Interoperability-Ready, Training-Focused Architecture for Command and Control Systems

Alejandro J. M. Repetto
CIDESO, DIGID - EST, IESE – Ejército Argentino
Ciudad Autónoma de Buenos Aires, Argentina
ajmrepetto@gmail.com

ABSTRACT

There are countless differences between a learning environment and a battlefield. Therefore, a unified architecture which allows for dynamic interaction between reality and simulation would be a priceless achievement. Embedding simulation components in C2 systems for training purposes has always been a challenge from the realism and interoperability standpoints. Prevailing interoperability and composability-oriented frameworks such as TENA are fit for large scale environments; yet they surpass the requirements for less-demanding scenarios and increase the complexity of the solution. The Interoperability-ready, Training-focused Architecture (ITA) addresses these issues by being based on nonmilitary-specific technical software solutions, reducing the complexity and, therefore, deployment costs.

ITA is a layered architecture built upon COTS/Open software solutions which allows for the coexistence and interaction of both C2 and simulation modules in the same virtual environment. Its layers are designed to improve interoperability among the involved technologies, from communications hardware to service coordination software. In the upper layers, the use of widespread technologies (e.g. P2P and SOA), with well-known military ontologies (e.g. JC3IEDM) permits the dynamic interaction between reality and simulation, enhancing the training capabilities of the whole system. Additionally, it makes ITA interoperable with other widespread frameworks (e.g. TENA), improving its scalability. The layers' independence also facilitates rapid development of new applications by developing small, self-sufficient pieces of software.

The aim of the present work is to report the results of using ITA in the Argentine Army. First, a description of the problem and analysis of possible solutions are outlined, arriving then at the specifications for architecture's layers and their associated technologies. Lastly, the outcome of the actual implementation is analyzed, evaluating the impact on both the development process and the user experience, analyzing pros, cons and potential improvements.

ABOUT THE AUTHORS

Alejandro J. M. Repetto, M.Sc, is a Team Leader of the research staff at the CIDESO (Software Research and Development Office of the Argentinean Army) since 2006. He received his first M.Sc with honors in Information Systems Engineering from the Escuela Superior Técnica del Ejército Argentino, Buenos Aires, Argentina in 2004, where he also completed a specialization post-graduate program in Cryptography and Computing Systems Security in 2007, and his second MSc. in Computer Systems Engineering from the Politecnico di Milano, Italy in 2008. He is alumni at the Alta Scuola Politecnica, a school for young talents at the Politecnico di Milano and Politecnico di Torino, Italy. He acted as system security engineer and as consultant in software engineering for the private area, always maintaining a focus on innovation. He acts as associate professor at the simulation department of the Escuela Superior Técnica del Ejército Argentino. His research focuses include grid computing, algorithms optimization and simulation modeling.

Interoperability-Ready, Training-Focused Architecture for Command and Control Systems

Alejandro J. M. Repetto
CIDESO, DIDEP - EST, IESE – Ejército Argentino
Ciudad Autónoma de Buenos Aires, Argentina
ajmrepetto@gmail.com

INTRODUCTION

Shifting from a learning environment to the battlefield has always been a challenge for combatants. Wargames have been widely employed for improving decision-making skills at mid and high command ranks in most armies. However, there are countless differences between the learning environment and the battlefield. Developing a transparent interoperability framework with a unified concept for both training and command and control (C2) is bound to reduce the impact of this transition. The construction of such a framework implies the development of not only a common graphical user interface (GUI) but also a complete set of tools and middleware, with its integration interfaces, which can dynamically connect different data sources (e.g. from the field –sensors and reports-, from simulation models or from specific C2 tools). Even though it is true that there are other interoperability frameworks in use in the military, *e.g.* TENA, HLA and DIS, ITA hopes to simplify systems integration by using standard open architectures such as Enterprise Services Bus (ESB), peer to peer (P2P) network and Service Oriented Architecture (SOA) approaches, proposing a new vision for this problem.

ITA is a complete, flexible and scalable architecture which supports both military operations (including the interaction among different branches of the armed forces – Army, Navy and Air Force– and National Security Forces – Cost Guard and Militarized Police–) and training. Guaranteeing interoperability among C2 systems and training systems from scratch is essential to succeed in the long term. Moreover, as training systems mostly oriented to Live, Virtual and Constructive (LVC) simulations are responsible for training and for decision-making support tools, so their integration with C2 is considered essential for the solution.

ITA, thus, is a flexible framework that merges C2 and training systems under a single architecture, using Commercial off- the-Shelf (COTS) technologies and well-known architectures, tailored to those setups where available military-specific interoperability frameworks exceed the requirements and add complexity to the solution.

Technical Context

The most widespread interoperability platforms have been studied and evaluated looking for a useful starting point to develop ITA (Millmore, 2006). Test & Training Enabling Architecture (TENA), High Level Architecture (HLA) and Distributed Interactive Simulation (DIS) have been considered in the studies. While being worldwide standards in reference to interoperability, two drawbacks have been detected for their incorporation into the Argentinean Army's solution: they exceed the requirements and they are complex tools that demand specialized skills, both for developing and maintaining. These frameworks are meant for bigger scale and more advanced military technologies. However, it is considered important to be aware of their interfaces in order to allow for future interactions with other forces and permit the evolution of the framework.

The purpose of ITA is to support the operations of the local Army, taking into account available technology and the size of the problem; yet interoperability with other frameworks is still considered to be fundamental. Therefore, the four guidelines for developing the platform can be summarized as follows:

1. Allow for interoperability among the C2, training and decision support systems within the Army.
2. Define clear interfaces to allow interaction among the Army, the other branches of the Armed Forces (National and Regional) and National Security Forces.
3. Use COTS technology in order to facilitate the development and operation.
4. Permit the evolution of the solution through the design of a flexible and scalable framework.

INTEROPERABILITY- READY, TRAINING-FOCUSED ARCHITECTURE

The Architectural Triad

ITA is built upon three different views: the *layered view*, the *component view* and the *interconnectivity view*, creating the so called Architectural Triad (AT). Each view addresses a specific design problem, and combined they describe the architecture in full. Additionally, there is a fourth view, the *procedures view*, which is tasked with blending the technical aspects with the doctrine and the procedures, adapting and updating them along with the technical advancements. This view will not be discussed in this work, which is mainly technical. Figure 1 shows a schematic view of ITA along with its three technical facets.

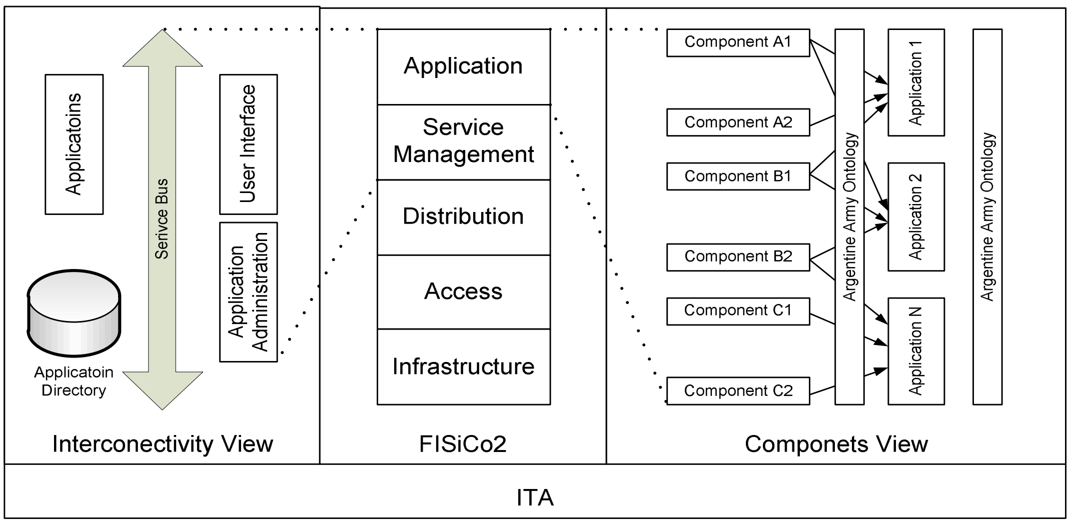

Figure 1. Interoperability-Ready, Training-Focused Architecture: Architectural Triad.

Although design, implementation and maintenance of the AT can be a complex endeavor, it is still simpler than for the available frameworks, and can allow small armies to include a Joint C2 and Training System (JC2TS) to its facilities at a low ownership cost (Pullen, et al., 2010). Furthermore, considering that it has been developed from the ground up but taking into account previous experiences like TENA, HLA and DIS, it sets up a novel concept for JC2TS for smaller-scale armies when compared to the US' or NATO's frameworks.

The *layered view* is divided into levels where each one is responsible for a group of tasks on its own, and communicates with its neighbors by well-defined interfaces; without going further into business details. This view outlines the system from a technical perspective. The layers specify how the information flows through the system, defining the way the data is exchanged. This solves the underlying technical problem of military communications by increasing reliability, robustness, security and flexibility; but it does not characterize the data's semantics. This view is fundamental to the systems' integration, interoperability and scalability, taking into account that layer independence and the interface hard-specification allow for implementation adaptability, increasing the architecture's flexibility.

In contrast to this, the *component view* main focus is on business logic. Components are pieces of software which either represent the reality, in simulation mode, or provide services to the users and collect data from sensors, in C2 mode. Components are, in fact, military applications designed in the most atomic-way possible, allowing for composability and enhancing the systems' scalability by enabling a rapid development strategy. This view is responsible for the interoperability, from the data semantics angle. The interfaces defined for the components should be unique for both simulation and C2. Doing so, the concept of a "plug-and-play" system, where users can switch on-the-fly from a training environment to battlefield applications can be achieved. The exchanged data must comply with a pre-defined standard in order to facilitate the internal interoperability, i.e. within the Argentine Army, including legacy systems, and the external interoperability, i.e. other forces and nations, but always behaving like black boxes where the fundamental issue is the interfaces' implementation. The adopted standard is the Argentine Army Ontology (AAO), which is based on the NATO's Joint C3 Information Exchange Data Model (JC3IEDM).

Lastly, the *interconnectivity view* consists of a broker-like architecture, which manages the components in a dynamic way. This perspective shows how they can be combined in order to build up different system configurations by mixing LVC simulation with reality, considering each unique operative situation. Using a service directory, it is capable of attaching components in a single user application for both C2 and training, being the JC2TS' functional core. This view enables the switching from a training environment to the combat field in a transparent way.

In summary: the AT, constituted by these three views, addresses the technical aspects, the business aspects and the interconnectivity aspects of JC2TS. It provides a complete framework for JC2TS, combining different design approaches and using well-known design strategies and open products.

Layered view: FISiCo2

The layered view, called FISiCo2 (Framework de Interoperabilidad para Sistemas de Comando y Control[1]), is in charge of the global description of the architecture. It handles robustness, reliability, security and low level interoperability, i.e. communication protocols and standards. It divides the problem in five different layers, each one independent from the others and with well-defined interfaces.

FISiCo2 tackles the problem inherent to any complex system by dividing it into several facets: (1) *infrastructure*, involving any kind of physical artifact, e.g. computers, radios, vehicles, UAV, etc.; (2) *access*, responsible for interpreting and managing the hardware through software representations, can be seen as a set of drivers; (3) *distribution*, in charge of the logical communication among system's endpoints; (4) *service management*, responsible for the interaction among subsystems; and (5) *application*, where all the software components and services can be found, forming a cloud-computing-like architecture.

The first layer involves ITA's physical aspects: It includes not only the communication's hardware, i.e. TCP/IP devices to HF or VHF military radios but also radars, vehicles (manned and unmanned), weapons, and any other kind of artifact that can be used in the JC2TS. Layer 2 is in charge of masking the complexity and the hardware's heterogeneity. In a nutshell, it links the hardware layer with the upper software layers by using a single communication protocol for any hardware, acting as a driver of sorts. Layer 3 is responsible for the logical communication among users and systems. It provides routing, security and AAA services (Authentication, Authorization and Access). The routing service is centered around a context-aware component which chooses the best path by weighting factors such as security, robustness and bandwidth of available links. Layer 4's objective is to manage the application directory, giving support to application integration and providing publication, discovery, invocation and management services, acting as a kind of orchestrator amongst different applications. It is mostly oriented towards reutilization, allowing for the composition of new applications by combining already-implemented services. Finally, the application layer is the space where all the services and components reside. Each one has its own architecture, but they interact by using layer 4's interface specifications. The applications should be as granular as possible, to improve the composability and maximizing the reutilization.

This top-down approach allows for dividing the problem of JC2TS by its technical challenges: infrastructure, access, distribution, orchestration and implementation. If the interfaces for each layer are correctly defined, determining the fundamental services that each of them should provide and their responsibilities, this generic view becomes fully independent from the technology, enhancing the system's development process, robustness and scalability.

Layer 1: Infrastructure

The Infrastructure layer involves all the physical systems necessary for running both C2 and training software. It includes computers and communication elements, such as modems, routers, switches, satellites,

[1] Interoperability framework for C2 Systems

vehicles, GPS and radars. All these elements should be considered in the architecture in order to design a coherent system. These artifacts determine restrictions and the system's functional and non-functional requirements.

Usage of HF or VHF radios for communications, for instance, imposes a hard restriction to the amount of data and the transmission speed capacities of the system, forcing designers and developers to pay special attention to dataflow optimization. A similar situation occurs with the system's use conditions: if a system must be robust enough to handle the loss of several hardware pieces, e.g. when using the system for C2, the software is required to distribute the information in a way that the system continues to operate no matter how many nodes are lost. These examples show why this layer, even if it does not include any software, impacts the architecture.

The technologies selected for implementation of ITA in the Argentine's Army are, so far, related to communications. ITA must permit information exchange using HF and VHF military radios that support ALE (Authomatic Link Establishment) mode, especially for C2 or mixed C2 training scenarios. This restriction, for example, introduces at least two requirements: the communications should be hybrid (TCP/IP and HF/VHF), and they should be as light as possible, all this while taking into account the radios' limited bandwidth. Additionally; radars, GPS, UAVs, UGVs and cipher hardware have been detected as legacy infrastructure that ITA should deal with in the near future.

Layer 2: Access

This layer is tasked with masking the heterogeneity of Layer 1's components. Even though the access layer is mainly focused on communications it also deals with the homogenization of other technologies such as GPS' protocols, radars' drivers and unmanned vehicles' controllers.

The main issue that ITA has to deal with is the usage of military radios (e.g. HF and VHF) as data transmitters. Frequently, military communication equipment has embedded optimized protocols to send and receive data packets. However, as these are proprietary implementations, in order to build ITA over open technologies it is necessary to use common standards, supported for most radios, in order to exchange data over these kinds of links.

STANAG 5066, NATO's standard for data applications over HF, is the most widely used protocol to transmit data packets using radio frequencies. A driver has been developed for transmitting data over HF links, standardizing the interfaces and granting independence between the communication's hardware and the software. Therefore, for each implementation there will be a driver to mask the hardware problem.

Additionally, STANAG 5066 describes the use of HMTP (HF Mail Transfer Protocol), which permits data exchange using an e-mail-like (SMTP) protocol. Also a driver using these specifications has been developed for data transmission, inserting the application information into the payload of an e-mail. The advantage of using HMTP, which is a standard as well, is that new generation military radios optimizes communication for this protocol, reducing transmission issues, like turnaround time.

Finally, in order to tackle the problem of concurrency, when many radios try to send messages to the command center (no matter if it is the HQ during a real operation or the datacenter during a simulation game), the Automatic Packet Reporting System (APRS) protocol has been adopted. APRS establishes a network of servers which acts as middleware between a radio network and a TCP/IP network. The APRS servers can be seen as proxies which represents the radios in the data network.

In summary, for homogenizing the radio data transmission, STANAG 5066, HTMP and APRS technology have been implemented. Other communication devices, such as satellites, have their own embedded drivers that allow for interoperability with TCP/IP.

Layer 3: Distribution

This layer was designed as an overlay network (Doval & O'Mahony, 2003), called ODiN (Overlay Distribution Network). It hides the heterogeneity of the infrastructure layer, using Layer 2's drivers, and adds robustness, security and priority management services to the network. ODiN presents a generic system that is not attached to any technology. Nevertheless, in order to fulfill the requirements, the architecture is based on a Peer-to-Peer (P2P) communication strategy.

The robustness of the P2P is based in the possibility of administering multipath connections in a dynamic fashion, avoiding the single point of failure problem. As all the network's participants act as possible hops in the communication between two endpoints, the number of paths for connecting two peers grows quadratically to the number of available peers. This feature is given by the P2P's self-organization ability.

Also, by the use of groups, hierarchies and access level management, P2P frameworks improve ITA's security, guaranteeing confidentiality. Moreover, these frameworks allow configuring sets of users by their interests or roles in the system, building virtual broadcast domains and optimizing the communications.

In order to put this idea into practice, it was necessary to implement a specific P2P infrastructure which could handle the requirements. In the beginning, developing a basic P2P platform from scratch was considered. Nonetheless, this idea has been discarded considering that the non-functional requirements started to grow rapidly. Security, scalability, robustness and transparency issues have arisen, and the systems turned extremely complex for being developed from zero. Based on this observation, two platforms have been analyzed: Sun Microsystems' JXTA and .NET My Services. JXTA was, finally, the chosen one.

Additionally to the JXTA implementation and associated to this layer, a Context Aware Data Manager (CADM) is planned. The CADM component increases the routing's intelligence by sensing the context in which the messages must be exchanged. It is aware of two kinds of contexts: technical and operational. The first refers to communication aspects such as available bandwidth, security and reliability of the links, etc. These characteristics are informed directly by the peers, which announce the status of the links that join two endpoints. The operational context refers to the situation in which the message is being sent. For instance the CADM can suggest the best encryption algorithm to use based on the position or the confidentiality level of the data-packet. This component has been designed in-house, using BDI Agents technology, and is being implemented using a standard BDI Java framework called JADE[1].

In summary, layer 3 is an overlay network, implemented in JXTA, which applies the P2P concept for message delivery. Additionally, using another open standard, JADE, a context-aware data manager is included in this layer, providing intelligent routing capacities to the solution.

[1] http://www.jadex.org

Layer 4: Service Management

Service management layer (SML) provides a Service Oriented Architecture (SOA) facility to FISiCo2 by implementing an Enterprise Service Bus (EBS)-like architecture (Leymann, 2005). This layer outlines how services interact among them. It defines the protocols for publishing, invoking and discovering services by maintaining a service directory. When a layer 5's application requests a service, the SML resolves the call by announcing the relative address of the service provider through ODiN, using the virtual P2P addresses. It is important to highlight that SML provides the mechanisms for the services' interaction and not the services themselves.

These mechanisms enhance the ITA's composablity feature, providing a way to generate new applications by combining available services (Charfi & Mezini, 2004). It also improves reliability and system performance by permitting service redundancy. If a service provider goes down, other available provider can respond, setting up a high-performance, failure-resilient cluster facility.

Additionally, the concept of SOA allows for developing components in different programming languages, choosing the best fitting language for each piece of software without caring about the technical integration.

Finally, this layer contributes to improve the manageability of FISiCo2, providing tools to easily configure the business rules for the services interaction, permitting the use of BPEL (Dobson, 2006).

SML's implementation has been also done with JXTA. JXTA provides an advertisement service for publishing available services. An advertisement is an XML document that describes JXTA's messages, peers, peer groups, or services. They are used for exchanging information about what services are available in the JXTA network (Brookshier, Govoni, Krishnan, & Soto, 2002).

The services' management is carried out by the peer groups. Peer groups are contexts for services that interoperate in a users' group domain. They offer discovery, membership, access, peer authentication, pipes, resolver and monitoring services to implement a full SOA infrastructure. Based on these services, JXTA provides interfaces to build group applications, improving the resource-sharing capabilities of the system.

It is important to remark that, even if the same technology has been chosen for both layer 3 and 4, they are conceptually different and their implementation is independent. They are logically divided by internal interfaces in order to enable future changes in any technology.

Layer 5: Application

For ITA's scope, the application's layer is the space where all business-related software cohabits. It does not include the infrastructure software which resides in the lower layers, such as drivers, P2P frameworks or SML; they are support systems but not C2TJS' applications. This layer is the one nearest to the end user and shapes a cloud computer-like vision in the sense of application collaboration.

Basic military software, such as the Battle Management System (BMS), Situation Aware System (SAS), Message Exchanging System (MES), Constructive Simulation System (CSS) and User Interfaces (UI), share here a common space, using the underlying layers for interaction amongst them. Even though all of them have their own internal software architecture, they must communicate using the protocols established by the SML.

Another restriction to the applications' software architecture defined by ITA is that all of them must maintain the focus on interoperability, using open standards. For instance, the MES uses the recommendations given by ACP123/STANAG 4406 (Combined Communications-Electronics Board, 2008), with data structures for exchanging messages between nations.

Also, all data structures implemented in both the systems' objects and the data models comply the NATO's Joint C3 Information Exchange Data Model (JC3IEDM) as a baseline ontology. Doing so, the interoperability with other military systems should be straightforward, optimizing the collaboration in joint military operations.

Components View

Applications are made from components. The FISiCo2's application layer generically describes the applications that a JC2TS should consider and how they must interact with each other by using the SML. However, in order to maximize the composability capacity of the architecture, the components view defines how a component must be developed in order to ensure reutilization.

The main recommendation given to architects by this definition is that all the applications must be designed as a sum of atomic and autonomous components, as suggested in (Schade, 2005). These pieces of software must have well-defined interfaces and the exchanged information should comply with the AAO.

Aligning all the components within a single ontology assures internal interoperability and allows for the development's parallelization. Interoperability is defined by the usage of common interfaces, with the components behaving as black boxes, permitting the development of applications without the need to delve deep into each piece of software. Furthermore, as the ontology defines data structures and concepts, it makes a parallel development possible, where many work teams can develop different parts of the software with the certainty they will be integrated without any problem.

The AAO is based on J3CIEDM for ensuring external or international interoperability. The components and their interfaces, when using data structures included in the J3CIEDM, fully comply with its recommendations. However, not all of the necessary structures are described in this standard. Hence, an extended standard has been implemented: the AAO. If any change or adaptation is needed in order to align the J3CIEDM to the AAO, it is then documented and the transformation interfaces are built alongside the development of the component.

In summary, this view implements a cloud computing concept, assuring the system's reusability, composability and interoperability, optimizing computational resources and development time.

Interconnectivity View

The aim of the *interconnectivity view* is to establish rules about how components interact among them. This view introduces a middleware capable of connecting on-demand sources of simulated and combat zone data, reaching a unique training capability where both C2 and training elements share the same information space.

This perspective functions within layers 4 and 5 of FISiCo2, where the applications and the orchestration are implemented. The design is a cloud computing-like platform, where groups of applications cooperate through common interfaces and standard languages in order to do both training and C2.

The strategy is to develop interfaces – web services or others, depending on the performance, security, robustness and technology requirements– for all the systems that we would like to integrate, i.e. legacies and new systems, using whenever possible common military language definitions such as JC3IEDM, Battle Management Language (BML) and Military Scenario Definition Language (MSDL), as recommended in (Tolk & Blais, 2005). By doing so, future interoperability capacity with other military systems is ensured.

All applications, consumers and producers, will communicate with each other using FISiCo2's Layer 4, the SML, using COTS technologies for cloud computing, e.g. Globus Toolkit, Alchemi, standard application servers, Service Oriented Architecture (SOA). In particular, SOA approaches allow for a set of self-sufficient systems to be managed in a relatively easy way and, at the same time, it enables the building more complex structures by combining them in a synergic way.

In order to gain a clearer understanding of this view, two scenarios will be proposed: a training scenario and a real operation scenario.

For the training scenario, we will consider a simulation exercise using a constructive simulation system. The brigade's commander and the staff are deployed in a training cabinet and the trainer sets up a virtual combat scenario. Through an interactive intuitive interface, called administrator application, the trainer chooses which simulation models the trainees will use. He gives instructions to the SML (layer 4) about which subsystems should be connected to the simulation game. Using ITA, the trainer can go beyond the former training's setup, for example, adding real data from the terrain, deploying a real unit in the field and having them execute orders given by the trainees. In this case, some logistics parameters, e.g. fuel consumption, will be real and not simulated, but others, such as ammunition or casualties, will be simulated using a single platform to coordinate the whole exercise, achieving a LVC simulation facility.

In the opposite situation, a real war scenario, the system operator will connect all the real data sources to the commander GUI. Nevertheless, some simulations models can be made available in order to support the decision making process of the commander and his staff. For instance, close combat simulations can be used in order to compare and contrast different tactics and decide on the best one for carrying out the actual attack; movement and logistic simulations can be carried out to give the commander the tools he needs to decide on the best path to follow.

These two examples try to illustrate how a set of isolated applications can work through common interfaces to produce high interoperability levels and synergy in both wartimes and training situation.

Thus, the interconnectivity view consists of four elements: the service repository, which has the addresses and the access methods for the services; the applications or the components available for each situation; the administration tool used to customize the GUIs for each user and situation; and the GUI itself, that will be the same for all system users, integrating applications as plug-ins.

ARCHITECTURE'S CHARACTERISTICS

Technical Aspects

Considering that ITA was meant for supporting the development of JC2TS, a focus on robustness, interoperability and usability is a must. Particularly for C2 systems, these parameters must be optimized to assure service continuity during real military operations. Nevertheless, other technical aspects such as scalability, maintainability and flexibility, have been considered during ITA's design.

Robustness
Robustness is provided mainly by FISiCo2's distribution layer, ODiN. This layer, which provides the P2P-like communication architecture, avoids the single point of failure problem, enables the multipath feature and offers self-organization ability. Basically, if a node within the network goes down, all other nodes continue working, no matter which was the one that failed.

Furthermore, SML enhances ITA's robustness by enabling service redundancy schemas. Service redundancy allows for more than one node to provide or to publish the same service, generating a fault resilient architecture. When a service is requested by a consumer, the SML returns the address of the node that is providing it. In case of redundancy, the SML provides the address of the first available or the less loaded node. If the service is critical, the nodes can be strategically distributed in order to assure operational continuity.

Interoperability

With regards to interoperability, basing the whole system on an international ontology, i.e. JC3IEDM, assures external interoperability enabling direct interaction with training and C2 systems from other countries. Furthermore, with the development of AAO, internal interoperability is guaranteed. The AAO guarantees semantic interoperability, defining the semantics of the data, unifying the concepts (Turnitsa & Tolk, 2006).

Even though the usage of standard ontologies for all data that flows through the system is an important characteristic, ITA also provides tools for service communication and interoperation. SML establishes a set of rules about how to interact and communicate, enhancing technical interoperability.

Finally, the interconnectivity view provides a fundamental tool called administration's application, which allows administrators to customize the user GUI by combining other applications. This view, therefore, guarantees the syntactic interoperability.

In summary, ITA is ready to interoperate with both internal and external systems with a very low effort due to its interoperability-oriented design.

Usability

Usability is improved by the deployment of a single GUI for all systems. The GUI has a single look & feel and is plug-in oriented. System's administrator configures on-demand applications that the operator will be able to use in his interface.

The basic GUI consists of a Geographic Information System (GIS) and a set of applications which permits interaction between the application and users. This GUI is technically defined in ITA's interconnectivity view.

A coherent interface dramatically improves usability considering that users have to adapt to a single system for both training and C2.

Maintainability

The complexity of a multipurpose system for JC2TS is always high. This complexity makes maintainability difficult, yet the division of responsibilities provided by the AT in general and by each view in particular optimizes this feature, reaching a balance between maintainability and functionality. Maintainability relies therefore on the compliance with definitions delineated by each view.

In addition, as the architecture is correctly separated in subsystems, applications and components, any modification to any of the said elements, as long as the interface remains unchanged, does not affect other parts of the system. This disaggregation enhances maintainability from the development point of view. However, the coordination of a considerable amount of components can still be a difficult part of the maintenance process.

Flexibility

ITA enhances JC2TS' flexibility. The architecture allows for adding new devices, developing new programs and adapting and changing technologies in a transparent way.

FISiCo2 improves both hardware and software flexibility. For instance, if there is a change in the communication technology (layer 1), by just developing a new driver (layer 2) the change will be absorbed by the architecture. Furthermore, the distribution and service management layers contribute to flexibility by breaking the network's hierarchies, managing node's connection and disconnection on-the-fly.

The interconnectivity view gives flexibility to ITA by the administrator's software. The administrator can customize the GUI for each user considering his responsibilities. As this configuration is totally on-demand, the flexibility is improved.

Finally, the composability capacity provided by the components view allows for software changes and adaptations in an effortless way, improving the adaptability.

Scalability

The scalability is guaranteed in all views by the loose dependency of their components. The independency among the layers in FISiCo2, among the components in the components view and among the services in the interconnection view makes ITA extremely scalable.

If new devices are added to the system, it rapidly adapts to the new configuration. Moreover, if any of the selected technologies for the implementation must be deprecated because of a growth in scale of the system, the upgrade is straightforward; any part of the system is easily replaceable as long as the interfaces remain the same.

The core of technical scalability in FISiCo2 is ODiN, that because of its unstructured addressing system, supports a virtually unlimited growth in the number of nodes interacting among them.

With respect to the software scalability, the component view facilitates the development of an unlimited number of atomic pieces of software which are the base for building applications. This feature allows for new information systems to be easily added to the cloud at a very low cost. Furthermore, as applications can be provided by any available technology, a service can be run over a single server or over a grid computing system, enhancing also the scalability in terms of computing power.

Finally, new applications are easy to integrate to the end user application by using the interconnection view. By reconfiguring the user interface, new features can be quickly made available to users.

Economic aspects

From the economic point of view, four major savings can be notice: savings in the development costs, scalability at low cost, licensing and a reduction in training costs.

Considering the optimization in reutilization due to the component-oriented architecture, developing new applications from scratch is extremely easy and, therefore, economic. Once a complete basic library of components is built, new applications will probably reutilize many of them combining this software in different ways to produce new outputs. Also, the unification of the GUI saves development time for future applications. These time-savings provide a budget reduction for developing new applications, exploiting composability and reutilization.

The system's scalability from a technical standpoint impacts the budget as well. Presenting an architecture that can easily evolve and grow produces a reduction of future costs related to technology changes and creation of new components.

All selected technologies for implementing ITA are open-source software. This implies a zero-cost licensing and the possibility of having support and updates through open communities. Furthermore, it reduces not only software costs but also administrative times and cost of public tenders for acquiring privative software.

Finally, a non-technical related economic aspect is a reduction in training costs and the potential savings in lives and materials during real operations. ITA allows for dynamic interaction between reality and simulation, providing flexible realistic scenarios for training. This training can be implemented in a distributed fashion all around the world, reducing the logistic costs associated with moving personnel from one place to other for instruction. Additionally, because of its realism, training is much more efficient and troops are better-trained producing a reduction in costs during real operation, avoiding misuse of material and preventing tactical errors during wartime.

Methodological concerns

Having a single data space, for both C2 and training systems, and the capacity of minimizing the line between reality and simulation make ITA to be an invaluable methodological tool. It simultaneously facilitates more efficient training and a higher command capacity for the army's staff.

The unification of concepts given by the ontologies produces an enormous interoperability capacity between training and C2, allowing for personnel to be trained in more realistic LVC scenarios, without risking neither life nor materials. They can conduct a virtual operation in the same external conditions of a real one in a transparent fashion. This improvement in the training technology produces a methodological advance in the way the personnel is instructed.

On the other side, as users are instructed in the same system they are going to use in the battlefield during wartime, decision making capacity and the conduction ability will be dramatically increased with respect to the present situation.

We can then conclude that two important methodological advances are provided by ITA: the capacity of training personnel anywhere without moving them from one place to other due to its distributed fashion; and the unification of systems concepts for C2 and training, which changes the very conception of training within the army.

CONCLUSIONS

ITA fulfills the proposed goals, setting up a common framework for command and control and training using open standards, assuring interoperability and improving personnel's instruction and commanding capacities.

It is composed of three views which enable internal and external interoperability at all levels, technical, semantic and syntactic. This fact assures future interaction with other forces and other nations, enhancing also the scalability and robustness of the solution.

Additionally, ITA allows for the dynamic interaction between simulation and C2, giving to the Argentine Army a unique tool for lowering the cost of operations and training but, at the same time, improving the quality of the instruction and the ability of personnel during wartime.

The proposed architecture is economically feasible and enhances important features such as usability, flexibility and scalability, which are fundamental for C2 systems. Furthermore, it stands a step beyond the state of the art in the local army, changing the way in which the personnel are trained.

In conclusion, ITA represents a great technical advance for the Argentine Army and an experience that can be taken by other armies which have the same scale problems and are not able to implement bigger frameworks such as TENA, HLA or DIS.

REFERENCES

Brookshier, D., Govoni, D., Krishnan, N., & Soto, J. C. (2002). Overview of JXTA. In *JXTA: Java P2P Programming.* Sams Publishing.

Charfi, A., & Mezini, M. (2004). Hybrid Web Service Composition: Business Processes Meet Business Rules. *2nd International Conference on Service Oriented Computing.*

Combined Communications-Electronics Board. (2008). *Interim Implementation Guide for ACP 123/STANAG 4406 Messaging Services Between Natins.*

Dobson, G. (2006). Using WS-BPEL to Implement Software Fault Tolerance for Web Services. *32nd EUROMICRO Conference on Software Engineering and Advanced Applications* (pp. 126 - 133). IEEE Computer Society.

Doval, D., & O'Mahony, D. (2003). Overlay Networks: A Scalable Alternative for P2P. *Internet Computing - IEEE* , 79-82.

Leymann, F. (2005). The (Service) Bus: Services Penetrate Everyday Life. *Service-Oriented Computing - ICSOC 2005* (pp. 12-20). Springer.

Millmore, C. (2006). *ARCHITECTURAL ALTERNATIVES for the Defence Integrated Simulation and Training Architecture.* The University of New South Wales, Australian Defence Force Academy .

Pullen, M., Levine, S., Heffner, K., Khimeche, L., Schade, U., Frey, M., et al. (2010). Integrating National C2 and Simulation Systems for BML Experimentation. *Euro Simulation Interoperability Workshop* .

Schade, U. (2005). *Towards a Higher Level of Interoperability: Ontology Components for Command and Control Systems.* Research Institute for Communication Information Processing, and Ergonomics, FGAN. NATO.

Tolk, A., & Blais, C. (2005). Taxonomies, Ontologies, and Battle Management Languages – Recommendations for the Coalition BML Study Group. *Spring Simulation Interoperability Workshop.* San Diego, CA.

Turnitsa, C., & Tolk, A. (2006). Ontology Applied – Techniques employing Ontological Representation for M&S. *IEEE Fall Simulation Interoperability.* IEEE CS Press.