Abstract—In compliance with the Department of National Defence and Canadian Forces Architecture Framework (DNDAF) and the Service Oriented Architecture (SOA), we develop a new web services implementation of a Multi Sensor Data Fusion (MSDF) System. The migration and the decomposition into web services of the Concept Analysis and Simulation Environment for Automatic Target Tracking and Identification (CASE_ATTI) test bed, developed at DRDC Valcartier, is realized using a spiral approach. A validation process of the web services is proposed by comparison with the results generated by the CASE_ATTI test bed. We also present some general guidelines that should be satisfied by a MSDF web services provider.

Keywords: Multi-sensor data fusion, web services, DNDAF, interoperability.

I. INTRODUCTION

In the last decades, government agencies, universities or industries, focused on Multi-Sensor Data Fusion (MSDF) systems for military and non-military applications. Bowman and Steinberg [1] and Waltz and Llinas [2] already addressed some questions about the design and the implementation of such MSDF systems. A lot of efforts were deployed to develop several implementations of these MSDF systems or of some components of them, using a diversity of architectures, platforms, applications and programming languages. One of the existing implementations of such a MSDF system is the Concept Analysis and Simulation Environment for Automatic Target Tracking and Identification (CASE_ATTI) test bed realized by Defence R&D Canada - Valcartier [3]. Other implementations can be found in [4] and more general surveys of the data fusion applications and products are reported in [5]–[7].

However, one of the problems invoked by several major actors in this field is the poor and heavy interoperability capabilities between the existing systems. We propose in this paper to use a web services approach within a Service-Oriented Architecture (SOA) [8] to decompose the CASE_ATTI test bed into services and thus, to create a more flexible and platform/application independent MSDF system, according also to the Department of National Defence and Canadian Forces Architecture Framework (DNDAF) [9] which is an extension of the US Department of Defence Architecture Framework (DoDAF) [10]–[12].

Section II presents a brief introduction to the CASE_ATTI test bed. Sections III and IV detail the DNDAF and the SOA architectures, respectively. Section V shows a brief decomposition of the CASE_ATTI test bed into essential services and proposes a spiral decomposition into more granular services. Section VI introduces the web services’ validation process. Section VII presents some challenges for the migration process from the CASE_ATTI test bed to a web services implementation. In Section VIII we present some characteristics of the web service provider. Section IX is the conclusion.

II. CASE_ATTI TEST BED

The CASE_ATTI test bed [3] developed by DRDC Valcartier, provides a highly modular, structured and flexible hardware/software simulation environment for the Multi-Sensor Data Fusion. The aim of this test bed is to study and compare various advanced MSDF concepts and schemes in order to demonstrate their applicability, feasibility and performance. A high level structure of the CASE_ATTI test bed is presented in Figure 1.

CASE_ATTI is composed by three main modules:
(a) A **Stimulation module** which emulates the behavior of real targets, sensor systems and the meteorological environment in an Anti-Air Warfare (AAW) context.

(b) A **Level 1 Data Fusion (L1DF) module** which provides the algorithm-level test and replacement capability required to study and compare the technical feasibility, applicability and performance of advanced, state-of-the-art L1DF techniques.

(c) A **Performance Evaluation (PE) module** which evaluates (using different metrics between different parameters) the performance of the algorithms from the L1DF module.

### III. DNDAF: A COMMON ARCHITECTURE GUIDE TO DESIGN MSDF SYSTEMS

In the last 30 years the size and the complexity of the information systems increased very quickly and the need of a logical construct (or architecture) for defining and controlling the interfaces and the integration of all the components of a system became more a necessity than an option. Several architecture frameworks were developed to deal with the complexity of the information systems. Zachman's framework (first introduced in [14] and later revised in [15]) is widely regarded as the first Architecture Framework, from which others evolved, such as the C4ISR Architecture Framework [16], DoDAF (v1.0 and v1.5 [10]–[12]), MoDAF [17], NAF, AGATE Architecture Framework, TOGAF [18], which have been used to some degree of effect as Systems Architecture Frameworks. The DoDAF broadened the applicability of the last version of Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance (C4ISR) architecture framework to all the mission areas rather than just to the C4ISR community and is heading toward a new type of information intensive warfare known as Net-Centric Warfare. Services are a key means to share information and capabilities in a Net-Centric Environment (NCE) through published service interfaces. A service is a self contained function in which the consumer does not know (or care) “how” the service is implemented. It only knows that the service performs “what” is defined by its published interface.

The Department of National Defence and Canadian Forces Architecture Framework (DNDAF) [9] adjusts the DoDAF, MoDAF and other already known architecture frameworks to the Canadian DND and the Canadian Forces requirements.

The DNDAF defines a set of 31 products for visualizing, understanding, and assimilating the broad scope and complexities of an architecture description through graphic, tabular, or textual means and to ensure uniformity and standardization in the documentation (from requirements to implementation) and communication of architecture. The products and elements defined within the DND Architecture Framework are organized under six views, each depicting certain perspectives of the architecture:

- **Common View (CV)**: Describes the global architecture plans, scope, definitions, context and taxonomies.
- **Operational View (OV)**: Focuses on the behaviors and functions describing the enterprise mission aspects
- **System View (SV)**: Describes the system and applications supporting the mission functions - relates systems and characteristics to operational needs.
- **Technical View (TV)**: Describes a set of standards governing the implementation, arrangement, interaction, and interdependence of systems.
- **Information View (IV)**: Provides the definitions and structures of information required to make decisions and to manage the resources.
- **Security View (SecV)**: Focuses on the attributes of the architecture that deal with the protection of the operational assets (including the information assets).

These views and their relationships (shown in Figure 2 as presented in [9]) enable one to measure the interoperability and the performance of the system and also the impact on mission and task effectiveness.

![Figure 2. Relationships between Views (from [9])](image)

### IV. WEB SERVICES WITHIN A SERVICE-ORIENTED ARCHITECTURE

A Service-Oriented Architecture (SOA) is “a way of sharing functions in a widespread and flexible way” [19] and “an architectural style where existing or new functionality are grouped into atomic services” [8].

The concept of a SOA is not a new concept. SOA supports the transformation of a complex application into a set of linked services (repeatable basic applications) that can be accessed when needed over a local network or over the internet. What is new in a SOA is the property of loose coupling, meaning that the implementation of a service is hidden from the caller of this service. Loosely coupled services can thus use incompatible system technologies and nevertheless they can easily be joined together to create composite services. The strength of such a system is that it reduces the risk that a change in one application/module will force a change in another application/module.

Service Oriented Architectures can be implemented using a wide range of technologies, such as SOAP, RPC, DCOM, CORBA, WCF, or Web Services, and a file system mechanism could be used to communicate data between services/processes.
Web services (WS) implement a Service-Oriented Architecture. A major focus of Web services is to make functional building blocks accessible over standard Internet protocols that are independent from platforms and programming languages. These services can be new applications or just wrapped around existing legacy systems to make them network-enabled [8]. Figure 3 shows how a web service system works. The numbers indicated in the diagram show the order of the interactions between the different actors involved in a web service system.

In the next sections we present a web services implementation, using a SOA Architecture, of a Multi-Sensor Data Fusion System. Our goal is to make functional building blocks accessible over standard Internet protocols that are independent from platforms and programming languages.

V. CASE ATTI DECOMPOSITION INTO WEB SERVICES

A target identification test scenario is presented in Figure 4. This graphic is associated to the DNDAF OV-1 view, which represents a High-Level Operational Concept Graphic.

The Systems/Services Functionality Description (DNDAF SV-4) is depicted in Figures 5 and 6, which present a basic decomposition of a MSDF system into crucial modules (from [20]) and a detailed decomposition (from [21]) respectively.

A first step decomposition of the CASE ATTI into web services should be made according to its main modules: the stimulation, the level 1 data fusion and the performance evaluation modules as presented in Figure 1.

Figure 3 shows an implementation of the Stimulation web service. A detailed block diagram of the stimulation module can be found in the left side of the Figure 6.

The Stimulation web service is based on the following steps:

1) the client connects to the stimulation web-server and obtains a client ID number;
2) the client provides to the stimulation web-server the stimulation CONFIG files, the input socket port, address and key for the report reception;
3) the stimulation web service extracts the CONFIG files in a separate directory and spawns a sensor module specifying the client’s socket port, address and key;
4) the client receives the contact reports (pop them from the socket) and can process them at will;
5) the client asks the stimulation web-server to close the stimulation simulation;
6) the stimulation web-server closes all processes and removes the CONFIG files associated to the completed simulation.

Figure 8 presents the functionality of a Tracking web service. The block diagram of the Tracking module can be found in the right side of the Figure 6.

The Tracking web service is based on the following steps:
1) the client connects to the tracking web-server and obtains a client ID number;
2) the client provides to the tracking web-server the tracking and connection CONFIG files, the input socket port, address and key for the reports reception;
3) the tracking web service extracts the CONFIG files in a separate directory and spawns the tracking module specifying the client's socket port, address and key;
4) the tracking web service asks for the contact buffers which are provided by the client using the "stimulation output" file;
5) the client receives all the buffers (pop them from the socket) and can process them at will;
6) the client asks the tracking web-server to close the tracking simulation;
7) the tracking web-server closes all processes and removes the CONFIG files associated to the completed simulation.

The Performance Evaluation (PE) Module from CASE/ATTI is divided in several parts: a 2D-3D viewer, a Performance Analysis Database (PADB) and a Measure Of Performances (MOP) module.

Figure 9 shows a first attempt of the PE Module' migration into a web service.

In the next step, using a top to bottom incremental decomposition, each WS created in the previous step could be broken up, if possible and necessary, into several web services, which may lead to a very fine granularity of services. For example, the L1DF web service could be broken up into more granular web services such as: tracking, data association, kinematic fusion and identity information fusion. The PE web service could be broken up into more granular web services such as: radial missed distance, accuracy of the filter calculated covariance, state estimator error, etc. Figure 10 shows such an incremental decomposition.
Different measures of uncertainty for evidence theory, probability theory or fuzzy sets theory should also be considered in the decomposition of a MSDF system into web services, even if they were not originally part of the CASE ATT test bed. More details about measures of uncertainty used in data/information fusion can be found in [22].

VI. VALIDATION OF THE SERVICES

In this section we only concentrate on the validation process of the main web services derived from the CASE ATT test bed (see Figure 1) which are the Stimulation, L1DF and Performance Evaluation web services. This validation step supposes to disconnect from the CASE ATT test bed one module at a time and use the associated web service instead. We then perform tests using the hybrid CASE ATT test bed and the real CASE ATT test bed and the results should match together.

Figure 11 shows how the independent stimulation and tracking web services can be used together to perform the same task as the corresponding modules from the CASE ATT test bed.

VII. CHALLENGES FOR THE WEB SERVICES

ARCHITECTURE

One of the challenges of CASE ATT’s decomposition into web services is to create an optimized environment for simulation and tests.

For example, in the last decades, several authors proposed a set of alternatives to Dempster’s rule of combination to better cope with conflicting evidences. Such alternatives are Yager’s rule, Dubois and Prade’s rule, PCR rule, etc. In a first step we have to consider the migration of each one of these modules as independent (stand-alone) web services.

However, we remark that Dempster’s rule and some of its alternatives are closely linked to the same conjunctive rule. Thus, realizing a comparison test between these combination rules, without being aware of the links between them, could cause an unnecessary computational overload, by realizing redundant computations. If in a simple test this situation is not critical, it can become critical in heavy Monte Carlo simulations.

The client of a MSDF web server should not be concerned about how the granular services are implemented. The MSDF web server should be able to detect the client’s usage and should adapt the services’ orchestration for an optimal usage.

VIII. WEB SERVICES PROVIDER

The goal of CASE ATT’s decomposition into web services is to make functional building blocks accessible over standard Internet protocols that are independent from platforms and programming languages. The decomposition of the CASE ATT test bed into granular web services is only the first step of this work.

Some of the services could not be real-time services (or almost real-time services) and will need important computation time before providing the results. The un-truncated Dempster’s rule of combination used to combine a lot of evidential information is only one example of a non real time service. To overcome any problem induced by the time-expensive execution of the services, the following aspects should also be considered in the design of the web services:

- The service provider should be able to assist several clients at the same time. Thus, a parallel service provider should be implemented. However, since it is not possible to provide an infinite number of parallel services, a First-Come-First-Served Queueing System should also be implemented.

- The service provider should be able to provide (upon request) an estimate of the execution time of a requested service. The service provider should give the option to the client to abort the service request if he considers the waiting too long.

- Some service requests could be introduced into a queueing system, and the service provider should be able to provide (upon request) an estimate of the remaining time until the requested services will be processed. The service provider should also give the option to the client to abort the services requests if he considers the waiting too long.

- Some service requests could necessitate important computation time. A client could be disconnected from the web server, but he should be able to collect the results of the requested service whenever he connects to the web server.
IX. CONCLUSIONS

In this paper we provide a web services approach in the design of a Multi-Sensor Data Fusion (MSDF) System. We use the CASE ATTI test bed for MSDF systems - developed at DRDC Valcartier, as a starting point of our migration toward a MSDF web services architecture. The design of the stimulation, the tracking and the performance evaluation web services was proposed here. A more granular decomposition into web services is still under study. A validation mechanism was also proposed. An optimized implementation of a MSDF web services architecture cannot be realized without addressing some challenging issues also presented in this paper.

ACKNOWLEDGMENT

This work was supported by DRDC Valcartier Canada under the contract number W7701-6-3797. The content of this paper is exclusively the responsibility of the authors.

REFERENCES