The Joint Live Virtual Constructive Data Translator Framework – Interoperability for a Seamless Joint Training Environment

Warren Bizub
Technical Division Chief
US Joint Forces Command Joint Warfighting Center

warren.bizub@jfcom.mil

Derek Bryan
Experimentation Analyst
US Pacific Command – J85

derek.bryan.ctr@pacom.mil

Edward Harvey
President
BMH Associates, Inc

eharvey@bmh.com

ABSTRACT

US Joint Forces Command's (USJFCOM) Joint National Training Capability (JNTC) program has been tasked by the Office of the Under Secretary of Defense (OUSD) for Personnel and Readiness (P&R) to develop and integrate a distributed, seamless Joint Training Environment (JTE) consisting of Live, Virtual, and Constructive (LVC) training technologies. Currently, LVC assets within the JNTC architecture are integrated through loosely defined protocols, multiple types and instances of protocol translators, and a collection of distributed messaging tools, most of which are either ill-defined, require technical expertise to use, or do not provide the level of interoperability necessary to meet OUSD P&R requirements. The Joint Training and Education Capability Group (JTECG) is studying future LVC integration concepts and tools that will address these limitations and result in more interoperable LVC systems. One such tool, the Joint LVC Data Translator (JLVCDT) Framework, will reduce the number and variety of protocol translators used to support Joint training, will support rapid development and integration of LVC protocols through the use of a scalable software architecture, and will act as a system and software platform for further research and development of a Common LVC Architecture (CLA). Developed in conjunction with the United States Services, the JLVCDT will reuse software and interfaces from existing Service and Joint tools within a more scalable and extensible application infrastructure. The JLVCDT will be easier to configure, deploy, and control than existing tools and will require less technical expertise to operate. It is anticipated that development of the JLVCDT will continue through 2007 resulting in an initial operating capability that can be used to support JNTC events (including multi-national partners)by late 2007to early 2008. This paper will describe the vision for the JLVCDT, the technical approach to developing the JLVCDT, and how the JLVCDT supports the CLA concept.

1.0 INTRODUCTION

The Department of Defense (DoD) Training Transformation (T2) program has identified three key capabilities designed to prepare Joint and Service personnel in Joint operations as part of the future warfighting environment: Joint Knowledge Development and Distribution Capability (JKDDC), JNTC, and Joint Assessment and Enabling Capability (JAEC). JKDDC focuses on individual training, JNTC focuses on collective training, and JAEC assesses the validity of both individual and collective training against Joint requirements. The combination of these capabilities will ensure that combatant commander forces are constantly evolving in order to achieve a higher level of Joint force readiness.

Bizub, W.; Bryan, D.; Harvey, E. (2006) The Joint Live Virtual Constructive Data Translator Framework – Interoperability for a Seamless Joint Training Environment. In *Transforming Training and Experimentation through Modelling and Simulation* (pp. 9-1 – 9-8). Meeting Proceedings RTO-MP-MSG-045, Paper 9. Neuilly-sur-Seine, France: RTO. Available from: http://www.rto.nato.int/abstracts.asp.

The T2 Implementation Plan (T2-I Plan) provides guidance to JKDDC, JNTC, and JAEC on how to transform current forces to support future Joint operations. JNTC, the focus of this paper, is instructed to "prepare forces by providing units and command staffs with an integrated live, virtual, and constructive training environment with appropriate joint context that allows accurate, timely, and relevant training in support of specific operational needs [1]." This statement suggests that an integrated Live, Virtual, and Constructive (LVC) training environment is a key enabler in achieving the JNTC mission. As such, the JNTC Technical Management team has been investigating, enhancing, and developing both current and future technologies to provide an integrated, seamless, and interoperable LVC training environment.

2.0 CURRENT CAPABILITIES

Currently, JNTC events use multiple types and instances of protocol translators to integrate the necessary LVC assets. Examples of these interfaces include High Level Architecture (HLA), Distributed Interactive Simulation (DIS), Test and Training Enabling Architecture (TENA), Tactical Data Links (TADIL), Command, Control, Communications, Computers and Intelligence (C4I), and digital and analog voice interfaces, among others. The majority of these interfaces are developed by different vendors, utilize different techniques for achieving their functions, and require technical subject matter experts to install, configure, test, operate, and maintain.

Table 1: JVTSE Gateway Plan

	Translator Types	# of Protocols	# of Instances	# of Operators
w/o JLVCDT	13	40	28	~10
w/ JLVCDT (near-term)	3	40	8	~4
w/ JLVCDT (long-term)	1	40	5	~2

Table 1 provides a summary of the 2005 Joint Virtual Training Special Event (JVTSE) gateway plan. The summary table includes high-level statistics of the gateway resources required to conduct the event both with and without a JLVCDT-like capability. Both near-term and long-term goals are listed for the JLVCDT capability. It is anticipated that this capability can significantly reduce the operations and maintenance costs required to integrate and employ LVC environments. The larger or more complex the LVC environment becomes, the greater the expected benefit. To meet this need, the JTECG Technical Management Team, in conjunction with the Services, is developing the JLVCDT Framework.

3.0 JLVCDT OVERVIEW

The JLVCDT will reduce the number and complexity of translators used in the JTE through the development and employment of an extensible translator framework. The Framework will provide a system and software architecture capable of rapidly integrating, configuring, controlling, and monitoring the execution of LVC Interface Modules (LIMs).

The JLVCDT Framework will be capable of being employed in several different operating modes and configurations. Examples of operating modes include the following:

1. Translation Only – The Translation Only mode provides protocol translation features only. This mode will be used most often when employing a remote JLVCDT instance or when translation performance has been determined to be a priority.

2. Graphic User Interface (GUI) Only – The GUI Only mode provides configuration, monitoring, and control of one or more JLVCDTs. Translations to/from LVC protocols do not occur in this mode.

3. Translation/GUI Combination – If desirable, the Translation and the GUI can be initialized as part of the same process. This mode contains all features described in the Translation Only and GUI Only operating modes.

Examples of configurations include the following:

1. One-to-One – Similar to currently employed translator capabilities, the JLVCDT will be capable of translating bi-directional, one-to-one LVC protocols (e.g., HLA <-> DIS).

2. One-to-Many – The JLVCDT will be capable of translating bi-directional, one-to-many LVC protocols (e.g., HLA <-> DIS/Link).

3. Many to Many – The JLVCDT will be capable of translating bi-directional, many-to-many LVC protocols (e.g., HLA/TENA <-> DIS/Link).

Additional configurations of one or more JLVCDT instances may be required in order to support specific hardware interfaces, network interfaces, and performance requirements. In these cases it will be possible for communication to occur between JLVCDT instances.

JLVCDT operators will have the ability to instantiate, configure, control, and monitor the execution and performance of one or more JLVCDT instances through the GUI on a distributed network. The GUI design will be based on well-understood human-computer interface principles in order to reduce the need for engineering-level support. In addition, the GUI will provide a set of tools and interfaces that support the test, integration, debugging, and run-time performance monitoring and control of LIMs [1].

3.1 Management Strategy

The JLVCDT project began in Fiscal Year (FY) 2005 and initially focused on requirements specification, the identification of reuse opportunities, and high-level design. In addition, JLVCDT goals and objectives were briefed to the Joint community (via JNTC) to ensure that the Services were supportive of the JLVCDT concept and to maximize reuse opportunities. Reuse of existing translators within the JLVCDT Framework is a key component of the JLVCDT management strategy. Acting as the lead for JLVCDT, the technical managers have provided initial feedback on requirements, design, and development priorities to the implementation team.

JNTC's current operations technical integration group has requested a near-term JLVCDT capability that will provide two-way HLA to DIS connectivity in support of JNTC events. The technical integration group currently employs multiple instances of these gateways to support interoperability between the Services' virtual and constructive simulation systems. The JLVCDT is intended to provide equal or better functional capabilities than current translators, but in a more common, usable and open software architecture.

Additional FY06 JLVCDT goals include initial implementations of TENA, TADIL, and C4I LIMs, as well as core architecture development. Of particular significance is the development of a C4I LIM that is capable of reducing redundancies in LVC-generated Common Operating Pictures (COP) such as the Global Command and Control System (GCCS). These redundancies are a result of legacy simulation systems not adhering to the 6016C Link-16 military standard. The JLVCDT will be capable of translating the non-compliant messages into compliant messages for a lower effort and cost than would be required to upgrade the legacy simulations. In addition, the JLVCDT Link-16 capabilities will allow other Link-16 systems to participate in the JTE and it will provide Theater Ballistic Missile (TBM) and space reporting for applicable platforms.

Anticipated long-term JLVCDT goals include the development of a GUI that supports configuration and control of remote JLVCDT instances, the publication of an Application Programmer's Interface (API) that will allow any organization to develop LIMs, and the development of custom LIMs to support program-specific LVC interfaces. Prioritization of long-term JLVCDT goals and objectives will occur each Fiscal Year.

4.0 JLVCDT DESIGN

The JLVCDT system and software architecture is open and extensible. One benefit of this approach is that other organizations will be able to enhance existing LIMs or develop new LIMs. This design feature will help ensure that the JLVCDT will support as many Joint and Service users as possible.

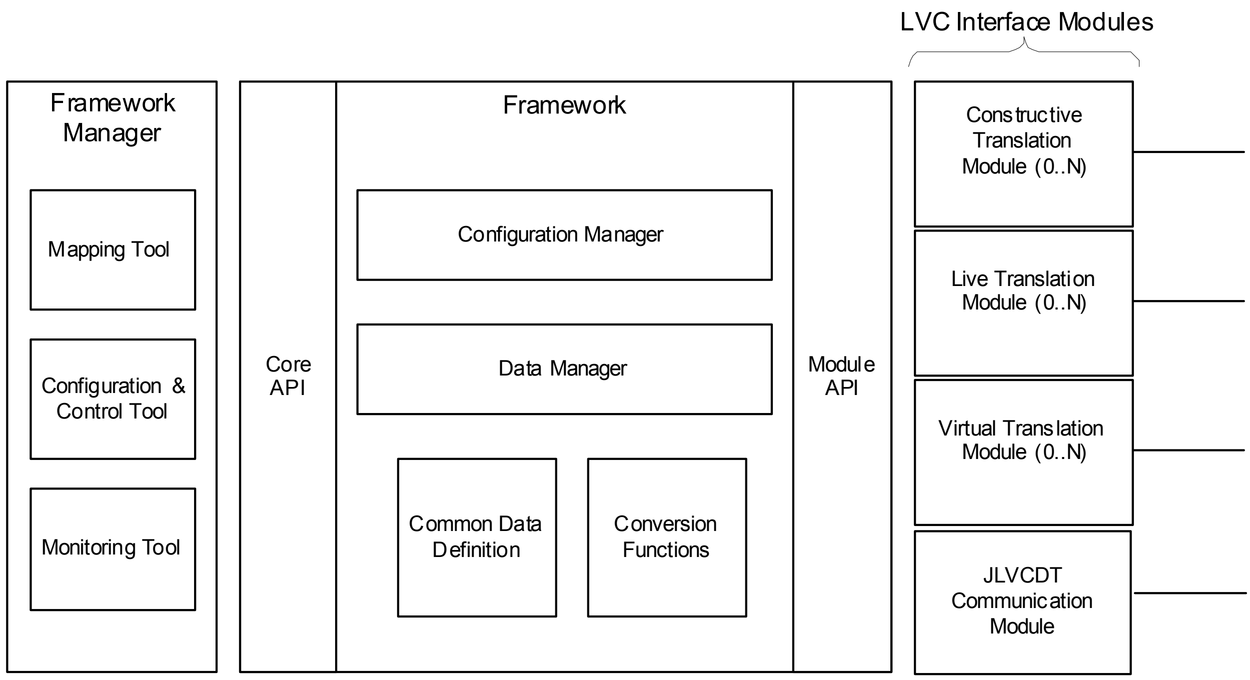

Figure 1: JLVCDT Component Architecture

Figure 1 depicts the component architecture of the JLVCDT Framework. The primary components of the JLVCDT Framework are the following [2]:

1. Framework Manager (FM) – A set of Graphical User Interface (GUI) tools that support the configuration, control, and deployment of the JLVCDT Framework.

    a. Mapping Tool – Maps data elements between protocols

    b. Configuration and Control Tool – Sets system and application parameters and controls run-time performance

    c. Monitoring Tool – Provides feedback on system, application, and network performance

2. Framework – The core of the JLVCDT system responsible for managing and mapping LIM data.

   a. Core API – The API used to integrate the Framework into other applications (e.g., the FM)

   b. Module API – The API used to integrate LIMs into the Framework

   c. Configuration Manager – The component responsible for LIM configuration and instantiation.

   d. Data Manager – The component that passes data between the Framework Core and the Module API/LIMs.

   e. Conversion Functions – Common functions used to convert data and data types between protocols

   f. Common Data Definition – The integrated data model that facilitates all data use, storage, and translation.

3. LVC Interface Modules (LIMs) – One or more instantiations of LVC interfaces.

4.1 LVC Interface Modules

New or existing LIMs may be integrated into the Framework. The determination of whether to use an existing LIM or develop a new LIM will be based on a combination of management and technical criteria including, but not limited to, time and effort required, technical risk, capabilities and limitations, extensibility, complexity, and maintainability.

Each translation LIM will provide basic services for its protocol including registering subscriptions, receiving data from its network interface and providing it to the Framework through the Module API, and packing and sending outgoing messages [2].

In addition to translation LIMs, the JLVCDT will also support application LIMs. The basic services and responsibilities for application LIMs are similar to those of translation LIMs. The primary differences being that the data passed from an application LIM into the Framework is not read from a network interface and no data received by an application LIM from the Framework is sent out via a network interface. Any application that needs access to the data being translated would be a candidate application LIM (e.g., a dead reckoning module, a logger, etc.).

Each LIM will subscribe to data and events through an observer interface. When a message or data set that the LIM is subscribed to comes into the Framework from any other LIM, the interested LIM will be notified so that it may process that message or data set appropriately. Each LIM will run in its own thread to avoid blocking the Framework as a result of network operations and data processing. If desired, any LIM could be separated its own process, e.g., to distribute LIMs onto other machines or to avoid namespace collisions within LIM protocol libraries.

4.2 Technical Benefits

In addition to the high-level, conceptual benefits of the JLVCDT system presented earlier in the paper, there are several low-level, technical benefits to the JLVCDT implementation worth noting. First, through the use of threading and a well-defined software infrastructure, the JLVCDT will support many-to-many translations. This capability will reduce the number of translators, operators, and computers required and will simplify the system and networking architectures of the environments where the JLVCDT is used. Next, through the use of the JLVCDT mapping tool, data mappings for each protocol will be exposed to JLVCDT users rather than being hidden in source code. In addition to the test, integration, and debugging benefits of this feature, this allows most mapping updates between protocols to be made by JLVCDT operators, rather than developers. This capability will reduce the time and level of expertise required to integrate and verify distributed LVC environments.

Perhaps the largest benefit of the JLVCDT implementation is the use of the Common Data Definition (CDD). The use of a CDD is desirable because it hides unimportant (and many times complex) internal details of LIMs, while simultaneously exposing their capabilities to other LIMs. Therefore, in order to integrate a new LIM, a developer need only have expertise in his or her LIM and how it maps to the CDD. This approach removes the requirement for a LIM developer to have detailed knowledge of other LIM data elements and mappings and allows rapid construction of many-to-many translators.

In addition to the technical benefits described above, the JLVCDT Framework will be developed using modern tools and techniques that will result in a stable, extensible, and usable system and software implementation. These benefits may not be as quantifiable as some of the others mentioned above but they are no less significant.

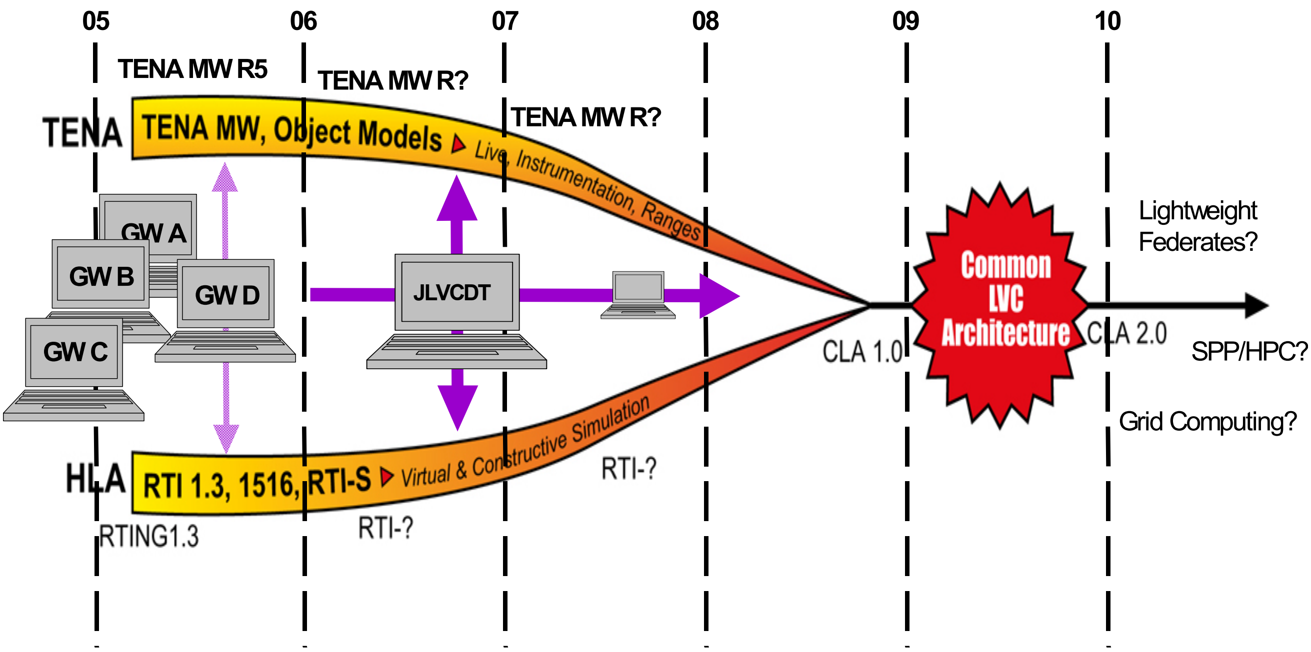

Figure 2: Common LVC Architecture Vision

5.0 COMMON LVC ARCHITECTURE

The JLVCDT will support the transition and transformation from JNTC initial operating capability to final operating capability. However, the implementation of the JLVCDT should not and will not preclude further research and development of integrated LVC concepts. Examples of additional integrated LVC concepts that the JTECG technical managers are currently investigating include eXtensible Battle Management Language (XBML), Command and Control Information Exchange Data Model (C2IEDM/JC3IEDM), Global Information Grid (GIG) and GIG Net-Centric Enterprise Services (NCES), web services, and others. Some of these concepts will increase the interoperability of data between systems while others will increase the availability of data. In either case, a higher level of interoperability between LVC systems is possible in the form of a Common LVC Architecture (CLA) as depicted in Figure 2.

A CLA will move beyond on-the-wire interoperability towards semantic interoperability and will be supported by on demand networks and services. One of the key tenets of semantic interoperability is that data is not only defined, but is comprehended by all systems that need to use it. The responsibility of data comprehension is shared between the data and the system, rather than with the system and the system developer. Transitioning to a CLA implementation will not be easy or inexpensive but it is something that must be investigated in order to adequately support JNTC T2 goals and objectives.

The JLVCDT is being designed to meet current LVC interoperability goals and those of anticipated future transitional capabilities. Through the use of its mapping tools and CDD, the JLVCDT is exposing the details of LVC interoperability to a larger audience. As additional LIMs are added to the JLVCDT, the CDD will expand, as will the mappings, the conversions, and the rules for using that data. Collectively, this information is defining the comprehension associated with the data that is currently hidden in the numerous types and instances of translators used in the JTE. Furthermore, developers will be able to use the JLVCDT to add LIMs in support of the CLA concept (e.g., C2IEDM). For these reasons, we believe that the JLVCDT is an excellent tool for defining, experimenting, and testing concepts in support of a CLA construct.

6.0 REFERENCES

[1] OUSD P&R, Director, Readiness and Training Policy and Programs (2005), "DoD Training Transformation Implementation Plan."

[2] Bogedain, Jerry; Mobley, Ken; and Teer, Brian (2005), "JLVCDT Architecture Description Document."