \rangle$, and set elements are capabilities; the goal is to find a small family of sets that covers the set of requirements (also a set of capabilities).

After finding the valid solutions, we drop those solutions that are not minimal, i.e., those containing more platforms than necessary. The result is a set of valid solutions $\mathcal{V}(T) = \{V_1, ..., V_n\}$, where each solution is a set of platform configurations, $V_i = \{\Pi_1^i, \ldots, \Pi_m^i\}$. We call the overall procedure the **Set Cover Matchmaker (SCM)**. SCM is a brute force algorithm, limiting the number of assets per package to avoid excessive computation.

Finally, we have introduced a mechanism to sort the set of solutions according to numeric criteria, such as the economic cost. For example, with the following task requirements: $\mathcal{R}^T$ = {ConstantSurveillance, IRINT, SIGINT}, SCM obtains the following ranking of solutions, with cheaper solutions ranked first.

- $V_1 = \{\langle$PredatorB, {IRCamera, SIGINTSensor}$\rangle\}$

- $V_2 = \{\langle$E-Hunter, {IRCamera}$\rangle$, $\langle$I-GNAT, {SIGINTSensor}$\rangle\}$
- $V_3 = \{\langle$GlobalHawk, {IRCamera}$\rangle$, $\langle$I-GNAT, {SIGINTSensor}$\rangle\}$

In this example, the first solution involves a single platform carrying two types of sensor, IR and SIGINT, and two solutions involving two platforms. The most expensive one is the one with the GlobalHawk, since that platform alone costs twice as much as any of the other platforms in the example. Other valid solutions are excluded from the final output because they are not minimal; for example $\{\langle$GlobalHawk,{IRCamera}$\rangle$, $\langle$Predator,{SIGINTSensor}$\rangle\}$ is a valid solution, but we can remove the GlobalHawk and use only the PredatorB, which is actually our first solution.

SCM is implemented using the Jena[5] and Pellet[6] packages to process and reason with our OWL DL MMF and ISR ontologies[7].

4 Optimal allocation of sensors to competing tasks

After the reasoning process has determined which types of sensors and platforms are appropriate for which tasks, the next step is to allocate instances of the assets to tasks. Resources are constrained by available inventory, so it may not be possible to satisfy all tasks, even within the various possible solutions determined by the fitness-for-purpose reasoning. At this stage, tasks are potentially in competition for resources. Since the tasks may vary in difficulty and importance, two kinds of decisions must be made. First, which tasks shall be attempted and which shall be abandoned? Second, for each attempted task, which resources should be assigned to it? Hence, algorithms to efficiently allocate assets are necessary.

However, the application of these algorithms in practice needs utility functions that attach a value to each pair $\langle$*task*, *resource*$\rangle$, so that the overall utility of different allocations of resources can be compared to decide which one is best. In our view, the utility functions should aggregate a number of qualitative and quantitative factors.

There are a number of proposals to apply quantitative metrics to assess the utility of alternative ISR assets, such as the Sensor Mix Model (SMM) [10], that takes into account numeric attributes such as the *dwelling time* (time on station), the area to be covered, the *sweep width* of the sensors mounted on the platform, etc. However, we have no knowledge of previous proposals to take into account qualitative capabilities, such as the ones represented in our ISR ontology (e.g. all-weather operation, day and night, foliage penetration, moving target detection capability, etc).

By introducing qualitative capabilities to the description of ISR resources and tasks, we are considerably extending the range of scenarios that can be modeled using purely numeric models. Existing frameworks are typically focused on a single resource type, or predefined resource packages, while in our approach we can find new resource packages on demand by dynamically reasoning about their different capabilities.

In our approach, we use the matchmaker before allocating resources so as to filter out solutions that are not fit-for-purpose because of their qualitative properties; in other words, we discard those solutions that are expected to be useless, and by doing so, we reduce the search space while running the allocation algorithms. Note that the majority of the qualitative attributes we have identified as relevant to the problem depend on the abstract asset types, that is, the classes of sensors and platforms available, while most

[5] http://jena.sourceforge.net/

[6] http://pellet.owldl.com/

[7] Available from http://www.csd.abdn.ac.uk/research/ita/sam/webpages/home.php

of the numeric attributes depend on actual instances of those classes deployed in the field (typically there will be several instances of the same asset class involved in a given situation).

In the rest of this section we briefly describe one of the models we have considered to enable the application of resource-allocation algorithm for multiple competing tasks:
Semi-Matching with Demands (SMD) [11, 12]. This model is based on the use of additive utility values for each pair asset-task, and different priorities and demands for each task, all of these being numeric attributes.

Problem instance: A weighted bipartite graph $G = \langle \mathcal{A}, \mathcal{T}, P, D, E \rangle$, where $\mathcal{A} = \{A_1, ..., A_n\}$ is a collection of assets, $\mathcal{T} = \{T_1, ..., T_m\}$ is a collection of tasks, $P = \{p_1, ..., p_m\}$ is a collection of task profit values, $D = \{d_1, ..., d_m\}$ is a collection of task demands, and $E = \{e_{ij} : i \in [1, n], j \in [1, m]\}$ is a collection of non-negative weights for the edges $\mathcal{A} \times \mathcal{T}$.

Goal: Find a semi-matching $F \subseteq \mathcal{A} \times \mathcal{T}$ (no two chosen edges share the same asset) maximizing the sum of the profits p_j of the satisfied tasks, where T_j is satisfied if its total utility reaches its demand, i.e., $\sum_{(A_i, T_j) \in F} e_{ij} \geq d_j$.

We note that the SMD problem as defined above is quite abstract. That means that the variables used may have different meanings and interpretations. The utility values e_{ij} can be calculated using different metrics and functions. The profits represent the contribution of every task to the global utility obtained from all tasks as a whole. The idea here is to give tasks different degrees of importance depending on their overall context. For example, if a task is critical for the success of other tasks, then the first task will have higher importance, and will be assigned a high profit value. Finally, demands represent a numeric notion of the amount of resources required by a task, and its particular interpretation is closely related to the metric used to calculate utility values; for example, if we use SMM demands will represent the area to be covered.

In order to integrate the SMD model and the fitting reasoner, we have so far considered only single-platform configurations. More precisely, the application of the model assumes:

1. Every asset $A_i \in \mathcal{A}$ is assumed to be an instantiation of a single platform configuration $\mathit{\Pi} = \langle P, \mathcal{S} \rangle$, which cannot be re-configured on demand to satisfy the requirements of a specific task.
2. For every asset-task pair (A_i, T_j), if A_i is an instantiation of a platform configuration ($\mathit{\Pi} \in V(T_j)$), then we apply a utility metric like SMM to obtain a particular utility value e_{ij}; otherwise e_{ij} is assumed to be zero and it is ignored by the allocation algorithms.

It is in this way that we can obtain an SMD problem instance and thereby take advantage of existing allocation algorithms. In ongoing research, we plan to extend SMD and its assignment algorithms to the more general setting of multi-platform solutions, as those resulting from the application of SCM. We emphasize that two separate computational problems must be solved. The first is to determine which possible combinations of assets are sufficiently fit-for-purpose for every different task, according to the classes of sensors and platforms available in the theater of operations. The second is to find a global assignment of particular instances of those assets, based on their actual utility values.

The challenge in the second problem is to maximize the global mission success given limited access to resources. Indeed, the SMD formulation is known to be NP-hard, as well as NP-hard to approximate in the most general setting [11]. Under the

assumption that the degree d of the problem instance (the maximum number of assets within range of any task) is limited, however, a simple greedy algorithm can guarantee a d-approximate solution. An alternative, knapsack-like formulation, SUM, has been described in [13].

5 Fabric for sensor deployment and delivery

We are currently working on the problem of deploying selected sensor instances on the network, so that they operate as a coherent system that can deliver the required information to users. Our main focus at present is to interface the fitting and allocation components with a particular sensor infrastructure — the Sensor Fabric — a prototype implementation of which has been built using commercial off-the-shelf components [14]. While the main aim of the fabric is to research and apply algorithms for the retrieval and dissemination of task-specific information across sensor networks, it provides three main components that complement the fitting and allocation work previous described:

1. *Sensor Catalogue*: provides a global inventory/registry of sensor instances, describing all known assets and their availability. It manages the visibility of, and access to, sensor data feeds. The Sensor Catalogue is used to select the sensors required to fulfill the requirements of a specific task.
2. *Topology Manager*: manages the network topology and inter-node communication, describing all known network nodes and their availability. The Topology Manager is used for node location and routing information.
3. *Fabric Manager*: responsible for the control (configuration and sensor-mission assignment) and monitoring of individual sensors and intermediate nodes, and establishing the communication channels between them. The Fabric Manager also provides a container for running in-network information fusion and filtering algorithms, and registering them as assets with the Sensor Catalogue.

Figure 9 sketches the Sensor Fabric architecture and the three main services provided: the Sensor Catalogue (SC), the Topology Manager (TM), and the Fabric Manager (FM). There is one instance each of the Sensor Catalogue and Topology Manager per Sensor Fabric, and one instance of the Fabric Manager per node. Sensor nodes publish data locally to a topic in a global topic name space, and similarly consumers subscribe locally to the same global topic name. The Fabric establishes the communication channel between the two automatically, and ensures that published messages are delivered correctly. In this example data published from sensors $S1$, $S2$, and $S3$ on node E is accessed by the client on node A as if it is local. The Fabric transparently delivers the data across the network between the nodes, E to B to A.

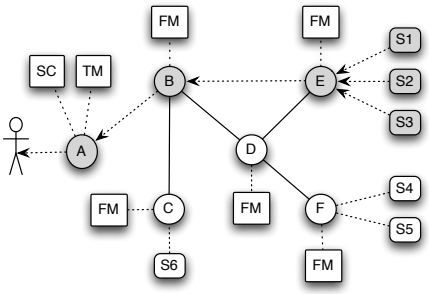

Fig. 9: Overview of the Fabric

We have defined interfaces that allow the fitting and allocation components to call on the Sensor Catalogue to obtain several kinds of information, including: available asset types (for fitting), available asset instances (for allocation), and current status of assets (obtained from the Fabric Manager and providing, for example, data to be used as

part of the computation of a utility value). Then, an interface to the Topology Manager allows us to specify how a set of selected instances needs to be configured to operate as a coherent system to deliver data to a user.

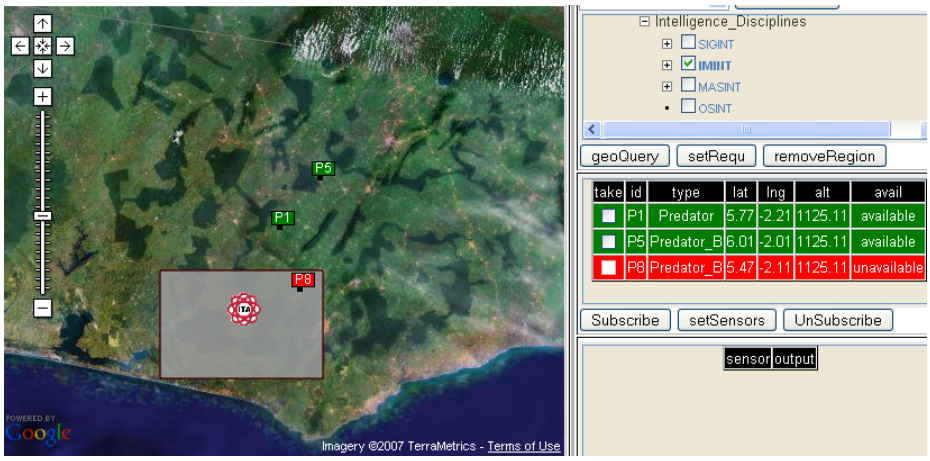

Fig. 10: Tool for integrating the reasoner and the Sensor Fabric

Figure 10 is a screenshot of a software prototype we are developing to enable the integration of the services provided by the reasoner and those provided by the Sensor Fabric. This screenshot shows the main user interface, which enables the specification of tasks, and supports users in the allocation of appropriate assets for their their tasks. Each task is characterised by qualitative requirements used by the reasoner to discover platforms that are fit-for-purpose. In addition, each task can define a geographic area of interest where some information has to be collected. The left part of the interface shows a map provided by Google Map Web services[8], where the user can specify the area of interest (overlapping rectangle) for a particular task. After selecting some requirements for the task, the user can retrieve the assets available that satisfy them, together with information on their availability and geographic location. In addition, the location of the assets is depicted on the map. In the example, we can see that user needs Imagery Intelligence (IMINT), which can be provided by three assets: one UAV of class Predator ($P1$) and two UAVs of class PredatorB ($P1$ and $P5$). In addition, we can see that one of the PredatorB platforms is not available, because it has crashed previously ($P8$). Therefore, in this case the user can select either $P1$ or $P5$ to accomplish the task. Multiple tasks can be defined, and then the assets committed to one task will not be available to other tasks. The next step is to integrate the allocation algorithms so as to maximize the global profit from multiple tasks.

6 Discussion and Conclusion

At present, we have initial implementations of the complete set of ontologies described in Section 2, a prototype of the semantic reasoner that performs the sensor-task fitting,

[8] http://code.google.com/apis/maps/

implementations of the SMD and SUM allocation algorithms outlined in Section 4, and an initial integration with the Sensor Fabric as described in Section 5.

Evaluation is ongoing: the fitting and Fabric components have been demonstrated and delivered to prospective US and UK users. Feedback on the overall approach has been positive, in particular:

- Grounding the framework on the Missions and Means Framework gives prospective users confidence in the overall fitting approach.
- The ability to draw on pre-existing sensor and task representations is seen as a key advantage of the ontology-centric approach.
- The extensibility of the multi-dimensional approach is viewed as a highly desirable and attractive element.

Moreover, the feedback obtained so far has highlighted a number of areas for further work:

- Allowing the user to specify ISR requirements at a higher level; currently, the ISR capabilities are framed too much in terms of intelligence types (IMINT, RADINT, etc, as shown in Figure 5(d)), and it would be desirable for users to specify "what they want" (e.g. detect vehicles at a particular location) rather than "how to get it" (IMINT, RADINT, etc). We are therefore working on mapping from such high-level information tasks to the kinds of intelligence suitable to satisfy them, as a step prior to the current fitting.
- Allowing the user to get explanations of the solutions recommended by the fitting process, and to explore "what if" alternatives.
- More detailed handling of deployable resources, to take better account of the cost of getting resources into the desired area when making fitting and allocation decisions.

In addition to these points, work on the integration is continuing; our longer-term goal is to use the fully-integrated set of components (sensor-task fitting, allocation, and deployment) to experiment with strategies to achieve maximum responsiveness and agility in assigning sensor assets to mission tasks in a distributed, dynamic, and multi-mission environment. The immediate challenges for these aspects of our work are to:

- expand and deepen the ontologies by working with domain experts in ISR, continuing to draw on pre-existing ontologies where possible;
- further develop the reasoning mechanisms used in the fitting process, including various approaches to non-exact matching allowing different ways to rank alternatives;
- fully integrate the fitting and allocation steps, including providing a richer treatment of utility.

Acknowledgements This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorised to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

1. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.P.: Semantic matching of web services capabilities. In: Proceedings of the First International Semantic Web Conference (ISWC 2002), London, UK, Springer-Verlag (2002) 333–347

2. Sheehan, J.H., Deitz, P.H., Bray, B.E., Harris, B.A., Wong, A.B.H.: The military missions and means framework. In: Proceedings of the Interservice/Industry Training and Simulation and Education Conference. (2003) 655–663
3. Robin, A., Havens, S., Cox, S., Ricker, J., Lake, R., Niedzwiadek, H.: OpenGIS© sensor model language (SensorML) implementation specification. Technical report, Open Geospatial Consortium Inc (2006)
4. Russomanno, D., Kothari, C., Thomas, O.: Building a sensor ontology: A practical approach leveraging ISO and OGC models. In: Proceedings of the International Conference on Artificial Intelligence. (2005) 637–643
5. McMullen, D., Reichherzer, T.: The common instrument middleware architecture (CIMA): Instrument ontology & applications. In: Proceedings of the 2nd Workshop on Formal Ontologies Meets Industry (FOMI 2006), Trento, Italy (2006) 655–663
6. Bermudez, L., Graybeal, J., Arko, R.: A marine platforms ontology: Experiences and lessons. In: Proceedings of the ISWC 2006 Workshop on Semantic Sensor Networks, Athens GA, USA (2006)
7. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F., eds.: Description Logic Handbook. Cambridge University Press (2003)
8. Castillo, G.J., Trastour, D., Bartolini, C.: Description logics for matchmaking of services. In: Proceedings of Workshop on Application of Description Logics (KI 2001). (2001)
9. Di Noia, T., Di Sciascio, E., Donini, F.M.: Semantic matchmaking as non-monotonic reasoning: A description logic approach. Journal of Artificial Intelligence Research (JAIR) **29** (2007) 269–307
10. Tutton, S.J.: Optimizing the allocation of sensor assets for the unit of action. Technical report, Naval Postgraduate School, California (2006) Master Thesis.
11. Bar-Noy, A., Brown, T., Johnson, M., La Porta, T., Liu, O., Rowaihy, H.: Assigning sensors to missions with demands. In: Proceedings of the 3rd International Workshop on Algorithmic Aspects of Wireless Sensor Networks. (2007)
12. Rowaihy, H., Johnson, M., Bar-Noy, T.B.A., La Porta, T.: Assigning Sensors to Competing Missions. Technical Report NAS-TR-0080-2007, Network and Security Research Center, Pennsylvania State University (October 2007)
13. Pizzocaro, D., Chalmers, S., Preece, A.: Sensor assignment in virtual environments using constraint programming. In: Proceedings 27th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence, Springer (2006) 333–338
14. Bergamaschi, F., Conway-Jones, D., Gibson, C., Stanford-Clark, A.: A distributed test framework for the validation of experimental algorithms using real and simulated sensors. In: First Annual Conference of the International Technology Alliance (ACITA 2007). (2007)

Appendix - Glossary of acronyms

ACINT:	Acoustic Intelligence
ISR:	Intelligence, Surveillance and Reconnaissance
FOPEN:	Foliage Penetration
HALE-UAV:	High Altitude Long Endurance UAV
IMINT:	Imagery Intelligence
LDRF:	Laser Designator and Range Finder
MALE-UAV:	Medium Altitude Long Endurance UAV
MMF:	Missions and Means Framework
SCM:	Set Cover Matchmaker
SIGINT:	Signals Intelligence
SMM:	Sensor Mix Model
UAV:	Unmanned Aerial Vehicle
VTOL:	Vertical Takeoff and Landing